Azure DevOps & Web Deploy doc created 02/05/2020 - Today is 06/02/2023

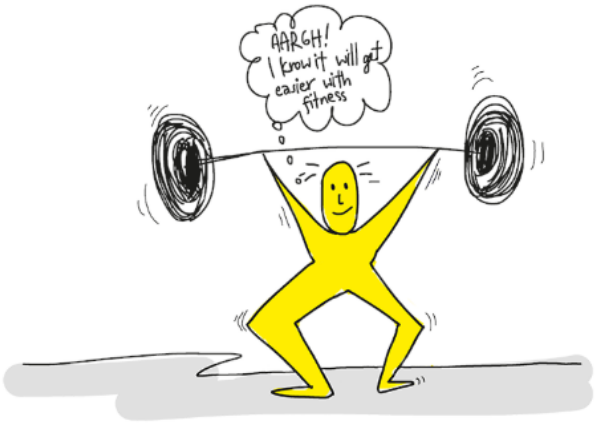
"A process designed to help teams work together more effectively"

The goal is to shorten cycle time. Start with the release pipeline. How long does it take you to deploy a change of one line of code or configuration?

A DevOps Definition of Done is working software collecting telemetry against the intended business objectives.

DevOps May Hurt at First

If it hurts, do it more often. Just like going to the gym, adopting new practices is likely to hurt at first. The more often you exercise the new practices, the easier they will become. And just like training at the gym, where you exercise large muscles before small muscles, adopt practices that have the greatest impact first and cross-train to develop synergy.



Azure Portal, Microsoft Teams, Azure DevOps: What's the connection?

- **Microsoft Teams** is a dedicated hub for teamwork, that brings your teams, content, conversations, chats, meetings, files, apps and tools from across Office 365 and Azure DevOps together into one place
- **DevOps** is more of a process of combining the developers and operations teams. It improvises on the communication, collaboration, integration between teams making them a single unit. This allows for quicker identification of bugs and fixes.

What is Azure DevOps?

https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops

- Agile development is building products in small incremental pieces so you can get real feedback along the way.
- Agile tools are project management tools designed to support an agile methodology, - Scrum, Kanban, Scrumban, or other hybrid agile methods.
- Some argue that Kanban is about getting things done, and Scrum is about talking about getting things done.

- Kanban is primarily concerned with process improvements, while Scrum is concerned with getting more work done faster.
- Hello World

****The "Work" Development Methodologies or Managing Process Work item types (WITs)**

Whether describing work to be done, impediments to release, test definitions, or other key items, Work items are the workhorse of modern projects.

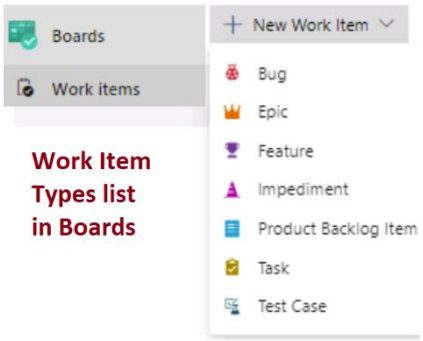
- Why choose using Agile or Waterfall project flow methods to drive product development.
- Use an Agile process called Scrum is to focus on delivering the business value in the shortest time.
- Difference between a Kanban board and a Scrum board.
- [About default processes and process templates - Azure Boards | Microsoft Learn]
- **Setting - Process selection chooses Work item types (WITs) – basically a column in a spreadsheet representing Bug, Feature etc.
- Board uses columns like Start In progress etc – to move WIT thru a time measurable Status of completion for each work item.

Process selection chooseW ork item types (WITs)

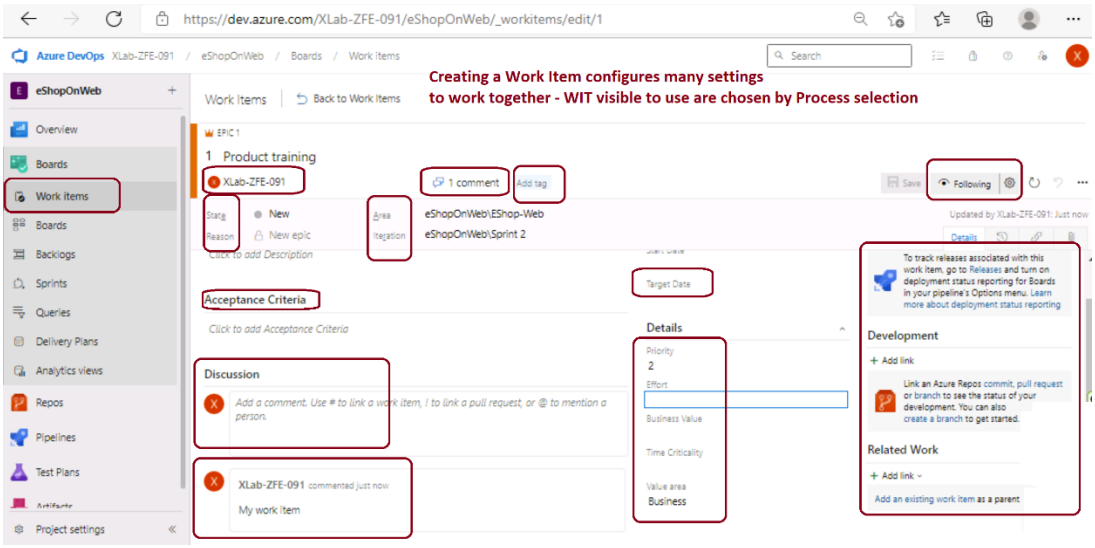
Main distinctions between the WITs and states used by the four default processes.

Tracking area	Basic	Agile	Scrum	CMMI
Workflow states	• To Do • Doing • Done	• New • Active • Resolved • Closed • Removed	• New • Approved • Committed • Done • Removed	• Proposed • Active • Resolved • Closed
Product planning (see note 1)	• Issue	• User story • Bug (optional)	• Product backlog item • Bug (optional)	• Requirement • Bug (optional)
Portfolio backlogs (2)	• Epic	• Epic • Feature	• Epic • Feature	• Epic • Feature
Task and sprint planning (3)	• Task	• Task • Bug (optional)	• Task • Bug (optional)	• Task • Bug (optional)
Bug backlog management (1)	• Issue	• Bug	• Bug	• Bug
Issue and risk management	• Issue	• Issue	• Impediment	• Issue • Risk • Review

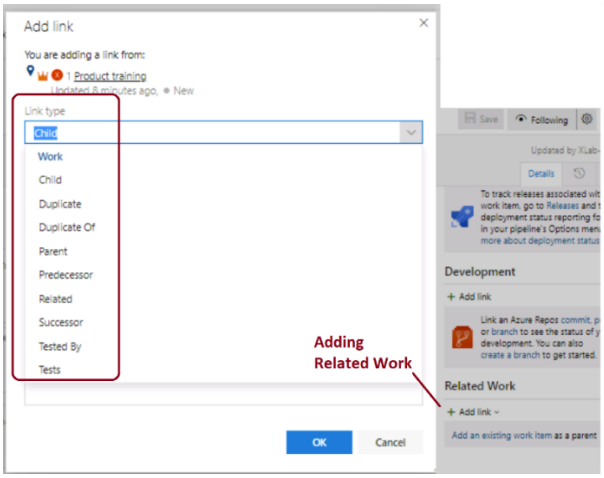
-



- **Note**: The work item form includes all of the relevant work item settings.
- This includes details about who it's assigned to, its status across many parameters, and all the associated information and history for how it has been handled since creation.
- One of the key areas to focus on is the **Related Work** lower right corner in screenshot.

- 

- Related Work

- 

- [Organize your product backlog in Azure Boards - Azure Boards | Microsoft Learn]

- Directional Link

- [Link work items to support traceability in Azure Boards - Azure Boards | Microsoft Learn]

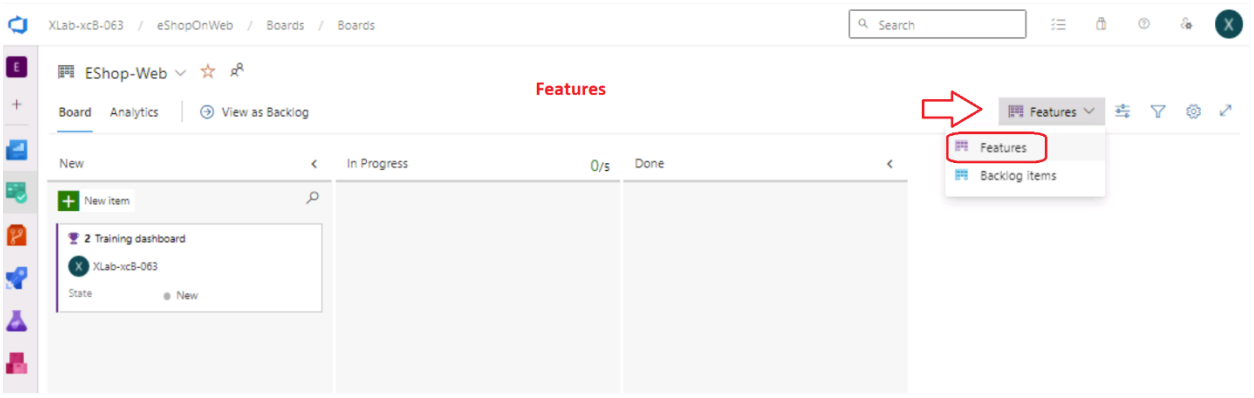
- Link user stories and issues to other work items in Azure Boards - Azure Boards | Microsoft Learn

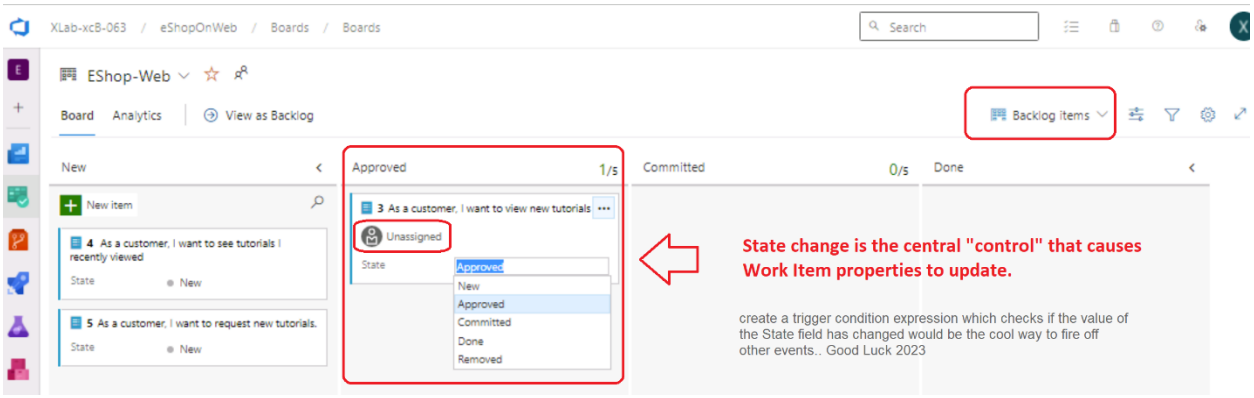
- Manage the various relationships between work items and other artifacts, like builds, commits, pull requests, and more.

- After you've added features or epics to your portfolio backlog, organize your backlog by mapping backlog items.

- Backlogs are automatically created when you create a project or add a team. Each team has access to their own product, portfolio, and sprint backlogs as described in About teams and Agile tools.

- Backlog Items is not the same as View as Backlog



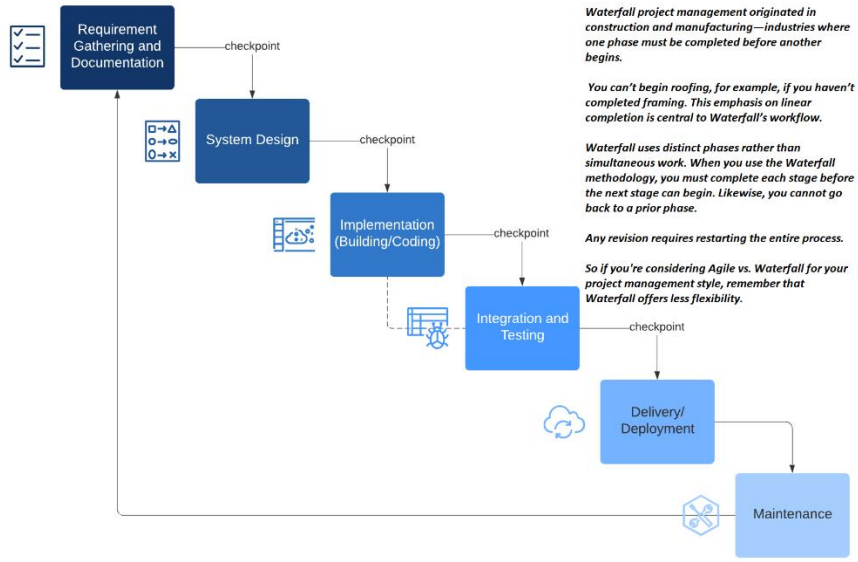


Work Item State Trigger – when WIT moves across columns of board

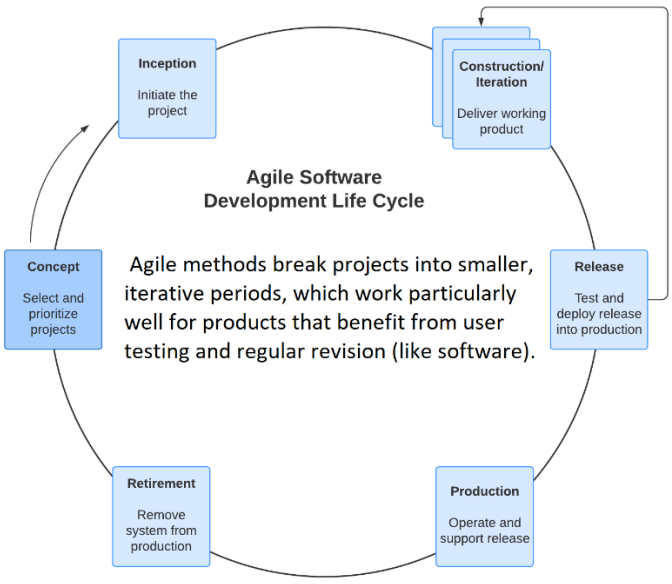
1. Azure DevOps Work Item state trigger – Expiscornovus
2. Customize the workflow of an inherited process - Azure DevOps Services | Microsoft Learn
3. Query by account, user, workflow, or board changes in Azure Boards - Azure Boards | Microsoft Learn
4. Azure DevOps - Connectors | Microsoft Learn – WIT updated trigger

5. Product backlog item (PBI) work item that is a child of the feature and shares its area and iteration.
6. Capacity tab of the Sprints: Capacity per day textbox, type 1 represents 1 hour of development work per day.
7. Work details
8. Remaining Work estimate to reflect the amount of time expected for each task.

Waterfall uses distinct phases rather than simultaneous work. When you use the Waterfall methodology, you must complete each stage before the next stage can begin. Likewise, you cannot go back to a prior phase. Any revision requires restarting the entire process

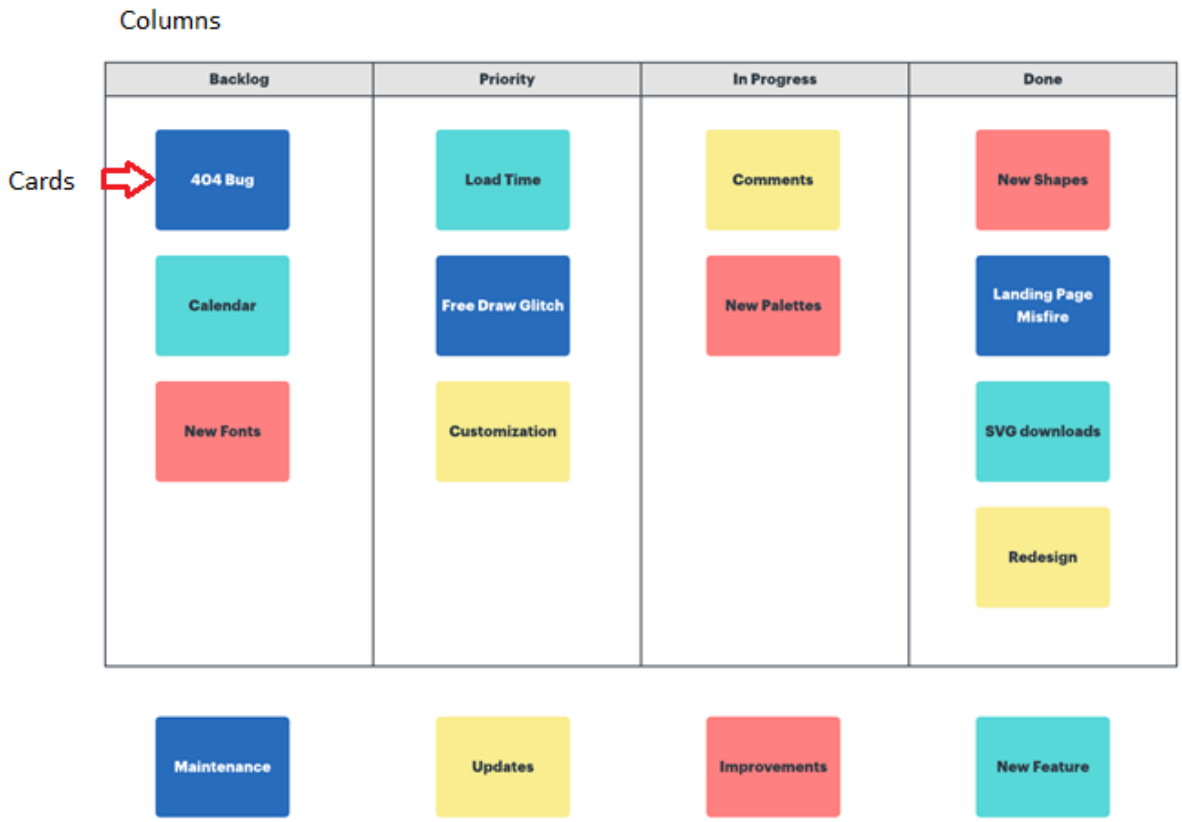


Agile project management refers to a category of methodologies that includes Scrum and Kanban; the fundamental differences between Agile and Waterfall project management is Agile focuses on adaptive, simultaneous workflows vs. linear workflows.



Kanban project management methodology is a visual system for managing work and improving workflow using a visual representation of process and work in order to identify and correct bottlenecks.

Everything is represented on a Kanban board. This board is traditionally a physical object—most often a whiteboard—that is easily viewable by the team.

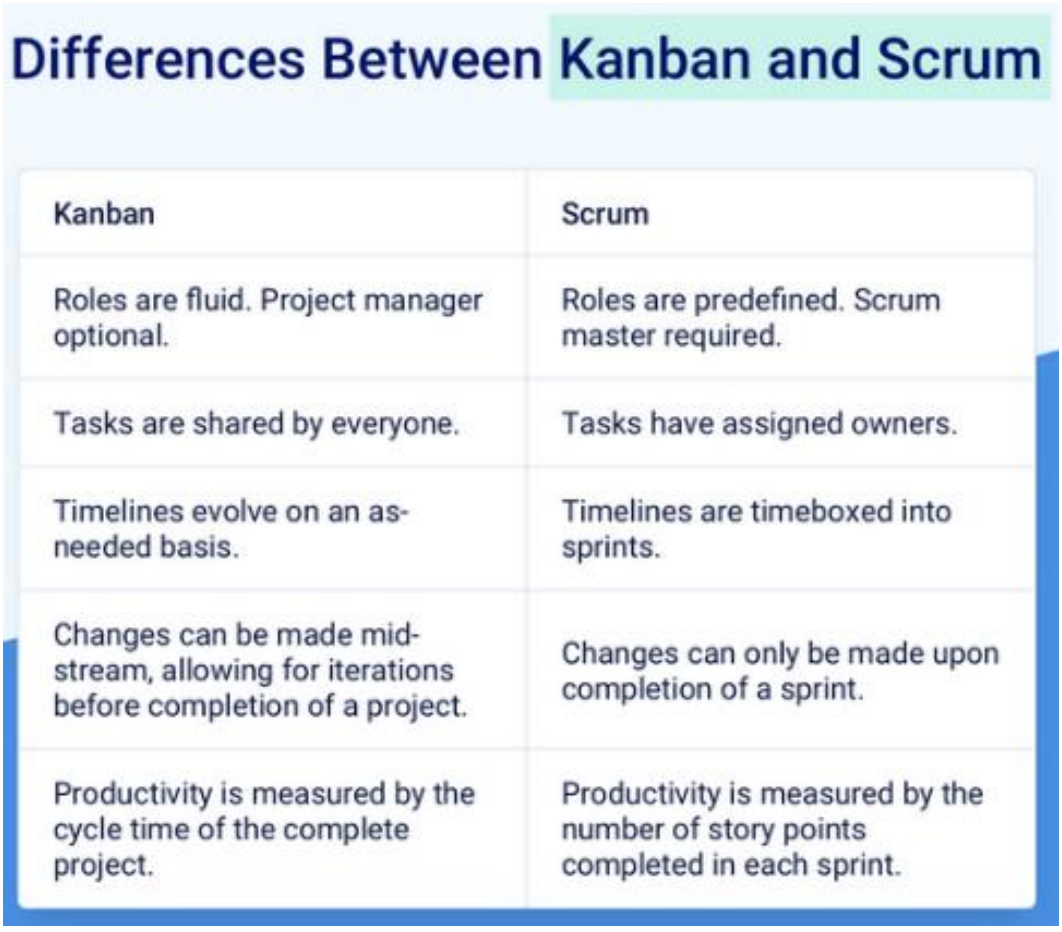


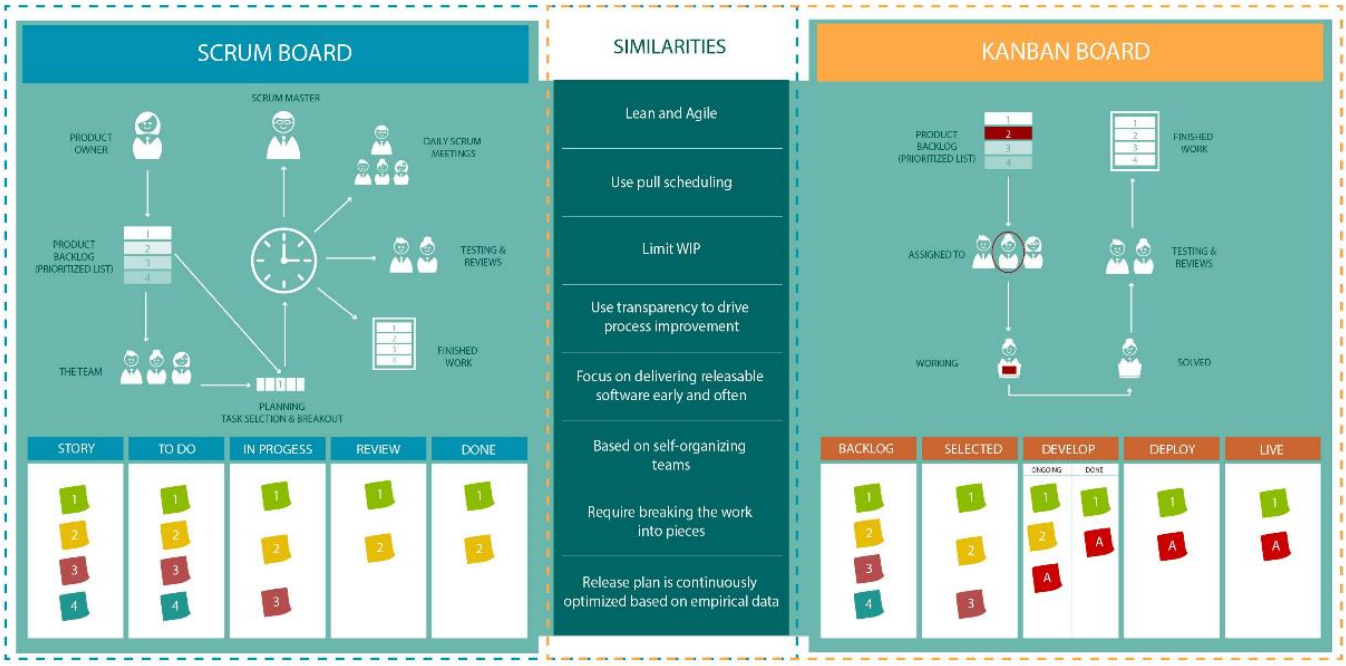
Scrum

- An agile framework; with its focused on a shorter and more frequent software delivery cycle. It mainly manages needs to gain a favorable outcome. It lays down a path on how operations and developer teams manage the needs in small batches and rapid releases.

1. Like Scrum, Kanban encourages work to be broken down into manageable chunks and uses a Kanban Board (very similar to the Scrum Board) to visualize that work as it progresses through the work flow.
    a. Where Scrum limits the amount of time allowed to accomplish a particular amount of work (by means of sprints),
    b. Kanban limits the amount of work allowed in any one condition (only so many tasks can be ongoing, only so many can be on the to-do list.)
2. Used by many kinds of teams and focuses on providing a visual representation of an agile team's workflow.
3. Primarily concerned with process improvements.

How is Kanban different from Scrum? – what is different in DevOps software operation between either choice?





Scrum Board and Kanban Board differences

https://www.cprime.com/2015/02/3-differences-between-scrum-and-kanban-you-need-to-know/

On a Scrum board

1. The columns are labeled to reflect
    a. Periods in the work flow beginning with the sprint backlog and ending with whatever fulfills the team's definition of done.
2. All the stories added to the board at the beginning of each sprint should be found in the final column at the end of that sprint or the sprint was unsuccessful.
3. After the sprint retrospective, the board is cleared and prepped for the next sprint.

On a Kanban board

1. The columns are labeled
    a. To show work flow states, but with one vital difference:
    b. Also publish the maximum number of stories allowed in each column at any one time.
        i. This enforces the team-determined limitations Kanban prescribes for each condition.
    c. Since each column has a limited number of allowed stories and there are no required time boxes (such as sprint length),
        i. There is no reason to reset the Kanban board as work progresses.
        ii. It will continue to flow for as long as the project continues,
        iii. With new stories being added as the need arises
        iv. And completed stories being re-evaluated should it be necessary.

Scrum is a tool used to organize work into small, manageable pieces that can be completed by a cross-functional team within a prescribed time period (called a sprint, generally 2-4 weeks long).

- An agile process specific to software development teams that allows us to focus on delivering the business value in the shortest time. (end dates defined – Done)
- Typically tackles complex knowledge work such as software development.
- Concerned with getting more work done faster.
- Scrum is a framework for developing and sustaining complex products.

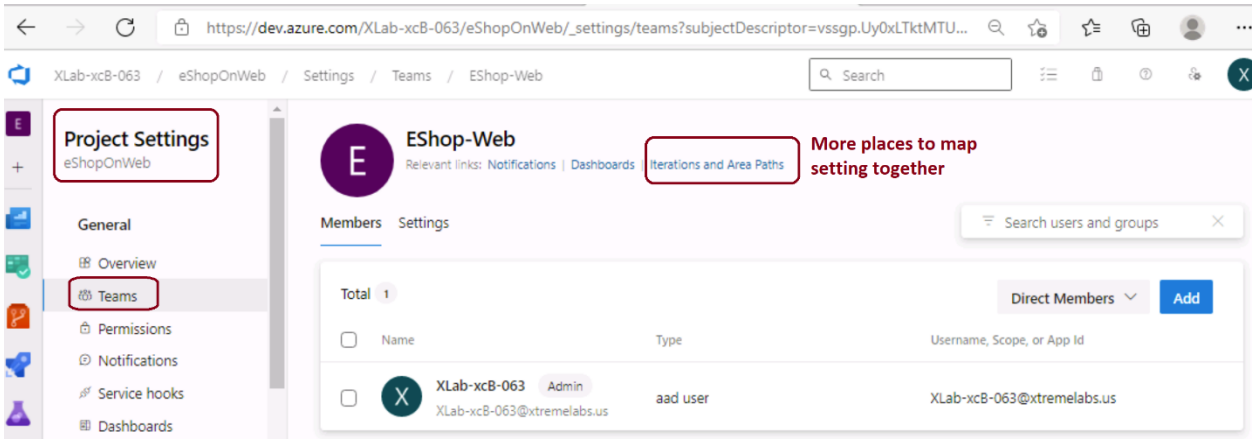
To plan, organize, administer, and optimize this process, Scrum relies on at least three prescribed roles:

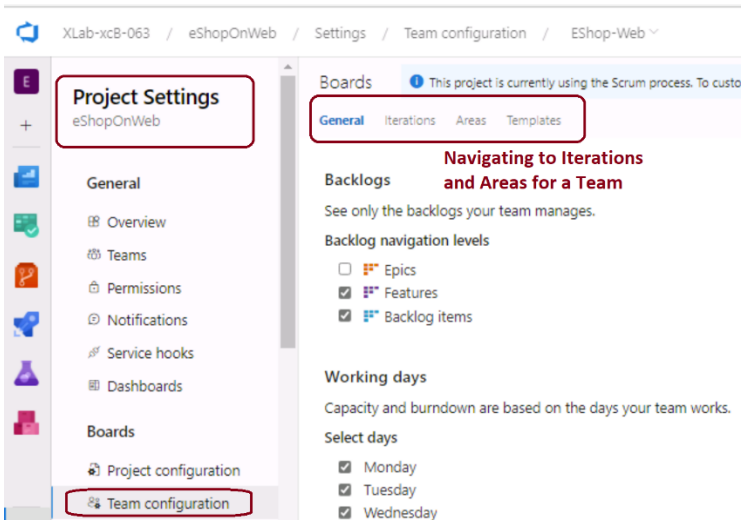
1. the Product Owner (responsible for initial planning, prioritizing, and communication with the rest of the company),
2. the Scrum Master (responsible for overseeing the process during each sprint),
3. Team Members (responsible to carry out the purpose of each sprint, such as producing software code.)
4. Another common tool used by scrum teams is the Scrum Board –
    a. A visual representation of the work flow
    b. Broken down into manageable chunks called "stories"
    c. With each story moved along the board from the "backlog" (the to-do list), into work-in-progress (WIP), and on to completion.

Sprints

1. Scrum uses two-week sprints to get work done.
2. Sprints are planned in advance, executed, and then reviewed at the end of the two-week period.
3. During sprint planning, the team creates a sprint backlog.
4. The team completes these backlog tasks during the sprint, managing the work among themselves.
5. Team members also hold a Daily Scrum which is a 15-minute time-boxed event for the Development Team to synchronize activities and create a plan for the next 24 hours.
    a. The Daily Scrum is held every day of the Sprint.
6. Contributors discuss any potential roadblocks interfering with project success.
7. Review the previous day's work and plans for the upcoming day's tasks.
8. Scrum meeting ensures the team works collaboratively and stays in sync.
9. [How are area and iteration paths used? - Azure DevOps | Microsoft Learn]

**Settings



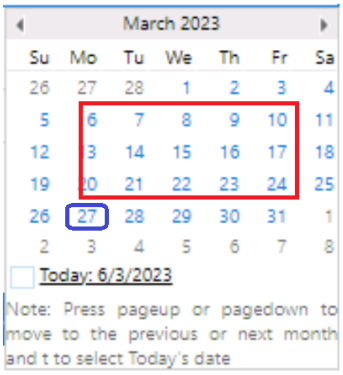


10. Dates
    a. Begin and end each sprint
    b. Specify the Start Date as the first work day of last week, and count 3 full work weeks for each sprint.
    c. For example, if March 6 is the first work day of the sprint, it goes until March 24th.
    d. Sprint 2 starts on March 27, which is 3 weeks out from March 6.

Sprint 1 - March 6 - until March 24th

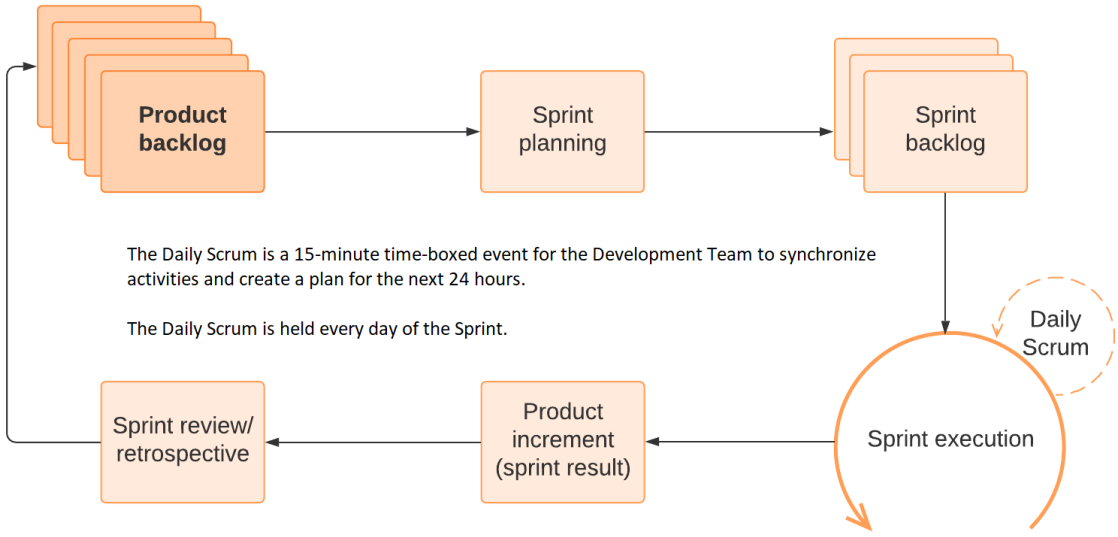
Sprint 2 starts on March 27

3 weeks out from March 6



e.

11. Area paths
    a. Group work items by team, product, or feature area.
    b. The default setting for all teams is to exclude sub-area paths.
    c. Include sub-areas so that the team gets visibility into all of the work items from all teams.
    d. Not include sub-areas automatically removes work items from their view as soon as they are assigned to one of the teams.
12. Iteration paths
    a. Group work into sprints, milestones, or other event-specific or time-related period.
13. [About teams & Agile tools - Azure DevOps | Microsoft Learn]



Scrum Master

1. A Scrum master links the team to the product owner.

2. Before beginning a project, the Scrum master works with the product owner to define requirements.
3. They then help the team plan sprints.
4. Once a sprint begins, the Scrum master helps remove any roadblocks that arise.
5. A Scrum master facilitates work rather than managing it.
6. The Scrum methodology encourages teams to manage their own productivity; the Scrum master merely helps them do so.

Burndown charts

1. Scrum uses a burndown chart during sprints to let team members see progress at a glance.
2. Rather than displaying completed tasks, the burndown chart visualizes what's left to be done.
3. Is continuously updated to help team members manage their workflow.

Creating the Kanban board

1. At its most basic, the Kanban board consists of clearly labelled columns that denote work yet to be done, work in progress, and finished work.
2. Include horizontal rows to designate swimlanes for work.
    a. Used in process flow diagrams, or flowcharts, that visually distinguishes job sharing and responsibilities for sub-processes of a business process.
3. All in-progress columns must have an explicit limit to the number of items that can be in them at any given time.
4. Items will physically move through the Kanban board's columns as they move through the work process.
5. These items are represented by Kanban cards.

Creating Kanban cards

1. Kanban cards are generally sticky notes, index cards, or other similarly sized papers that represent a project.
2. Record pertinent information on these cards, such as due dates, relevant team members, or dates that the card moves columns on the Kanban board.
3. Avoid including too many details — the board and cards should serve primarily as a visual representation of work rather than a record of project details.

Using the Kanban board

1. Cards are initially placed in the "to do" column, commonly called the "backlog."
2. When work begins on a project, the card moves to the first in-progress column.
3. *Work is always pulled through the Kanban board rather than pushed*
    a. A push would occur when the delivering team is finished with their task, regardless of the receiving team's bandwidth.
    b. A pull depends entirely on the receiving team's bandwidth and happens according to each column's WIP limits.

WIP limits

1. The Kanban methodology requires strict limits on the amount of ***work in progress*** at any given time.
2. Teams assign a limit to the number of cards in any active-work columns.
3. When the limit is met, no new work can enter the column until a task is completed and moved to the next column.
4. This system helps teams identify bottlenecks, and it encourages individual contributors to rally together to fix them.

Creating Bottlenecks

1. Example, a limit of two items in your dev column, nothing can be added to that column until there is one item or no items left.
    a. If design has finished a project but dev is already at capacity, the project continues to sit with design, creating a bottleneck.
    b. When dev finishes a project and it moves to the next column, they can then pull a card from design.
2. As bottlenecks happen, your team should work together to eliminate them. In order to effectively use the Kanban board, you must also consider how to adjust your process to prevent future bottlenecks.
3. As your team visualizes its workflow, problems will become readily visible. Your team should adjust process as necessary, essentially using experimentation to perfect your workflow. You might have to adjust the column limits, rethink swimlanes, or make other changes.
4. As you refer to the Kanban board, you can make small, effective changes based on the data in front of you.

Continuous improvement

1. The goal of the Kanban methodology is to improve the team's process.
2. The team meets periodically to discuss changes that need to be made, and the data displayed on the Kanban board guides these discussions.
3. When held regularly, these meetings help the team continuously correct and adjust their process. This improves workflow without making any sudden or dramatic changes, ensuring easy Kanban implementation on nearly any team.

Basic principles of Kanban

As you follow the process described above, make sure to keep these principles in mind.

1. Visualize your workflow.

Keep your Kanban board highly visible, and update it frequently. An accurate visible representation of your process allows you to spot problems and address them, in addition to helping your team improve communication regarding project status.

2. Limit work in progress.

Set a WIP limit for each column, and stick to it. *This limit prevents a loss of efficiency due to multitasking or re-prioritizing*. You will likely have to experiment to find this appropriate limit, but a common rule of thumb is to start with two items for each person working in that column.

3. Improve continuously.

- The Kanban methodology works to achieve *kaizen*, or continuous improvement.

- *Kaizen* is a concept referring to business activities that continuously improve all functions and involve all employees from the CEO to the assembly line workers. Kaizen (改善) is the Sino-Japanese word for "improvement"

- You should use your board to identify problems and make changes to fix them.
- Some users suggest that Kanban is more accurately a process refinement system rather than a project management system.

Essential practices of Kanban

If you have decided to incorporate Kanban methodology into your project management style, review these four steps to ensure that this workflow increases efficiency on your team.

1. Make explicit process policies.

Kanban methodology only works if everyone on the team knows how it functions. Once you define WIP limits, make them visible on the board. Ensure that all team members understand how work is pulled through the board, and establish expectations for addressing bottlenecks.

2. Manage the flow.

Don't simply visualize your workflow—manage it. Define important metrics, such as how long it takes an item to move through the board, and use these metrics to improve your process. When you see bottlenecks, take steps to prevent them in the future. Actively engage with the information you get from the Kanban board rather than passively observing it.

3. Use feedback loops.

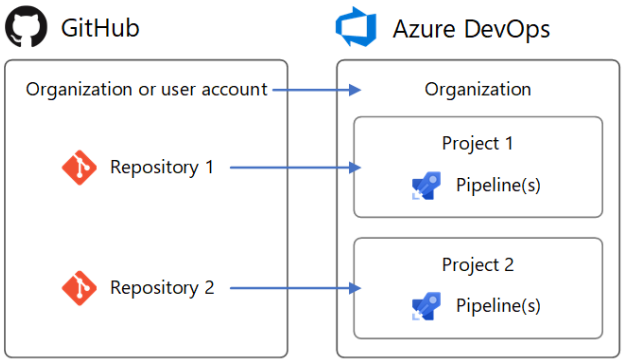
Decide upon regular intervals to review the data on your Kanban board. Set up a review at a defined time (every two weeks, for example), as well as whenever problems become clear on the board, such as bottlenecks. Regular feedback ensures continual improvement.

4. Improve collaboratively.

Kanban methodology relies heavily upon the investment of team members to make it work. So work with your team to address problems and correct bottlenecks. This helps you improve processes in ways that a manager alone might not consider, and it keeps team members invested in making things work.

Social Coding at GitHub: Why do I - Branch, Fork, Clone, Pull, Merge, Commit

- GitHub and Azure Pipelines are two independent services that integrate; have their own organization and user management; Replicate the organization and users from GitHub to Azure Pipelines.
- GitHub is organizations and user accounts that contain repositories vs. Azure DevOps' is organizations that contain projects.



- To set up an identical structure in Azure DevOps:

  1. Create a DevOps organization named after your GitHub organization or user account. It will have a URL like `https://dev.azure.com/your-organization`.
  2. In the DevOps organization, create projects named after your repositories. They'll have URLs like `https://dev.azure.com/your-organization/your-repository`.
  3. In the DevOps Project, create pipelines named after the GitHub organization and repository they build, such as `your-organization.your-repository`. Then, it's clear which repositories they're for.

  Following this pattern, your GitHub repositories and Azure DevOps Projects will have matching URL paths. For example:

Service	URL
GitHub	`https://github.com/python/cpython`
Azure DevOps	`https://dev.azure.com/python/cpython`

What is the app structure that allows granular code changes?

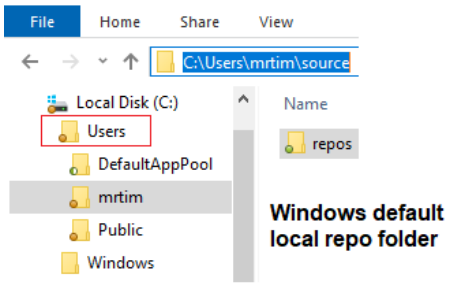
1. Repository or repo
   a. Build GitHub repositories - Azure Pipelines | Microsoft Learn
      i. Azure Pipelines can automatically build and validate every pull request and commit to your GitHub repository.
   b. LFS - Work with large files in your Git repo - Azure Repos | Microsoft Learn
   c. A location that stores files for development – in this case GitHub Version Control
   d. A place where the history of your work is stored
   e. Each Git repo has its own set of permissions and branches to isolate itself from other work in your project.
   f. Types of GitHub accounts - GitHub Docs
   g. GitHub's products - GitHub Docs
   h. Where is your repo? Git, Azure, Local – other?

i. Azure Repo has folders

The repository is organized the following way:

- **.ado** folder contains Azure DevOps YAML pipelines
- **.devcontainer** folder container setup to develop using containers (either locally in VS Code or GitHub Codespaces)
- **.azure** folder contains Bicep&ARM infrastructure as code templates used in some lab scenarios.
- **.github** folder container YAML GitHub workflow definitions.
- **src** folder contains the .NET 6 website used on the lab scenarios.

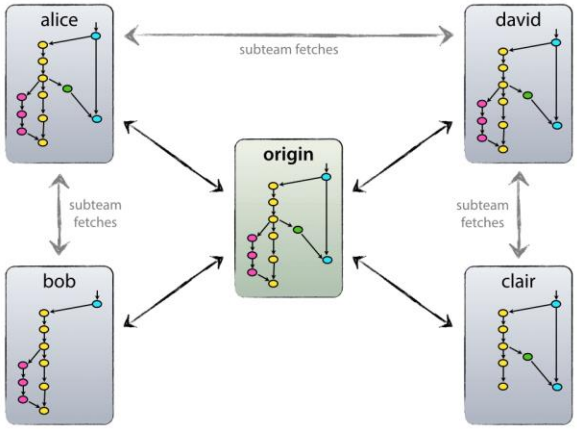
j.

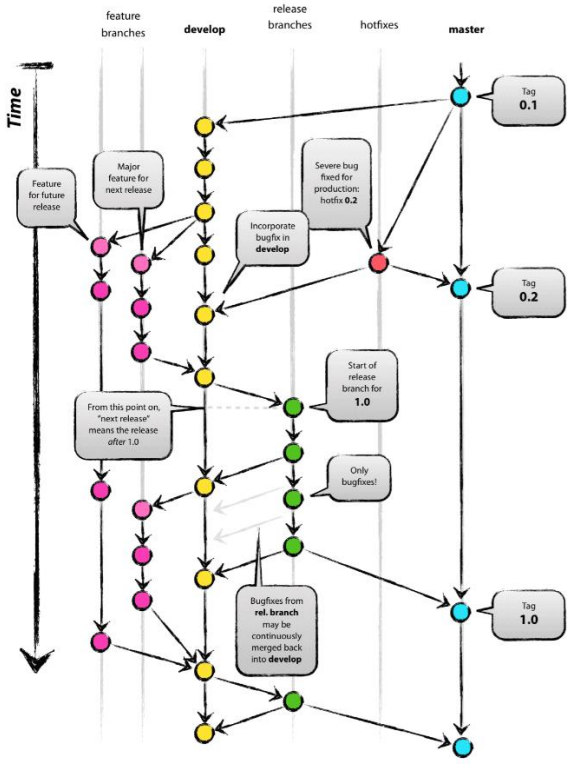


k. C:\Users\UserName\source\repos

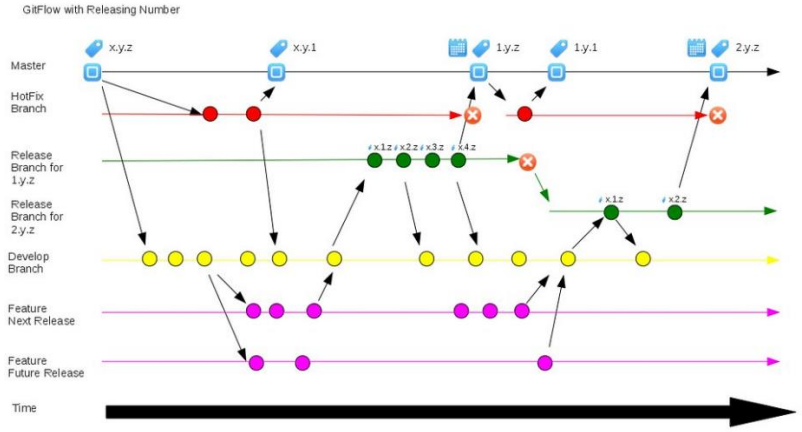
2. Mono-repo
   a. Source code is kept in a single repository
3. Multiple-repo
   a. Each project has its own repository
4. Master (repo)
   a. The base or root copy of the application
5. Clone
   a. To work locally on your PC clone the project.
6. Partial clone
   a. A performance optimization that "allows Git to function without having a complete copy of the repository.
   b. The goal of this work is to allow Git better handle extremely large repositories."
7. Commit
   a. Is a snapshot of all your files at a point in time.
   b. If a file has not changed from one commit to the next, Git uses the previously stored file.
   c. This design *differs* from other systems which store an initial version of a file and keep a record of deltas over time.
8. Branch
   a. Branching is the way to work on different versions of a repository at one time.
   b. Instead of copying files from directory to directory, Git stores a branch as a reference to a commit.
   c. WorkflowUndo mistakes and errors with a workflow

d. A branch represents the tip of a series of commits not a container for commits.
e. You can keep working on your branch regardless of the work that is happening in other branches.
f. A common practice in a lot of companies is to have one branch where the code is ready to shipped and another branch for testing.
g. Committing changes to a branch will not affect other branches
h. Can share branches with others without having to merge the changes into the main project.

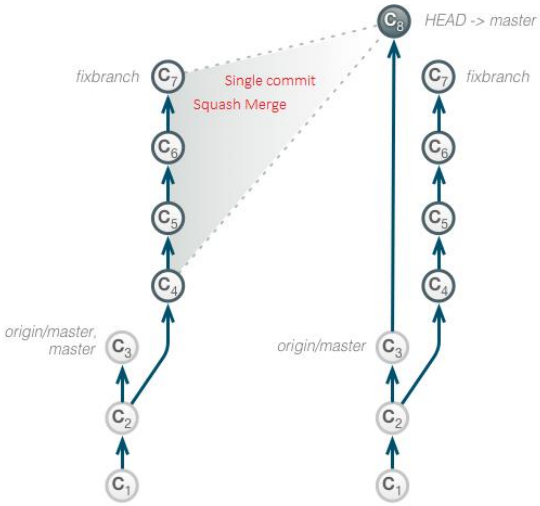
9. Branching Workflow strategy
   a. Use feature branches for all new features and bug fixes.
   b. Merge feature branches into the master branch using pull requests.
   c. Keep a high quality, up-to-date master branch.
   d. https://nvie.com/posts/a-successful-git-branching-model/
   e. 

   f. 

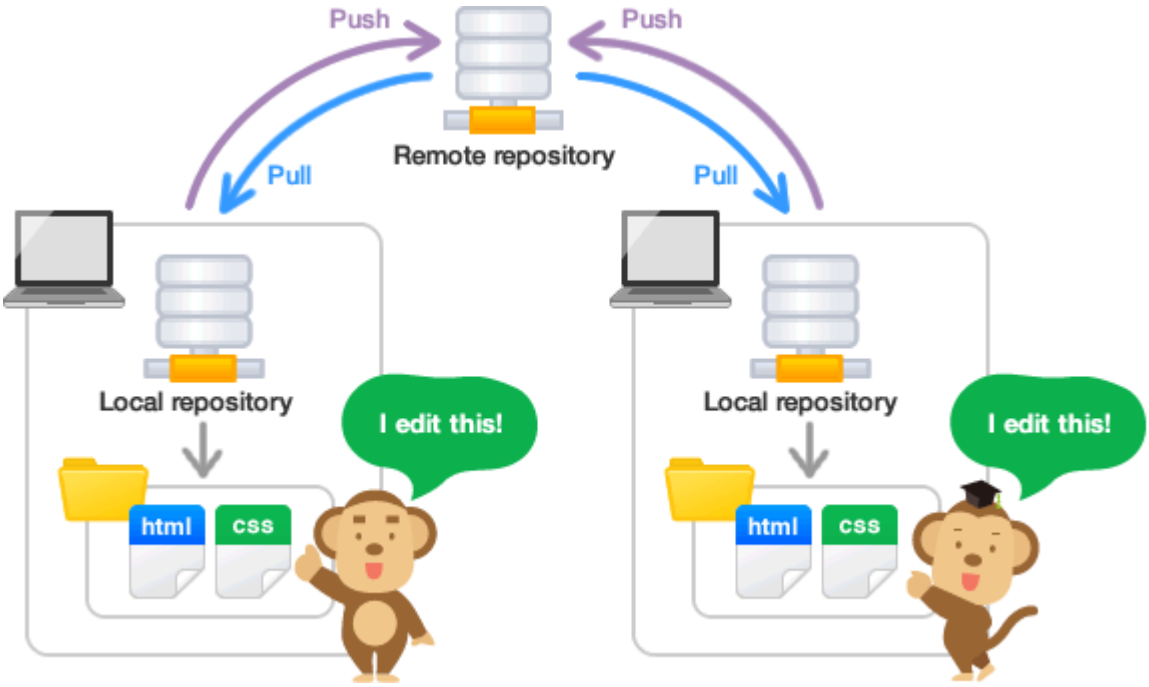
g. Distributed GIT: https://git-scm.com/book/en/v2/Distributed-Git-Distributed-Workflows#ch05-distributed-git

h.


10. GitFlow
    a. Enable parallel development by isolating new development from finished work.
    b. New development, features and non-emergency bug fixes are done in feature branches
    c. Only merged back into main body of code when the developer(s) when ready for release.
11. GitHooks
12. GitVersion
    a. A tool to help achieve Semantic Versioning
    b. Calculates the version based on:
        i. The nearest tag
        ii. The commit messages between this commit and the nearest tag
        iii. The name of the branch
13. A fork is a copy of a repository.
    a. Where is the fork – Local or off premise?
    b. Forking is at the core of social coding at GitHub.
    c. Forking a repository allows you to freely experiment with changes without affecting the original project.
    d. Most commonly, forks are used to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea.
    e. A successfully forked project repository only exists on GitHub.
14. Pull requests
    a. Let you tell others about changes you've pushed to a branch in a repository on GitHub.
    b. Once a pull request is opened, you can discuss and review the potential changes with collaborators
    c. Add follow-up commits before your changes are merged into the base branch.
    d. Merge a pull request
15. Fork-based pull request workflows
    a. Allows anybody to contribute
16. Inner Source
    a. Brings all the benefits of open-source software development inside your firewall
17. Rebasing

   a. The process of taking all the changes that were committed on one branch and applying them to a new branch.
   b. Run `git rebase-i` the -i option to rewrite, replace, delete, and merge individual commits in the history.
18. Stage and commit your changes
19. Squash merge
   a. A merge option that allows you to condense the Git history of topic branches when you complete a pull request.
      i. Instead of each commit on the topic branch being added to the history of the default branch
   b. All the file changes are added to a single new commit on the default branch.
   c. 

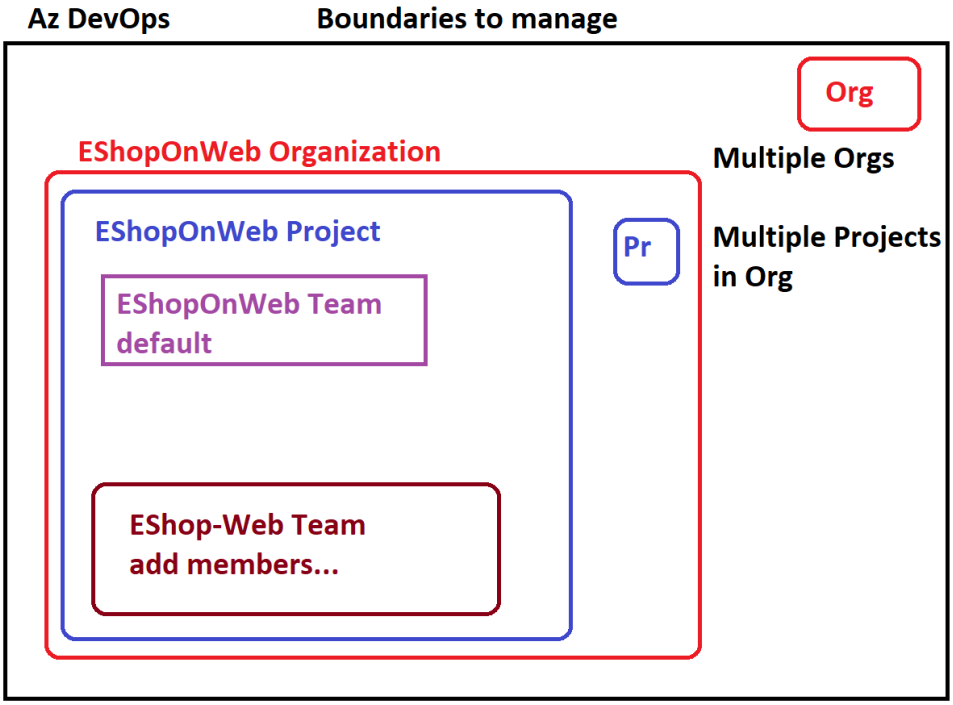
20. Stashing
   a. Takes the dirty state of your working directory — that is, your modified tracked files and staged changes — and saves it on a stack of unfinished changes that you can reapply at any time (even on a different branch).
21. `git clean`
   a. Remove cruft that has been generated by merges or external tools or to remove build artifacts in order to run a clean build.
   b. Be pretty careful with this command, since it's designed to remove files from your working directory that are not tracked.
22. `git stash –all`
   a. Remove everything but save it in a stash.
23. Commit messages
   a. Tell you why a change was made
24. https://docs.gitlab.com/
25. Tip migration
   a. Importing only the latest version of the source code
26. space


Push
Push
Remote repository
Pull
Pull
Local repository
I edit this!
html
css
Local repository
I edit this!
html
css

Login Links

1. Azure Portal:
2. https://azure.microsoft.com/en-us/account/
3. Azure DevOps:
4. https://azure.microsoft.com/en-us/services/devops/
5. Microsoft Teams:
    a. https://teams.microsoft.com/
    b. Microsoft Teams is a dedicated hub for teamwork, that brings your teams, content, conversations, chats, meetings, files, apps and tools from across Office 365 and Azure DevOps together into one place.
6. space

Az Devops Boundaries



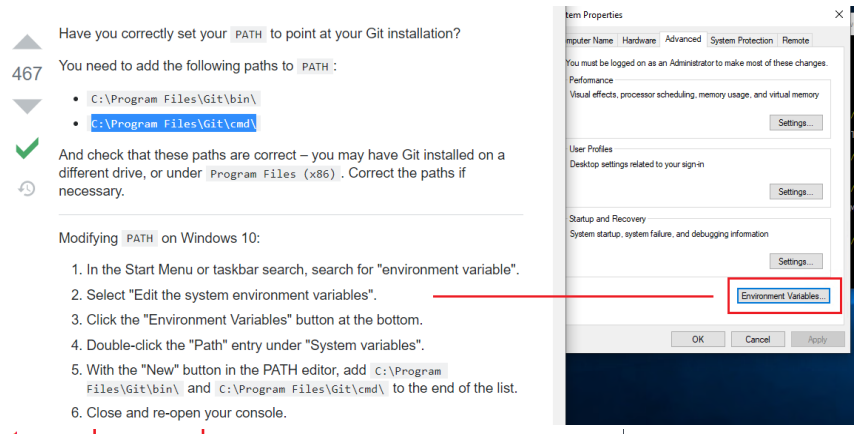
Setup Dev Sandbox

1. Docker (Windows, Linux or Mac)
    a. https://docs.docker.com/docker-for-windows/
2. GitHub Setup
    a. https://help.github.com/en/github/getting-started-with-github/quickstart

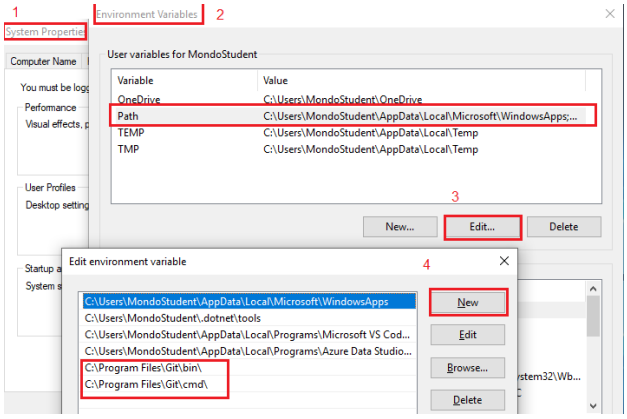
```
C:\Users\MondoStudent\Source\Repos\myWebApp>git init

'git' is not recognized as an internal or external command,
operable program or batch file.
```

    b. Install Git SCM to Windows - https://gitforwindows.org/
    c. Add path if CMD line errors or use

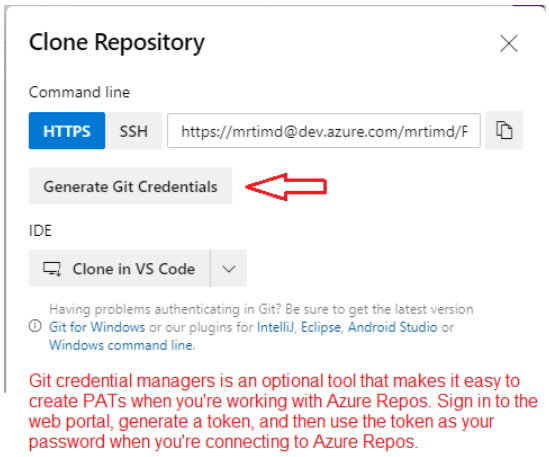
d.



e.

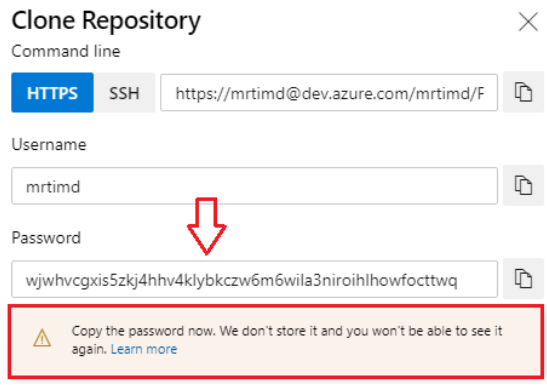


f. https://stackoverflow.com/questions/4492979/git-is-not-recognized-as-an-internal-or-external-command

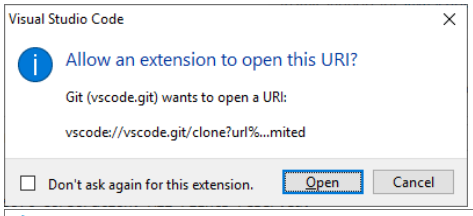


g.

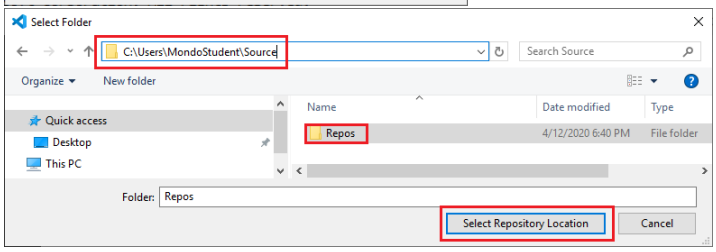
h.

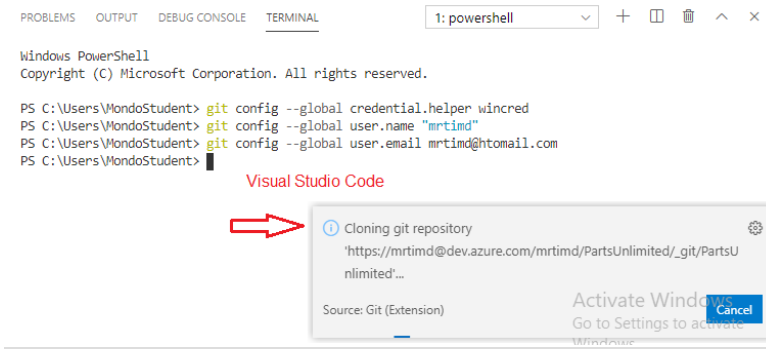


i.

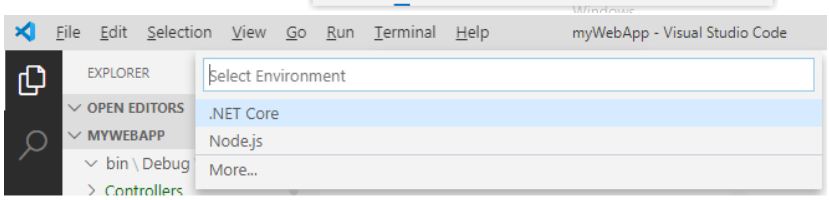


j.

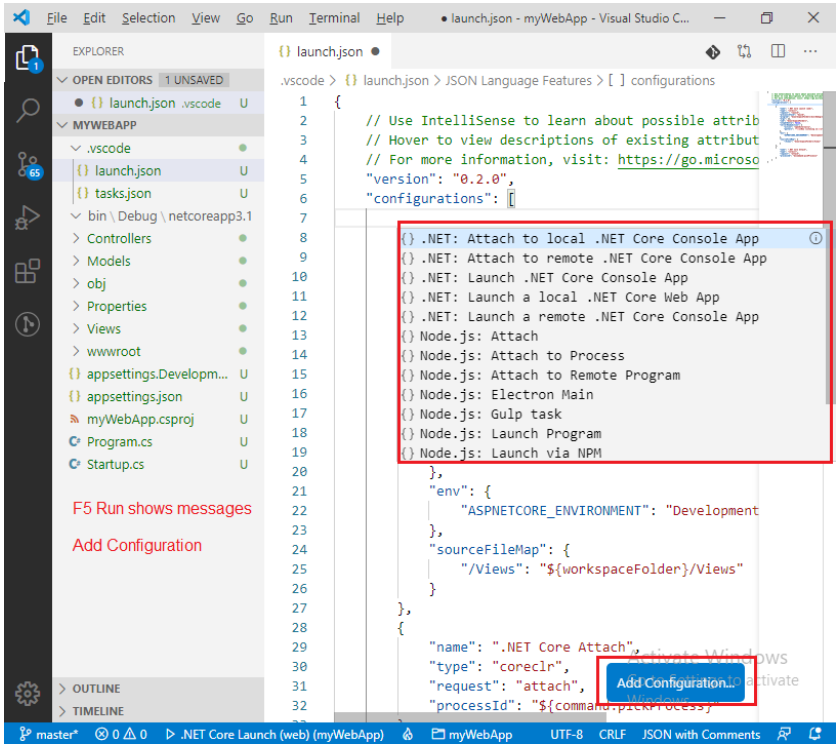




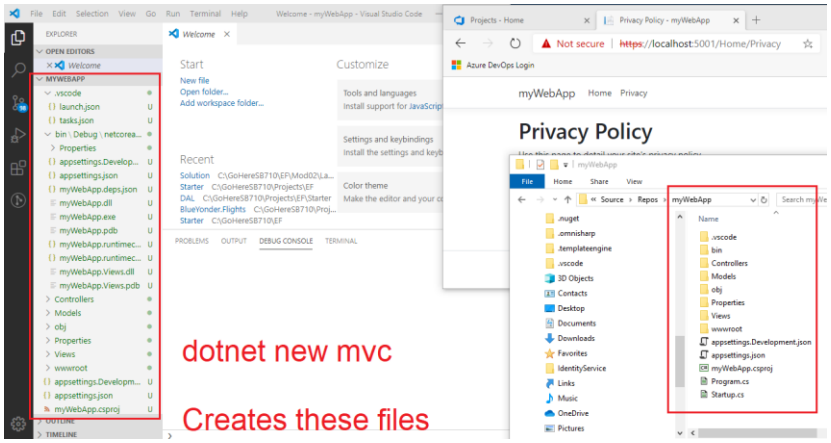
k.

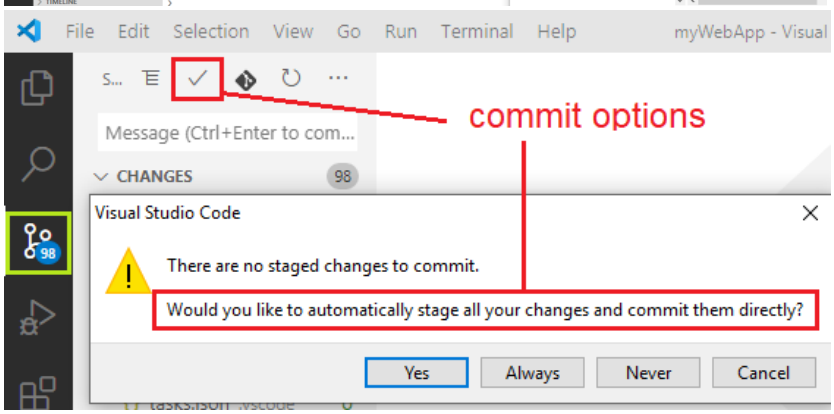


l. F5 Start shows these options

m. 

n. Cmd line *dotnet new mvc* creates this application structure. View it with Windows File Explorer, Visual Studio Code.

o. 

p. 

q.

```
Select Command Prompt
* feature-devops-home-page
  master

C:\Users\MondoStudent\Source\Repos\myWebApp>git status
On branch feature-devops-home-page
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .vscode/
        Controllers/
        Models/
        Program.cs
        Properties/
        Startup.cs
        Views/Home/Index.cshtml
        Views/Shared/
        Views/_ViewImports.cshtml
        Views/_ViewStart.cshtml
        appsettings.Development.json
        appsettings.json
        bin/
        myWebApp.csproj
        obj/
        wwwroot/

nothing added to commit but untracked files present (use "git add" to track)

C:\Users\MondoStudent\Source\Repos\myWebApp>
```

Git status

r.

```
C:\Users\MondoStudent\Source\Repos\myWebApp>git status
On branch feature-devops-home-page
nothing to commit, working tree clean
```

```
C:\Users\MondoStudent\Source\Repos\myWebApp>git checkout master
Unlink of file 'bin/Debug/netcoreapp3.1/myWebApp.Views.dll' failed. Should I try again? (y/n)
```

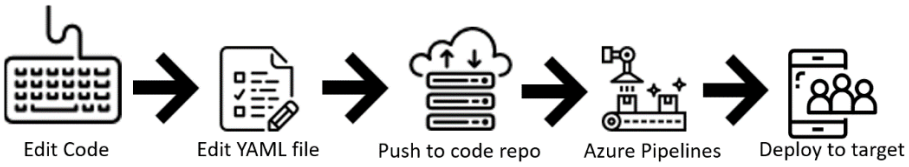
s. is it open in an editor? - VS Code

3. Microsoft Teams with Azure DevOps Services (Collaborate, Communicate and Celebrate):
   a. https://www.azuredevopslabs.com/labs/vstsextend/teams/
   b. Azure DevOps Services integration with Microsoft Teams provides a comprehensive chat and collaborative experience across the development cycle.
   c. Teams can easily stay informed of important activities in your Azure DevOps team projects with notifications and alerts on work items, pull requests, code commits, build and release.
4. Integrating Microsoft Teams with Azure DevOps Services
   a. https://www.azuredevopslabs.com/labs/vstsextend/teams/#integrating-microsoft-teams-with-azure-devops-services
5. Azure DevOps Kanban board & Dashboards in Teams:
   a. https://www.azuredevopslabs.com/labs/vstsextend/teams/#azure-devops-kanban-board--dashboards-in-teams
6. Logs
   a. Log Analytics workspace
   b. **Process of creating a diagnostic setting to send resource logs to a Log Analytics workspace where you can analyze them with log queries.
   c. **Collect resource logs from an Azure resource - Azure Monitor | Microsoft Learn

   d. Data is retrieved from a Log Analytics workspace by using a log query written in Kusto Query Language (KQL).
   e. Tutorial - Create a log query alert for an Azure resource - Azure Monitor | Microsoft Learn
   f. Install the Log Analytics Agent on the virtual machines.
   g. Container Monitoring solution in Azure Monitor - Azure Monitor | Microsoft Learn
   h. View and manage your Docker and Windows container hosts in a single location.
   i. Space
7. GitHub
   a. Run Git commands in a script - Azure Pipelines | Microsoft Learn
8. Microsoft-hosted agent
   a. Maintenance and upgrades are taken care of for you. Each time you run a pipeline, you get a fresh virtual machine for each job in the pipeline.
   b. The virtual machine is discarded after one job (which means any change that a job makes to the virtual machine file system, such as checking out code, will be unavailable to the next job). Microsoft-hosted agents can run jobs directly on the VM or in a container.
9. space

Azure Pipelines

You define your pipeline in a YAML file called azure-pipelines.yml with the rest of your app.

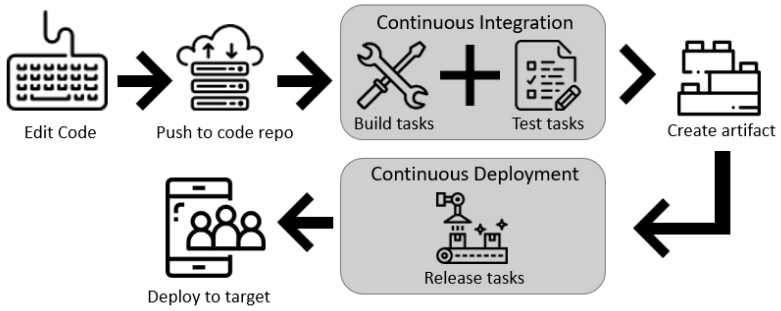


The pipeline is versioned with your code. It follows the same branching structure. You get validation of your changes through code reviews in pull requests and branch build policies.

Every branch you use can modify the pipeline by modifying the azure-pipelines.yml file.
Follow these basic steps:

1. Configure Azure Pipelines to use your Git repo.
2. Edit your azure-pipelines.yml file to define your build.
3. Push your code to your version control repository. This action kicks off the default trigger to build and deploy and then monitor the results.

Create and configure pipelines in the Azure DevOps web portal with the Classic user interface editor. You define a *build pipeline* to build and test your code, and then to publish artifacts. You also define a *release pipeline* to consume and deploy those artifacts to deployment targets.



Follow these basic steps:

1. Configure Azure Pipelines to use your Git repo.
2. Use the Azure Pipelines classic editor to create and configure your build and release pipelines.
3. Push your code to your version control repository. This action triggers your pipeline and runs tasks such as building or testing code.

The build creates an artifact that's used by the rest of your pipeline to run tasks such as deploying to staging or production.

A release is a package or container containing a versioned set of artifacts specified in a release pipeline in your CI/CD process.

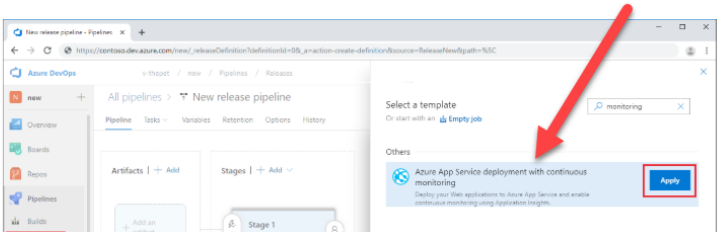
10. Tutorial - Create your first pipeline
    a. Using Azure Pipelines to build a GitHub repository
    b. Create your first pipeline - Azure Pipelines | Microsoft Learn
    c. Continuous Monitoring
    d. **Settings - Enable Continuous Monitoring for the release pipelines.

Azure Pipelines integrates with Azure Application Insights to allow continuous monitoring of your DevOps release pipeline throughout the software development lifecycle.

With continuous monitoring, release pipelines can incorporate monitoring data from Application Insights and other Azure resources. When the release pipeline detects an Application Insights alert, the pipeline can gate or roll back the deployment until the alert is resolved. If all checks pass, deployments can proceed automatically from test all the way to production, without the need for manual intervention.

Configure continuous monitoring

1. In Azure DevOps, select an organization and project.
2. On the left menu of the project page, select **Pipelines** > **Releases**.
3. Drop down the arrow next to **New** and select **New release pipeline**. Or, if you don't have a pipeline yet, select **New pipeline** on the page that appears.
4. On the **Select a template** pane, search for and select **Azure App Service deployment with continuous monitoring**, and then select **Apply**.

11. 

Deployment groups – pipelines - target the application release on a set of virtual machines

12. Use deployment groups in a release pipeline - Azure Pipelines | Microsoft Learn
13. Feature Flags -good discussion
    a. Introduction to feature toggles - Training | Microsoft Learn
    b. Feature toggles are a great alternative to branching as well.
    c. Branching is what we do in our version control system.
    d. To keep features isolated, we maintain a separate branch.
    e. is an IF statement.
    f. Progressive experimentation with feature flags - Azure DevOps | Microsoft Learn
14. Space

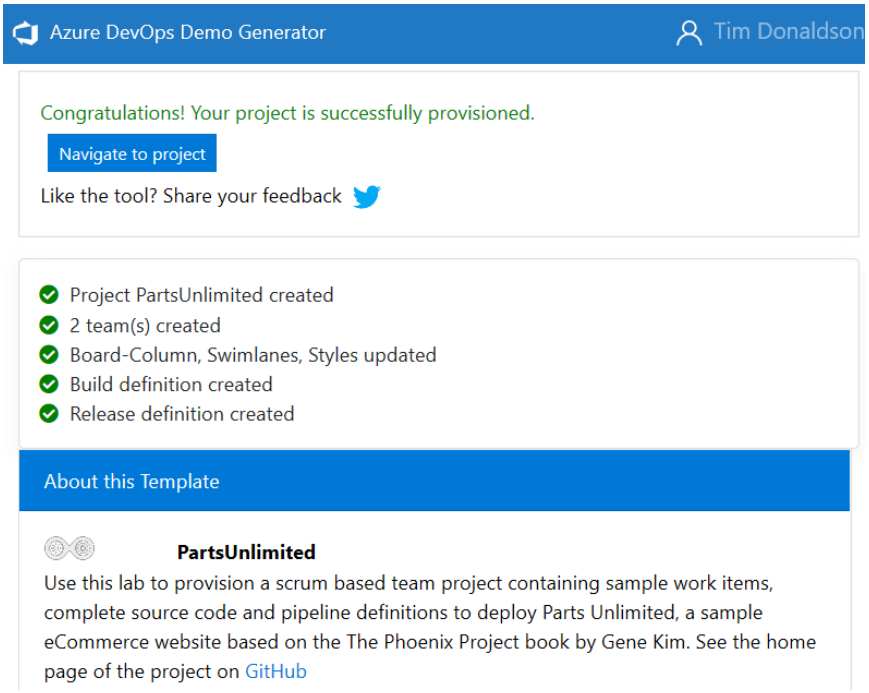
Exam AZ-400: Microsoft Azure DevOps Solutions Exam

https://docs.microsoft.com/en-us/learn/certifications/exams/az-400

Lab Prerequisites

Certain Azure DevOps labs require a preconfigured Parts Unlimited team project.

1. https://www.azuredevopslabs.com/labs/azuredevops/prereq/
2. Navigate to https://azuredevopsdemogenerator.azurewebsites.net
3. This utility site will automate the process of creating a new Azure DevOps project within your account that is prepopulated with content (work items, repos, etc.) required for the lab.
4. Setup error: Connect to Organizational Account
    a. Do you see users anywhere when assigning things?



1. Get started creating and populating demo Azure DevOps Services projects
    a. https://docs.microsoft.com/en-us/azure/devops/demo-gen/use-demo-generator-v2?view=azure-devops&viewFallbackFrom=vsts
2. About the Azure DevOps Services Demo Generator
    a. https://docs.microsoft.com/en-us/azure/devops/demo-gen/?view=azure-devops
3. Azure DevOps Demo Generator
    a. https://azuredevopsdemogenerator.azurewebsites.net/
4. Configuring the Parts Unlimited solution in Visual Studio
    a. https://www.azuredevopslabs.com/labs/azuredevops/prereq/#task-2-configuring-the-parts-unlimited-solution-in-visual-studio
5. space

Lab instructions example:

1. Version Controlling with Git in Visual Studio Code and Azure DevOps | Azure DevOps Hands-on-Labs (azuredevopslabs.com)

Work with Organizations

1. Create and configure organization FAQs:
    a. https://docs.microsoft.com/en-us/azure/devops/organizations/accounts/faq-configure-customize-organization?view=azure-devops
    b. Connect your Azure DevOps organization to Azure Active Directory (Azure AD) so you can sign in with the same username and password that you use with Microsoft services.
2. Plan your organizational structure:
    a. https://docs.microsoft.com/en-us/azure/devops/user-guide/plan-your-azure-devops-org-structure?view=azure-devops
3. Configure a hierarchy of teams:
    a. https://docs.microsoft.com/en-us/azure/devops/boards/plans/configure-hierarchical-teams?view=azure-devops
4. space

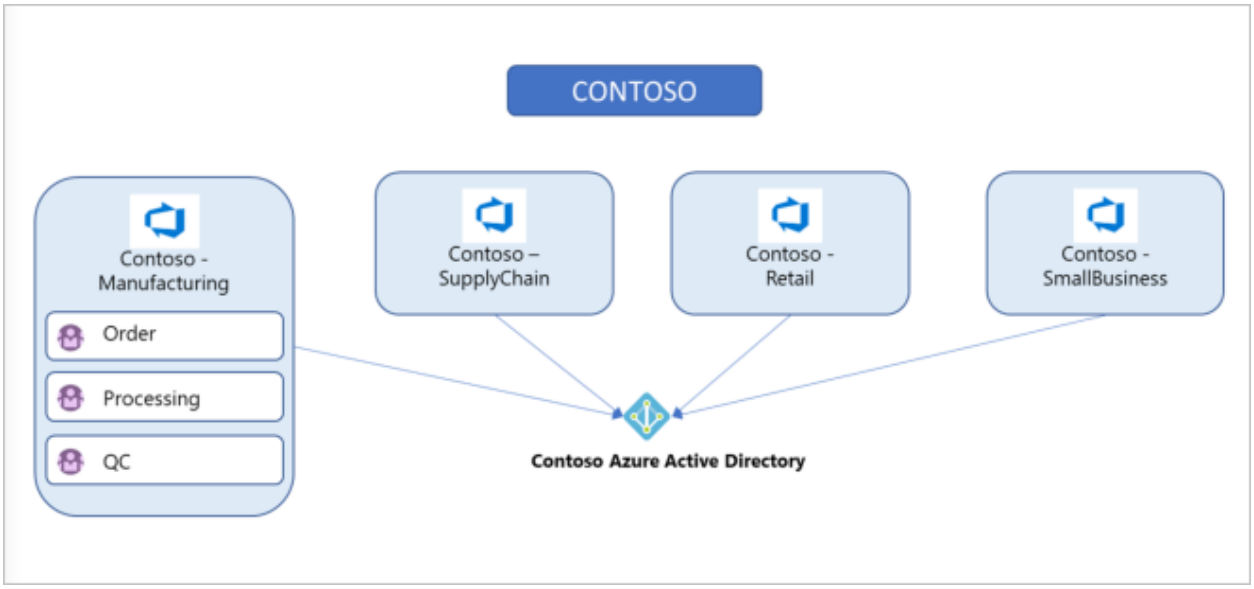
Work with Projects

A project in Azure DevOps 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

In the following image, the Contoso company has four projects within their Contoso-Manufacturing organization.



How many projects do you need?

You need at least one project to start using an Azure DevOps service, such as Azure Boards, Azure Repos, or Azure Pipelines. When you create your organization, a default project is created for you. In your default project, there's a code repo to start working in, backlog to track work, and at least one pipeline to begin automating build and release.

Within an organization, you can do either of the following approaches:

- Create a single project that contains many repos and teams
- Create many projects, each with its own set of teams, repos, builds, work items, and other elements

Even if you have many teams working on hundreds of different applications and software projects, you can manage them within a single project in Azure DevOps.

However, if you want 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 can be connected to many Azure DevOps organizations.

Single Project

A single project puts all of the work at the same "portfolio" level for the entire organization. Your work has the same set of repos and iteration paths.

A single project allows teams to share source repos, build definitions, release definitions, reports, and package feeds.

You might have a large product or service that's managed by many teams. Those teams have tight inter-dependencies on each other across the product life cycle.

You create a project and divide the work using teams and area paths.

This setup gives your teams visibility into each other's work, so the organization stays aligned. Your teams use the same taxonomy for work item tracking, making it easier to communicate and stay consistent.

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

Project Complexity

Note the common topics; builds, teams, policies, pipelines, repos, Azure, Deployment groups

Project name	Project details
Project 1	Project1 will provide support for incremental builds and third-party SDK components
Project 2	Project2 will use an automatic build policy. A small team of developers named Team2 will work independently on changes to the project. The Team2 members will not have permissions to Project2.
Project 3	Project3 will be integrated with SonarQube
Project 4	Project4 will provide support for a build pipeline that creates a Docker image and pushes the image to the Azure Container Registry. Project4 will use an existing Dockerfile.
Project 5	Project5 will contain a Git repository in Azure Reports and a continuous integration trigger that will initiate a build in response to any change except for changes within /folder1 of the repository.
Project 6	Project6 will provide support for build and deployment pipelines. Deployment will be allowed only if the number of current work items representing active software bugs is 0.
Project 7	Project7 will contain a target deployment group named Group7 that maps to Pool7. Project7 will use Azure Automation State Configuration to maintain the desired state of the computers in Group7.

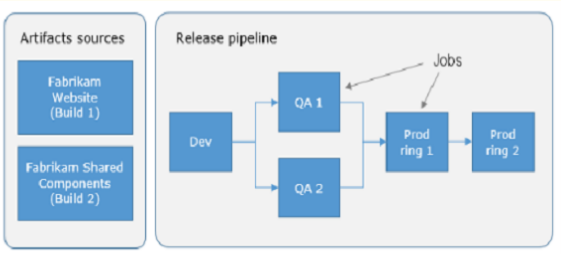
Tech Requirements Example

Technical requirements Scenario Examples

The company's investment planning applications suite must meet the following requirements:

- New incoming connections through the firewall must be minimized.
- Members of a group named Developers must be able to install packages.
- The principle of least privilege must be used for all permission assignments.
- A branching strategy that supports developing new functionality in isolation must be used.
- Members of a group named Team Leaders must be able to create new packages and edit the permissions of package feeds.
- Visual Studio App Center must be used to centralize the reporting of mobile application crashes and device types in use.
- By default, all releases must remain available for 30 days, except for production releases, which must be kept for 60 days.
- Code quality and release quality are critical. During release, deployments must not proceed between stages if any active bugs are logged against the release.
- The mobile applications must be able to call the share pricing service of the existing retirement fund management system. Until the system is upgraded, the service will only support basic authentication over HTTPS.
- The required operating system configuration for the test servers changes weekly. Azure Automation State Configuration must be used to ensure that the operating system on each test server is configured the same way when the servers are created and checked periodically.

An example of a release pipeline that can be modeled through a release pipeline in shown below:



In this example, a release of a website is created by collecting specific versions of two builds (artifacts), each from a different build pipeline. The release is first deployed to a Dev stage and then forked to two QA stages in parallel. If the deployment succeeds in both the QA stages, the release is deployed to Prod ring 1 and then to Prod ring 2. Each production ring represents multiple instances of the same website deployed at various locations around the globe.

References: https://docs.microsoft.com/en-us/azure/devops/pipelines/release

Steps/Workflows

The forking workflow

1. Create a fork
2. Clone it locally
3. Make your changes locally and push them to a branch
4. Create and complete a PR to upstream
5. Sync your fork to the latest from upstream

space

Azure DevOps Vocabulary & Whatnot Continued

1. Application Insights
    a. Funnels: Application Insights funnels - Azure Monitor | Microsoft Learn
        i. Customers are progressing through the entire process or ending the process at some point.
        ii. The progression through a series of steps in a web application is known as a funnel.
    b. User Flows: Application Insights User Flows analyzes navigation flows - Azure Monitor | Microsoft Learn
        i. Churn the most: How users move between the pages and features of your site.
    c. Impact: Application Insights usage impact - Azure Monitor | Microsoft Learn
        i. Load times and other properties influence conversion rates
    d. space
2. Diffable
    a. People who do the same jobs as other people but just do them differently. Quicker, more efficiently.
3. Sharing boundary
    a. A Containers "edges" or scope that is accessible.
4. Azure Ecosystem
    a. Global Azure offers more than 100 services available in 54 regions around the globe.
        i. Products available by region:
        ii. https://azure.microsoft.com/en-us/global-infrastructure/services/
        iii. Use public cloud services for on-demand, self-service computing resources to migrate and modernize existing apps and to build new cloud-native apps.
    b. Azure Stack Edge - Accelerate machine learning workloads and run containerized apps or virtualized workloads on-premises, on a cloud-managed appliance.
    c. Azure Stack HCI - Run virtualized apps on-premises, replace and consolidate aging server infrastructure, and connect to Azure for cloud services.
    d. Azure Stack Hub - Run cloud apps on-premises, when disconnected, or to meet regulatory requirements, using consistent Azure services.
    e. https://docs.microsoft.com/en-us/azure-stack/operator/compare-azure-azure-stack?view=azs-2002
5. Organization: (Grouping into containers based on)
    a. An organization in Azure DevOps is a mechanism for organizing and connecting groups of related projects.
    b. Examples are business divisions, regional divisions, or other enterprise structure. You can choose one organization for your entire company, or separate organizations for specific business units, or an organization just for you.
    c. Can't find your directory, contact your Azure AD administrator and request that they add you as a member to the Azure AD.
    d. https://docs.microsoft.com/en-us/azure/devops/organizations/accounts/connect-organization-to-azure-ad?view=azure-devops#connect-your-organization-to-azure-ad
6. Template
    a. Single

b. Main
c. To deploy complex solutions, you can break your template into many related templates, and then deploy them together through a main template.
d. The related templates can be separate files or template syntax that is embedded within the main template.
e. Using linked and nested templates when deploying Azure resources
    i. https://docs.microsoft.com/en-us/azure/azure-resource-manager/templates/linked-templates

7. Visual Studio Code
8. Visual Studio

Code – Syntax – ARM – JSON – PowerShell – Az CLI - Bash

9. Deploy resources with PowerShell and template - Azure Resource Manager | Microsoft Learn
    a. Note deploy to a Resource Group as parent container.
10. YAML
    a. (rhymes with "camel") A human-friendly, cross language, Unicode based data serialization language designed around the common native data structures of agile programming languages.
    b. Originally stood for Yet Another Markup Language at that time; was not really a markup language (marking up various elements of a text document) but a serialization language (textual representation of typed/cyclical data graphs).
    c. Was backronymed it to mean YAML Ain't Markup Language.
    d. Build GitHub repositories - Azure Pipelines | Microsoft Learn

YAML Copy

```yaml
# specific branch build
trigger:
  branches:
    include:
    - master
    - releases/*
    exclude:
    - releases/old*
```

In the above example, the pipeline will be triggered if a change is pushed to master or to any releases branch. However, it won't be triggered if a change is made to a releases branch that starts with `old`.

    e.
    f. Azure Pipelines can automatically build and validate every pull request and commit to your GitHub repository.
11. Centralized Source control
    There is a single central copy of your project and programmers commit their changes to this central copy
    a. CVS (Concurrent Versions System)
    b. SVN (SubVersioN)
        i. Atomic commits allow each commit to be applied in full or not at all. Open source version control system.

ii. https://subversion.apache.org/

c. Team Foundation Version Control (TFVC)
   i. A centralized version control system.
   ii. Typically, team members have only one version of each file on their dev machines.
   iii. Historical data is maintained only on the server.
   iv. Branches are path-based and created on the server.

d. Perforce
   i. Version control, application lifecycle management, agile planning, and static analysis.
   ii. Perforce Software is also the parent company of Perfecto and Rogue Wave.
   iii. https://www.perforce.com/

12. Distributed Source control

Every developer clones a copy of a repository and has the full history of the project

a. Mercurial
   i. Every clone contains the whole project history, so most actions are local, fast and convenient.
   ii. The term repository refers to the directory named .hg (dot hg) in the repository root directory.
      1. The repository root directory is the parent directory of the .hg directory.
      2. Mercurial stores its internal data structures – the metadata – inside that .hg directory.
   iii. Working directory
      1. The top-level directory in a repository sometimes called the "working copy."
   iv. https://www.mercurial-scm.org/

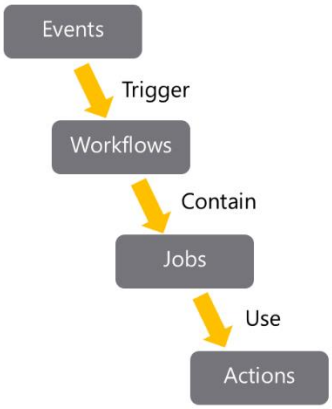
b. Git SCM
   i. Git is a distributed version-control system for tracking changes in source code during software development.
   ii. SCM in the context of git means Source Code Management System rather than a software configuration management system.
   iii. Pay as you go pricing
      1. https://github.com/features/actions#pricing-details

c. Bazaar

13. Workflows
   a. GitHub tracks events that occur. Events can trigger the start of workflows.
   b. Workflows can also start on cron-based schedules and can be triggered by events outside of GitHub.
   c. They can be manually triggered.
   d. Workflows are the unit of automation. They contain Jobs.
   e. The job defines the location in which the actions will run, like which runner to use.

14. Actions

a. Are the mechanism used to provide workflow automation within the GitHub environment used to build continuous integration (CI) and continuous deployment (CD) solutions.
b. Defined in YAML and stay within GitHub repositories.
c. Executed on "runners," either hosted by GitHub or self-hosted.
    i. When you execute jobs, the steps execute on a Runner.
    ii. JavaScript actions will be faster (no container needs to be used)
    iii. Don't use self-hosted runners in public repos a significant security risk.
d. What are Actions? - Training | Microsoft Learn
    i. Actions can produce console output directly from within the GitHub UI.
    ii. **GitHub - skills/hello-github-actions: Create a GitHub Action and use it in a workflow.
    iii. create workflow files, trigger workflows, and find workflow logs.
        - Organize and identify workflow files.
        - Add executable scripts.
        - Create workflow and action blocks.
        - Trigger workflows.
        - Discover workflow logs.
    iv.



e.
f. Events can trigger the start of workflows.
g.

15. GitHub App authentication
    a. https://developer.github.com/apps/building-github-apps/authenticating-with-github-apps/
16. Personal access tokens (PATs)
    a. Alternate passwords that you can use to authenticate into Azure DevOps.
    b. To create a Personal Access Token, browse https://github.com/settings/tokens, then click "Generate new token".

Approval

17. Approval Process
    a. Based on a condition.
    b. Policy approvals must be completed within 5 hours.
    c. After implementation deployments start failing after 2 hours.

    d. Why?
18. Code quality restriction
    a. Define approvals at the start of a stage (pre-deployment approvers)
    b. At the end of a stage (post-deployment approvers)
    c. Or both.
    d. https://docs.microsoft.com/en-us/azure/devops/pipelines/release/approvals/approvals?view=azure-devops
19. NDepend
    a. A Visual Studio extension that assesses the amount of technical debt that a developer has added during a recent development period, typically in the last hour
20. Resharper Code Quality Analysis
    a. A command line tool and can be set to automatically fail builds when code quality issues are found
21. Visual Studio App Center
    a. Automate and manage the lifecycle of your iOS, Android, Windows, and macOS apps.
    b. Ship apps more frequently, at higher-quality, and with greater confidence.
22. Tagging a release
23. OWASP
    a. Open Web Application Security Project (OWASP) is a nonprofit foundation that works to improve the security of software.
    b. https://owasp.org/
24. Azure CLI
25. Maven
    a. A tool that can now be used for building and managing any Java-based project
    b. https://maven.apache.org/what-is-maven.html
26. ASF member
    a. Apache Software Foundation
    b. https://www.apache.org/foundation/how-it-works.html
27. Xcode
    a. Is the Apple Integrated Development Environment (IDE) that you use to create applications for Apple products, such as iPads, iPhones, and Macs.
28. Azure Security Center
    a. Security Posture Management
    b. What's the difference between Azure Security Center, Azure Defender and Azure Sentinel? - Microsoft Community Hub
    c. Azure Defender: Advanced Workload Protection
        i. Defender for Cloud's menu, open the Workload protections dashboard.
    d. *Network Map - an interactive graphical view of the network topology of your Azure workloads and the traffic routes. By default, the topology map displays resources that have network recommendations with high or medium severity.
    e. Protecting your network resources in Microsoft Defender for Cloud | Microsoft Learn
29. Docker Container
    a. Detect any image vulnerabilities for the images stored in Azure Container Registry.
    b.

30. Azure Container Registry
    a. Allows you to store images for all types of container deployments including DC/OS, Docker Swarm, Kubernetes, and Azure services such as App Service, Batch, Service Fabric, and others.
    b. DevOps team can manage the configuration of apps isolated from the configuration of the hosting environment.
    c. Container registries are versioned repositories where container artifacts are stored.
31. Npm
32. Nuget
33. GNU
    a. An operating system and an extensive collection of computer software. GNU is composed wholly of free software, most of which is licensed under the GNU Project's own General Public License.
    b. Bash is the GNU Project's shell
        i. https://www.gnu.org/software/bash/
    c. Git for Windows provides a BASH emulation used to run Git from the command line
    d. https://gitforwindows.org/
34. Artifact repositories
35. Test management
36. Work management
37. Project metrics
38. KPIs
39. DevOps measurements (e.g., cycle time, lead time, WIP limit)
40. Agile techniques and practices
    a. Continuous integration and continuous deployment (CI/CD)
41. Feature flag lifecycle
42. Technical debt
    a. A concept in software development that reflects the implied cost of additional rework caused by choosing an easy (limited) solution now instead of using a better approach that would take longer. (also known as design debt or code debt)
    b. Never enough time to do it right but always enough time to do it over.
43. Static code analysis
44. Infrastructure security validation
45. Gates (Release Gates)
    a. Gates allow automatic collection of health signals from external services, and then promote the release when all the signals are successful at the same time or stop the deployment on timeout.
    b. Typically, gates are used in connection with incident management, problem management, change management, monitoring, and external approval systems.

Scenarios for gates

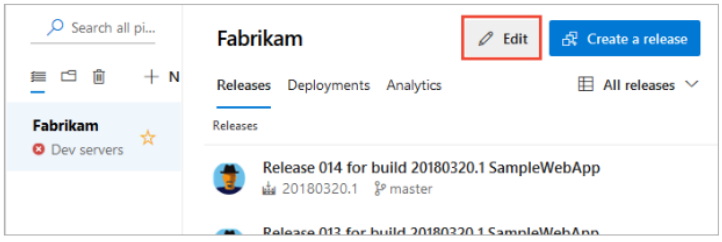
Some scenarios and use cases for gates are:

- **Incident and issues management**. Ensure the required status for work items, incidents, and issues. For example, ensure deployment occurs only if no priority zero bugs exist, and validation that there are no active incidents takes place after deployment.
- **Seek approvals outside Azure Pipelines**. Notify non-Azure Pipelines users such as legal approval departments, auditors, or IT managers about a deployment by integrating with approval collaboration systems such as Microsoft Teams or Slack, and waiting for the approval to complete.
- **Quality validation**. Query metrics from tests on the build artifacts such as pass rate or code coverage and deploy only if they are within required thresholds.
- **Security scan on artifacts**. Ensure security scans such as anti-virus checking, code signing, and policy checking for build artifacts have completed. A gate might initiate the scan and wait for it to complete, or just check for completion.
- **User experience relative to baseline**. Using product telemetry, ensure the user experience hasn't regressed from the baseline state. The experience level before the deployment could be considered a baseline.
- **Change management**. Wait for change management procedures in a system such as ServiceNow complete before the deployment occurs.
- **Infrastructure health**. Execute monitoring and validate the infrastructure against compliance rules after deployment, or wait for healthy resource utilization and a positive security report.

Most of the health parameters vary over time, regularly changing their status from healthy to unhealthy and back to healthy. To account for such variations, all the gates are periodically re-evaluated until all of them are successful at the same time. The release execution and deployment does not proceed if all gates do not succeed in the same interval and before the configured timeout.

c.
d. Pre-deployment
e. Post-deployment
f. Release deployment control using gates
g. [Control releases with deployment gates - Azure Pipelines | Microsoft Learn]
h. Query for Work Items
i. **Settings - Construct queries to search for values of interest which can then be used.

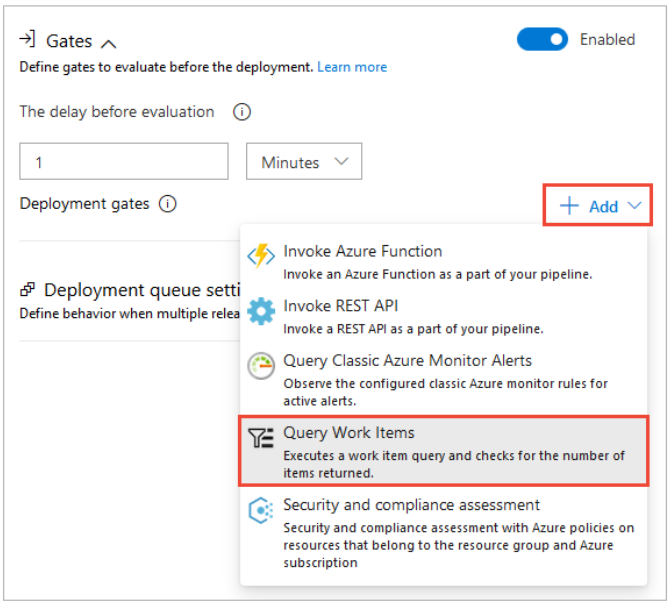
1. In the **Releases** tab of **Azure Pipelines**, select your release pipeline and choose **Edit** to open the pipeline editor.



j.

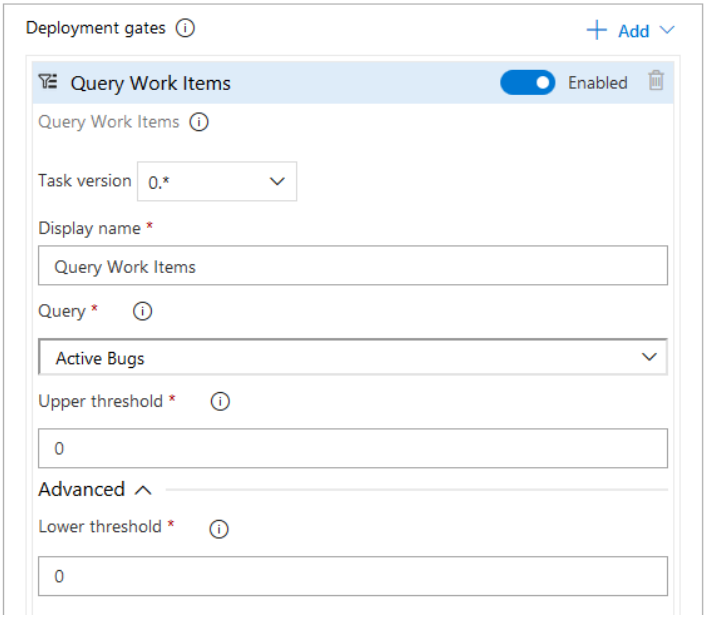
4. Choose + **Add** and select the **Query Work Items** gate.

k.



5. Configure the gate by selecting an existing work item query. You can use one of the built-in Azure Pipelines and TFS queries, or create your own query. Depending on how many work items you expect it to return, set the maximum and minimum thresholds (run the query in the **Work** hub if you're not sure what to expect).

l.

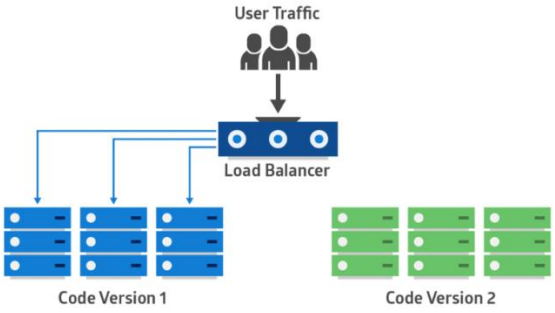


46. Azure Artifacts
    a. Upstream sources
    b. Enable you to use a single feed to store both the packages you produce and the packages you consume from "remote feeds": both public feeds (e.g. npmjs.com, nuget.org, Maven Central, and PyPI) and authenticated feeds (i.e. other Azure DevOps Services feeds in your organization or in organizations in your Azure Active Directory (AAD) tenant).

    c. Hosting a private PowerShell repository
        i. Share PowerShell scripts across teams to promote collaboration and maximize effectiveness. By storing PowerShell modules
    d. https://docs.microsoft.com/en-us/azure/devops/artifacts/concepts/upstream-sources
47. License management strategy
    a. VSTS users
    b. Concurrent pipelines
    c. Test environments
    d. Open source software licensing
    e. Third-party DevOps tools and services
    f. Package management licensing
48. WhiteSource
    a. https://www.azuredevopslabs.com/labs/vstsextend/WhiteSource/
    b. Continuous open source software security and compliance management.
49. BlackDuck
50. Smart Detection
    a. Automatically warns you of potential performance problems and failure anomalies in your web application.
    b. NOT security problems.
    c. https://docs.microsoft.com/en-us/azure/azure-monitor/app/proactive-diagnostics
51. Azure App Service Web Apps
52. End-to-end traceability
53. Monitoring and feedback to development teams
    a. Monitor availability with URL ping tests - Azure Monitor | Microsoft Learn
54. Authentication and access
55. MFA- multi-factor authentication is used to prompt users for additional verification.
56. Branching models
    a. Fork strategies
    b. Branch policies
57. Branches
    a. Create new branches to isolate changes for a feature or a bug fix from your master branch and other work.
58. Branches and Merges
    a. Works through pull requests, so the commit history of your development doesn't necessarily form a straight, chronological line.
    b. When you use history to compare versions, think in terms of file changes between two commits instead of file changes between two points in time.
    c. A recent change to a file in the master branch may have come from a commit created two weeks ago
        i. In a feature branch but was only merged yesterday.
59. Code flow strategy
60. Azure pipeline configuration (e.g., agent queues, service endpoints, pools, webhooks)
61. Pull request
62. Device sets

63. Distribution groups
    a. A Distribution Group represents a set of users that can be managed jointly and can have common access to releases.
    b. Example of Distribution Groups can be teams of users, like the QA Team or External Beta Testers, or can represent stages or rings of releases, such as Staging.
    c. https://docs.microsoft.com/en-us/appcenter/distribution/groups
64. Azure Key Vault
    a. Secrets
    b. Tokens
    c. Certificates
65. Cobertura
    a. Java tool that calculates the percentage of code accessed by tests. It can be used to identify which parts of your Java program are lacking test coverage. It is based on jcoverage.
    b. Can publish results to Azure Devops.
66. Web App
67. Azure Kubernetes Service
68. Containers
    a. The container exists solely to deliver the software or services, and should be optimized for that purpose.
    b. https://www.sumologic.com/blog/snowflake-configurations-and-devops-automation/
69. Test suites and categories
70. ACR Tasks
    a. A suite of features within Azure Container Registry.
    b. Provides cloud-based container image building for platforms including Linux, Windows, and ARM, and can automate OS and framework patching for your Docker containers.
71. Package-lock.json
72. Security analysis tools
73. Application Security Project
74. Build tools and configuration
    a. Azure Pipelines
    b. Jenkins
        i. A lightweight build automation tool.
        ii. A plugin is required to include support for TFS in order to automate a TFS – managed project build and publish.
        iii. Uses "build" as a term for copying the files from source control.
        iv. Publish means copying the files to various other directories other than the "build" directory.
    c. Multi-agent builds
    d. Automated build workflow
    e. Snowflake server:
        i. Partial automation is a snowflake environment.
        ii. A snowflake configuration is ad-hoc and "unique" to the environment at large.
        iii. DevOps, drop unique configurations, focus on full-automation.

Deployments

75. Timeout
    a. is exceeded for an approval process, the deployment is rejected.
76. Frequency
77. Overview of Azure Blueprints - Azure Blueprints | Microsoft Learn
78. Deploy to Azure SQL Database - Azure Pipelines | Microsoft Learn
    a. DACPACs
79. Azure DNS
    a. Load Balancing
80. Azure Traffic Manager
    a. Traffic Manager - Cloud Based DNS Load Balancing | Microsoft Azure
    b. Blue-Green deployments using Azure Traffic Manager | Azure Blog | Microsoft Azure
    c. Azure Traffic Manager - traffic routing methods | Microsoft Learn
    d. Priority traffic-routing method
    e. Weighted traffic-routing method
    f. Performance traffic-routing method
    g. Geographic traffic-routing method
81. Deployment pattern
    a. "a named strategy for solving a recurring problem"
    b. Deployment pipelines tie together configuration management, continuous integration and test and deployment automation that improves software quality, stability, and reduce time and cost required to make incremental changes.
82. Deployment strategies:
    a. https://dev.to/mostlyjason/intro-to-deployment-strategies-blue-green-canary-and-more-3a3
    b. Big bang deployments update whole or large parts of an application in one fell swoop.
    c. Rolling deployments
        i. An application's new version gradually replaces the previous over a period of time.
    d. Blue-Breen deployments, Red-Black or A/B Deployment (Boolean Tim sez)
        i. One is the currently-running production environment receiving all user traffic (depicted as Blue). The other is a clone of it, but idle (Green). Both use the same database back-end and app configuration:
    e. 

    f. Before blue-green deployment

i. The new version of the application is deployed in the green environment and tested for functionality and performance. Once the testing results are successful, application traffic is routed from blue to green. Green then becomes the new production.

g. Canary deployments
   i. With canary deployment, you deploy a new application code in a small part of the production infrastructure. Once the application is signed off for release, only a few users are routed to it.
   ii. This minimizes any impact.
h. Progressive exposure deployments
i. Endpoints
j. https://continuousdelivery.com/implementing/patterns/
k. https://octopus.com/docs/deployment-patterns
l. Waterfall model: https://en.wikipedia.org/wiki/Waterfall_model

83. Erlang vs Hack: What are the differences?
    a. Erlang:
       i. A programming language used to build massively scalable soft real-time systems with requirements on high availability.
       ii. Some of Erlang's uses are in telecoms, banking, e-commerce, computer telephony and instant messaging.
       iii. Erlang's runtime system has built-in support for concurrency, distribution and fault tolerance.
       iv. OTP is set of Erlang libraries and design principles providing middle-ware to develop these systems;
    b. Hack:
       i. A programming language for HHVM that interoperates seamlessly with PHP.
       ii. Hack provides instantaneous type checking via a local server that watches the filesystem.
       iii. It typically runs in less than 200 milliseconds, making it easy to integrate into your development workflow without introducing a noticeable delay.
    c. "Real time, distributed applications" is the primary reason why developers consider Erlang over the competitors
    d. "Interoperates seamlessly with php" was stated as the key factor in picking Hack.
84. Azure Pipelines
    a. Implement continuous integration and continuous delivery (CI/CD) for the app and platform of your choice.
       i. https://docs.microsoft.com/en-us/azure/devops/pipelines/?view=azure-devops
       ii. **https://docs.microsoft.com/en-us/azure/devops/pipelines/get-started/what-is-azure-pipelines?view=azure-devops
    b. Each time you run a pipeline you get a fresh virtual machine. The virtual machine is discarded after one use.
    c. Hosted-agent which is provided by Microsoft.
       i. Software-as-a-service.
       ii. Don't have access to configure or these servers these servers.

   d. Self-hosted agent which is a private agent
      i. Could be on-premises because this could be your machine on your own premises or maybe your machine on the cloud at any could provider.
      ii. Control everything about this machine.
85. Agent pools
   a. Scoped to the Azure DevOps organization
   b. Can share an agent pool across projects
86. Multi-stage builds
   a. Use multiple FROM statements in your Dockerfile.
   b. Each FROM instruction can use a different base, and each of them begins a new stage of the build.
   c. Selectively copy artifacts from one stage to another
   d. Leaving behind everything you don't want in the final image.
   e. https://docs.docker.com/develop/develop-images/multistage-build/#use-multi-stage-builds#use-multi-stage-builds
87. SonarQube
   a. An automatic code review tool to detect bugs, vulnerabilities and code smells in your code.
   b. It can integrate with your existing workflow to enable continuous code inspection across your project branches and pull requests.
   c. https://docs.sonarqube.org/latest/
   d. White Source Bolt
   e. Open Web
88. Code Smell
   a. In computer programming, a code smell is any characteristic in the source code of a program that possibly indicates a deeper problem.
89. Health signals for release approvals
90. Release gates
91. Release pipeline
   a. Azure Kubernetes Service
   b. Service Fabric
   c. WebApp
   d. A fully-automated, script-driven DevOps pipeline works best when the elements that make up the pipeline are uniform.
92. Multi-phase release pipelines
93. Tasks and templates - task and variable groups
94. Artifact management
95. GPLv3
   a. GNU General Public License is a series of widely used free software licenses that guarantee end users the freedom to run, study, share, and modify the software.
   b. https://www.gnu.org/licenses/quick-guide-gplv3.en.html
96. Package security and license rating
   a. Black Duck
   b. White Source

97. Package feeds
98. Infrastructure as Code (IaC)
    a. Transient infrastructure
    b. Infrastructure state drift
    c. Configuration drift happens when production or primary hardware and software infrastructure configurations "drift" or become different in some way from a recovery or secondary configuration or visa versa.
    d. Detecting and Managing Drift with Terraform: https://www.hashicorp.com/blog/detecting-and-managing-drift-with-terraform/
99. Visual Designer
    a. Classic Pipelines vs. YAML
100. Resource templates
    a. ARM Templates
    b. Terraform
    c. Chef
    d. Puppet
    e. Ansible
        i. Orchestration is about bringing together disparate things into a coherent whole.
101. Azure Kubernetes Service
    a. Managing and Deploying Containerized Apps
    b. Deployment file for publishing to Azure Kubernetes Service
    c. Kubectl
        i. https://kubernetes.io/docs/reference/kubectl/cheatsheet/
    d. Helm
        i. Helm Charts help you define, install, and upgrade even the most complex Kubernetes application.
        ii. https://helm.sh/
    e. Develop a scaling plan
102. PowerShell Desired State Configuration (DSC)
103. VM Agent with custom script extensions
104. Measure end-user satisfaction
    a. Send a Smile
    b. app analytics
    c. Twitter
    d. Reddit
    e. Help Desk
105. Feature usage tracking
    a. Implement tools to track usage and flow - Training | Microsoft Learn
106. **HAProxy** is free, open source software that provides a high availability load balancer and proxy server for TCP and HTTP-based applications that spreads requests across multiple servers.[1] It is written in C[2] and has a reputation for being fast and efficient (in terms of processor and memory usage)
107. Azure Load Balancer
108. Service principal

a. Defines the access policy and permissions

Create a service principal and store credentials

Use the az ad sp create-for-rbac command to create the service principal, and az keyvault secret set to store the service principal's **password** in the vault:

Azure CLI | Copy | Try It

```
# Create service principal, store its password in AKV (the registry *passwor
az keyvault secret set \
  --vault-name $AKV_NAME \
  --name $ACR_NAME-pull-pwd \
  --value $(az ad sp create-for-rbac \
               --name $ACR_NAME-pull \
               --scopes $(az acr show --name $ACR_NAME --query id --output
               --role acrpull \
               --query password \
               --output tsv)
```

b.

109. Ticketing systems with development team's work management system
   a. IT Service Management connector
   b. ServiceNow Cloud Management
      i. https://www.servicenow.com/
   c. App Insights work items
110. Telemetry

ARM Bicep CI/CD

111. Azure Resource Manager (ARM) templates
   a. Two ways to deploy templates with Azure Pipelines?
   b. CI/CD with Azure Pipelines and templates - Azure Resource Manager | Microsoft Learn
   c. Tutorial - Create and deploy template - Azure Resource Manager | Microsoft Learn
   d. **Continuous integration with Azure Pipelines - Azure Resource Manager | Microsoft Learn
   e. -TemplateFile
   f. Deploy resources with PowerShell and template - Azure Resource Manager | Microsoft Learn

Deploy local template

The following example creates a resource group, and deploys a template from your local machine. The name of the resource group can only include alphanumeric characters, periods, underscores, hyphens, and parenthesis. It can be up to 90 characters. It can't end in a period.

Azure PowerShell | Copy

```
$resourceGroupName = Read-Host -Prompt "Enter the Resource Group name"
$location = Read-Host -Prompt "Enter the location (i.e. centralus)"

New-AzResourceGroup -Name $resourceGroupName -Location $location
New-AzResourceGroupDeployment -ResourceGroupName $resourceGroupName `
  -TemplateFile c:\MyTemplates\azuredeploy.json
```

   g.

112. cruft
   a. Trash, debris, or other unwanted matter that accumulates over time.

b. Unnecessary digital information that accumulates over time, such as unneeded files or obsolete lines of code in software: "By removing cruft, you can recover valuable disk space ... and reduce the chance of software conflicts"

113. Work item types (WITs)

114. Maven is a build automation tool used primarily for Java projects.
   a. Maven can also be used to build and manage projects written in C#, Ruby, Scala, and other languages.
   b. The Maven project is hosted by the Apache Software Foundation.
   c. https://maven.apache.org/index.html

115. PMD
   a. Is a source code analyzer. https://pmd.github.io/
   b. It finds common programming flaws like unused variables, empty catch blocks, unnecessary object creation, and so forth.
   c. Supports Java, JavaScript, Salesforce.com Apex and Visualforce, PLSQL, Apache Velocity, XML, XSL.
   d. Includes CPD, the copy-paste-detector.
      i. Finds duplicated code in Java, C, C++, C#, Groovy, PHP, Ruby, Fortran, JavaScript, PLSQL, Apache Velocity, Scala, Objective C, Matlab, Python, Go, Swift and Salesforce.com Apex and Visualforce.

116. Process: A process defines the building blocks of the work tracking system. Whenever you create a project, you select the process you want your project to use.

117. Swimlane diagrams

118. Bitbucket Pipelines & Deployments
   a. Bitbucket Cloud is a Git based code hosting and collaboration tool, built for teams.
   b. Bitbuckets best-in-class Jira and Trello integrations are designed to bring the entire software team together to execute on a project.
   c. Integrated CI/CD for Bitbucket Cloud that's trivial to set up, automating your code from test to production

119. Trello is the visual way for teams to collaborate on any project
   a. https://www.atlassian.com/software/trello

120. Jira
   a. Plan, track, and manage your agile and software development projects in Jira. Customize your workflow, collaborate, and release great software.

Azure Automation

121. Azure Automation State Configuration
   a. An Azure service that allows you to write, manage, and compile PowerShell Desired State Configuration (DSC) configurations.
   b. Also imports DSC Resources, and assigns configurations to target nodes, all in the cloud.
   c. Azure Automation State Configuration overview | Microsoft Learn
   d. Get started with Azure Automation State Configuration | Microsoft Learn

122. Register-AzureRmAutomationDscNode
   a. https://docs.microsoft.com/en-us/powershell/module/azurerm.automation/register-azurermautomationdscnode?view=azurermps-6.13.0

b. Registers an Azure virtual machine as an APS Desired State Configuration (DSC) node in an Azure Automation account.

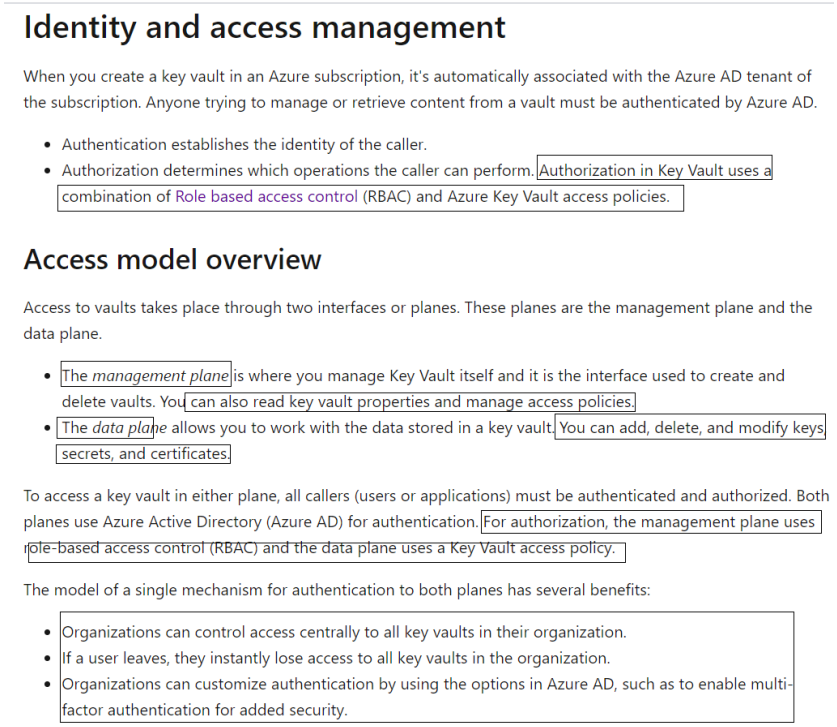
Analytics

1. Analytics widgets - Azure DevOps | Microsoft Learn
    a. Burndown widget remaining work across multiple teams and multiple sprints. Scope of work
    b. Burnup widget completed work across multiple teams and multiple sprints.
    c. Sprint Burndown to mitigate risk and check for scope creep.
    d. Cumulative Flow Diagram (CFD) count of work items (over time) for each column of a Kanban board – Bottlenecks?
    e. Cycle Time how long does it take my team to build a feature or fix a bug?
    f. Lead Time measures the total time elapsed from the creation of work items to their completion.
    g. Velocity: compare the work delivered against your plan and track work that's completed late.
    h. Test Results Trend (Advanced) track the daily count of tests, pass rates, and test duration. Quality: find outliers in your test results; longer to run; affect pass rate.
2. space

DSC Configurations

    a. DSC Configurations - PowerShell | Microsoft Learn
    b. Desired State PowerShell Management platform: infrastructure configuration as code.
    c. PSDesiredStateConfiguration
3. Azure Storage Analytics
    a. control-plane events - manage resources in your subscription.
    b. Control plane and data plane operations - Azure Resource Manager | Microsoft Learn
    c. data plane - use the data inside the key vault data plane; Key vault access policy
    d. Example: You create a storage account through the control plane. You use the data plane to read and write data in the storage account.
    e. URL is https://management.azure.com
    f. Enable access to Azure Resource Manager for template deployment

g.



Identity and access management

When you create a key vault in an Azure subscription, it's automatically associated with the Azure AD tenant of the subscription. Anyone trying to manage or retrieve content from a vault must be authenticated by Azure AD.

- Authentication establishes the identity of the caller.
- Authorization determines which operations the caller can perform. Authorization in Key Vault uses a combination of Role based access control (RBAC) and Azure Key Vault access policies.

Access model overview

Access to vaults takes place through two interfaces or planes. These planes are the management plane and the data plane.

- The *management plane* is where you manage Key Vault itself and it is the interface used to create and delete vaults. You can also read key vault properties and manage access policies.
- The *data plane* allows you to work with the data stored in a key vault. You can add, delete, and modify keys, secrets, and certificates.

To access a key vault in either plane, all callers (users or applications) must be authenticated and authorized. Both planes use Azure Active Directory (Azure AD) for authentication. For authorization, the management plane uses role-based access control (RBAC) and the data plane uses a Key Vault access policy.

The model of a single mechanism for authentication to both planes has several benefits:

- Organizations can control access centrally to all key vaults in their organization.
- If a user leaves, they instantly lose access to all key vaults in the organization.
- Organizations can customize authentication by using the options in Azure AD, such as to enable multi-factor authentication for added security.

4. Desired State Configuration (DSC)
    a. Resources do the work of ensuring that the target node is in the state defined by the configuration.
    b. For a DSC configuration to be applied to a node it must first be compiled into a MOF file.
5. Service Hook
    a. If you use Jenkins to build your apps, you can store your code in Azure DevOps Services and continue to use Jenkins for your continuous integration builds.
    b. You can trigger a Jenkins build when you push code to your project's Git repository or when you check in code to Team Foundation version control.
    c. https://docs.microsoft.com/en-us/azure/devops/service-hooks/services/jenkins?view=azure-devops
6. Cobertura
    a. A free Java tool that calculates the percentage of code accessed by tests. It can be used to identify which parts of your Java program are lacking test coverage. It is based on jcoverage
    b. https://cobertura.github.io/cobertura/ .
7. Apache Ant
    a. A Java library and command-line tool whose mission is to drive processes described in build files as targets and extension points dependent upon each other.
    b. The main known usage of Ant is the build of Java applications.
    c. Ant supplies a number of built-in tasks allowing to compile, assemble, test and run Java applications.
8. Slack Integration - Visual Studio Marketplace

   a. llow users to subscribe to notifications and monitor their pipelines, repositories and work items. Users can take actions like Approve deployments or create new work items in Azure Boards from Slack.
   b. Azure Pipelines with Slack - Azure Pipelines | Microsoft Learn
   c. Track the events occurring
   d. Use the Azure Boards app with Slack - Azure Boards | Microsoft Learn
   e. Create work items and monitor work item activity in your Azure Boards project from your Slack channel.
   f. Azure Repos with Slack - Azure Repos | Microsoft Learn
   g. Receive notifications in your channel whenever code is pushed/checked in and whenever a pull request (PR) is created, updated or a merge is attempted.
9. Selenium
10. Testing
11. CI/CD
   a. Securing Azure Pipelines - Azure Pipelines | Microsoft Learn
   b. Penetration Testing
12. Unit testing
   a. The practice of testing small pieces of code, typically individual functions, alone and isolated.
13. Functional testing
   a. Is defined as the testing of complete functionality of some application.
   b. In practice with web apps, this means using some tool to automate a browser, which is then used to click around on the pages to test the application.
14. Integration tests
   a. Similar to unit tests, but there's one big difference: while unit tests are isolated from other components, integration tests are not.
   b. For example, a unit test for database access code would not talk to a real database, but an integration test would.
15. Variable group
   a. Used to store values that you want to control and make available across multiple pipelines.
   b. Use variable groups to store secrets and other values that might need to be passed into a YAML pipeline.
   c. Variable groups are defined and managed in the Library page under Pipelines.
   d. https://docs.microsoft.com/en-us/azure/devops/pipelines/library/variable-groups?view=azure-devops&tabs=yaml
16. Space

Case Photoapps – note steps in Project Details

Overview
Photonapps is an online training provider

Requirements
Photonapps has currently undertaken several development projects. These projects will develop applications which will be hosted in Azure.

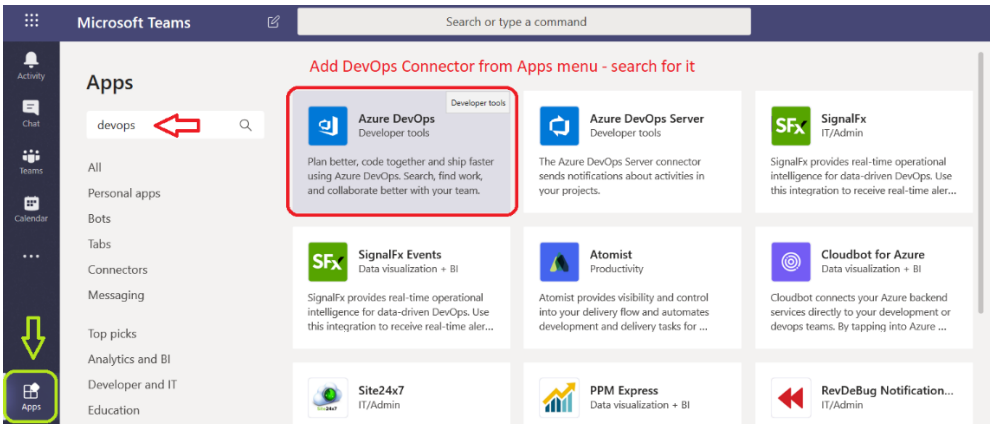
Photonapps goes ahead and set's up an Azure subscription and creates a Devops organization. The Devops organization currently consists of a Docker extension and an agent pool named photonappppool. The deployment pool contains 5 Azure Virtual Machines that run Windows Server 2016.

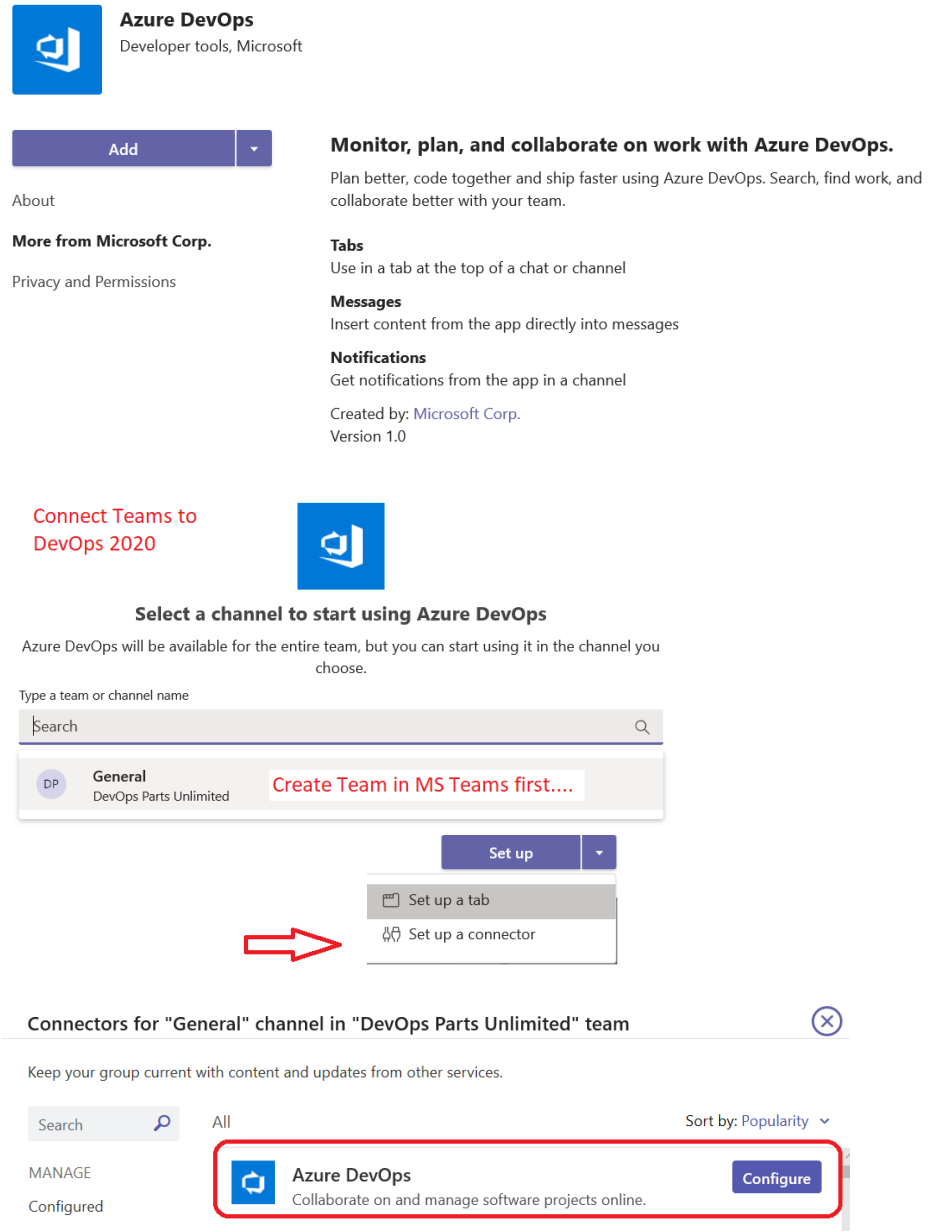
Below are the different projects which are going to be hosted in Azure Devops

Project Name	Project Details
photonappA	This project will ensure support for incremental builds
photonappB	This project will have integration with SonarQube
photonappC	This project will ensure to have a build pipeline that would create a Docker image. The image would be pushed to Azure Container Registry.
photonappD	This project would contain a Git repository. A trigger would be in place to initiate a build from any commit changes to the repository. The trigger must not consider any changes within a folder named /data in the repository.
photonappE	This project will contain a target deployment group called photonappgroup. This project will use Azure Automation State Configuration to maintain the desired state for computers in photonappgroup
photonappF	This project will have support for build and deployment pipelines. It needs to be ensured that the deployments are only allowed if the number of active bugs is 0.

Add Microsoft Teams Connection

What may show as Connectors in some media – click Apps, Search DevOps – follow the wizard.





Azure DevOps: https://docs.microsoft.com/en-us/connectors/visualstudioteamservices/

References

15. What is DevOps?
    a. https://docs.microsoft.com/en-us/azure/devops/learn/what-is-devops
16. What is Azure DevOps? – Note where you are in the TOC.
    a. https://docs.microsoft.com/en-us/azure/devops/user-guide/what-is-azure-devops?view=azure-devops
17. Azure DevOps Server was formerly named Visual Studio Team Foundation Server (TFS).

   a. https://docs.microsoft.com/en-us/azure/devops/user-guide/what-is-azure-devops?view=azure-devops
18. Differences between global Azure, Azure Stack Hub, and Azure Stack HCI:
   a. https://docs.microsoft.com/en-us/azure-stack/operator/compare-azure-azure-stack?view=azs-2002
19. Plan your organizational structure:
   a. https://docs.microsoft.com/en-us/azure/devops/user-guide/plan-your-azure-devops-org-structure?view=azure-devops
20. QuickStart: Sign up, sign in to Azure DevOps:
   a. https://docs.microsoft.com/en-us/azure/devops/user-guide/sign-up-invite-teammates?toc=%2Fazure%2Fdevops%2Fget-started%2Ftoc.json&bc=%2Fazure%2Fdevops%2Fget-started%2Fbreadcrumb%2Ftoc.json&view=azure-devops
21. Azure DevOps Services integration with Microsoft Teams
   a. Provides a comprehensive chat and collaborative experience across the development cycle. Teams can easily stay informed of important activities in your Azure DevOps team projects with notifications and alerts on work items, pull requests, code commits, build and release.
   b. https://www.azuredevopslabs.com/labs/vstsextend/teams/
22. Exam AZ-400: Microsoft Azure DevOps Solutions:
   a. https://docs.microsoft.com/en-us/learn/certifications/azure-devops#certification-exams
   b. Prerequisite 1 exam
23. On-premises XML process customization:
   a. https://docs.microsoft.com/en-us/azure/devops/reference/on-premises-xml-process-model?view=azure-devops-2019
24. Azure Pipelines:
   a. Automate your builds and deployments with Pipelines so you spend less time with the nuts and bolts and more time being creative
25. MCE: Microsoft DevOps Engineer:
   a. https://docs.microsoft.com/en-us/learn/certifications/azure-devops
26. Agile vs. Waterfall vs. Kanban vs. Scrum: What's the Difference?
   a. https://www.lucidchart.com/blog/agile-vs-waterfall-vs-kanban-vs-scrum
   b. https://www.lucidchart.com/blog/how-kanban-methodology-can-improve-your-team
27. Space

Noise – Setup Demo Generator Options

- **Identity (read)**
- Grants the ability to read identities and groups.

- **Work items (full)**
- Grants full access to work items, queries, backlogs, plans, and work item tracking metadata. Also provides the ability to receive notifications about work item events via service hooks.

- **Build (read and execute)**

- Grants the ability to access build artifacts, including build results, definitions, and requests, and the ability to queue a build, update build properties, and the ability to receive notifications about build events via service hooks.

- **Code (read, write, and manage)**
- Grants the ability to read, update, and delete source code, access metadata about commits, changesets, branches, and other version control artifacts. Also grants the ability to create and manage code repositories, create and manage pull requests and code reviews, and to receive notifications about version control events via service hooks.

- **Agent Pools (read, manage)**
- Grants the ability to manage pools, queues, and agents

- **Test management (read and write)**
- Grants the ability to read, create, and update test plans, cases, results and other test management related artifacts.

- **Extensions (read and manage)**
- Grants the ability to install, uninstall, and perform other administrative actions on installed extensions.

- **Release (read, write, execute and manage)**
- Grants the ability to read, update and delete release artifacts, including folders, releases, release definitions and release environment and the ability to queue and approve a new release.

- **Task Groups (read, create and manage)**
- Grants the ability to read, create and manage taskgroups.

- **Variable Groups (read, create and manage)**
- Grants the ability to read, create and manage variablegroups.

- **Service Endpoints (read, query and manage)**
- Grants the ability to read, query and manage service endpoints.

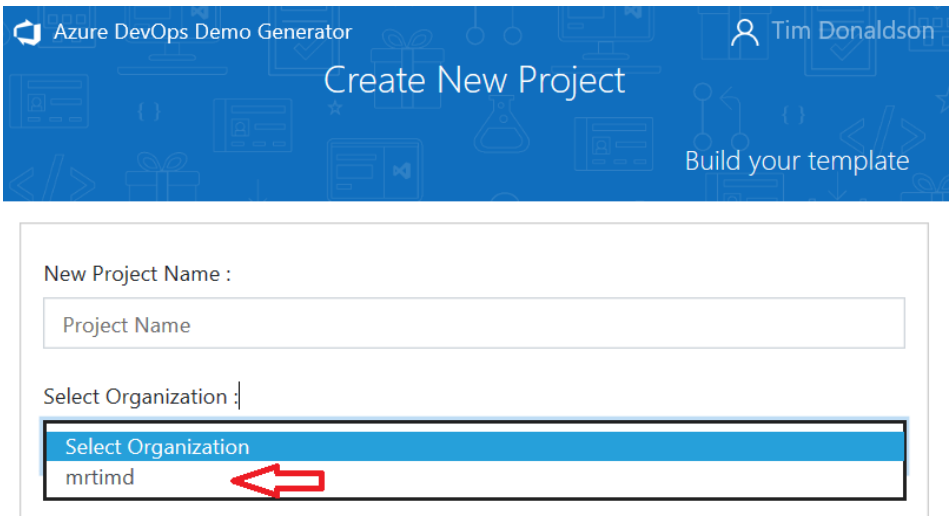
- **Project and team (read, write and manage)**
- Grants the ability to create, read, update, and delete projects and teams.

- **Team dashboards (manage)**
- Grants the ability to manage team dashboard information

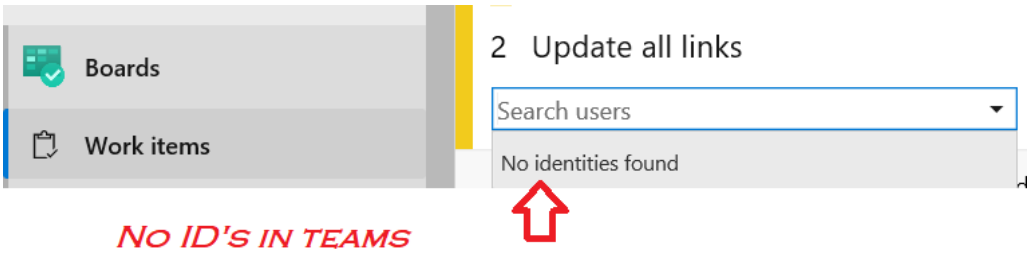
- **Wiki (read and write)**
- Grants the ability to read, create and updates wikis, wiki pages and wiki attachments.

Setup Issues

Could not load your organizations. Please check if the logged in Id contains the Azure DevOps Organizations or change the directory in profile page and try again.



After logging into DevOps and Portal with same organizational account I got a value in the drop down.



Code shots

[Key Vault secret with template - Azure Resource Manager | Microsoft Learn]

```
{
    "apiVersion": "2018-05-01",
    "name": "dynamicSecret",
    "type": "Microsoft.Resources/deployments",
    "properties": {
        "mode": "Incremental",
        "templateLink": {
            "contentVersion": "1.0.0.0",
            "uri": "[uri(parameters('_artifactsLocation'), concat('./nested
        },
        "parameters": {
            "location": {
                "value": "[parameters('location')]"
            },
            "adminLogin": {
                "value": "ghuser"
            },
            "adminPassword": {
                "reference": {
                    "keyVault": {
                        "id": "[resourceId(parameters('vaultSubscription'),
                    },
                    "secretName": "[parameters('secretName')]"
                }
            }
        }
    }
}
```

References

- Course AZ-400T00--A: Designing and Implementing Microsoft DevOps solutions - Training | Microsoft Learn
- Exam AZ-400: Designing and Implementing Microsoft DevOps Solutions - Certifications | Microsoft Learn