Azure DevOps Documentation

Azure DevOps:

Azure DevOps offers developer services that help teams to plan their work, collaborate on code development, and build and deploy applications. It allows organizations to create and improve products at a much faster rate than other softwares**.**

Services:

- Azure Repos
- Azure Pipelines
- Azure Boards
- Azure Test Plans
- Azure Artifacts

Azure DevOps Architecture:

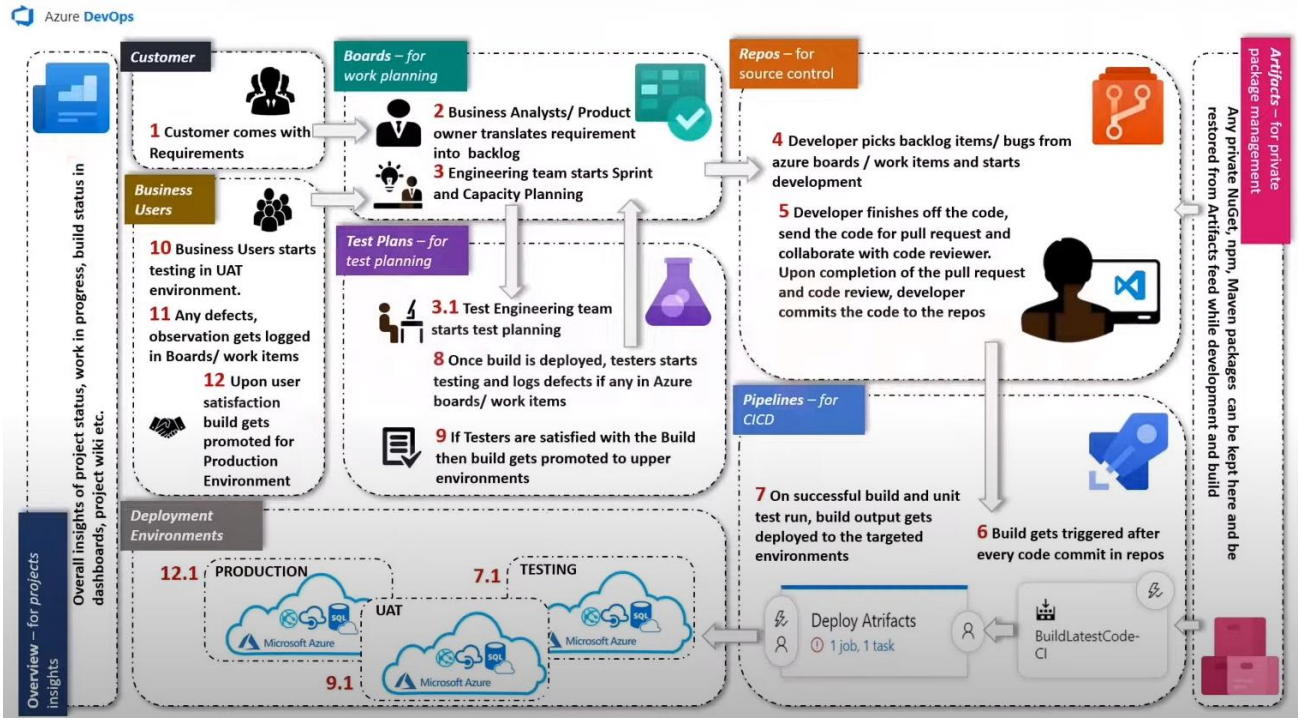


Fig: Azure DevOps Architecture

Azure Repos:

Azure Repos is a set of version control tools that you can use to manage your code. Whether your software project is large or small, using version control as soon as possible is a good idea.

Version Control tool:

- Git
- TFVC (Team Foundation Version Control)

Git is a distributed version control system, meaning that your local copy of code is a complete version control repository. These fully functional local repositories make it is easy to work offline or remotely. You commit your work locally, and then sync your copy of the repository with the copy on the server.

TFVC is a centralized version control system. Typically, team members have only one version of each file on their dev machines. Historical data is maintained only on the server. Branches are path-based and created on the server.

Branch: Branches are lightweight references that keep a history of commits and provide a way to isolate changes for a feature from your main branch and other work.

Commit: A commit is a group of changes saved to your local repository. You can share these changes to the remote repository by pushing.

Fork: A fork is a complete copy of a repository, including all files, commits, and branches(optionally).

Branch policies: The help protect the important branches in your development.

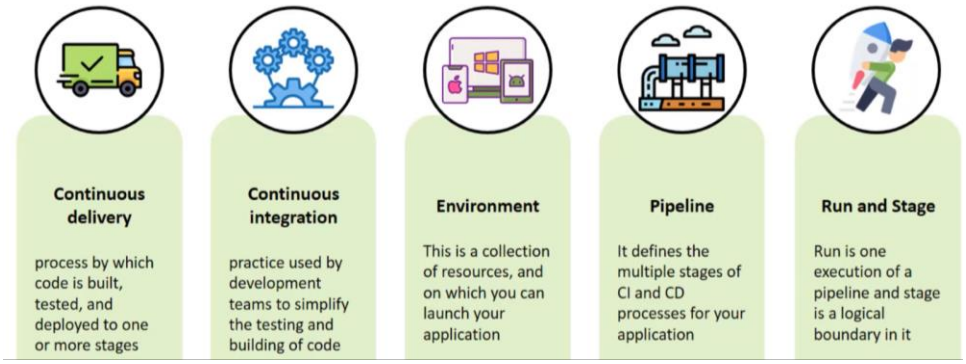
- Isolate work in progress from the completed work in your main branch.
- Guarantee changes build before they get to main.
- Limit who can contribute to specific branches.
- Enforce who can create branches and the naming guidelines for the branches.

Azure Pipelines:

Azure Pipelines automatically builds and tests code projects to make them available to others. Azure Pipelines combines continuous integration (CI) and continuous delivery (CD) to test and build your code and ship it to any target.

Continuous Integration (CI) is the practice used by development teams of automating merging and testing code. Implementing CI helps to catch bugs early in the development cycle, which makes them less expensive to fix.

Continuous Delivery (CD) is a process by which code is built, tested, and deployed to one or more test and production environments.



Azure Boards:

Azure Boards provides software development teams with the interactive and customizable tools they need to manage their software projects. It provides a rich set of capabilities including native support for Agile, Scrum, and Kanban processes, calendar views, configurable dashboards, and integrated reporting.

- Track user stories, bugs, features, and epics.
- Use interactive backlogs, boards, lists, and calendar views.
- Use GitHub, track work in Azure Boards.
- Implement Agile, Scrum, and Kanban processes.
- Gain visibility through end-to-end traceability.
- Support independent, autonomous teams.

Azure Test Plans:

Azure Test Plans provides rich and powerful tools everyone in the team can use to drive quality and collaboration throughout the development process.

Azure Artifacts:

Azure Artifacts enables developers to share their code efficiently and manage all their packages from one place.

Process of a building project:

1. Creating an Azure DevOps organization.
2. Creating an Azure DevOps project.
3. Integrating GitHub with the DevOps project and Azure Boards.
4. Creating a CI/CD pipeline for any application with Azure Pipelines and Repos (Build, Staging and production)
5. Use Azure Test and Feedback tool to identify and create work items and manual testing using Azure Test Plans
6. Creating a dashboard for an overview of the DevOps project.