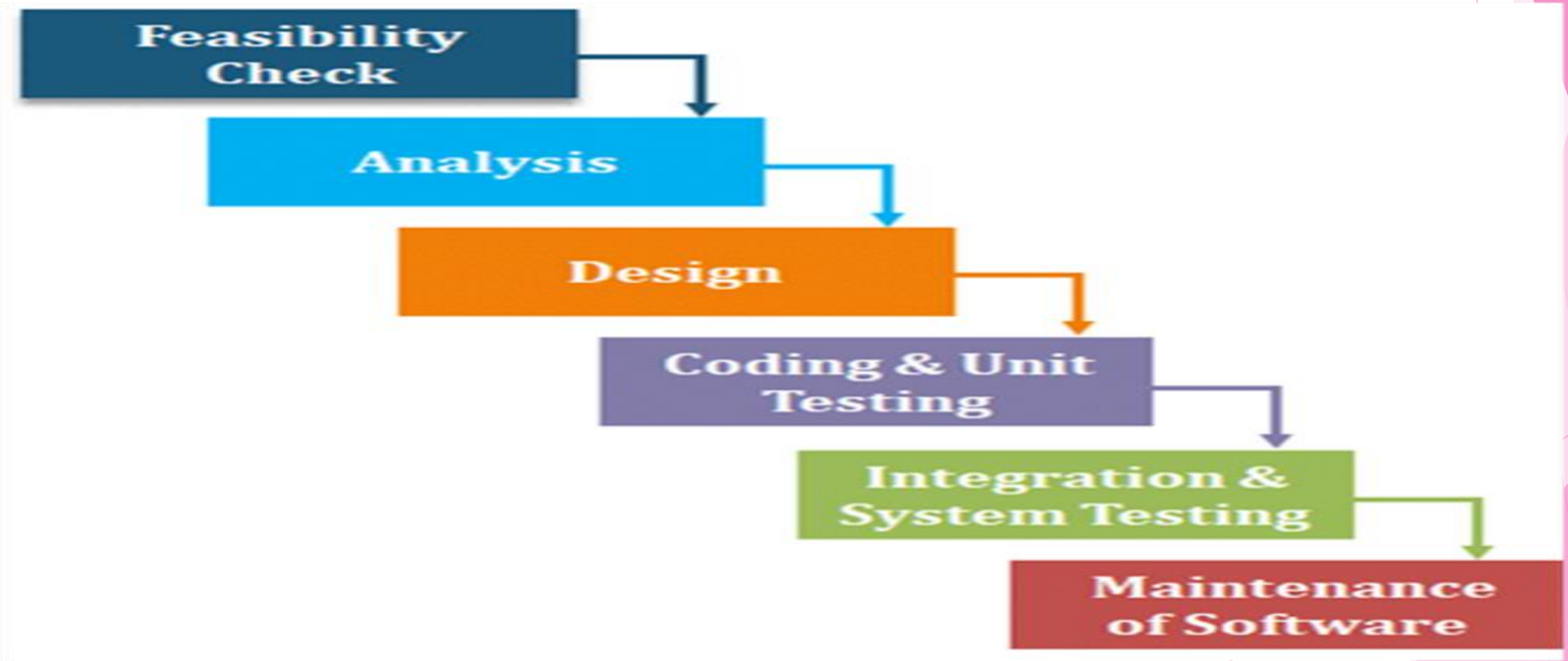
Azure devops

SDLC


Feasibility Check
Analysis
Design
Coding & Unit Testing
Integration & System Testing
Maintenance of Software

Waterfall model

- Although the waterfall model was highly influential and continues to serve as the basis for many applications today
- It had significant drawbacks:
  - Testing was done only after the complete development had been done.
  - If a client had any feedback, it would take a lot of time and money to rebuild the application.
  - It was the best method to follow if the client knew exactly what they wanted without any changes in between.

Agile method

- To overcome these problems, the Agile method came into existence.
- But, there were problems with this method too. Here, the cons were the following:
  - From developers to clients, everyone had to be in sync in order to proceed with development.
  - With no clear vision of the end product, it was difficult to navigate through the right track. Often there would be setbacks which used to lead to delays in development.
  - With no proper documentation, it was difficult to trace back or to cross-check.
- Like Agile was used because there were cons in the waterfall model, DevOps was the solution to all the problems the Agile model had.
- Development and operations teams treaded differently until DevOps came into existence.
- Automation is one of its main advantages due to which efficiency also increases.
- DevOps pays equal attention to all phases, from development to deployment.

DevOps

- A methodology or a practice that
  - brings together development (Dev) and operations (Ops)
  - to teams for deploying efficient applications
  - while shortening the development life cycle overall.

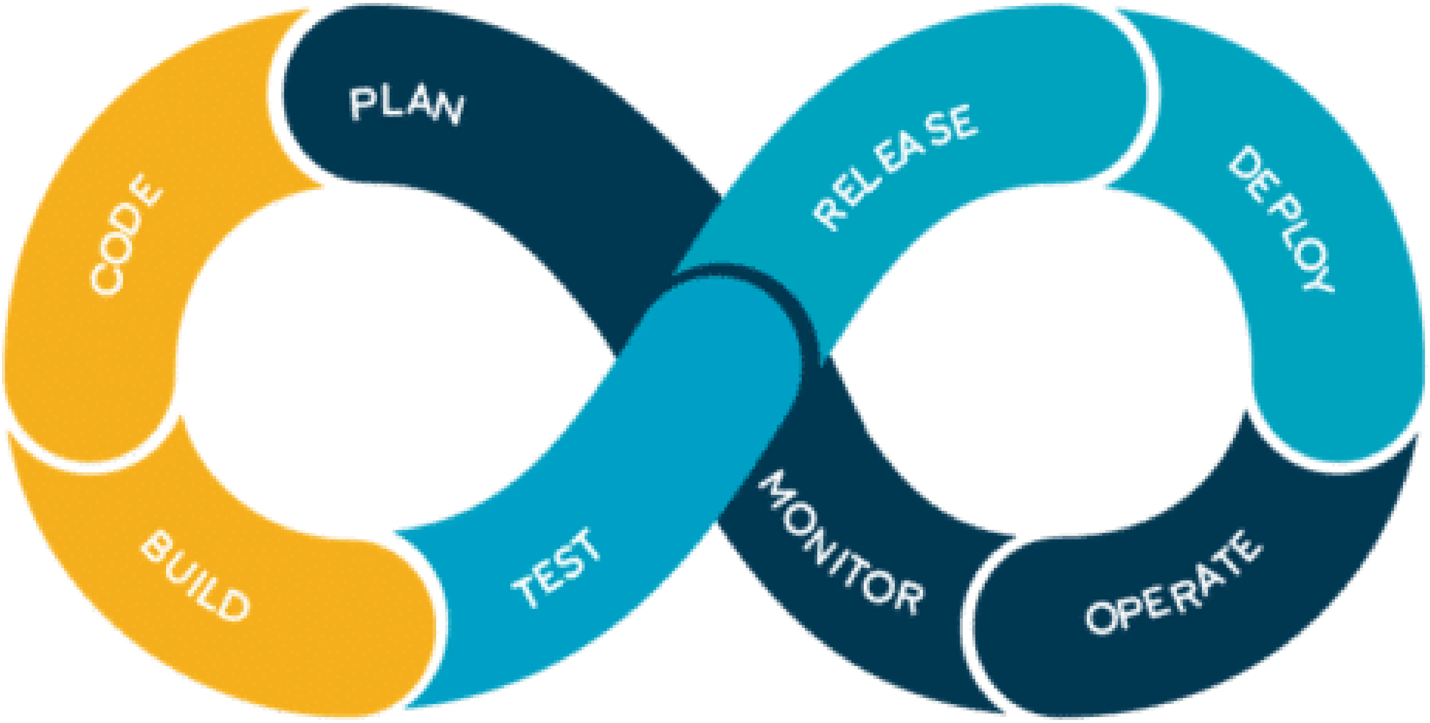
DevOps Lifecycle

- DevOps focuses on bringing all the development, operations, and IT infrastructure guys, including Developers, Testers, System Admins, and QAs, under one roof.
- Hence, all these people together are called DevOps Engineers.
- DevOps Engineers share the end-to-end responsibility of
  - gathering information,
    setting up the infrastructure,
  - developing, testing,
  - deploying,
  - continuously monitoring, and
  - fetching feedback from end-users.
- This process of developing, testing, deploying, and monitoring keeps on repeating for better results.

Azure DevOps

- comes with two options:
  - Azure DevOps Services
  - Azure DevOps Server

DevOps Lifecycle



Devops Lifecycle

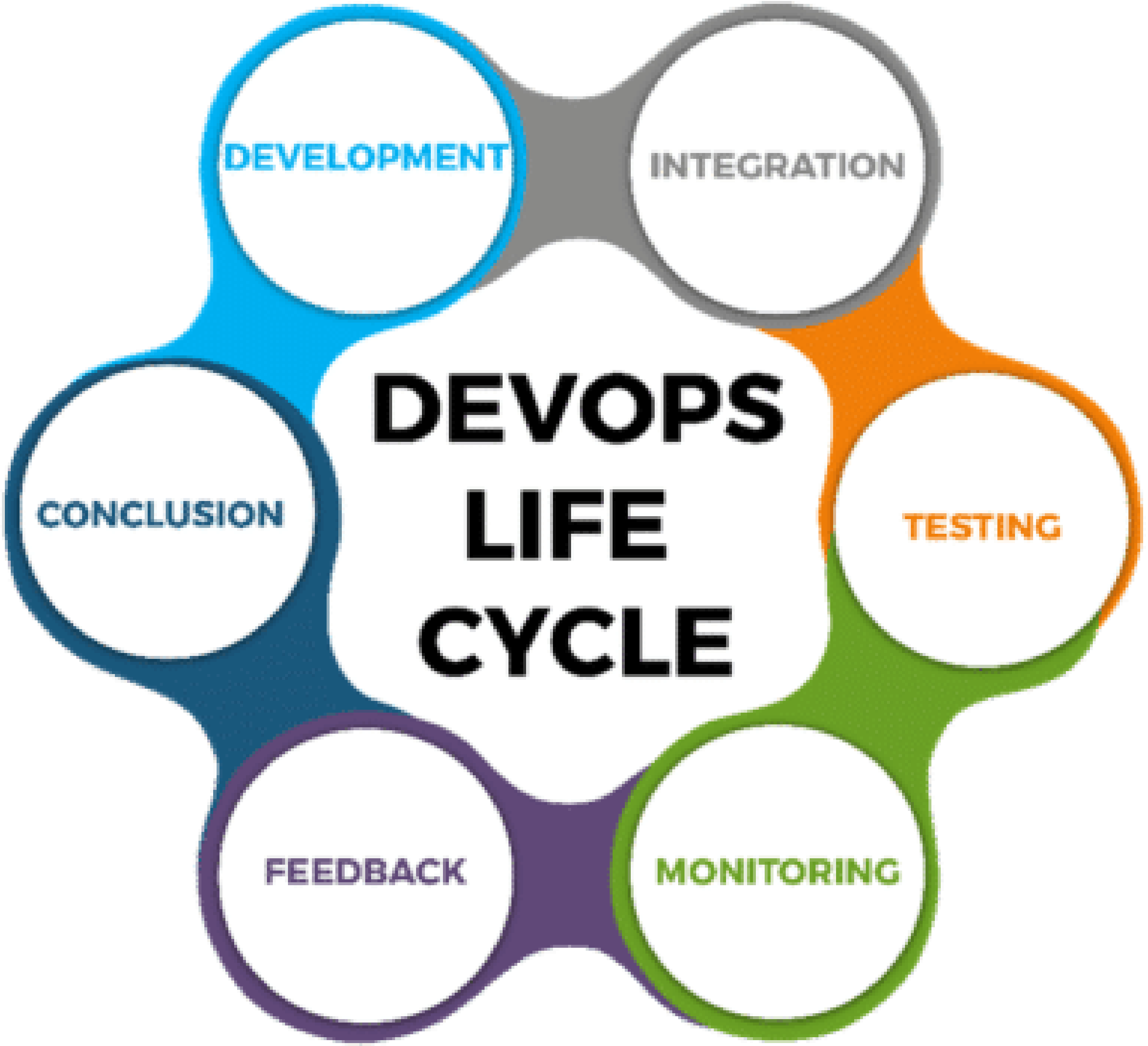
- Code: The first step in the DevOps life cycle is coding, where developers build the code on any platform
- Build: Developers build the version of their program in any extension depending upon the language they are using
- Test: For DevOps to be successful, the testing process must be automated using any DevOps automation tool like Selenium
- Release: A process for managing, planning, scheduling, and controlling the build in different environments after testing and before deployment
- Deploy: This phase gets all artifacts/code files of the application ready and deploys/executes them on the server
- Operate: The application is run after its deployment, where clients use it in real-world scenarios.
- Monitor: This phase helps in providing crucial information that basically helps ensure service uptime and optimal performance
- Plan: The planning stage gathers information from the monitoring stage and, as per feedback, implements the changes for better performance

- the different stages in the DevOps life cycle that contributes to the consistent software development life cycle (SDLC):

- Continuous Development
- Continuous Integration
- Continuous Testing
- Continuous Monitoring
- Virtualization and Containerization

Stages in the DevOps life cycle

- The different stages that contributes to the consistent software development life cycle (SDLC):
  - Continuous Development
  - Continuous Integration
  - Continuous Testing
  - Continuous Monitoring
  - Virtualization and Containerization


DEVELOPMENT
INTEGRATION
DEVOPS LIFE CYCLE
CONCLUSION
TESTING
FEEDBACK
MONITORING

Continuous Development

- In the Waterfall model, our software product gets broken into multiple pieces or sub-parts for making the development cycles shorter, but in this stage of DevOps, the software is getting developed continuously.

- Tools used: As we code and build in this stage, we can use GIT to maintain different versions of the code. To build/package the code into an executable file, we can use a reliable tool, namely, Maven.

Continuous Integration

- In this stage, if our code is supporting new functionality, it is integrated with the existing code continuously. As the continuous development keeps on, the existing code needs to be integrated with the latest one '**continuously**,' and the changed code should ensure that there are no errors in the current environment for it to work smoothly.
- **Tools used**: **Jenkins** is the tool that is used for continuous integration. Here, we can pull the latest code from the GIT repository, of which we can produce the build and deploy it on the test or the production server.

Continuous Testing

- In the continuous testing stage, our developed software is getting tested continuously to detect bugs using several automation tools.

- Tools used: For the QA/Testing purpose, we can use many automated tools, and the tool used widely for automation testing is Selenium as it lets QAs test the codes in parallel to ensure that there is no error, incompetencies, or flaws in the software.

Continuous Monitoring

- A very crucial part of the DevOps life cycle where it provides important information that helps us ensure service uptime and optimal performance.
- The operations team gets results from reliable monitoring tools to detect and fix the bugs/flaws in the application.
- Tools used:
  - Such as Nagios, Splunk, ELK Stack, and Sensu are used for monitoring the application.
  - They help us monitor our applications and servers closely to check their health and whether they are operating actively.
  - Any major issue detected by these tools is forwarded to the development team to fix in the continuous development phase.

DevOps Benefits

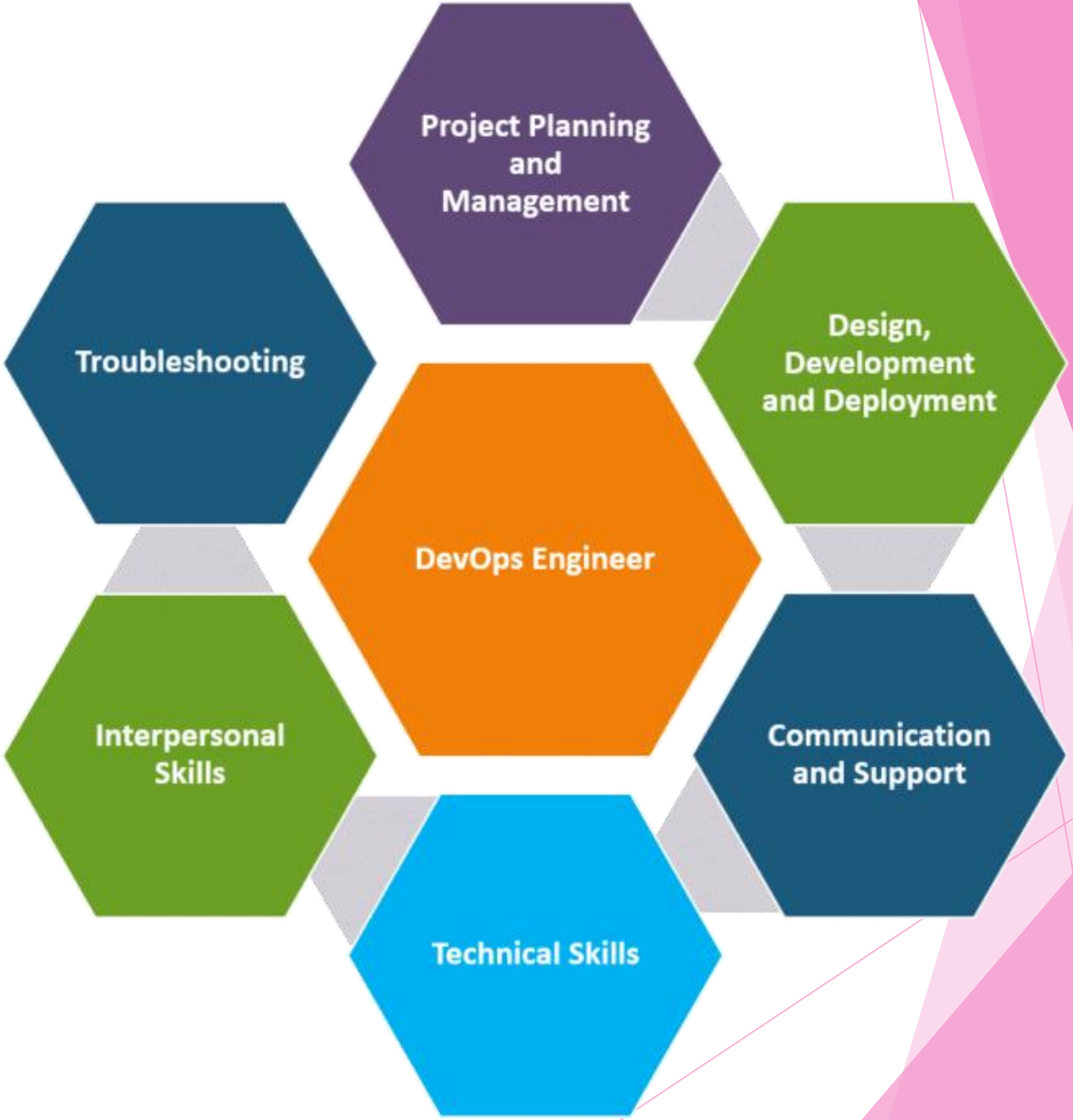
- After being successfully implemented in SDLC, now DevOps is considered the key to speeding up various cloud platforms. Its all-rounder performance has attracted aspirants to build a career in this domain, and hence having sound knowledge is imperative for them.

- DevOps is a contemporary approach that lets companies utilize numerous benefits. Some of the major DevOps benefits are as follows:

Devops benefits

Breaking Silos
Continuous Improvement
Minimized Failures
Creativity & Innovation
Performance-oriented Culture

Roles and responsibility of devops engr


Project Planning and Management
Troubleshooting
Design, Development and Deployment
DevOps Engineer
Interpersonal Skills
Communication and Support
Technical Skills

Azure DevOps

- Azure DevOps comes with two options:
  - Azure DevOps Services
  - Azure DevOps Server

Azure DevOps Services Vs. Server

Azure DevOps Services	Azure DevOps Server
It is a cloud offering.	It is an on-premise offering.
It offers two options for scaling and scoping data: organizations and projects.	It offers three options for scaling and scoping data: deployment, project collections, and projects.
You can connect over the public network.	You can connect to the intranet server.
The access level must be assigned to each user.	Access levels must be set based on the license.

Azure DevOps Services

Azure DevOps
Azure Boards
Azure Repos
Azure Pipelines
Azure Test Plans
Azure Artifacts

Azure Boards

- Azure boards make it easy to track tasks, bugs, and features.
- There are three types of work items:
  - Epic
  - Issue
  - Task
- An epic work item tracks requirements or features; issue tracks bugs or smaller changes, and task tracks even smaller works done. It is easy to add or update your work status and, with a drag-and-drop feature, you can prioritize your works.

Azure Repos

- Before talking about Azure Repos, let’s, first, discuss what a version control system is.
- Version control systems are software that helps you track changes that you have made in our code, by taking snapshots of our files. One popular example is Git.
- Azure Repos is a collection of version control tools that are used to manage our code and provides two types of version control:
  - Git
  - Team Foundation Version Control (TFVC)
- Git is a distributed version control system which means that the local copy that we have is a complete repository allowing us to work offline as well.
- In TFVC, the historical data is kept only on the server since it is a centralized version control system.

Azure Pipeline

- Azure Pipeline is a combination of continuous integration (CI) and continuous delivery (CD) which we can use to create and test our code automatically and give access to other users.
- We can produce consistent and quality code with CI and CD.
- We can work with programming languages like Python, Ruby, Java, PHP, C#, and Go.
- To use Azure Pipelines, our source code must be stored in a version control system such as Git.

Azure Test Plans

- We can improve the quality of our code by testing it.
- With Azure Test Plans, we can
  - test our code manually or exploratorily as well.
  - request, provide, and track feedback also.
  - perform unit and functional testing. Running tests continuously is also possible with Continuous testing.

Azure Artifacts

- A store that has all our artifacts that were produced while developing and deploying.
- Artifacts are executable files, i.e., they do not contain any code.
- We can use multiple feeds to organize and control access to our packages.
  - A feed is a container for packages that helps in consuming and publishing.
  - Azure Artifacts provide a fast, secure, and easy feed of binary packages.
- Azure Artifacts ensure that our pipelines are fully integrated package management.
- We can also create packages like Maven, npm, and NuGet.

Azure DevOps Tools

- Azure Pipelines:
  - enables the implementation of continuous integration and continuous deployment (CI/CD) processes,
  - facilitating the seamless building, testing, and deployment of applications to diverse platforms and cloud environments.
- Azure Boards:
  - With this tool, teams can effectively plan, track, and engage in collaborative discussions regarding work items.
  - provides features like Kanban boards, backlogs, team dashboards, and customizable reporting to enhance productivity and transparency.

Azure DevOps Tools

- Azure Monitor:
    - offers comprehensive observability capabilities, allowing users to gain valuable insights into the performance of their applications, infrastructure, and network.
    - It facilitates monitoring, diagnostics, and troubleshooting to ensure efficient operations.
- Visual Studio:
    - This IDE is specifically designed for creating robust and scalable applications intended for Azure.
    - It offers a rich set of features and tools to enhance the development experience and streamline application deployment.

Azure DevOps Tools

- Azure Kubernetes Service:
  - provides a managed Kubernetes environment in Azure, enabling developers to deploy, scale, and manage containerized applications effortlessly.
  - It simplifies the orchestration and management of containers, allowing for efficient application scaling and resource utilization within Azure.

Azure DevOps Benefit

- Reliability:
  - Azure DevOps is reliable, scalable, and globally available.
- Accessibility:
  - Azure DevOps users get access to new features every three weeks. Microsoft is transparent.
- Flexibility:
  - If our DevOps team doesn’t want or need the full suite of services, they can acquire them independently.
- End of upgrade cycle:
  - For organizations running on-premises CI/CD tooling, upgrading is difficult. Using SaaS, it is an easy task.