DevOps?

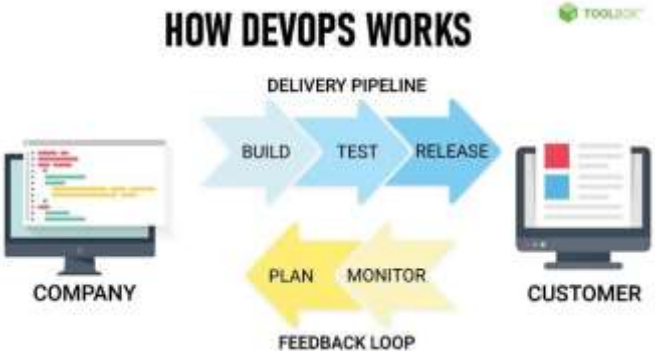
The word "DevOps" was coined in 2009 by Patrick Debois, who became one of its gurus. The term was formed by combining "development" and "operations," which provides a starting point for understanding exactly what people typically mean when they say "DevOps." Notably, DevOps isn't a process or a technology or a standard. Many devotees refer to DevOps as a "culture"—a viewpoint that New Relic favors. We also use the term "DevOps movement" when talking about topics such as adoption rates and trends for the future, and "DevOps environment" to refer to an IT organization that has adopted a DevOps culture.

"DevOps represents a change in IT culture, focusing on rapid IT service delivery through the adoption of agile, lean practices in the context of a system-oriented approach. DevOps emphasizes people (and culture), and seeks to improve collaboration between operations and development teams. DevOps implementations utilize technology— especially automation tools that can leverage an increasingly programmable and dynamic infrastructure from a life cycle perspective."

DevOps is defined as a combination of processes and tools created to facilitate organizations in delivering services and applications much faster than they can through conventional software development processes. It helps increase customers' confidence in the applications that an organization offers, thereby allowing the company to flourish and achieve its business goals faster.

Under the DevOps model, development and operations teams work in constant cohesion throughout the entire project lifecycle, starting right from development to deployment. When security is the main focus, the quality assurance team is tightly knitted with the DevOps team throughout the app lifecycle. In this situation, some DevOps teams are also referred to as **DevSecOps**. Close coordination with the QA team ensures that no loopholes are left unchecked in the provided service/app.

How DevOps Works?



A DevOps process can be summarized as an infinite loop that comprises the following stages — build, test, and release through the delivery pipeline and plan and monitor through feedback, which resets the loop again. With such an amazing combination, teams use tech stack and tooling that assists them in reliably developing apps. Moreover, going away from the norm, teams use automated processes here. DevOps tools also allow engineers to complete different tasks independently. Be it provisioning infrastructure or deploying code, they can accomplish these tasks without being dependent on one another. As such, the DevOps model accelerates the overall application development process.

Key Goals and Benefits of DevOps

Goals of DevOps

The fast-paced growth of the IT industry and continuous advancements in technology make it critical to set DevOps goals that are experimental and challenging for companies to compete and thrive in the market. Here are the key goals and principles that every successful DevOps program has in common.

1. Ensures effective collaboration between teams: Effective collaboration in any process relies on shared ownership. During the development process, all those involved should embrace the fact that everyone is equally responsible for the entire development process. Whether it is development, testing, or deployment, each team member should be involved. They should understand that they have an equal stake in the final outcome. In the DevOps paradigm, passing of work from one team to another is completely defined and broken down. This accelerates the entire process of development since collaboration between all the teams involved is streamlined.
2. Creates scalable infrastructure platforms: The primary focus of DevOps is to create a sustainable infrastructure for applications that make them highly scalable. According to the demands of the modern-day business world, scalable apps have become an absolute necessity. In an ideal situation, the process of scaling should be reliable and fully automated. As a result, the app will have the ability to adapt to any situation when a marketing effort goes viral. With the app being scalable, it can adjust itself to large traffic volumes and provide an immaculate user experience.
3. Builds on-demand release capabilities: Companies must focus on keeping their software in a 'releasable' state. Continuous delivery will allow the software to add new features and go live at any stage. DevOps aims to automate the process of release management because it has a plethora of advantages. Automated release management is predictable, fast, and very consistent. Moreover, through automation, companies can release new versions as per their requirements. Automated release management also has complete and thorough audit trials, as these are essential for compliance purposes.
4. Provides faster feedback: Automating monotonous tasks such as testing and reporting will accelerate the process of rapid feedback. Since the development team will know what has to change, it can roll out the updated version faster. In addition, the team can better understand the impact of the changes that it has done in the software lifecycle. A concrete understanding of changes will assist team members in working efficiently in tandem. With rapid feedback, the operations team and developers can make better decisions collectively and enhance the app's performance.

Benefits of DevOps

DevOps helps organizations deliver added value to their customers. Here are some compelling benefits of DevOps.

1. Smarter work and faster release: With DevOps, your development team can release the required deliverables quickly. Faster release of deliverables will keep you miles ahead of your competitors, which is very important in today's cut-throat business realm. Businesses should understand that if their review cycle is not automated, it will slow down the release process. Moreover, the inclusion of disparate tools will lead to context switching and higher costs. Thus, DevOps can help rectify this worrisome business situation.
2. Quick resolution of issues: In a business world where speed and accuracy are paramount, a fast feedback loop will help you thrive. With DevOps, the communication process becomes seamless, and, as such, it minimizes the time required to solve issues. Without open communication, key issues can slip out of mind, which will have serious repercussions in the long run. DevOps fosters open

communication that helps resolve issues, thus unblocking the release pipeline faster.

3. Better collaboration between teams: DevOps paves the way for more dynamic and round-the-clock communication between teams. It renders an environment for mutual collaboration and integration among teams that are distributed globally. Eliminating the traditional departmental barriers between teams forms a new sense of ownership, wherein each team member feels equally responsible for meeting delivery timelines. This collaboration contributes to happier and more engaged employees.
4. Fostering innovative mindsets: With DevOps, deployment phases of the application are more relaxed as compared to traditional methods. This is because it streamlines the entire process, ensures that there are no lapses in quality, and allows on-time and efficient release. Thus, as everything is in order, the development team is more at peace. This allows it to think out of the box and provide additional value to the user. Having a development team with an innovative mindset is a boon for any business organization. An innovative approach, in itself, has immense scope and leads to better quality and resolution of issues at hand. Thus, through DevOps, the process of expanding the horizon of an app becomes much easier.
5. Faster threat detection: Automated and continuous testing of the code will make the process of threat detection faster. As developers can locate problem areas at an early stage, they can then resolve them faster. Thus, DevOps is a vital cog in maintaining and enhancing the quality and performance of an app. As the overall build of the app is in capable hands, teams working together are empowered to share feedback as and when necessary.
6. Increased customer satisfaction: Customer satisfaction is paramount in any day and age, irrespective of the business one is involved in. DevOps is known for enhancing customer experience, which ultimately increases the level of customer satisfaction. Dissatisfied customers are never a good sign for any business. Feedback loops are an important component of DevOps. These loops empower end users to track the progress of app development at various stages.
7. In addition, they can suggest changes (if any) or give their inputs to make the app more customer-centric. Due to their dynamic nature, feedback loops help developers and customers remain on the same page. Moreover, DevOps accelerates the process of app development, which eventually lessens the delivery timer. This has a positive impact on the customer satisfaction ratio.
8. Providing the much-needed edge: Along with staying true to their development process, companies need to ensure that they sustain themselves in the cut-throat competition. Implementing DevOps can be your trump card to provide your organization with that much-needed edge. Competitive advantage is necessary, as it can become the deciding factor in the popularity of an application in many cases. Some factors set expert businesses apart from mediocre ones:

    - Top-quality features
    - Quicker and timely software releases
    - Maximizing return on investments
    - Listening to constructive feedbacka

Difference between Agile and DevOps

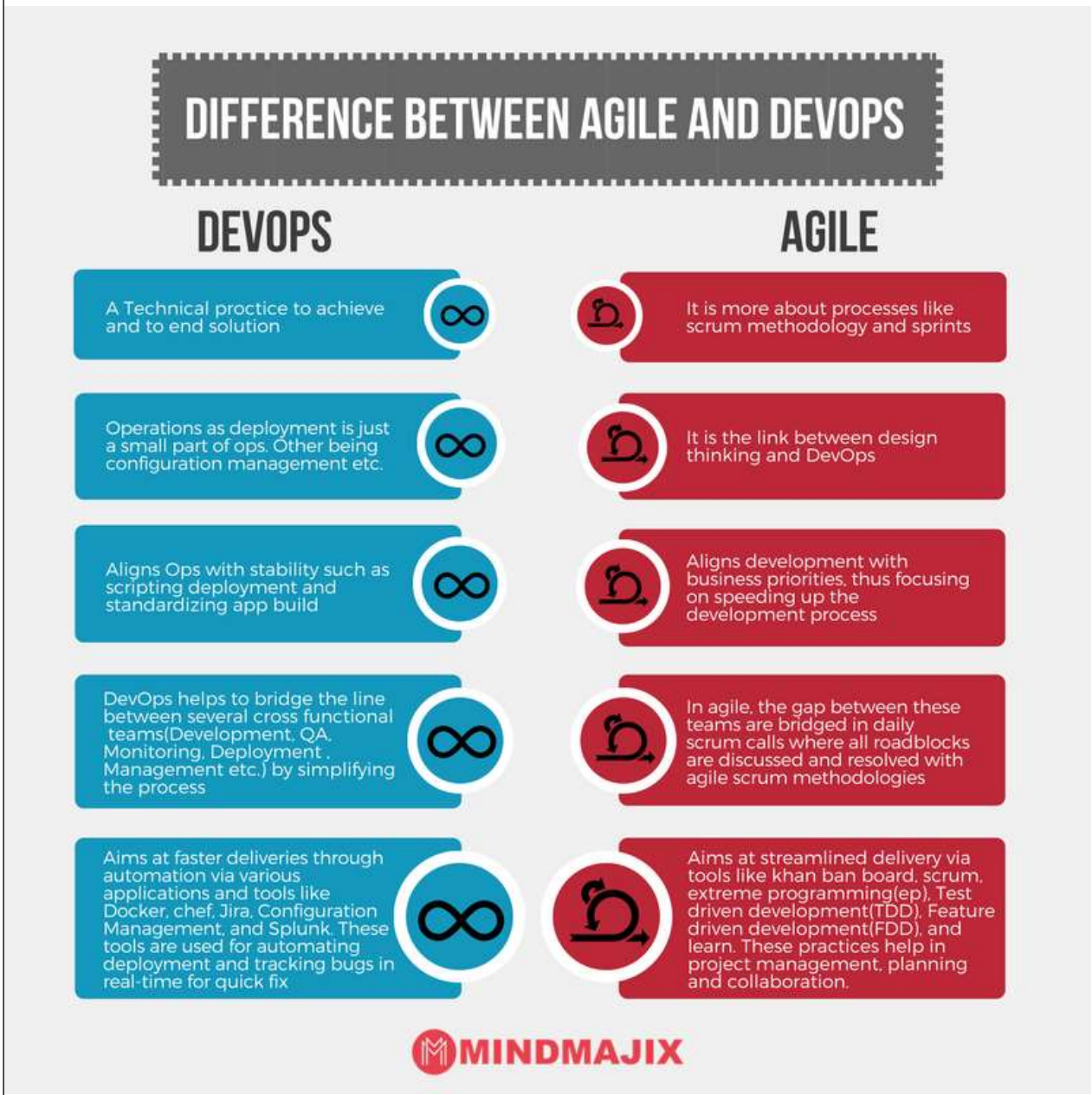
Agile: Agile program advancement comprises different approaches to computer program improvement beneath which prerequisites and arrangements advance through the collaborative exertion of self-organizing and cross-functional groups and their

customer/end client.

DevOps: DevOps could be a set of hones that combines program improvement and information-technology operations which points to abbreviating the framework's advancement life cycle and giving nonstop conveyance with tall program quality.

Example: Facebook's mobile app which is updated every two weeks effectively tells users you can have what you want and you can have it. Now ever wondered how Facebook was able to do social smoothing? It's the DevOps philosophy that helps facebook and sure that apps aren't outdated and that users get the best experience on Facebook. Facebook accomplishes this true code ownership model that makes its developers responsible that includes testing and supporting through production and delivery for each kernel of code.
They write and update their true policies like this but Facebook has developed a DevOps culture and has successfully accelerated its development lifecycle.

Difference between Agile and DevOps:



DevOps Tools

1. Git (GitLab, GitHub, Bitbucket)

Git remains indispensable in software development and DevOps due to its pivotal role in version control, collaborative coding, and efficient project management. As technology has accelerated, the need for streamlined and organized code management has never been greater.

Git empowers developers to collaborate on codebases, effortlessly creating and merging branches for new features and bug fixes. Its distributed nature ensures developers can work seamlessly offline, an increasingly valuable feature in today's remote and distributed work environments.

Additionally, Git facilitates the tracking of code modifications, making it easier to identify when and why specific changes were made, a critical aspect of maintaining code quality and security. Software development is essential in driving innovation and advancing progress, and Git maintains its prominent position as the bedrock of efficient, cooperative, and secure coding methodologies.

2. Maven

Due to its enduring significance in managing project dependencies, building, and project lifecycle management, Maven remains a pivotal tool in SD and DevOps. As a robust build automation and project management tool, Maven simplifies the complexities of Java-based project development by streamlining the compilation, testing, packaging, and distribution processes. It ensures consistent and reproducible builds, making it easier for development teams to collaborate efficiently and deliver high-quality software.

Maven's role in managing dependencies and facilitating continuous integration and deployment remains crucial. Its ability to handle complex build scenarios and integrate seamlessly with modern DevOps practices makes it indispensable for ensuring software projects' reliability, maintainability, and scalability in 2024 and beyond.

3. Jenkins

Its importance lies in its role as a powerful automation server that enables continuous integration and continuous delivery (CI/CD) pipelines. Jenkins streamlines software development by automating tasks such as building, testing, and deploying code changes, ensuring that software is delivered quickly and highly. With the growing complexity of modern applications, the need for efficient CI/CD processes has become even more paramount.

Jenkins provides flexibility, extensibility, and a vast library of plugins that cater to a wide range of technologies and tools, making it adaptable to diverse development environments. As organizations prioritize speed, reliability, and collaboration in their software development practices, Jenkins stands as a cornerstone tool, enabling teams to achieve seamless automation and efficient delivery of software solutions.

4. Chef

Chef, a powerful automation platform, is crucial in managing infrastructure as code. Chef empowers organizations to achieve scalability, reliability, and speed seamlessly. By allowing the automation of server provisioning, configuration, and maintenance, Chef enhances efficiency and consistency across the entire infrastructure, reducing manual errors and ensuring that infrastructure remains desired.

Moreover, Chef integrates smoothly with various cloud providers, containerization technologies, and other DevOps tools, making it adaptable to the ever-evolving tech landscape. As organizations prioritize agility and scalability, Chef remains a vital tool

in automating complex infrastructure tasks and enabling DevOps teams to focus on innovation and delivery.

5. Puppet

Puppet is essential because it simplifies the management and orchestration of complex IT infrastructures by allowing administrators to define infrastructure as code. It ensures consistency and repeatability in configuration across servers, cloud instances, and containers. Businesses increasingly rely on diverse, dynamic, and hybrid infrastructures.

Puppet's importance lies in its ability to streamline provisioning, configuration, and continuous compliance, thus reducing operational complexity, minimizing errors, and accelerating software delivery. Puppet continues to empower organizations to efficiently manage and scale their infrastructure while maintaining high levels of security and compliance, making it a crucial tool for DevOps teams.

6. Ansible

Ansible is a powerful and widely adopted automation and configuration management tool important in 2024 for several reasons. This tool stands out for its simplicity and versatility. It empowers organizations to automate repetitive tasks, provisioning of infrastructure, and configuration management across diverse environments, making it an invaluable asset for DevOps and IT teams.

Furthermore, Ansible's agentless architecture, declarative language, and a vast library of pre-built modules make it accessible to both beginners and seasoned professionals. As organizations prioritize efficiency, scalability, and the rapid deployment of applications and services, Ansible remains an indispensable DevOps toolkit, helping teams streamline operations, enhance security, and maintain infrastructure at scale, all while reducing manual errors and increasing agility in a fast-paced technological landscape.

7. Docker

Docker is crucial in modern software development and DevOps practices. It can simplify and streamline the management of applications across various environments. Docker containers encapsulate an app and its dependencies, ensuring consistent and reproducible deployments from development to production.

This technology enhances portability and scalability, accelerates development cycles, and reduces the "it works on my machine" problem. In a rapidly evolving software landscape, Docker's containerization approach remains crucial for achieving efficient, isolated, and highly flexible application deployment, making it an essential component of DevOps and continuous delivery pipelines.

8. Kubernetes

Kubernetes, often abbreviated as K8s, play a central role in modern software development and operations. Its importance lies in its ability to orchestrate, manage, and automate containerized applications at scale. As organizations increasingly embrace microservices architectures and containerization for their applications, Kubernetes provides the essential infrastructure for deploying, scaling, and maintaining these containers efficiently.

The tool’s resilience, self-healing capabilities, and support for hybrid and multi-cloud environments make it vital for achieving agility, reliability, and cost-effectiveness in application deployment. It serves as the backbone of cloud-native ecosystems, enabling organizations to accelerate software delivery, improve resource utilization, and respond effectively to the evolving demands of the digital landscape.

9. Slack

Slack is a crucial tool for businesses and organizations worldwide. Its significance lies in facilitating seamless communication and collaboration among teams, whether working in the same office or remotely. Slack's real-time messaging, file sharing, and integration capabilities streamline workflow, enhance productivity and keep teams connected across different time zones and locations.

As the work landscape evolves, with more companies embracing hybrid and remote work models, Slack is a vital hub for quick decision-making, project coordination, and knowledge sharing. With an ever-expanding ecosystem of integrations and features, Slack remains at the forefront of modern workplace communication, making it essential for businesses to stay agile, efficient, and competitive.

10. AWS Cloud Computing and Storage in DevOps

AWS (Amazon Web Services) Cloud Computing and Storage are crucial in DevOps because they provide scalable, flexible, and cost-effective infrastructure for DevOps practices. AWS offers many services, including compute resources, databases, container orchestration, and serverless computing, which align perfectly with modern software development and deployment demands.

Organizations adopt DevOps to accelerate software delivery. AWS provides the foundation for rapidly deploying and scaling applications, supporting continuous integration and continuous delivery (CI/CD) pipelines, and automating infrastructure provisioning through tools like AWS CloudFormation.

Furthermore, AWS's storage solutions enable efficient data management, backup, and recovery, ensuring the resilience and reliability required for DevOps operations. As cloud technology evolves, AWS remains at the forefront, enabling DevOps teams to focus on innovation and efficiency.

11. Azure Cloud Computing and Storage in DevOps

Azure Cloud Computing and Storage will be pivotal in DevOps practices in 2024 and beyond. Azure offers a comprehensive cloud ecosystem that enables organizations to scale their infrastructure, deploy applications, and store data efficiently. Azure provides essential services for continuous integration and continuous deployment (CI/CD), automation, monitoring, and security. Its cloud computing capabilities facilitate the provisioning of resources on demand, ensuring that development and testing environments are readily available.

Azure's storage solutions, including Azure Blob Storage, Azure Files, and Azure SQL Database, enable secure data storage and retrieval, supporting the data-driven aspects of DevOps. Besides, Azure's integration with DevOps tools like Azure DevOps Services streamlines the software development lifecycle, enhancing collaboration and automation.

12. GCP Cloud Computing and Storage in DevOps

Google Cloud Platform (GCP) offers robust cloud computing and storage solutions. GCP provides a scalable, reliable, and highly available infrastructure essential for modern DevOps practices. With its comprehensive set of services, including Google Compute Engine, Google Kubernetes Engine, Cloud Storage, and BigQuery, GCP empowers DevOps teams to build, deploy, and manage applications easily. Its emphasis on automation, infrastructure as code, and container orchestration aligns seamlessly with DevOps principles.

Moreover, GCP's cutting-edge technologies, such as AI and machine learning capabilities, provide DevOps practitioners with advanced tools for monitoring, analytics, and automation, making it a powerful choice for organizations seeking to optimize their software development

and delivery processes.

13. Monitoring, Alerting, and Incident Response Tools: SignalFx

Monitoring, alerting, and incident response tools like SignalFx are pivotal in DevOps and software development. As software systems become complex and distributed, the need for real-time visibility into performance and the ability to respond swiftly to incidents is significant. SignalFx excels in this regard by providing advanced monitoring and observability solutions that enable organizations to detect anomalies, trace issues across microservices proactively, and set up intelligent alerts.

As applications scale, cloud-native architectures become the norm, and user expectations for reliability grow, SignalFx's capabilities are crucial. It empowers DevOps teams to ensure high availability, optimize resource utilization, and maintain a seamless user experience by identifying and addressing performance issues before they impact end-users. It is one of the most essential tools for modern software operations.

14. Appdynamics

AppDynamics, a leading application performance management and monitoring platform, remains critically important as it ensures the optimal performance of modern digital businesses. As organizations rely on complex and distributed software systems, proactively monitoring, troubleshooting, and optimizing these applications becomes essential.
AppDynamics provides real-time visibility into application performance, allowing businesses to swiftly identify bottlenecks, latency issues, and errors.

With the ever-growing complexity of applications, the importance of AppDynamics lies in its ability to empower organizations to deliver exceptional user experiences, maintain application reliability, and swiftly respond to performance issues, thereby ensuring the continued success and competitiveness of digital businesses.

15. Raygun

It is a crucial tool in software development and DevOps because it ensures application reliability and performance. Raygun is an application monitoring and error-tracking platform that empowers development teams to identify, diagnose, and resolve real-time issues.

With software systems growing in complexity and the increased demand for seamless user experiences, Raygun's importance lies in providing actionable insights into application errors and performance bottlenecks. It enables organizations to proactively address issues, reduce downtime, and enhance user satisfaction, leading to higher software quality and improved customer experiences.

Software is central to businesses across industries. Raygun's role in maintaining application health and facilitating rapid issue resolution makes it a fundamental tool for DevOps professionals and software developers.

16. Splunk Cloud

Splunk Cloud helps organizations gain critical insights from the ever-expanding volume of data generated in today's digital landscape. As businesses increasingly rely on data-driven decision-making, Splunk Cloud stands out as a robust and scalable platform for monitoring, searching, analyzing, and visualizing machine-generated data. Its importance lies in providing real-time visibility into the health and performance of complex systems, applications, and infrastructures, enabling rapid incident detection and response.

As cybersecurity threats evolve, Splunk Cloud's advanced security analytics and threat detection capabilities remain indispensable for safeguarding against cyberattacks and ensuring data integrity. In a world where data is a strategic asset, Splunk Cloud's role in

harnessing the power of data for operational excellence and security cannot be overstated.

17. Selenium

It remains a vital tool in software testing and automation due to its enduring relevance in ensuring the quality of web applications. As technology evolves, web applications become increasingly complex, requiring thorough testing across various browsers and platforms.

With its robust automation capabilities and extensive browser compatibility, Selenium allows developers and QA teams to automate repetitive testing tasks efficiently, conduct cross-browser testing, and ensure that web applications function flawlessly across diverse environments.

Its open-source nature, active community support, and integration with other DevOps tools make Selenium a go-to choice for organizations striving for continuous delivery and the rapid deployment of high-quality software, a cornerstone of modern software development practices.

18. Gremlin

Gremlin is an essential tool in chaos engineering, which has become increasingly critical for ensuring the resilience and reliability of modern software systems. As technology advances and complex distributed systems become the norm, the potential for unexpected failures and outages also rises.

Gremlin allows organizations to proactively identify weaknesses and vulnerabilities in their infrastructure and applications by simulating controlled failures, such as network disruptions, service outages, and resource constraints.

By intentionally inducing chaos and monitoring the system's response, teams can uncover weaknesses before they lead to costly downtime or security breaches. Gremlin facilitates organizations to build more robust, fault-tolerant systems that can withstand real-world challenges and deliver uninterrupted services to users.

19. ServiceNow

ServiceNow is a vital platform for organizations seeking to streamline their IT service management and beyond. Its significance lies in its ability to provide a unified, cloud-based solution for automating and optimizing various business processes, including ITSM, ITOM, HR, customer service, and more.

Due to the rapid digitization of services, remote work, and the growing complexity of technology infrastructures, ServiceNow offers a comprehensive approach to managing workflows, resolving issues, and delivering services efficiently. Its intelligent automation capabilities, analytics, and AI-driven insights empower organizations to enhance productivity, agility, and customer satisfaction while reducing operational costs.

ServiceNow's role in orchestrating and integrating diverse systems and processes makes it an indispensable tool for driving digital transformation and ensuring smooth operations in the ever-evolving business landscape of 2024.

20. Status Service Updates: The Status Page

"Status Service Updates: The Status Page" is a critical tool for organizations and businesses of all sizes. In today's world, where online services and applications are integral to operations, ensuring their availability and reliability is essential. It provides real-time information to users and stakeholders about the operational status of services, applications, and infrastructure. The Status Page plays a crucial role in transparency, trust-building, and customer satisfaction by promptly communicating service disruptions, planned maintenance, and incident resolutions.

Downtime can often lead to significant financial losses and damage to a company's reputation, so having a practical Status Page becomes not just a convenience but a necessity. It allows organizations to showcase their commitment to transparency and responsiveness in addressing service-related issues, ultimately fostering stronger customer relationships and trust.

21. ELK (Elasticsearch, Logstash and Kibana)

ELK, which stands for Elasticsearch, Logstash, and Kibana, continues to shine in DevOps and IT operations. This powerful trio of tools remains essential for organizations seeking effective log management, monitoring, and data visualization. Elasticsearch is a highly scalable and fast search engine that enables real-time data indexing and search.

Logstash facilitates the collection, processing, and transformation of log data from various sources, making it compatible with Elasticsearch. Kibana, on the other hand, provides a user-friendly interface for visualizing and analyzing data, offering customizable dashboards and powerful data exploration capabilities.

ELK's significance in 2024 lies in its ability to empower organizations with comprehensive insights into their systems, applications, and infrastructure. It ultimately facilitates quick problem resolution, proactive monitoring, and data-driven decision-making in an increasingly complex and fast-paced technological landscape.

22. GitLab CI/CD

GitLab CI/CD's significance lies in its ability to automate the complete software delivery pipeline, from code changes to deployment, in a single integrated environment. GitLab CI/CD ensures rapid and reliable delivery of software updates. It enables continuous integration (CI) by automatically building and testing code changes, allowing teams to catch issues early in the development cycle.

Furthermore, the continuous deployment (CD) aspect automates the release and deployment process, reducing the risk of human errors and enabling organizations to deliver features and updates to users swiftly and confidently. GitLab CI/CD's importance is further accentuated as businesses seek to accelerate digital transformation efforts, respond rapidly to changing market demands, and maintain a competitive edge through efficient and automated software delivery practices.

23. Scripting

Scripting remains vital due to its pivotal role in automating and streamlining various aspects of software development, system administration, and DevOps practices. Scripting languages like Python, Bash, and PowerShell empower tech professionals to write code that can execute repetitive tasks, manipulate data, and orchestrate complex processes efficiently.

Scripting facilitates rapid prototyping, configuration management, and the creation of automated deployment pipelines. It enhances productivity, ensures consistency and reduces human error in tasks ranging from software testing and deployment to infrastructure provisioning and monitoring. As organizations increasingly embrace DevOps and cloud-native technologies, scripting stays competitive and adaptive in the tech landscape.

24. Terraform

Terraform plays a crucial role in modern infrastructure provisioning and management. It allows organizations to define and deploy infrastructure as code, enabling the automated creation and configuration of cloud resources, containers, and other infrastructure components. Cloud computing, microservices, and containerization have become the norm in

2024. Terraform provides the agility and scalability required to keep up with the dynamic demands of modern applications.

Terraform's importance lies in its ability to bring consistency, version control, and automation to infrastructure operations, thereby reducing manual errors, streamlining DevOps workflows, and facilitating applications' rapid and reliable deployment in an increasingly complex and cloud-centric environment. As organizations adopt cloud-native technologies, Terraform remains essential to ensure efficient and consistent infrastructure management.

25. Phantom

Phantom enhances security automation and incident response capabilities. In today's rapidly evolving threat landscape, organizations face a constant barrage of cybersecurity incidents, and the ability to respond swiftly and effectively is necessary. It provides a platform for automating security workflows, from detecting and investigating potential threats to orchestrating responses and mitigating risks.

Phantom's importance lies in its capacity to reduce response times, increase consistency in incident handling, and free up manual resources from repetitive tasks. With the growing complexity of cyber threats, Phantom empowers security teams to defend against attacks and safeguard critical assets proactively.

26. Nagios

Nagios, an open-source monitoring and alerting system, remains vital due to its enduring significance in maintaining the reliability and performance of IT infrastructure and applications. Organizations increasingly rely on complex systems and services. Nagios plays a crucial role by providing real-time monitoring and alerting capabilities, allowing IT teams to detect and address issues before they impact users or cause system outages.

Its versatility, extensibility, and support for both on-premises and cloud environments make Nagios a valuable tool for ensuring critical systems' availability, stability, and security, aligning perfectly with the demands of modern IT operations and DevOps practices.

27. Vagrant

Vagrant continues to play a crucial role in software development and DevOps. It is a tool that simplifies creating and managing reproducible development environments. Its importance lies in its ability to provide developers and DevOps teams with a consistent and isolated environment for software development, testing, and deployment.

With the ever-evolving complexity of software stacks, dependencies, and infrastructure configurations, Vagrant remains essential in ensuring these environments are easily shareable, scalable, and maintainable. It allows developers to work seamlessly across various operating systems and provides a standardized setup that minimizes compatibility issues.

28. Sentry

Sentry plays a critical role in modern software development and DevOps practices. With software applications' increasing complexity and scale, identifying and addressing errors and issues has become crucial.

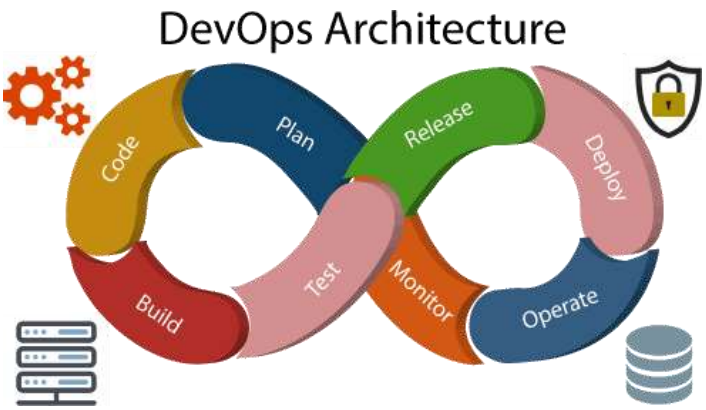
Sentry is vital because it provides real-time error tracking and monitoring, allowing development teams to proactively detect and diagnose issues, whether they occur in production or during development. Its importance is minimizing downtime, improving user experience, and maintaining software systems' overall health and reliability.

29. Gradle

Gradle continues to be a vital tool in software development and DevOps. Gradle is an advanced build automation system that plays a crucial role in managing dependencies, building projects, and orchestrating complex workflows efficiently. Its importance lies in its versatility and scalability, as it caters to various project sizes and types.

Gradle's ability to easily handle multi-language, multi-project builds and its support for plugin-based customization make it indispensable in modern software development. As organizations increasingly adopt microservices architectures and cloud-native technologies, Gradle's capabilities are instrumental in managing the complexity of building, testing, and deploying applications across diverse environments.

DevOps Architecture

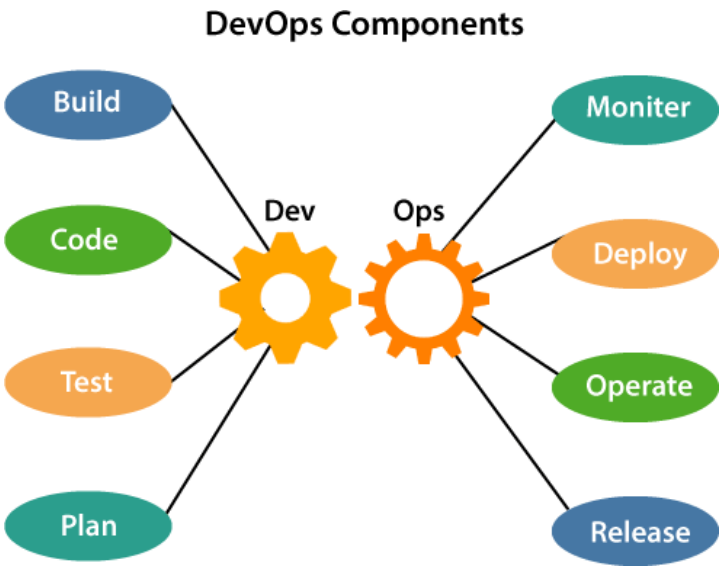


Development and operations both play essential roles in order to deliver applications. The deployment comprises analyzing the **requirements, designing, developing**, and **testing** of the software components or frameworks.

The operation consists of the administrative processes, services, and support for the software. When both the development and operations are combined with collaborating, then the DevOps architecture is the solution to fix the gap between deployment and operation terms; therefore, delivery can be faster.

DevOps architecture is used for the applications hosted on the cloud platform and large distributed applications. Agile Development is used in the DevOps architecture so that integration and delivery can be contiguous. When the development and operations team works separately from each other, then it is time-consuming to **design, test**, and **deploy**. And if the terms are not in sync with each other, then it may cause a delay in the delivery. So DevOps enables the teams to change their shortcomings and increases productivity.

Below are the various components that are used in the DevOps architecture:



1) Build: Without DevOps, the cost of the consumption of the resources was evaluated based on the pre-defined individual usage with fixed hardware allocation. And with DevOps, the usage of cloud, sharing of resources comes into the picture, and the build is dependent upon the user's need,

which is a mechanism to control the usage of resources or capacity.

2) Code: Many good practices such as Git enables the code to be used, which ensures writing the code for business, helps to track changes, getting notified about the reason behind the difference in the actual and the expected output, and if necessary reverting to the original code developed. The code can be appropriately arranged in **files, folders**, etc. And they can be reused.

3) Test: The application will be ready for production after testing. In the case of manual testing, it consumes more time in testing and moving the code to the output. The testing can be automated, which decreases the time for testing so that the time to deploy the code to production can be reduced as automating the running of the scripts will remove many manual steps.

4) Plan: DevOps use Agile methodology to plan the development. With the operations and development team in sync, it helps in organizing the work to plan accordingly to increase productivity.

5) Monitor: Continuous monitoring is used to identify any risk of failure. Also, it helps in tracking the system accurately so that the health of the application can be checked. The monitoring becomes more comfortable with services where the log data may get monitored through many third-party tools such as **Splunk**.

6) Deploy: Many systems can support the scheduler for automated deployment. The cloud management platform enables users to capture accurate insights and view the optimization scenario, analytics on trends by the deployment of dashboards.

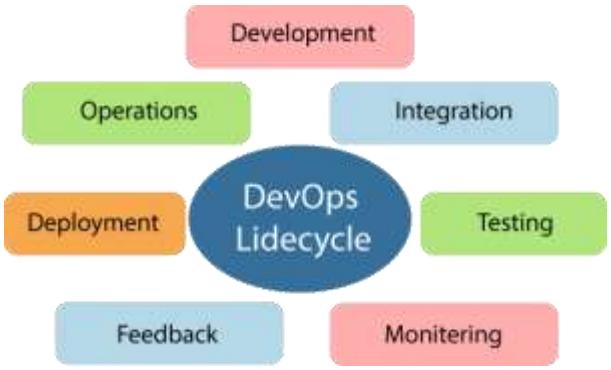
7) Operate: DevOps changes the way traditional approach of developing and testing separately. The teams operate in a collaborative way where both the teams actively participate throughout the service lifecycle. The operation team interacts with developers, and they come up with a monitoring plan which serves the IT and business requirements.

8) Release: Deployment to an environment can be done by automation. But when the deployment is made to the production environment, it is done by manual triggering. Many processes involved in release management commonly used to do the deployment in the production environment manually to lessen the impact on the customers.

DevOps Lifecycle

DevOps defines an agile relationship between operations and Development. It is a process that is practiced by the development team and operational engineers Learning DevOps is not complete without understanding the DevOps lifecycle phases. The DevOps lifecycle includes seven phases as given below:

together from beginning to the final stage of the product.



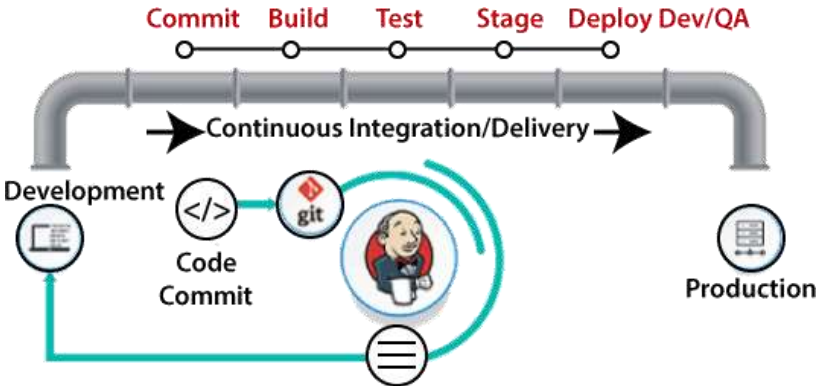
1) Continuous Development

This phase involves the planning and coding of the software. The vision of the project is decided during the planning phase. And the developers begin developing the code for the application. There are no DevOps tools that are required for planning, but there are several tools for maintaining the code.

2) Continuous Integration

This stage is the heart of the entire DevOps lifecycle. It is a software development practice in which the developers require to commit changes to the source code more frequently. This may be on a daily or weekly basis. Then every commit is built, and this allows early detection of problems if they are present. Building code is not only involved compilation, but it also includes **unit testing, integration testing, code review**, and **packaging**.

The code supporting new functionality is continuously integrated with the existing code. Therefore, there is continuous development of software. The updated code needs to be integrated continuously and smoothly with the systems to reflect changes to the end-users.



Jenkins is a popular tool used in this phase. Whenever there is a change in the Git repository, then Jenkins fetches the updated code and prepares a build of that code, which is an executable file in the form of war or jar. Then this build is forwarded to the test server or the production server.

3) Continuous Testing

This phase, where the developed software is continuously testing for bugs. For constant testing, automation testing tools such as **TestNG, JUnit, Selenium**, etc are used. These tools allow QAs to test multiple code-bases thoroughly in parallel to ensure that there is no flaw in the functionality. In this phase, **Docker** Containers can be used for simulating the test environment.

It may occur in the form of documentation files or maybe produce large-scale data about the application parameters when it is in a continuous use position. The system errors such as server not reachable, low memory, etc are resolved in this phase. It maintains the security and availability of the service.

Automation testing saves a lot of time and effort for executing the tests instead of doing this manually. Apart from that, report generation is a big plus. The task of evaluating the test cases

that failed in a test suite gets simpler. Also, we can schedule the execution of the test cases at predefined times. After testing, the code is continuously integrated with the existing code.

4) Continuous Monitoring

Monitoring is a phase that involves all the operational factors of the entire DevOps process, where important information about the use of the software is recorded and carefully processed to find out trends and identify problem areas. Usually, the monitoring is integrated within the operational capabilities of the software application.

5) Continuous Feedback

The application development is consistently improved by analyzing the results from the operations of the software. This is carried out by placing the critical phase of constant feedback between the operations and the development of the next version of the current software application.

The continuity is the essential factor in the DevOps as it removes the unnecessary steps which are required to take a software application from development, using it to find out its issues and then producing a better version. It kills the efficiency that may be possible with the app and reduce the number of interested customers.

6) Continuous Deployment

In this phase, the code is deployed to the production servers. Also, it is essential to ensure that the code is correctly used on all the servers. The new code is deployed continuously, and configuration management tools play an essential role in executing tasks frequently and quickly. Here are some popular tools which are used in this phase, such as **Chef, Puppet, Ansible**, and **SaltStack**.

Containerization tools are also playing an essential role in the deployment phase. **Vagrant** and **Docker** are popular tools that are used for this purpose. These tools help to produce consistency across development, staging, testing, and production environment. They also help in scaling up and scaling down instances softly.

Containerization tools help to maintain consistency across the environments where the application is tested, developed, and deployed. There is no chance of errors or failure in the production environment as they package and replicate the same dependencies and packages used in the testing, development, and staging environment. It makes the application easy to run on different computers.

7) Continuous Operations

All DevOps operations are based on the continuity with complete automation of the release process and allow the organization to accelerate the overall time to market continuingly. It is clear from the discussion that continuity is the critical factor in the DevOps in removing steps that often distract the development, take it longer to detect issues and produce a better version of the product after several months. With DevOps, we can make any software product more efficient and increase the overall count of interested customers in your product.

AWS

AWS stands for Amazon Web Services, It is an expanded cloud computing platform provided by Amazon Company. AWS provides a wide range of services with a pay-as-per-use pricing model over the Internet such as Storage, Computing power, Databases, Machine Learning services, and much more. AWS facilitates for both businesses and individual users with effectively hosting the applications, storing the data securely, and making use of a wide variety of tools and services improving management flexibility for IT resources.

Advantages & Features of AWS:

1. **Cost savings:** One of the biggest benefits of AWS is that it can help businesses save money. As mentioned previously, **businesses can avoid the high upfront costs of traditional infrastructure with AWS** and pay only for the resources they use. Traditionally, businesses had to invest in hardware and software upfront, which often led to overspending.

Let's look at this for example – if a business needs to run a website that gets 1000 visitors per day, they would need to purchase and maintain enough servers to support this traffic. With AWS, the business only pays for the compute resources they use when someone visits their website. This can result in significant cost savings.

2. **Flexibility:** Another key benefit of AWS is its flexibility. Businesses are able to customize their virtual environment – whether the operating system, database, programming language, or something else – to meet their specific needs. Especially in today's climate, the **migration process to the cloud** should be as frictionless as possible – and AWS makes that possible. Regardless of your use case or industry, **AWS can be tailored to fit your needs**, whether you're looking for a single cloud-hosted application or an entire suite of integrated solutions.

3. **Reliability:** AWS is known for being reliable, with an **uptime of 99.9%**. This makes it a **great platform for mission-critical applications that need to be available 24/7**. AWS also offers the ability to deploy resources across multiple availability zones for even greater reliability. The cloud platform also has a number of features that make it easier to ensure reliability, such as autoscaling and auto-healing. Autoscaling allows businesses to automatically scale their resources up or down based on demand, while auto-healing enables them to quickly identify and replace any faulty components.

4. **Security:** Businesses can take advantage of advanced security features, such as identity and access management, to help protect their data. Their tough infrastructure with an end-to-end approach is designed to withstand attacks and AWS provides customers with tools to help them monitor and respond to threats. When it comes to storage, Amazon S3 provides customers with a secure and reliable way to store and access data. The service is designed to be highly scalable and resilient, with built-in redundancy. Fine-grain identity and access controls can be applied to S3 buckets and objects, giving customers control over who has access to their data. **Security tasks can be automated with AWS CloudFormation**, making it easier for businesses to manage their security policies. And, you can rest easy knowing that AWS takes privacy seriously, with comprehensive customer data protection and compliance measures.

5. **Compliance:** By compliance, we mean that certain businesses are required to follow specific regulations. Financial services companies in the United States, for example, must comply with the **Sarbanes-Oxley Act**, while healthcare, education, and energy companies must comply with HIPAA and other regulations. AWS provides a number of compliance-related features and services, such as data encryption and identity and access management, to help businesses meet these requirements.

6. **High-Performance:** Interested in delivering your applications quickly and efficiently? Taking advantage of AWS features such as **auto-scaling and load balancing** will help ensure your applications are always available and running optimally. AWS can help businesses improve their performance by offering a variety of cloud-based services, including **Amazon Elastic Compute Cloud (EC2)**, which provides high-performance computing resources, and **Amazon CloudFront**, which delivers content quickly and securely to users around the world. Others include machine learning (ML) and analytics services, such as Amazon SageMaker and Amazon Athena. These services provide the tools businesses need to quickly and easily analyze their data for insights. Fast networking in the cloud is also possible with AWS, thanks to its Elastic Load Balancing (ELB) and Amazon Virtual Private Cloud (VPC). With ELB, businesses can balance their workloads across multiple instances for increased performance, while VPC allows businesses to create isolated private networks in the cloud.

7. **Developer Tools:** Developer tools are designed to make it easier for developers to create, deploy, and manage applications – and AWS provides developers with what they need to build applications quickly and easily. By leveraging developer tools, **developers can save time and money by automating tedious tasks**. They also benefit from access to **AWS's extensive library of pre-built applications** that can help them get their projects off the ground quickly. Services such as Amazon Elastic Beanstalk and Amazon CloudFormation can help them automate the process of creating and deploying applications. Other ways developers can improve productivity with AWS include using AWS CodeCommit to store and manage source code.
8. **Integration:** Thanks to its many integrations with other Amazon services, as well as third-party services, AWS makes it easy for businesses to get started with cloud computing. AWS provides a wide range of services that can be easily integrated into existing business infrastructure. This allows businesses to add new features and capabilities without having to make major changes or invest in new hardware or software. For instance, if a business wants to add mobile capabilities to its website, it **can take advantage of Amazon's Mobile SDK and Web Services**. These tools allow businesses to quickly develop and deploy mobile apps that connect directly with their existing infrastructure.
9. **Management Console**: The AWS management console is a web-based interface that provides users with a simple way to interact with and manage their AWS resources – essentially a place where you can access and manage everything on the cloud. It provides a graphical view of all the resources associated with an account, as well as tools for creating and configuring new resources. Compared to traditional command-line interfaces, the **AWS management console saves time and makes it easier for users to get the most out of their AWS services**. Not only that, but *your business gets access to 350+ free digital training courses through the **AWS Academy***, covering topics such as cloud fundamentals, DevOps, security, and big data.
   This means you can train your employees on how to use AWS, and in turn, help them become more efficient at their jobs.

10. **Scalability:** With an on-demand service, businesses can quickly spin up new servers as needed with just a few clicks. This makes it much easier to scale resources up or down as demand changes, allowing businesses to save costs and maintain performance even during peak periods. For example, if a business is expecting a sudden surge in traffic due to an advertising campaign or seasonal event, they can easily add more capacity to their server infrastructure to handle the increased load. **Bru Textiles**, a specialty textile company in Belgium, was able to quickly scale its infrastructure by leveraging AWS. Bru Textiles went digital to grow and offer new services. Embracing technology, they brought in digital twin technology to give their customers an idea of the texture and essence of their physical fabrics.

AWS Applications

- **Storage and Backup:** Storage and backup are important for any Cloud Computing service. AWS provides you with reliable storage services like Amazon Simple Storage Service to store large-scale data and backup services like AWS Backup to take backups of this data, which is stored in other AWS services. AWS stores the data in three different availability zones so that if one fails, you can still access your data. This makes AWS storage reliable and easily accessible. Therefore, companies with huge application data to store and backup securely can use AWS.
- **Big Data:** One of the biggest challenges faced by companies these days is Big Data. The companies are struggling to store their large amounts of data using traditional methods. With AWS Big Data storage services, they can manage to store their data even if the data limit increases unexpectedly as AWS provides virtually unlimited data storage with scale-in and scale-out options. AWS offers easy access and faster data retrieval as well. For data processing, it offers services like EMR, with which the companies can easily set up, operate, and scale their big data. Therefore, efficiently storing and managing Big Data is among the top AWS applications.
- **Enterprise IT:** AWS is a one-stop solution for any IT business. Many features of it such as secure storage, scalability, flexibility, and elasticity support companies to innovate faster than ever before. Using AWS for IT enterprises makes them profitable in terms of both money and time. As AWS maintains its cloud architecture, it need not waste time and money on professionals to do the same.
- **Social Networking:** Social networking is essential for businesses in the present-day scenario where Digital Marketing is key, and it is easier with AWS. Companies can connect with customers and stakeholders and communicate through social networking sites and develop their business. Services like AWS social networking engine, which is powered by TurnKey GNU/Linux (HVM) AMI stack, are used for performance and scalability to help companies build a suitable social networking site and gain profits.
- **Mobile Apps:** Mobile applications are embedded with day-to-day life. With AWS, you have the facility to create an app in your desired programming language. You can also keep up the applications that are consistently accessible and solid with high compute, storage, database, and application services. You can take advantage of AWS auto-scaling and managed relational database service for the better performance of your apps.
- **Websites:** AWS offers a wide range of website hosting options to create the best website for customers. Its services like Amazon Lightsail have everything, such as a virtual machine, SSD-based storage, data transfer, DNS management, and a static IP, to launch a website in such a way that the user can manage the website easily. Amazon EC2, AWS Lambda, Elastic Load Balancing, AWS Amplify, Amazon S3, etc. also help users build reliable and scalable websites.

- **Gaming:** AWS has been serving many gaming studios. Combining Amazon EC2 and S3 services with CloudFront enables gaming websites to deliver high-quality gaming experiences to their customers regardless of location.

Use Cases of AWS

- **Netflix**

Netflix is an entertainment platform that started in the United States, but eventually, it expanded to many countries and soon became popular. However, once Netflix confronted the scalability problem because of the sudden increase in viewers. That made Netflix choose AWS services. Netflix reports that when it started using AWS services like DynamoDB and Cassandra for its distributed databases, it could handle the data easily. So, scalability is a great advantage of AWS. Netflix has adapted around 100,000 server instances from AWS for computing and storage databases, analytics, recommendation engines, and video transcoding as well.

- **McDonald's**

McDonald's is the world's largest fast-food company that serves around 64 million people per day. The growth of this company has gone to another level when it started home deliveries. By utilizing AWS services, McDonald's created a platform that integrates local restaurants with delivery partners such as Uber Eats. Scalability is also a reason for the company to choose AWS services. Moreover, with AWS Microservices Architecture, McDonald's platform can scale 20,000 orders per second and integrate with the global partners easily.

- **Airbnb**

Airbnb is an international online marketplace for rental homes. This platform connects people who are looking for rental accommodation with those who want to rent out their houses. Quite soon, Airbnb became unable to handle the constant streaming of data on the website from its customers. That is when it started using Amazon EC2 service and Elastic Load Balancing, which distributes incoming traffic to multiple Amazon EC2 instances. In this way, Airbnb could avoid traffic, and customers could use the online platform without any disruption.

- **Novartis**

Novartis is the best example for AWS use cases in healthcare. Novartis is one of the world's largest healthcare companies that provides solutions for patients' well-being. It adapted Amazon EC2 services and built a platform using other services such as Amazon Simple Storage Service, Amazon Elastic Block Store, and four availability zones. Data Analysts of Novartis are taking advantage of the AWS services and still implementing new solutions for the patients.

- **Expedia**

Expedia is a worldwide online travel agency that has always focused on the constant development and innovation of its platform to offer an extraordinary user experience for its clients. Since 2010, Expedia has been using AWS services to build a standard deployment model for better infrastructure as AWS offers the best data security through different availability zones.

- **Samsung**

If you are using Samsung mobile phones, then you may know about the Samsung app store. For setting up the apps stacked in its store, the company started using AWS services. Using AWS app development services, Samsung wanted to provide its customers with the facility to download the apps anywhere without any network traffic.

- **NASA**

NASA (National Aeronautics and Space Administration) has always wondered about creating a library to present people with all its achievements through pictures and videos of space. Later on, it created such platforms, but because it had 10 different NASA centers, it couldn't provide the best experience for viewers. So, all it wanted was to create an easy-access platform for

people to search for and view images and videos. Then, NASA started adopting many services from AWS to solve this problem, which included Amazon Elastic Compute Cloud, Elastic Load Balancing, Amazon Simple Storage Service, Amazon Simple Queue Service, etc. Among these, Amazon S3 helped the company store all the incoming data such as photos, videos, and audio files without any hassle.

- **Facebook**

Facebook, without a doubt, is a widespread social media platform. To build a scalable application, Facebook used services such as Amazon Elastic Compute Cloud, Amazon Simple Storage Service, Amazon Relational Database Service, Amazon SimpleDB, Amazon CloudFront, Amazon Simple Queue Service, etc. Amazon RDS helps the platform to make it easy to set up, operate, and scale the database in the cloud.

Various Services offered by AWS

- Amazon EC2 (Elastic Cloud computing)
- Amazon RDS (Relational Database Services)
- Bonus Service: Amazon Connect
- Amazon S3 (Simple Storage Service)
- Amazon Lambda
- Amazon Cognito
- Amazon Glacier
- Amazon SNS (Simple Notification Service)
- Bonus Service: Amazon Lex
- Amazon Lightsail
- Amazon VPC (Virtual Private Cloud)
- Amazon Kinesis
- Amazon Inspector
- Amazon Auto-scaling
- Amazon IAM (Identity and Access Management)
- Dynamo DB
- Amazon SQS (Simple Queue Service)
- Amazon ElastiCache
- Amazon Chime
- AWS Athena
- CodeCatalyst
- Web Application Firewall
- AWS Amplify
- AWS Rekognition
- AWS QuickSight
- AWS Cloudformation
- AWS Management Console

The Important Cloud Services according to various categories that are provided by AWS are given below :

1. Compute

- **Amazon EC2:** Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It allows organizations to obtain and configure virtual compute capacity in the cloud. You can select from a variety of operating systems and resource configurations like memory, CPU, and storage that are required for your

application. Amazon EC2 enables you to increase or decrease capacity within minutes. You can use one or hundreds or even thousands of server instances simultaneously. Because this is all controlled with web service APIs, your application can automatically scale itself up and down depending on its needs. Amazon EC2 is integrated with most AWS services, such as Amazon Simple Storage Service (Amazon S3), Amazon Relational Database Service (Amazon RDS), and Amazon Virtual Private Cloud (Amazon VPC) to provide a complete, secure solution for computing applications. Amazon EC2 is an example of Infrastructure as a Service(IaaS). EC2 delivers secure, reliable, cost-effective compute and high-performance compute infrastructure so as to meet the needs of demanding businesses. Amazon EC2 is one of the easiest ways of providing servers on AWS Cloud and also the access to Operating system.

- **AWS Lambda:** AWS Lambda is a serverless, event-driven compute service that allows you to run code without managing servers. You pay only for the compute time you consume and there is no charge when your code is not running. With AWS Lambda, you can run code for any type of application with zero administration. Just upload your code, and Lambda takes care of everything required to run and scale your code with high availability. You can set up your code to automatically trigger from other AWS services, or you can call it directly from any web or mobile app. But triggering Lambda is possible with over 200 AWS services. You can only pay for what you have used. The compute time that you consume, you are needed to pay for it. You just only need to upload your code and everything required to run will take care of by Lambda and it automatically scales your code with high availability.
- **AWS Elastic Beanstalk:** AWS Elastic Beanstalk is a Platform as a Service that facilitates quick deployment of your applications by providing all the application services that you need for your application. Beanstalk is a plug-and-play platform that allows working with multiple programming languages and environments. Elastic Beanstalk supports a large range of platforms like Node js, Java, PHP, Python, and Ruby. So, you can develop your application to meet your requirements and simply deploy it on Elastic Beanstalk. The main aim to use AWS Elastic Beanstalk is to allow you to focus on the deployment and management of your applications. You can simply upload your code, and AWS Elastic Beanstalk automatically handles the deployment, from capacity provisioning, load balancing, and auto-scaling to application health monitoring. At the same time, you retain full control over the AWS resources powering your application and can access the underlying resources at any time.

2. Networking

- **Amazon VPC:** Amazon VPC is your network environment in the cloud. It allows you to create a private network within the AWS cloud that uses many of the same concepts and constructs as an on-premises network. Amazon VPC also gives you complete control of the network configuration. Customers can define normal networking configuration items such as IP address ranges, subnet creation, route table creation, network gateways, and security settings. Amazon VPC is an AWS foundational service and integrates with numerous AWS services. For instance, Amazon EC2 instances are deployed into your

Amazon VPC. Similarly, Amazon Relational Database Service (Amazon RDS) database instances deploy into your Amazon VPC, where the database is protected by the structure of the network just like your on-premises network. You can easily launch AWS resources into a virtual network by Amazon Virtual Private Cloud. An isolated virtual network environment in the AWS cloud is created by Amazon VPC.

- **Amazon Route 53:** Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. It is designed to give developers and businesses an extremely reliable and cost-effective way to route end users to Internet applications by translating human-readable names, such as www.geeksforgeeks.com, into the numeric IP addresses that computers use to connect to each other. Amazon Route 53 is fully compliant with IPv6 as well.

3. Storage

- **Amazon S3 (Simple Storage Service):** Amazon Simple Storage Service (Amazon S3) is object storage with a simple web service interface to store and retrieve any amount of data from anywhere on the web. It is designed to provide an infinite amount of storage and it is delivered with 99.999999999% durability. You can use Amazon S3 as primary storage for cloud-native applications as a target for backup and recovery and disaster recovery. It offers industry-leading scalability, data availability, security, and performance. It's simple to move large volumes of data into or out of Amazon S3 with Amazon's cloud data migration options. Once data is stored in Amazon S3, it can be automatically tiered into lower cost, longer-term cloud storage classes like Amazon S3 Standard – Infrequent Access and Amazon Glacier for archiving.
- **Amazon Glacier:** Amazon Glacier is a secure, durable, and extremely low-cost storage service for data archiving and long-term backup. Data stored in Amazon Glacier takes several hours to retrieve, which is why it's ideal for archiving. The fastest access to your archive data is via Amazon Glacier.

4. Databases

- **Amazon RDS (Relational Database Service):** Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while managing time-consuming database administration tasks, freeing you up to focus on your applications and business. You can find Amazon RDS is also available on several database instance types – optimized for memory, performance, or I/O. Amazon RDS provides you with six familiar database engines to choose from, including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle, and Microsoft SQL Server.
- **Amazon DynamoDB (Non-Relational Database):** Amazon DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale. It is a fully managed database and supports both document and key-value data models. When you create a database table that can store and retrieve any amount of data you can simply use Amazon DynamoDB that will serve any level of requested traffic. Its flexible data model and reliable performance make it a great fit for mobile, web, gaming, Internet of Things (IoT), and many other applications. DynamoDB provides many features like
    - built-in security

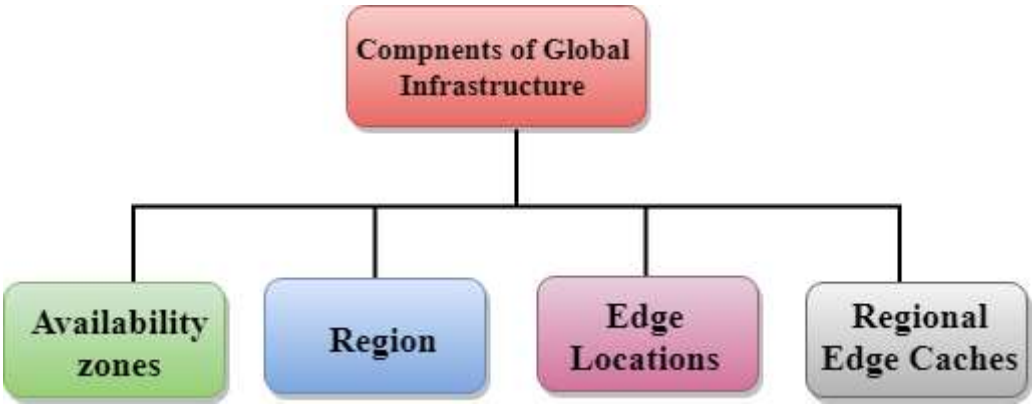
- backups
- automated multi-region replication
- in-memory caching
- data export tools.

Global Infrastructure of AWS

- AWS is a cloud computing platform which is globally available.
- Global infrastructure is a region around the world in which AWS is based. Global infrastructure is a bunch of high-level IT services which is shown below:
- AWS is available in 19 regions, and 57 availability zones in December 2018 and 5 more regions 15 more availability zones for 2019.

The following are the components that make up the AWS infrastructure:

- Availability Zones
- Region
- Edge locations
- Regional Edge Caches



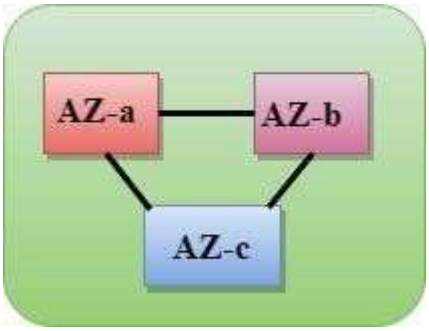
Availability zone as a Data Center

- An availability zone is a facility that can be somewhere in a country or in a city. Inside this facility, i.e., Data Centre, we can have multiple servers, switches, load balancing, firewalls. The things which interact with the cloud sits inside the data centers.
- An availability zone can be a several data centers, but if they are close together, they are counted as 1 availability zone.

Region

- A region is a geographical area. Each region consists of 2 more availability zones.

- A region is a collection of data centers which are completely isolated from other regions.
- A region consists of more than two availability zones connected to each other through links.



- Availability zones are connected through redundant and isolated metro fibers.

Edge Locations

- Edge locations are the endpoints for AWS used for caching content.
- Edge locations consist of CloudFront, Amazon's Content Delivery Network (CDN).
- Edge locations are more than regions. Currently, there are over 150 edge locations.
- Edge location is not a region but a small location that AWS have. It is used for caching the content.
- Edge locations are mainly located in most of the major cities to distribute the content to end users with reduced latency.
- For example, some user accesses your website from Singapore; then this request would be redirected to the edge location closest to Singapore where cached data can be read.

Regional Edge Cache

- AWS announced a new type of edge location in November 2016, known as a Regional Edge Cache.
- Regional Edge cache lies between CloudFront Origin servers and the edge locations.
- A regional edge cache has a large cache than an individual edge location.
- Data is removed from the cache at the edge location while the data is retained at the Regional Edge Caches.
- When the user requests the data, then data is no longer available at the edge location. Therefore, the edge location retrieves the cached data from the Regional edge cache instead of the Origin servers that have high latency.

What Is Cloud Computing?

Cloud computing is the use of hardware and software components in an off-premises location to deliver a service to a network. Users can access files and applications from any device that can access the internet.

Some features and capabilities include:

- Cloud providers can pull the computing resources to provide services to multiple customers with the help of a multi-tenant model
- Cloud computing proves an on-demand self-service, which helps administrators monitor performance
- Servers are maintained easily and there is nearly zero downtime
- Users can access cloud data and upload it on the cloud from any device with a solid internet connection
- Cloud environments can be modified according to the user's requirements and is easily accessible
- Clouds are highly secure, making data breaches more unlikely
- Migrating to the cloud eliminates the need to buy on-premises infrastructure
- It offers pay-as-you-go pricing, meaning you only pay for the resources you use

1. **Infrastructure as a Service:** IaaS delivers virtualized computing resources over the Internet. Users can rent virtual machines, storage, and networking infrastructure, allowing for easy scalability without investing in physical hardware. Examples include AWS EC2 and Azure **Virtual Machines**.
2. **Platform as a Service:** PaaS offers a robust platform for developers to build, deploy, and manage apps without worrying about the underlying infrastructure. It simplifies application development and deployment, with services like Google App Engine and Heroku leading the way.
3. **Software as a Service:** SaaS offers software applications on a subscription basis, accessible via a web browser. Users don't need to install or maintain software locally, making it ideal for collaboration tools (e.g., **Microsoft** 365, Google Workspace) and CRM systems (e.g., Salesforce).
4. **Function as a Service:** FaaS allows developers to execute code responding to events without managing servers. It's highly scalable and cost-efficient, exemplified by **AWS Lambda** and Azure Functions. FaaS is also known as serverless computing.
5. **Container as a Service:** CaaS enables the deployment and management of containerized applications using orchestration tools like Kubernetes. It provides portability and scalability for applications across different cloud environments.

Difference between main cloud computing services

Terms	IAAS	PAAS	SAAS
Stands for	Infrastructure as a service.	Platform as a service.	Software as a service.
Uses	IAAS is used by network architects.	PAAS is used by developers.	SAAS is used by the end user.
Access	IAAS gives access to the resources like virtual machines and virtual storage.	PAAS gives access to run time environment to deployment and development tools for application.	SAAS gives access to the end user.
Model	It is a service model that provides virtualized computing resources over the internet.	It is a cloud computing model that delivers tools that are used for the development of applications.	It is a service model in cloud computing that hosts software to make it available to clients.
Technical understanding.	It requires technical knowledge.	Some knowledge is required for the basic setup.	There is no requirement about technicalities company handles everything.
Popularity	It is popular among developers and researchers.	It is popular among developers who focus on the development of apps and scripts.	It is popular among consumers and companies, such as file sharing, email, and networking.
Percentage rise	It has around a 12% increment.	It has around 32% increment.	It has about a 27 % rise in the cloud computing model.
Usage	Used by the skilled developer to develop unique applications.	Used by mid-level developers to build applications.	Used among the users of entertainment.
Cloud services.	Amazon Web Services, sun, vCloud Express.	Facebook, and Google search engine.	MS Office web, Facebook and Google Apps.
Enterprise services.	AWS virtual private cloud.	Microsoft Azure.	IBM cloud analysis.
Outsourced cloud services.	Salesforce	Force.com, Gigaspaces.	AWS, Terremark
User Controls	Operating System, Runtime, Middleware, and Application data	Data of the application	Nothing

Google cloud services

Google offers a seven wide range of Services:

- **Compute**
- **Networking**
- **Storage and Databases**
- **Big Data**
- **Machine Learning**
- **Identity & Security**
- **Management and Developer Tools**

1. **Compute:** GCP provides a scalable range of computing options you can tailor to match your needs. It provides highly customizable virtual machines. and the option to deploy your code directly or via containers.

    - Google Compute Engine
    - Google App Engine
    - Google Kubernetes Engine
    - Google Cloud Container Registry
    - Cloud Functions

2. **Networking:** The Storage domain includes services related to **networking**, it includes the following services

    - Google Virtual Private Cloud (VPC)
    - Google Cloud Load Balancing
    - Content Delivery Network
    - What is Google Cloud Connect
    - Google Cloud DNS
    - What is Google Cloud Web Hosting

3. **Storage and Databases:** The Storage domain includes services related to data **storage**, it includes the following services

    - Google Cloud Storage
    - Cloud SQL
    - Cloud Bigtable
    - Google Cloud Datastore
    - Persistent Disk

4. **Big Data:** The Storage domain includes services related to **big data**, it includes the following services

    - Google BigQuery
    - Google Cloud Dataproc
    - Google Cloud Datalab
    - Google Cloud Pub/Sub

5. **Cloud AI:** The Storage domain includes services related to **machine learning,** it includes the following services

    - Cloud Machine Learning
    - Vision API
    - Speech API

- Natural Language API
- Translation API
- Jobs API

6. **Identity & Security:** The Storage domain includes services related to **security,** it includes the following services

    - Cloud Resource Manager
    - Cloud IAM
    - Cloud Security Scanner
    - Cloud Platform Security

7. **Management Tools:** The Storage domain includes services related to **monitoring and management**, it includes the following services

    - Stackdriver
    - Monitoring
    - Logging
    - Error Reporting
    - Trace
    - Cloud Console

8. **Developer Tools:** The Storage domain includes services related to **development**, it includes the following services
    - Cloud SDK
    - Deployment Manager
    - Cloud Source Repositories
    - Cloud Test Lab

AZURE

Azure is Microsoft's cloud platform, just like Google has its Google Cloud and Amazon has its Amazon Web Service or AWS.000. Generally, it is a platform through which we can use Microsoft's resources. For example, to set up a huge server, we will require huge investment, effort, physical space, and so on. In such situations, Microsoft Azure comes to our rescue. It will provide us with virtual machines, fast processing of data, analytical and monitoring tools, and so on to make our work simpler. The pricing of Azure is also simpler and cost-effective. Popularly termed as *"Pay As You Go"*, which means how much you use, pay only for that.

Microsoft Azure Used for

- **Deployment Of applications:** You can develop and deploy the application in the azure cloud by using the service called Azure App Service and Azure Functions after deploying the applications end users can access it.
- **Identity and Access Managment:** The application and data which is deployed and stored in the Microsoft Azure can be secured with the help of Identity and Access Managment. It's commonly used for single sign-on, multi-factor authentication, and identity governance.
- **Data Storage and Databases:** You can store the data in Microsoft azure in service like blob storage for unstructured data, table storage for NoSQL

data, file storage, and Azure SQL Database for relational databases. The service can be scaled depending on the amount of data we are getting.

- **DevOps and Continuous Integration/Continuous Deployment (CI/CD):** Azure DevOps will provide some tools like ncluding version control, build automation, release management, and application monitoring

Following are some of the services Microsoft Azure offers:

1. **Compute:** Includes Virtual Machines, Virtual Machine Scale Sets, Functions for serverless computing, Batch for containerized batch workloads, Service Fabric for microservices and container orchestration, and Cloud Services for building cloud-based apps and APIs.
2. **Networking:** With Azure, you can use a variety of networking tools, like the Virtual Network, which can connect to on-premise data centers; Load Balancer; Application Gateway; VPN Gateway; Azure DNS for domain hosting, Content Delivery Network, Traffic Manager, ExpressRoute dedicated private network fiber connections; and Network Watcher monitoring and diagnostics
3. **Storage:** Includes Blob, Queue, File, and Disk Storage, as well as a Data Lake Store, Backup, and Site Recovery, among others.
4. **Web + Mobile:** Creating Web + Mobile applications is very easy as it includes several services for building and deploying applications.
5. **Containers:** Azure has a property that includes Container Service, which supports Kubernetes, DC/OS or Docker Swarm, and Container Registry, as well as tools for microservices.
6. **Databases:** Azure also included several SQL-based databases and related tools.
7. **Data + Analytics:** Azure has some big data tools like HDInsight for Hadoop Spark, R Server, HBase, and Storm clusters
8. **AI + Cognitive Services:** With Azure developing applications with artificial intelligence capabilities, like the Computer Vision API, Face API, Bing Web Search, Video Indexer, and Language Understanding Intelligent.
9. **Internet of Things:** Includes IoT Hub and IoT Edge services that can be combined with a variety of machine learning, analytics, and communications services.
10. **Security + Identity:** Includes Security Center, Azure Active Directory, Key Vault, and Multi-Factor Authentication Services.
11. **Developer Tools:** Includes cloud development services like Visual Studio Team Services, Azure DevTest Labs, HockeyApp mobile app deployment and monitoring, Xamarin cross-platform mobile development, and more.

Difference between AWS (Amazon Web Services), Google Cloud, and Azure

	AWS	Google Cloud	Azure
Technology	EC2 (Elastic Compute Cloud)	Google Compute Engine(GCE)	VHD (Virtual Hard Disk)
Databases Supported	AWS fully supports relational and NoSQL databases and Big Data.	Technologies pioneered by Google, like Big Query, Big Table, and Hadoop, are databases, and Big Data,naturally fully supported.	Azure supports both relational and NoSQL through Windows AzureTable and HDInsight.
Pricing	Per hour — rounded up.	Per minute — rounded up	Per minute — rounded up.
Models	On demand, reserved spot.	On demand — sustained use.	Per minute- rounded up commitments(Pre-paid or monthly)
Difficulties	Many enterprises find it difficult to understand the company cost structure.	Fewer features and services.	Less "Enterprise-ready.
Storage Services	Simple Storage Service(S3) Elastic Block Storage. Elastic File storage.	Blob Storage Queue Storage. File Storage Disk Storage. Data Lake Store	Cloud storage. Persistent Disk Transfer appliance.
Machine Learing	Sage maker. Lex. polly.And many more	Machine learning Azure Bot service Cognitive service	Cloud speech AI Cloud Video Intelligence. Cloud Machine learning engine

GIT

Git is a distributed version control system (DVCS) that helps manage and track changes in source code during software development. It was created by Linus Torvalds in 2005 and has become one of the most widely used version control systems in the software development industry.

Some key concepts and features of Git:

- **Version Control:** Git allows developers to keep track of changes made to their code over time. This includes modifications, additions, and deletions of files.
- **Distributed System:** Git is a distributed version control system, meaning that each developer has a complete copy of the entire repository, including its full history. This allows developers to work independently and merge their changes when necessary.
- **Branching:** Git enables developers to create branches, which are essentially separate lines of development. This allows for the parallel development of features or bug fixes without affecting the main codebase.

- **Merging:** Git provides tools for merging changes from one branch into another. This is essential when multiple developers are working on different branches and need to bring their changes together.
- **History Tracking:** Git maintains a complete history of changes made to the codebase. Developers can view, revert, or analyze changes made over time.
- **Remote Repositories:** Git supports remote repositories, allowing developers to collaborate with others by pushing and pulling changes to and from a shared repository. Platforms like GitHub, GitLab, and Bitbucket provide hosting services for Git repositories.
- **Staging Area:** Git uses a staging area (also known as the index) to prepare and review changes before committing them to the repository. This allows developers to selectively include or exclude specific changes.
- **Open Source:** Git is an open-source tool, and its source code is freely available for modification and distribution.

Various GIT Components:

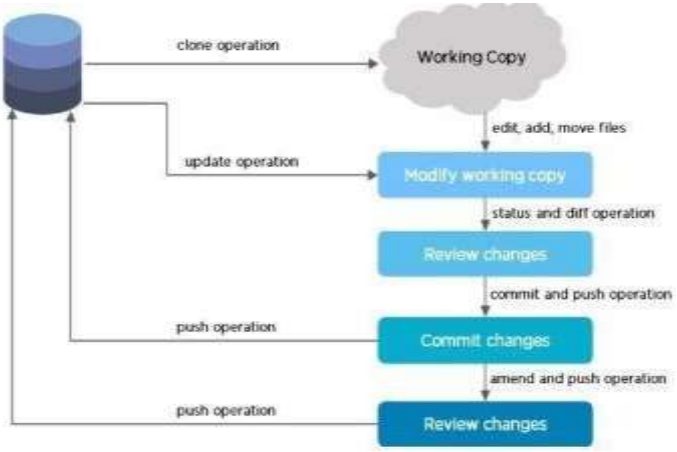
Git is composed of several key components that work together to enable version control and collaborative development. Here are the main components of Git:

- **Repository (Repo):** A repository is a directory or storage space where your project and its version history are stored. It contains all the files and directories associated with your project, along with the metadata and configuration information.
- **Working Directory:** The working directory is the directory on your local machine where you manipulate files and make changes to your project. It is essentially your local copy of the repository.
- **Index (Staging Area):** The index, also known as the staging area, is a middle ground where changes are prepared before being committed to the repository. It allows you to selectively stage changes, which means you can choose which modifications to include in the next commit.
- **Commit:** A commit is a snapshot of the changes made to the files in the repository. It represents a specific point in the project's history and is accompanied by a commit message that describes the changes.
- **Branch:** A branch is a parallel line of development within a repository. It allows developers to work on different features or bug fixes simultaneously without affecting the main codebase. Branches can be merged to incorporate changes into other branches.
- **Head:** HEAD is a reference to the latest commit in the currently checked-out branch. It essentially points to the tip of the branch you are currently on.
- **Remote:** A remote is a version of the repository stored on a different server. Git supports collaboration by allowing developers to push and pull changes between their local repository and remote repositories. Platforms like GitHub, GitLab, and Bitbucket are examples of remote repositories.
- **Clone:** Cloning is the process of creating a copy of a remote repository on your local machine. This allows you to start working on your own copy of a project.
- **Fetch:** The fetch operation retrieves changes from a remote repository but does not automatically merge them into your working directory. It is useful for reviewing changes before deciding to merge.
- **Pull:** Pull is a combination of fetch and merge. It retrieves changes from a remote repository and automatically merges them into your working directory.

- **Push:** Push is the operation that sends your committed changes to a remote repository, making them accessible to others.

Git workflow

The Git workflow is divided into three states:



1. **Working directory** - Modify files in your working directory
2. **Staging area (Index)** - Stage the files and add snapshots of them to your staging area
3. **Git directory (Repository) -** Perform a commit that stores the snapshots permanently to your Git directory. Checkout any existing version, make changes, stage them and commit.

Git Flow is a structured branching model designed for projects with well-defined release cycles and a need for strict quality control.

Branches:

The branching model described is commonly known as the Gitflow Workflow. It's a branching strategy that defines a strict branching model

designed to facilitate collaboration and streamline the release process. Let's go into detail about each branch:

1. ***Master Branch:*** The `master` branch represents the main codebase and contains production-ready code. This branch is typically stable and should only include thoroughly tested and approved changes. Each commit on the `master` branch represents a new version or release of the software.
2. ***Develop Branch:*** The `develop` branch is an integration branch where various feature branches are merged. It serves as a staging area for testing new features and ensuring they work well together before merging into the `master` branch. This branch may have ongoing development work and is not necessarily always in a production-ready state.
3. ***Feature Branches:*** Feature branches are created for developing new features or implementing changes. These branches are typically based on the `develop` branch. Once a feature is complete, the branch is merged back into the `develop` branch. Feature branches allow developers to work on specific tasks without affecting the main codebase.
4. ***Release Branch:*** The `release` branch is created when the `develop` branch reaches a point where it is ready for a production release. This branch is used for final testing, bug fixes, and preparing the code for deployment. No new features should be added to the release branch. Once the release is deemed stable, it is merged into both the `master` branch and the `develop` branch.
5. ***Hotfix Branch:*** The `hotfix` branch is used to quickly address critical issues or bugs in the production code. It is created directly from the `master` branch. Hotfixes are intended to be small and focused on resolving the specific issue at hand. Once the hotfix is

complete, it is merged into both the `master` branch and the `develop` branch to ensure that the fix is applied to future releases.

Here is the typical flow:

- Developers work on feature branches based on the `develop` branch.
- Completed features are merged into the `develop` branch.
- When ready for a release, a `release` branch is created from `develop`.
- The release branch undergoes testing and bug fixes.
- The release branch is merged into both `master` and `develop` once it's stable.
- If a critical issue arises in production, a `hotfix` branch is created from `master`.
- The hotfix is merged into both `master` and `develop` to keep both branches in sync.
- This Gitflow Workflow helps maintain a structured development process, ensuring that features are developed, tested, and released in a controlled manner.

Example Scenario:

- Imagine you're working on a large software project with a team of 10 developers. You have a major release planned for every six months.
- You create a "***feature/Ticket-Id***" branch to develop a new login system.
- Once the feature is complete, it's merged into the "***develop***" branch for integration and testing.
- As the release date approaches, you create a ***"release/v1.0"*** branch to freeze code for the upcoming release.
- Any critical issues discovered in the production environment are fixed in ***"hotfix"*** branch and merged into ***"master"*** and ***"develop."***

GIT INSTALLATION

Git for Windows stand-alone installer

- Download the latest Git for Windows installer.
- When you've successfully started the installer, you should see the **Git Setup** wizard screen. Follow the **Next** and **Finish** prompts to complete the installation. The default options are pretty sensible for most users.
- Open a Command Prompt (or Git Bash if during installation you elected not to use Git from the Windows Command Prompt).

- Run the following commands to configure your Git username and email using the following commands, replacing Emma's name with your own. These details will be associated with any commits that you create:

  $ git config --global user.name "CIT_CHENNAI"
  $ git config --global user.email CITCHENNAI@atlassian.com

- *Optional: Install the Git credential helper on Windows*

Bitbucket supports pushing and pulling over HTTP to your remote Git repositories on Bitbucket. Every time you interact with the remote repository, you must supply a username/password combination. You can store these credentials, instead of supplying the combination every time, with the Git Credential Manager for Windows.

BASIC COMMANDS OF GIT:

Some basic Git commands along with their syntax and examples:

- **Initialize a Repository:**

  Syntax: `git init`

  Example: `git init`

- **Clone a Repository:**

  Syntax: `git clone <repository_url>`

  Example: `git clone https://github.com/example/repository.git`

- **Check Repository Status:**

  Syntax: `git status`

  Example: `git status`

- **Add Changes to Staging Area:**

  Syntax: `git add <file(s)>`

  Example: `git add file.txt`

- **Commit Changes:**

  Syntax: `git commit -m "Commit message"`

  Example: `git commit -m "Add new feature"`

- **Create a New Branch:**

  Syntax: `git branch <branch_name>`

  Example: `git branch feature-branch`

- **Switch to a Branch:**

  Syntax: `git checkout <branch_name>`

  Example: `git checkout feature-branch` **OR**

  Syntax: `git switch <branch_name>` (Git version 2.23 and later)

Example: `git switch feature-branch`

- **Create and Switch to a New Branch:**

  Syntax: `git checkout -b <new_branch_name>`

  Example: `git checkout -b new-feature` **OR**

  Syntax: `git switch -c <new_branch_name>` (Git version 2.23 and later)

  Example: `git switch -c new-feature`

- **Merge Changes from One Branch to Another:**

  Syntax: `git merge <branch_name>`

  Example: `git merge feature-branch`

- **View the Commit History:**

  Syntax: `git log`

  Example: `git log`

- **Push Changes to a Remote Repository:**

  Syntax: `git push <remote_name> <branch_name>`

  Example: `git push origin master`

- **Pull Changes from a Remote Repository:**

  Syntax: `git pull <remote_name> <branch_name>`

  Example: `git pull origin master`

- **Show the Differences Between Working Directory and Staging Area:**

  Syntax: `git diff`

  Example: `git diff`

- **Show the Differences Between Staging Area and Last Commit:**

  Syntax: `git diff --cached`

  Example: `git diff --cached`

- **Show the Differences Between Working Directory and Last Commit:**

  Syntax: `git diff HEAD`

  Example: `git diff HEAD`

GITHUB

GitHub is an increasingly popular programming resource used for code sharing. It's a social networking site for programmers that many companies and organizations use to facilitate project management and collaboration. According to statistics collected in October 2020, it is the most prominent source code host, with over 60 million new repositories created in 2020 and boasting over 56 million total developers.

GitHub is a Git repository hosting service that provides a web-based graphical interface. It is the world's largest coding community. Putting a code or a project into GitHub brings it increased, widespread exposure. Programmers can find source codes in many different languages and use the command-line interface, Git, to make and keep track of any changes. GitHub helps every team member work together on a project from any location while facilitating collaboration. You can also review previous versions created at an earlier point in time.

GitHub's Features?

1. **Easy Project Management:** GitHub is a place where project managers and developers come together to coordinate, track, and update their work so that projects are transparent and stay on schedule.

2. **Increased Safety** With Packages Packages can be published privately, within the team, or publicly to the open-source community. The packages can be used or reused by downloading them from GitHub.

3. **Effective Team Management** GitHub helps all the team members stay on the same page and organized. Moderation tools like Issue and Pull Request Locking help the team to focus on the code.

4. **Improved Code Writing** Pull requests help the organizations to review, develop, and propose new code. Team members can discuss any implementations and proposals through these before changing the source code.

5. **Increased Code Safety** GitHub uses dedicated tools to identify and analyze vulnerabilities to the code that other tools tend to miss. Development teams everywhere work together to secure the software supply chain, from start to finish.

6. **Easy Code Hosting** All the code and documentation are in one place. There are millions of repositories on GitHub, and each repository has its own tools to help you host and release code.

HOSTING SERVICE FOR GIT REPOSITORY

When it comes to hosting Git repositories, various platforms provide a robust infrastructure for collaborative development, version control, and project management. Choosing the right hosting service depends on factors like ease of use, collaboration features, and integration capabilities. Here are some popular Git hosting services widely utilized in the software development community:

1. **GitHub:** GitHub stands out as one of the most prevalent Git hosting platforms, offering a user-friendly interface, powerful collaboration features, and seamless integration with various tools. It serves as an ideal choice for open-source projects, private repositories, and team collaboration.
2. **GitLab:** GitLab is a comprehensive web-based Git repository manager that not only provides source code management but also includes features like continuous integration. It caters to both cloud-based and self-hosted solutions, giving users flexibility in hosting their repositories.
3. **Bitbucket:** Owned by Atlassian, Bitbucket is another popular Git repository hosting service. Supporting both Git and Mercurial repositories, it offers features like code collaboration, issue tracking, and continuous integration. Bitbucket is often preferred by teams using other Atlassian tools such as Jira and Confluence.
4. **GitKraken Glo Boards:** GitKraken Glo Boards is an integrated task and issue tracking service linked with GitKraken, a Git client. This platform allows teams to manage tasks directly associated with their Git repositories and provides a visual approach to monitoring project progress.
5. **SourceForge:** SourceForge, with a long history, hosts open-source software projects and offers version control, bug tracking, and project management tools. While not as prominent as some other options, it remains a viable choice for numerous projects.
6. **AWS CodeCommit:** As part of Amazon Web Services (AWS), AWS CodeCommit is a fully managed source control service. It seamlessly integrates with other AWS services and provides a secure and scalable environment for hosting Git repositories.

Selecting the most suitable Git hosting service depends on your team's requirements, project size, and preferences for cloud-based or self-hosted solutions. Each platform has its strengths, catering to specific use cases within the software development landscape.

Difference between GIT and GITHUB

Git and GitHub are related concepts but serve different purposes in the context of version control and collaborative software development.

Git: Git is a distributed version control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows.

GitHub: GitHub is a web-based Git repository hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.

S.No.	Git	GitHub
1	Git is a software.	GitHub is a service.
2	Git is a command-line tool	GitHub is a graphical user interface
3	Git is installed locally on the system	GitHub is hosted on the web
4	Git is maintained by linux.	GitHub is maintained by Microsoft.
5	Git is focused on version control and code sharing.	GitHub is focused on centralized source code hosting.
6	Git is a version control system to manage source code history.	GitHub is a hosting service for Git repositories.
7	Git was first released in 2005.	GHub was launched in 2008.
8	Git has no user management feature.	GitHub has a built-in user management feature.
9	Git is open-source licensed.	GitHub includes a free-tier and pay-for-use tier.
10	Git has minimal external tool configuration.	GitHub has an active marketplace for tool integration.
11	Git provides a Desktop interface named Git Gui.	GitHub provides a Desktop interface named GitHub Desktop.
12	Git competes with CVS, Azure DevOps Server, Subversion, Mercurial, etc.	GitHub competes with GitLab, Bit Bucket, AWS Code Commit, etc.