SOFTWARE ENGINEERING

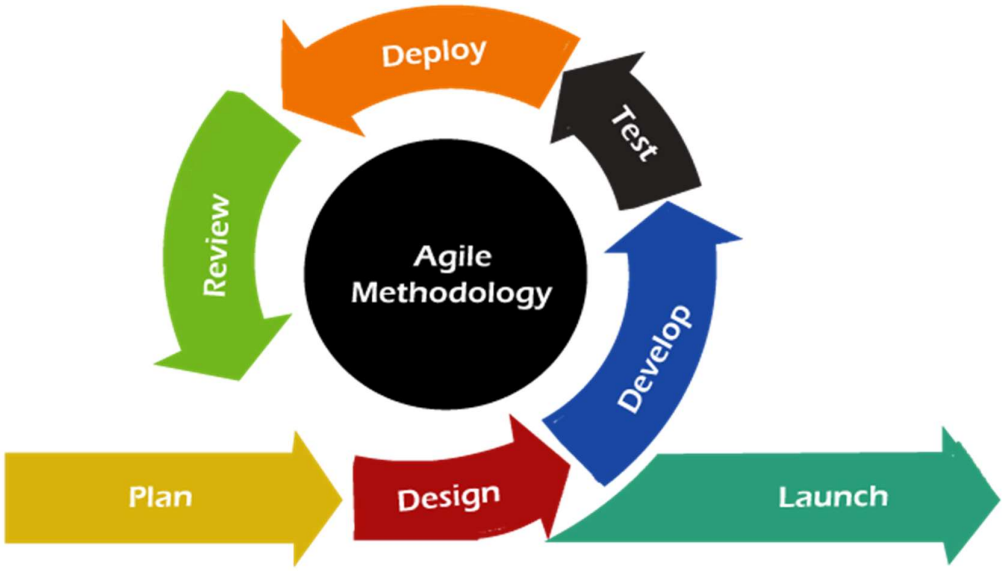
UNIT – 1

TOPIC – 8

SCRUM FRAMEWORK AND INTRODUCTION TO DEVOPS

I. Various Agile Frameworks

Agile is an approach to software development where work is divided into small, manageable parts called frameworks. These frameworks help teams deliver projects in a flexible and adaptive manner.



Some popular Agile frameworks are:

a) Rational Unified Process (RUP): Focuses on repeated cycles of development.
b) Adaptive Software Development (ASD): Emphasizes continuous learning and adaptation.
c) Feature Driven Development (FDD): Centers on building the project based on specific features.
d) Crystal Clear: Prioritizes simplicity and frequent delivery.
e) Dynamic Systems Development Method (DSDM): Offers a structured approach to rapid software development.
f) Extreme Programming (XP): Promotes continuous testing and customer feedback.
g) Scrum: A popular, team-based method that breaks work into short, manageable chunks.

II. SCRUM Framework

What is SCRUM?

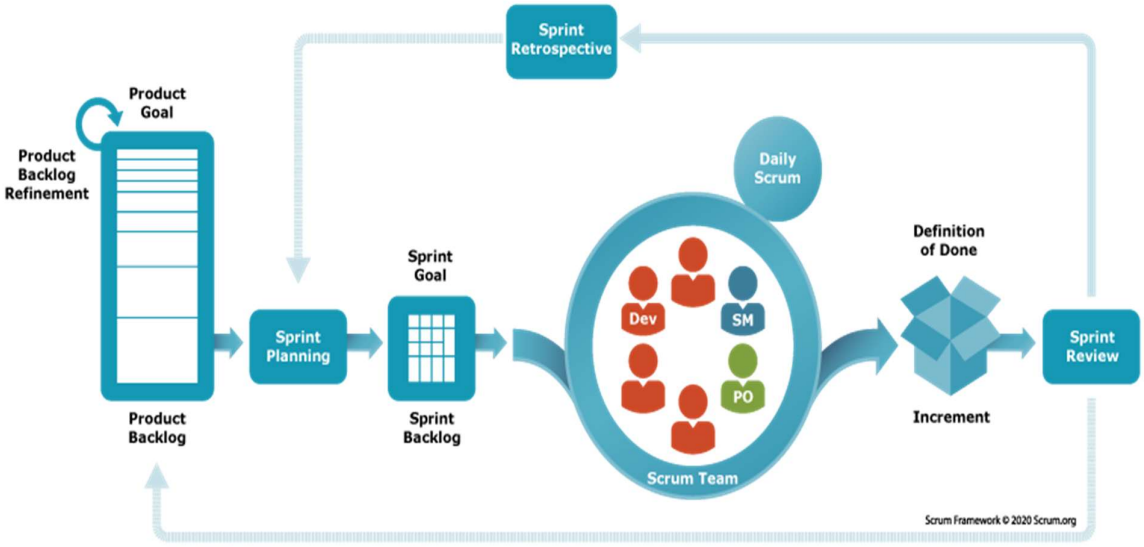
Scrum is a way to manage and organize the process of developing software. It's part of the Agile family, which means it helps teams stay flexible and open to changes. The main idea behind Scrum is to break big tasks into smaller, manageable pieces and complete them in Sprints—a set period of time (usually 2 weeks). Scrum encourages continuous improvement and frequent communication to make sure the team is always on track.

History of Scrum

Scrum was created in the early 1990s by two developers named Ken Schwaber and Jeff Sutherland. They wanted to help teams work together better and deliver software faster. They were inspired by how teams in Rugby work together in small groups to achieve their goals, so they used the word "Scrum" from Rugby.

In 1995, they officially introduced the Scrum Framework at a conference, and since then, it has become very popular. Today, Scrum is used not only in software development but also in other industries because it helps teams work efficiently and deliver results quickly.

Key Roles in Scrum:



Product Owner:

The Product Owner is like the manager of the product. They understand what the customer needs and create a list of tasks, called the Product Backlog. The Product Backlog contains

everything that needs to be done to create the final product, and it's the Product Owner's job to prioritize these tasks.

Scrum Master:

The Scrum Master acts as a coach for the team. They make sure the Scrum process is being followed and help the team stay organized. The Scrum Master removes any obstacles that might slow the team down, like solving problems and setting up meetings.

Scrum/Development Team:

This is the group of developers who do the actual work of building the product. They handle the technical tasks and work on the items from the Product Backlog.

Scrum Artifacts

Artifacts in Scrum are tools or documents that help keep the project organized.

Product Backlog:

This is a to-do list for the whole project. Created and managed by the Product Owner, it contains all the tasks, features, and improvements needed for the final product. The Product Backlog is constantly updated with new ideas, and tasks are prioritized based on what's most important.

Sprint Backlog:

A smaller, focused to-do list for each Sprint. The team picks tasks from the Product Backlog that they can finish during the Sprint (2 weeks or so). This list becomes the team's main focus for that Sprint.

Increment:

The Increment is the collection of all the work that has been completed in the current Sprint, plus everything from past Sprints. After each Sprint, the team should have a working version of the product that includes all the new tasks they finished. It's like reaching a small milestone.

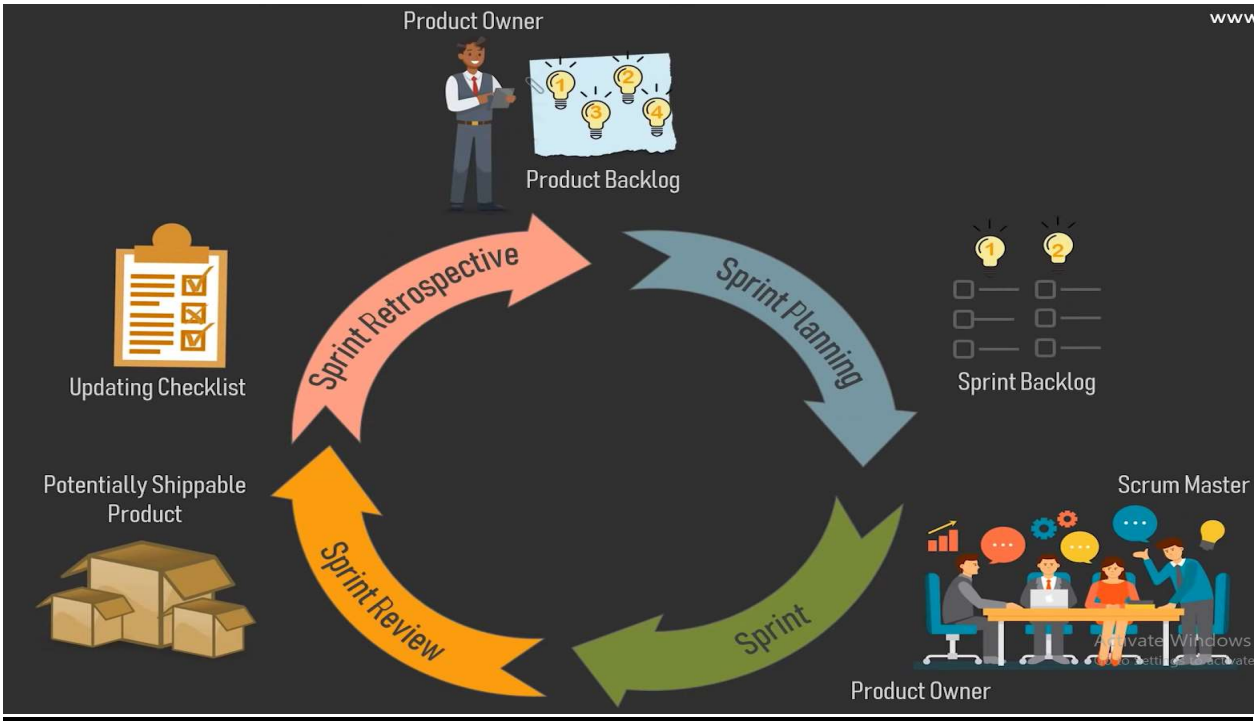
Burn-Down Chart:

This is a visual chart that tracks how much work is left compared to the time remaining in the Sprint. It helps the team see if they're on track to finish on time, and if they're falling behind, they can make changes to speed up. Ideally, the line on the chart should go downwards, showing that tasks are being completed.

Scrum Board (Task Board):

A visual tool that shows the status of tasks during the Sprint. Tasks are organized in columns like "To Do," "In Progress," and "Done." As tasks get completed, they are moved across the board. This helps everyone see what work has been done and what still needs to be finished.

Main Elements in Scrum:



- **Product Backlog:** A list of everything that needs to be done for the project, organized by the Product Owner in order of priority.
- **Sprint Backlog:** A smaller list of tasks chosen from the Product Backlog that the team will work on during the current Sprint.
- **Daily Scrum:** A short 15-minute meeting held every day. In this meeting, the team discusses:
  - What they worked on yesterday.
  - What they will work on today.
  - Any problems they are facing.
  - This keeps the team aligned and focused.

- **Sprint Review:** At the end of each Sprint, the team presents what they have completed to the Product Owner and other stakeholders. Feedback is gathered, and changes may be made for the next Sprint.
- **Sprint Retrospective:** After the Sprint Review, the team reflects on how the Sprint went and discusses what went well and what could be improved for the next Sprint.

Advantages of Scrum:

• **Quick Progress:** By breaking tasks into smaller parts, work is done faster, and parts of the product are ready quickly.

• **Flexibility:** Scrum allows for changes after every Sprint, so the team can make adjustments based on customer feedback.

• **Team Collaboration:** Everyone works together closely, which leads to better communication and problem-solving.

• **Continuous Improvement:** After each Sprint, the team reflects on what went well and what didn't. This helps them improve for the next Sprint.

Disadvantages of Scrum:

- **Confusing Roles:** Sometimes, team members might be unsure about their responsibilities, especially if they're new to Scrum.
- **Difficult to Manage Changes During a Sprint**: If a customer requests big changes in the middle of a Sprint, it can disrupt the team's workflow.
- **Requires Discipline:** Scrum needs the team to be very organized and disciplined in keeping up with daily meetings and task updates.
- **Not Ideal for Large Projects:** Scrum might not work well for extremely large projects, as it may not offer enough structure to handle everything.
- **Dependency on the Product Owner:** If the Product Owner is unavailable or slow to make decisions, the whole team can be delayed because they rely on the Product Owner to prioritize tasks.

Agile vs DevOps

- Agile is a software development approach that has been around for a few decades. It focuses on fast development cycles, ensuring that software is built and tested quickly.
- DevOps, introduced in the mid-2000s, also supports fast software development, but it goes beyond development. It includes continuous operations, meaning the software is constantly monitored to ensure it works smoothly with no downtime.

Differences Between Agile and DevOps

- Agile stops after development, testing, and deployment. There is no ongoing monitoring of the software.
- In DevOps, continuous monitoring ensures the software is always operational, with no issues for users.
- Agile typically has separate teams for development, testing, and operations, while DevOps merges all roles, making each individual responsible for all phases of the software life cycle.

Introduction to DevOps

The evolution of software development by comparing three approaches: Waterfall, Agile, and DevOps:

1. Waterfall Model (Traditional)

Best when all the requirements for the project are clear and won't change. The product is well-defined and stable from the start. Waterfall is a step-by-step method. You finish one step before moving to the next (like a waterfall flowing downward).

2. Agile Development (Modern Approach)

Best when the project's needs or requirements might change often. You need to develop the product quickly. Agile is flexible and works in short cycles, allowing for quick changes and improvements.

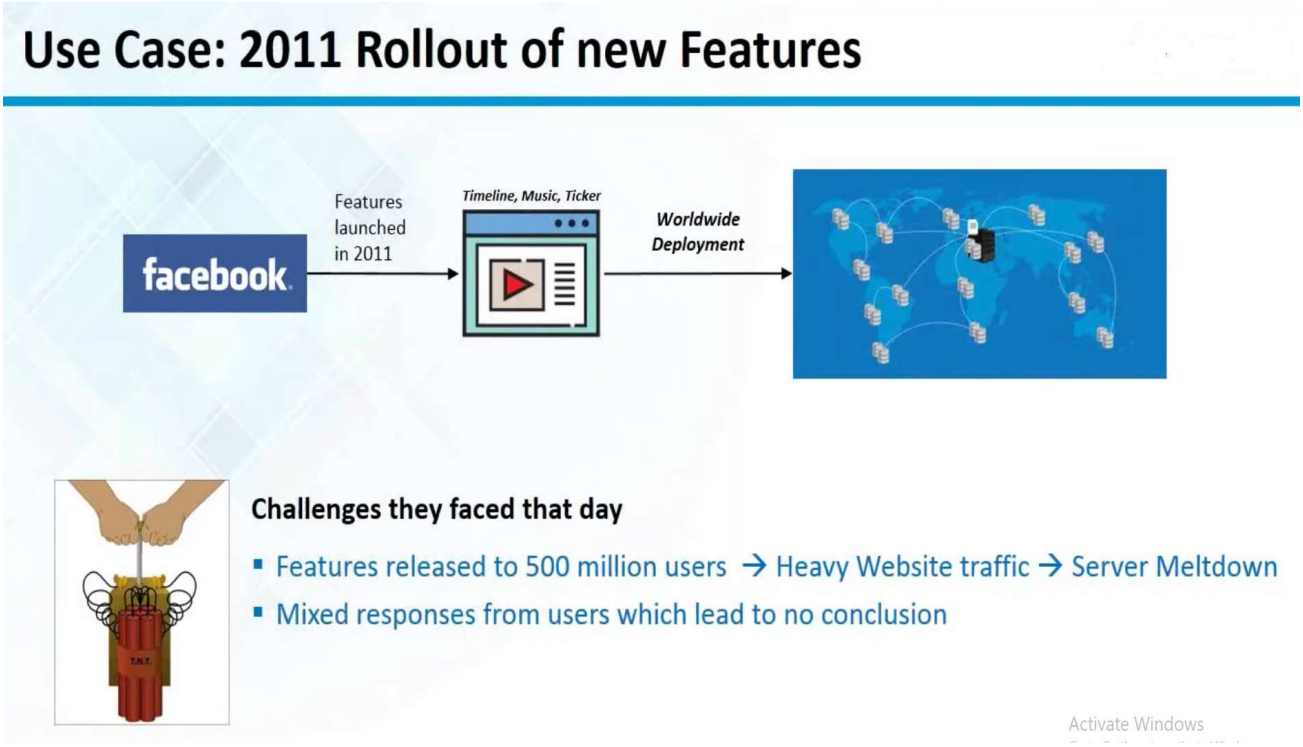
3. DevOps Approach (Latest Trend)

Best when requirements are still changing frequently, just like in Agile. Both the development and operations (maintenance and monitoring) need to happen quickly and smoothly. DevOps combines development and operations teams to ensure faster development and continuous monitoring of the product to keep it running without issues.

Use case

Facebook's 2011 Feature Rollout and How DevOps Helped

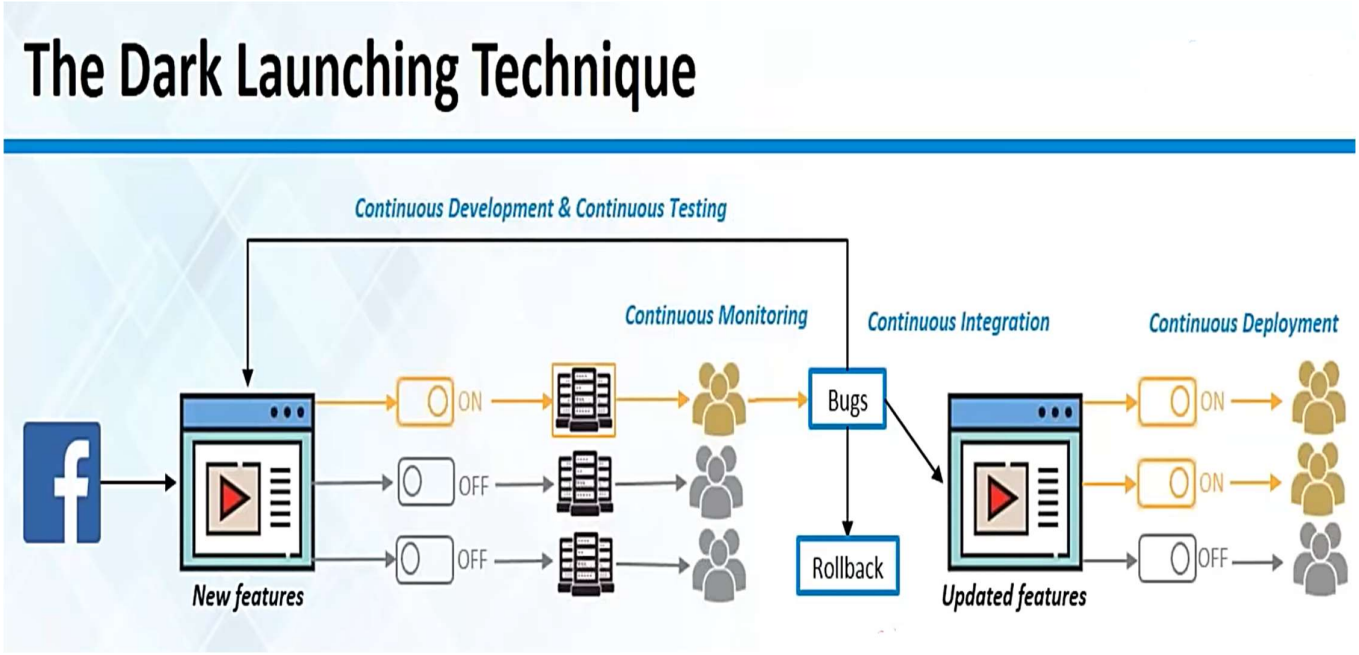
In 2011, Facebook introduced new features like **Timeline**, **Music**, and **Ticker**. They launched these features to **500 million users** all at once. However, the sudden increase in traffic was too much for Facebook's **servers**, causing a **server meltdown** where parts of the site stopped working. Users also gave **mixed feedback**, making it hard for Facebook to understand what to improve.



How DevOps Helped Fix These Issues

To solve these problems, Facebook used **DevOps**. DevOps brings together the teams that develop new features and the teams that manage the servers. This helped Facebook **fix the server problems quickly** by working together closely.

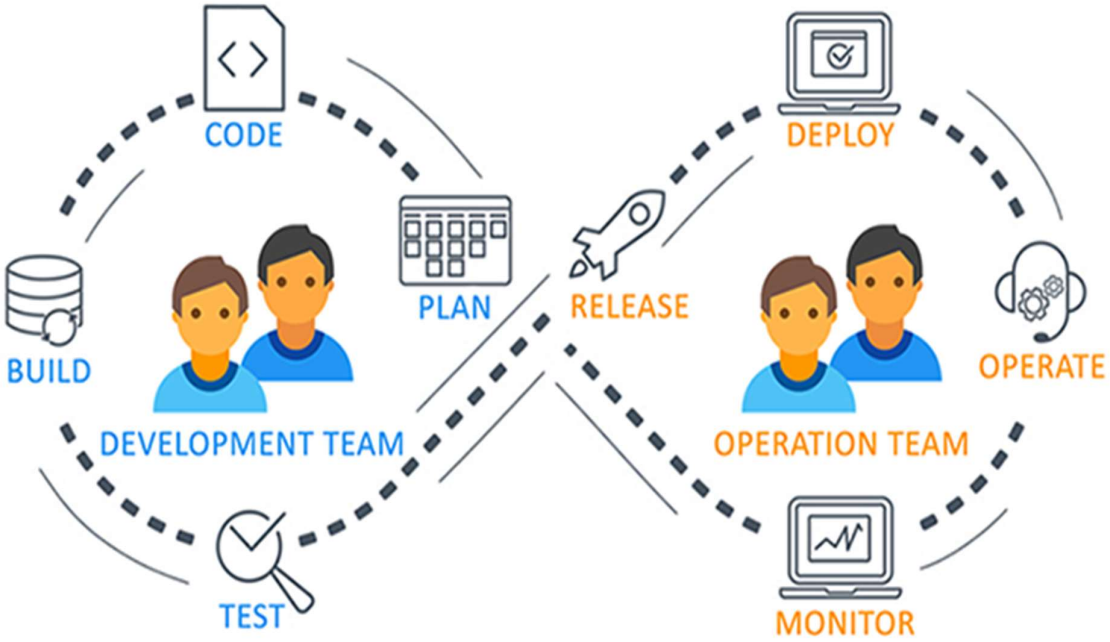
Facebook also used a technique called **dark launching**, which is part of the DevOps approach. Instead of giving the new features to everyone at once, they tested them with a **small group of users** first. This helped Facebook see how the servers handled the traffic and understand what users thought before releasing the features to everyone.



Using **DevOps** and **dark launching**, Facebook was able to avoid further big problems, gather better feedback, and make improvements without causing the website to crash again.

What is DevOps?

DevOps is a methodology that combines software development (Dev) and IT operations (Ops). Its main goal is to shorten the systems development lifecycle while delivering high-quality software. DevOps relies on practices like continuous development, integration, and monitoring to streamline the production process.



Key Features:

- **Continuous Development**: Ongoing and iterative updates to software code.
- **Continuous Testing**: Automated testing at every stage to catch bugs early.
- **Continuous Integration**: Regularly merging code changes to avoid conflicts.
- **Continuous Deployment**: Automating the release of software updates.
- **Continuous Monitoring**: Tracking performance and issues post-deployment.

DevOps Lifecycle

Description:

The DevOps lifecycle is a continuous, iterative process that links various stages of software development and operations. The goal is to create a seamless flow between coding, building, testing, deploying, and monitoring.

Key Phases:

1. **Plan**: Define the software objectives, features, and roadmap.
2. **Code**: Write the actual program according to the plan.
3. **Build**: Convert the code into executable software.
4. **Test**: Automate tests to ensure that the code works as expected.

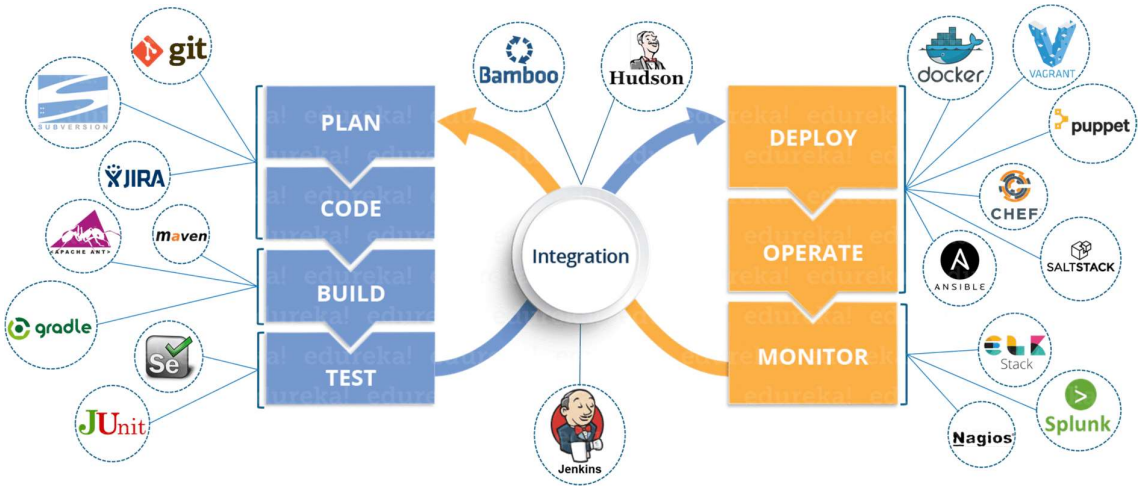
5. **Deploy**: Move the application to a production environment for users.
6. **Operate**: Ensure the software runs efficiently with minimal downtime.
7. **Monitor**: Continuously observe the system for performance and issues.

Integration: Throughout this lifecycle, integration ensures that each phase communicates smoothly, allowing for faster iterations and improvements.

DevOps Tools

Description:

DevOps tools automate processes across the lifecycle, ensuring that each step from coding to deployment is quick and efficient. These tools enhance collaboration between teams and ensure continuous workflows.



Key Tools:

- **Version Control Tools (Git, Subversion)**: Manage code versions and collaborate across teams.
- **Build Tools (Maven, Apache Ant)**: Automate the building and compilation of code.
- **Planning Tools (JIRA)**: Help manage tasks and track project progress.
- **Continuous Integration Tools (Jenkins, Bamboo, Hudson)**: Automate code integration and build processes.
- **Deployment Tools (Docker, Vagrant, Kubernetes)**: Manage containerized applications and deploy updates.
- **Monitoring Tools (Nagios, Splunk, ELK Stack)**: Track system performance and detect issues in real-time.

Each tool specializes in automating a specific phase of the DevOps lifecycle, ensuring smoother, faster, and more reliable workflows.