Team 1

DEV OPS

Team 1

UPDATES EVERY DAY HOW ?!

AGENDA

- Introduction.
- Core DevOps Principles and Practices.
- DevOps Tools and Technologies.
- DevOps Pipeline and Workflow.
- Challenges and Solutions in DevOp Implementation
- Future of DevOps and Conclusion
- Q&A and discussion

Team 1

INTRODUCTION TO DEVOPS

- DevOps combines "Development" and "Operations."
- It's a culture of collaboration to produce software faster and with higher quality.
- Focuses on automation and continuous delivery.

Team 1

WHAT IS DEVOPS ?

- DevOps unifies development and IT teams.
- Goal: Increase collaboration, automation, and deliver software faster.
- Emphasis on team alignment and breaking down silos.

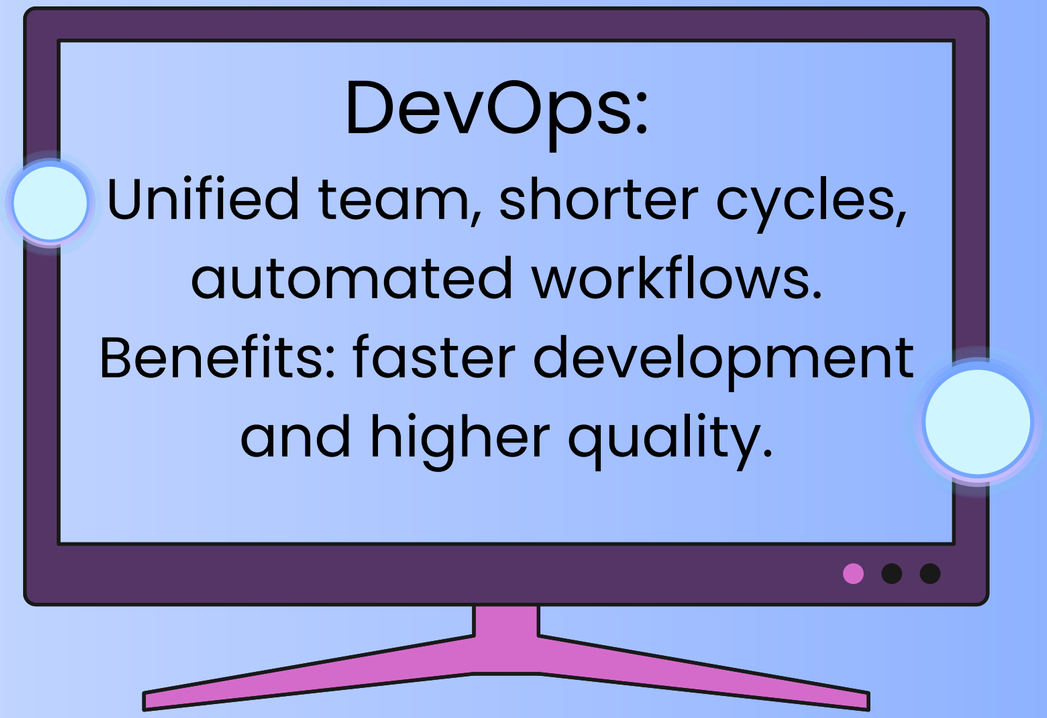
Team 1

HISTORY OF DEVOPS

- Emerged in the 2000s to meet demands for faster software updates.
- Inspired by Agile methodologies focused on speed and iteration.
- Became a standard practice to bridge the gap between development and operations.

Team 1

TRADITIONAL IT VS. DEVOPS



Team 1

BENEFITS OF DEVOPS

- Faster Time to Market: Faster releases.
- Improved Quality: Early error detection with automation.
- Enhanced Team Collaboration: Less friction, better teamwork.

Team 1

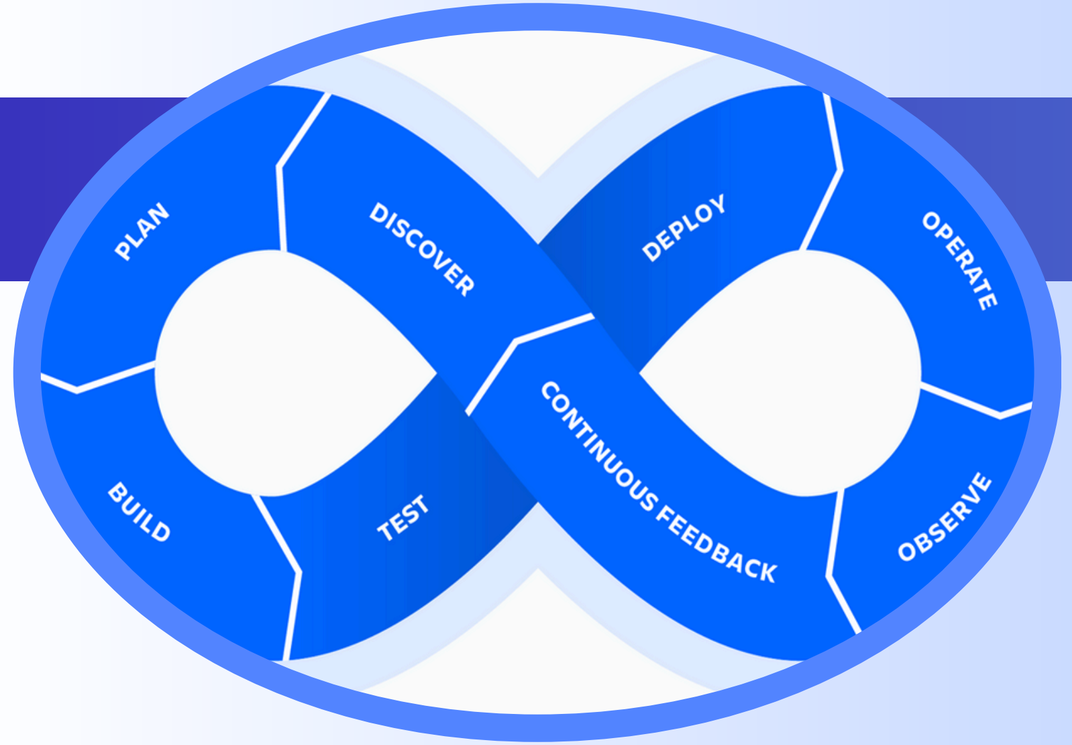
GOALS OF DEVOPS

- Continuous Delivery:
- Regular, reliable updates.
- Automation:
- Save time, reduce errors.
- Security and Compliance:
- Build security into the process.

Team 1

DEVOPS

PRINCIPLES AND PRACTICES



To realize the full potential of DevOps, teams should follow key DevOps principles

DevOps is more than just development and operations teams working together. It’s more than tools and practices. DevOps is a mindset, a cultural shift, where teams adopt new ways of working.

Team 1

1-COLLABORATION

- The key premise behind DevOps is collaboration. Development and operations teams coalesce into a functional team that communicates, shares feedback, and collaborates.

- By encouraging open communication, teams can gain understanding of each other's challenges and processes, leading to greater efficiency. Holding regular meetings, planning sessions together.

Team 1

2-AUTOMATION

- An essential practice of DevOps is to automate as much of the software development lifecycle as possible. This gives developers more time to write code and develop new features. Automation reduces human errors and increase team productivity.

- With automated processes, teams achieve continuous improvement with short iteration times, which allows them to quickly respond to customer feedback.

Team 1

3. CONTINUOUS INTEGRATION (CI)

- CI is the practice of integrating code changes into a shared repository several times a day. Each integration is automatically tested to detect and fix issues early.

- Benefits of CI: Reduced integration problems, faster bug detection, and improved software quality.

Team 1

4. CONTINUOUS DELIVERY (CD)

- CD extends CI by automatically deploying code changes to production after passing automated tests. This ensures that new features and bug fixes reach users quickly.

- Benefits of CD: Faster time to market, reduced manual intervention, and increased confidence in the deployment process.

Team 1

5. INFRASTRUCTURE AS CODE (IAC)

- IaC involves managing and provisioning computing resources using code and configuration files rather than manual processes. Tools like Terraform and AWS CloudFormation are commonly used.

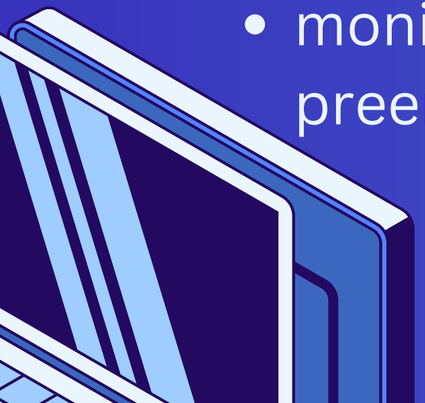
- IaC enables teams to create reproducible and consistent environments, which minimizes discrepancies between development, testing, and production.

Team 1

6. MONITORING AND LOGGING

- By collecting real-time data on application performance, resource utilization, and user behavior, teams gain insights into software performance. This data-driven approach allows for proactive issue detection, faster troubleshooting, and informed decision-making.

- monitoring tools can alert the team to sudden spikes in server load, enabling preemptive scaling of resources and preventing potential outages.



Team 1

7. LEAN

- Lean principles focus on maximizing value while minimizing waste. This involves optimizing processes, and removing unnecessary steps in the workflow.

- By applying lean practices, teams can streamline operations, improve cycle times, and enhance product quality. This focus on efficiency not only reduces costs but also encourages a culture of continuous improvement, where teams are always looking for ways to enhance their workflows.

Team 1

DEVOPS TOOLS & TECHNOLOGIES

Team 1

1-COLLABORATION AND COMMUNICATION TOOLS

- These tools enhance teamwork and transparency between development and operations.

Team 1

TOOLS

- **Slack**
  - Channels
  - Messaging multimedia.
  - Integrations
  - Searchable Archives
  - Customization
- **Jira**
  - Bug Tracking
  - Software Development
  - Team Collaboration
- **GitHub**

Team 1

2- AUTOMATION

- Automation is central to DevOps, enabling repetitive tasks to be performed consistently and efficiently.
- it applies to builds, testing, deployments, and beyond.

Team 1

TOOLS

- **Jenkins**

Jenkins works as a (CI/CD) tool to automate software development tasks.

Here's how it functions:

1. **Trigger**: A build is triggered by code changes, a schedule, or manually.
2. **Build**: Jenkins compiles code, runs unit tests, and packages the application.

Team 1

JENKINS CONT..

3.**Test**: Automated tests are executed to ensure quality.

4.**Deploy**: Artifacts are deployed to staging or production environments.

5.**Plugins**: Extend its functionality (e.g., Docker, Git).

6.**Monitoring**: Logs and dashboards track job performance.

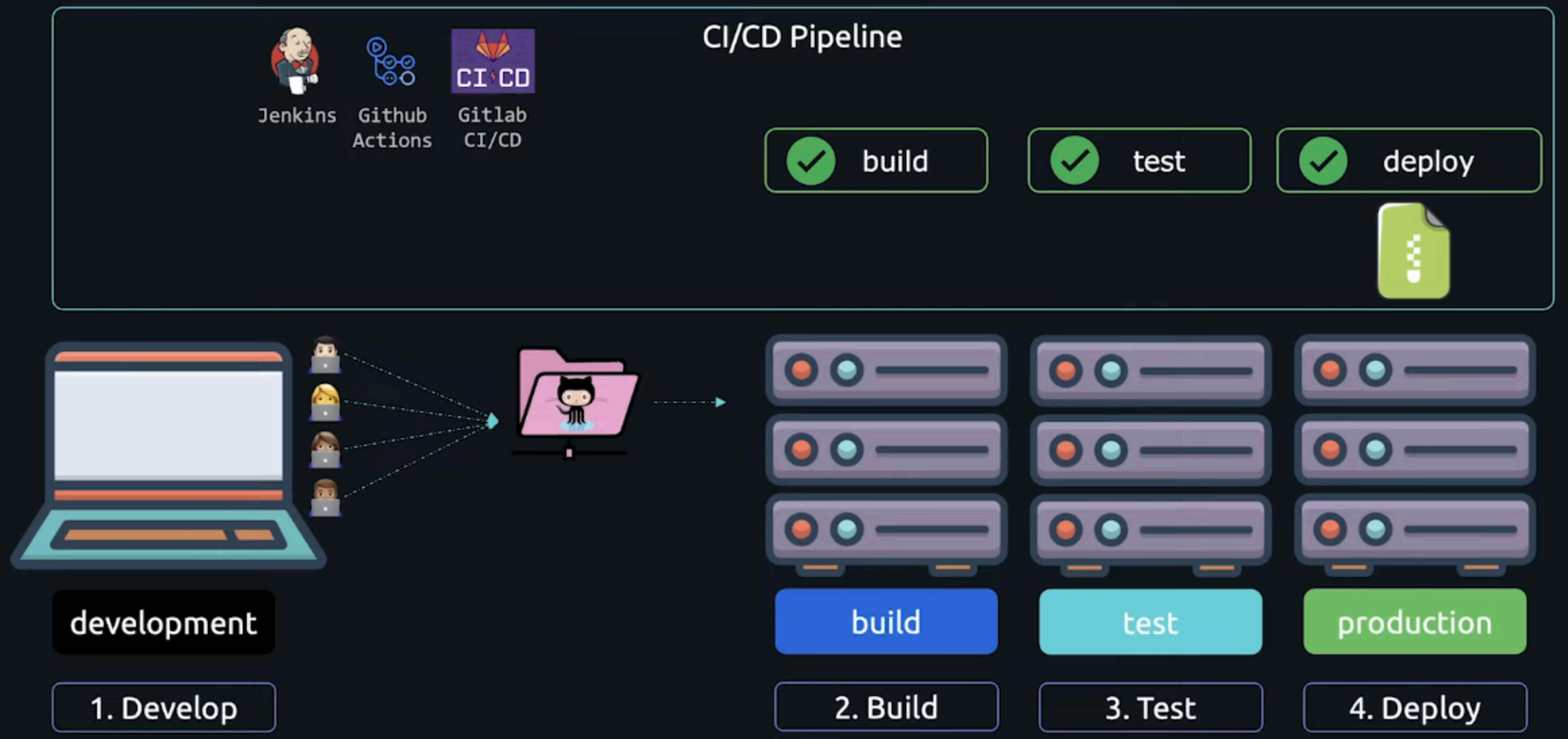
Jenkins uses pipelines defined in code (Jenkinsfile) for repeatable workflows.

Team 1

JENKINS CONT..

Common Tools Jenkins Integrates With

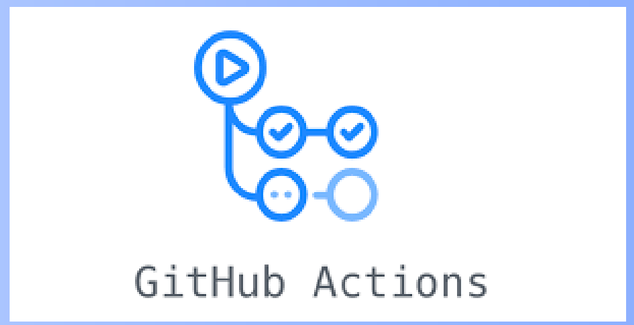
- **Version Control:** Git, GitHub, GitLab, Bitbucket.
- **Build Tools:** Maven, Gradle, Ant.
- **Containers:** Docker, Kubernetes.
- **Testing:** JUnit, Selenium, TestNG.
- **Cloud:** AWS, Azure, GCP.



Team 1

SIMILAR CI/CD TOOLS

- GitHub Actions
- GitLab CI/CD
- CircleCI

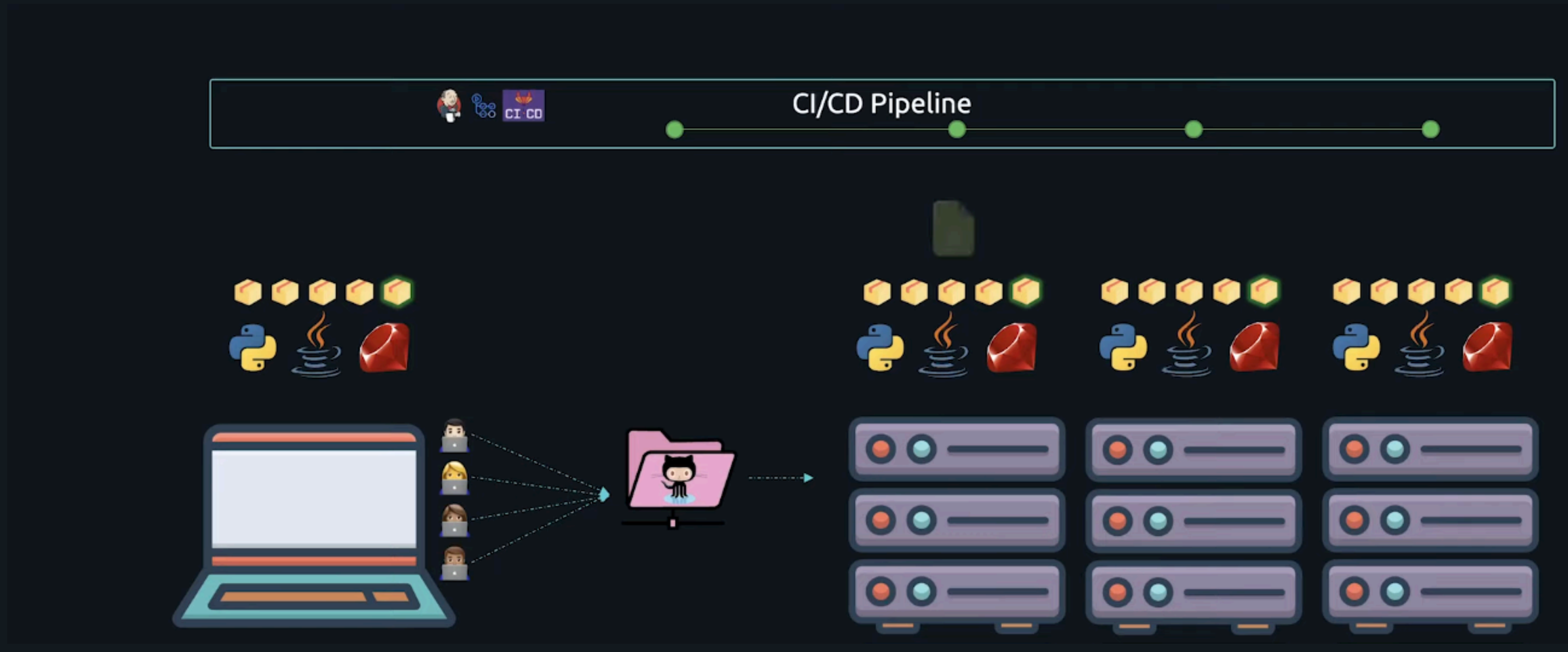


Team 1

3-CONTAINERIZATION

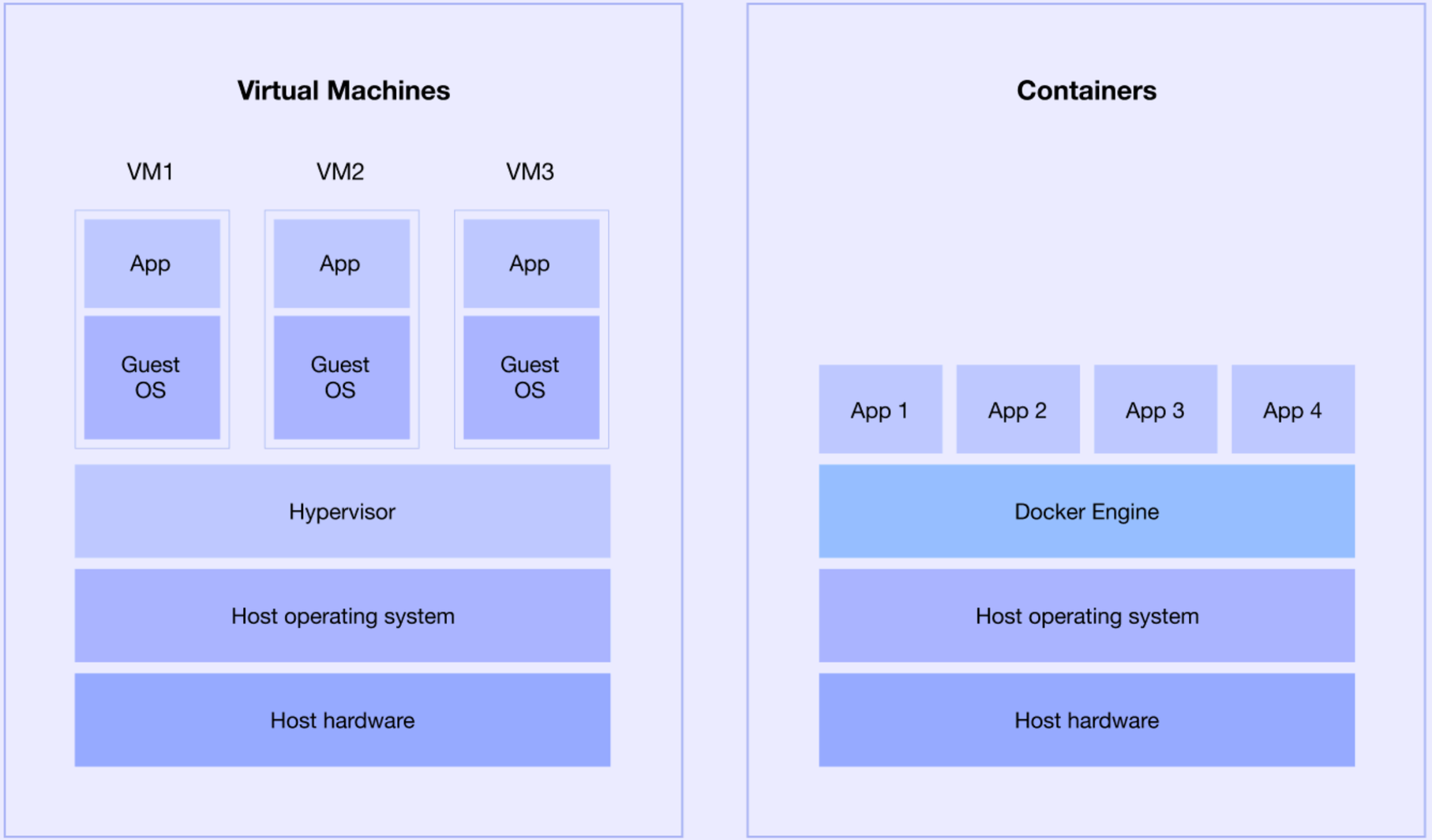
- **Containerization ensures consistent runtime environments across different stages of software development, from development to production.**

Team 1

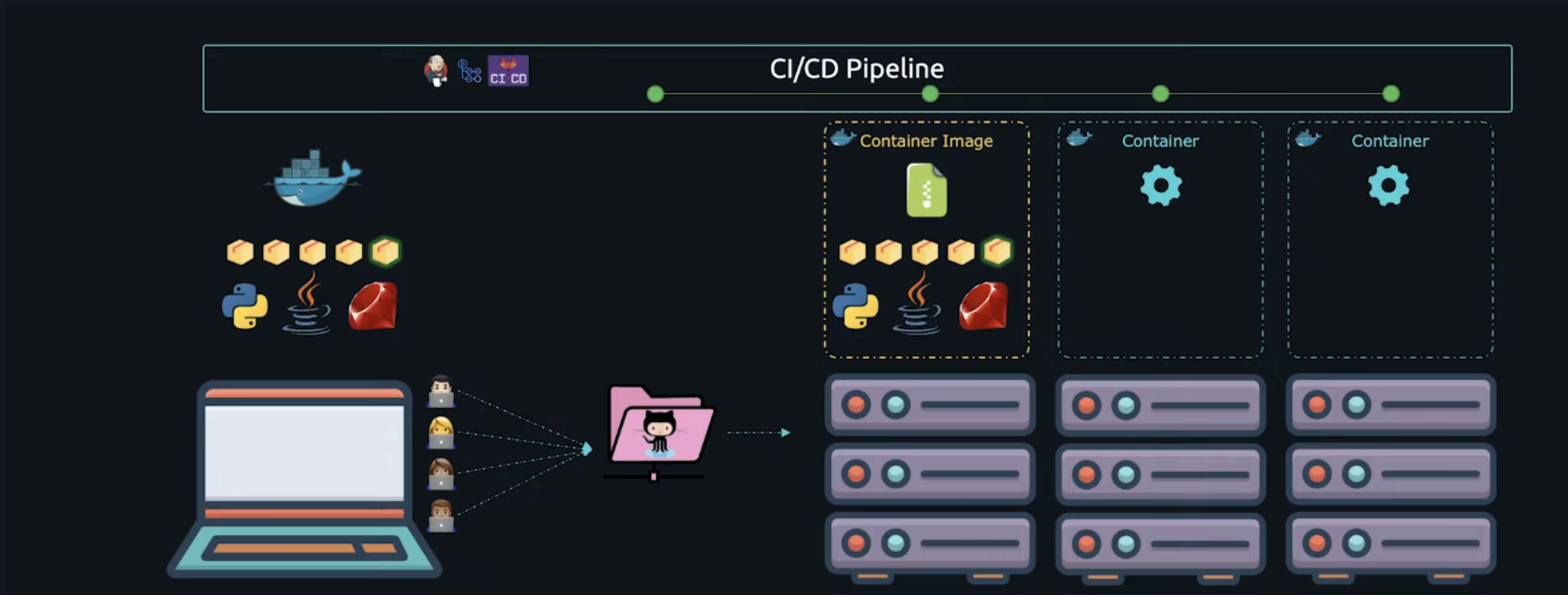

CI/CD Pipeline
CI CD

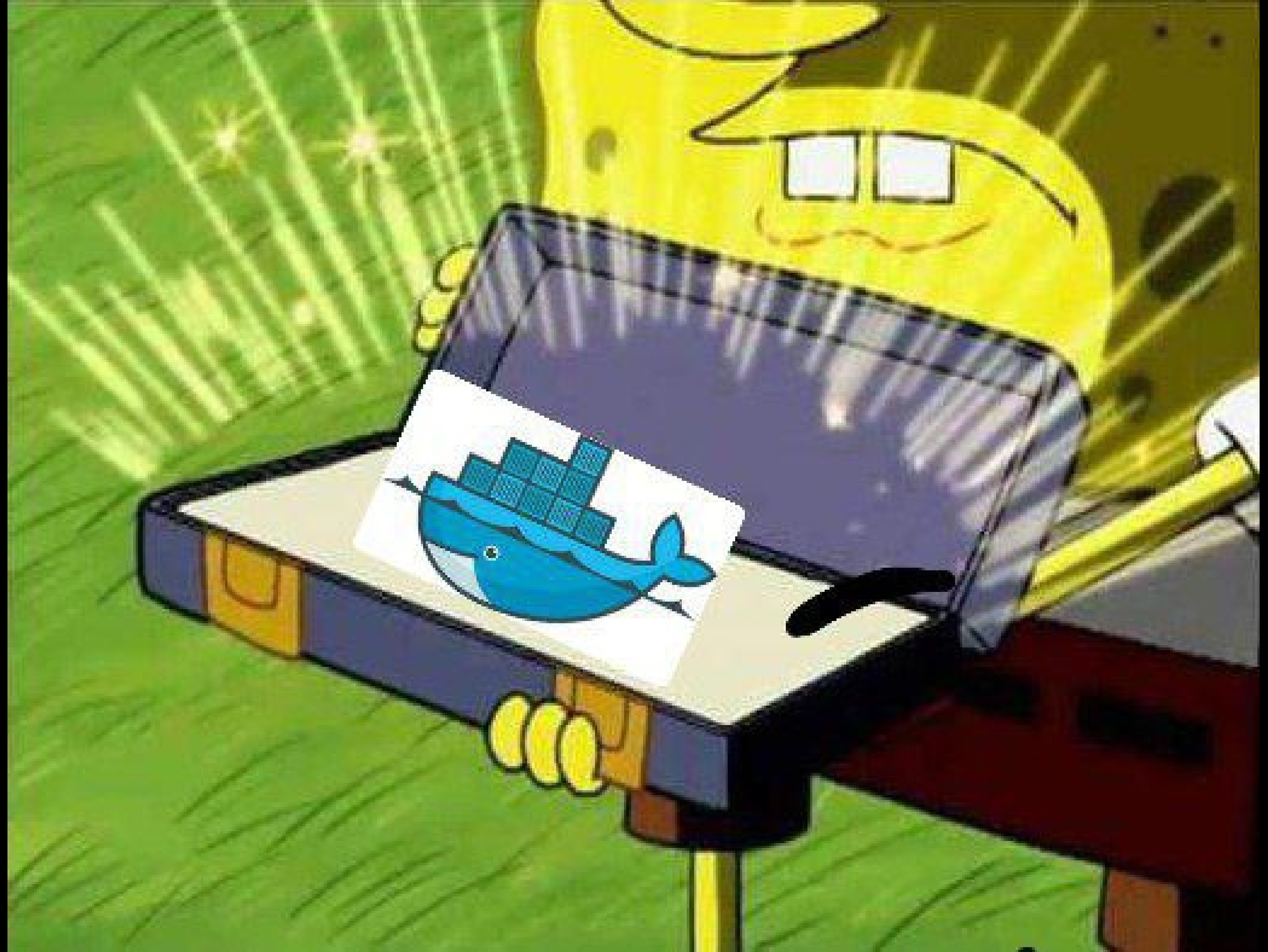
Team 1

Virtual machines versus containers



SIMFORM


CI/CD Pipeline
CI CD
Container Image
Container
Container



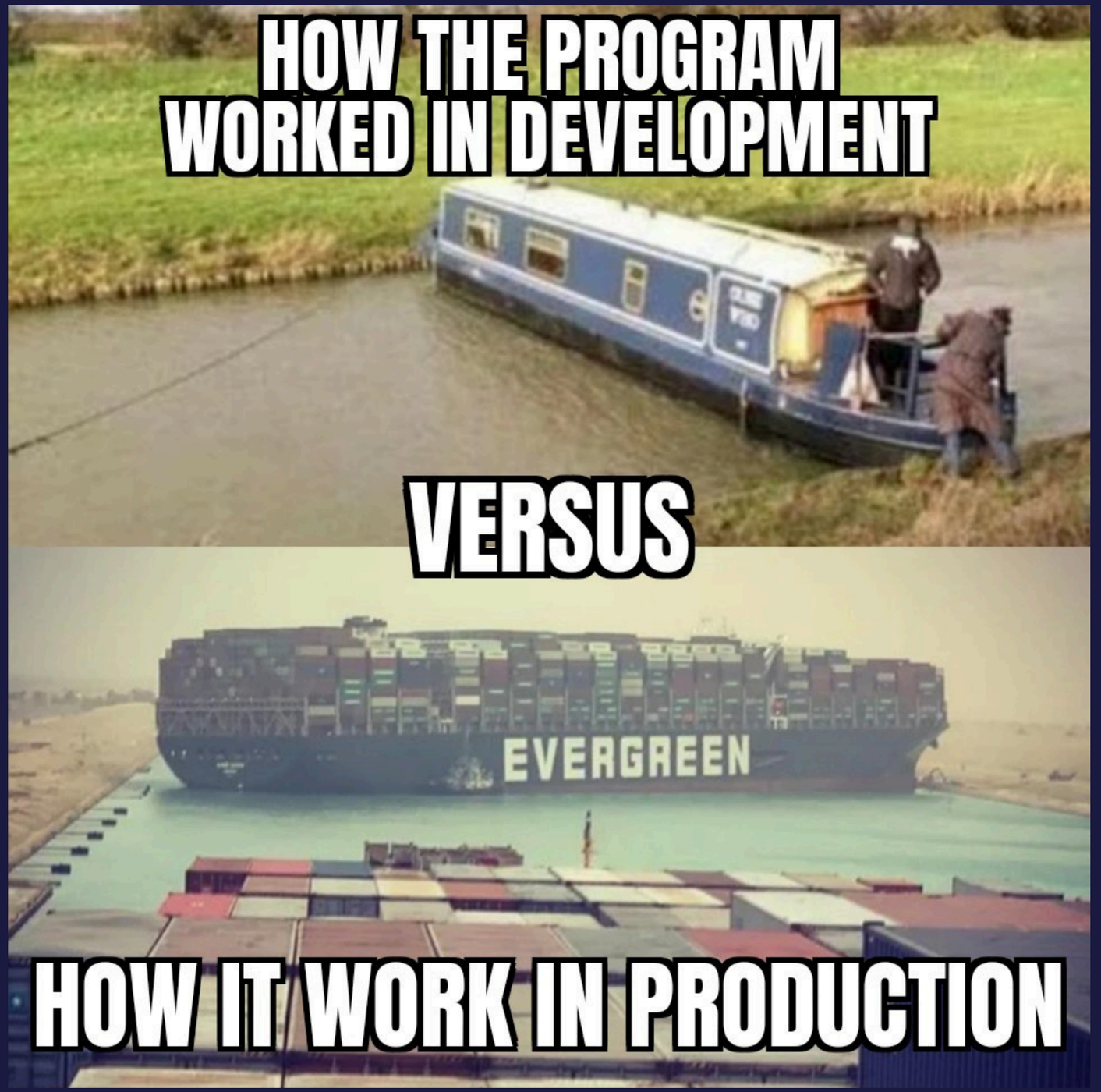
TOOLS

- Docker works by utilizing containerization technology to package applications and their dependencies into containers.

Team 1

HOW IT OPERATES?

- **Images**: Developers create Docker images, which are templates containing the application code, runtime, libraries, and dependencies.
- **Containers**: Containers are lightweight, isolated instances of images They share the host OS kernel, making them efficient and portable.
- **Docker Engine**: Manages container lifecycle: building, running, and stopping containers.


HOW THE PROGRAM
WORKED IN DEVELOPMENT
VERSUS
EVERGREEN
HOW IT WORK IN PRODUCTION

Team 1

4-CONTAINER ORCHESTRATION

- **Orchestration manages and scales containerized applications.**

Team 1

WHY?

- Containers solve the problem of deploying applications consistently across different environments, but managing many containers across multiple servers can be complex.

Team 1

KUBERNETES

- Kubernetes automates tasks like scaling, networking, and monitoring to simplify container management.

Team 1

KUBERNETES FEATURES

- **Automated Scaling:** Dynamically adjust the number of containers based on resource usage or traffic.
- **Load Balancing:** Distributes incoming traffic across multiple containers to ensure reliability.
- **Self-Healing:** Automatically restarts failed containers, replaces unresponsive ones, and reschedules them on healthy nodes.
- **Service Discovery:** Makes services available to other applications without needing manual configuration of IPs or DNS names.

DEVOPS Team 1

5-INFRASTRUCTURE AS CODE (IAC)

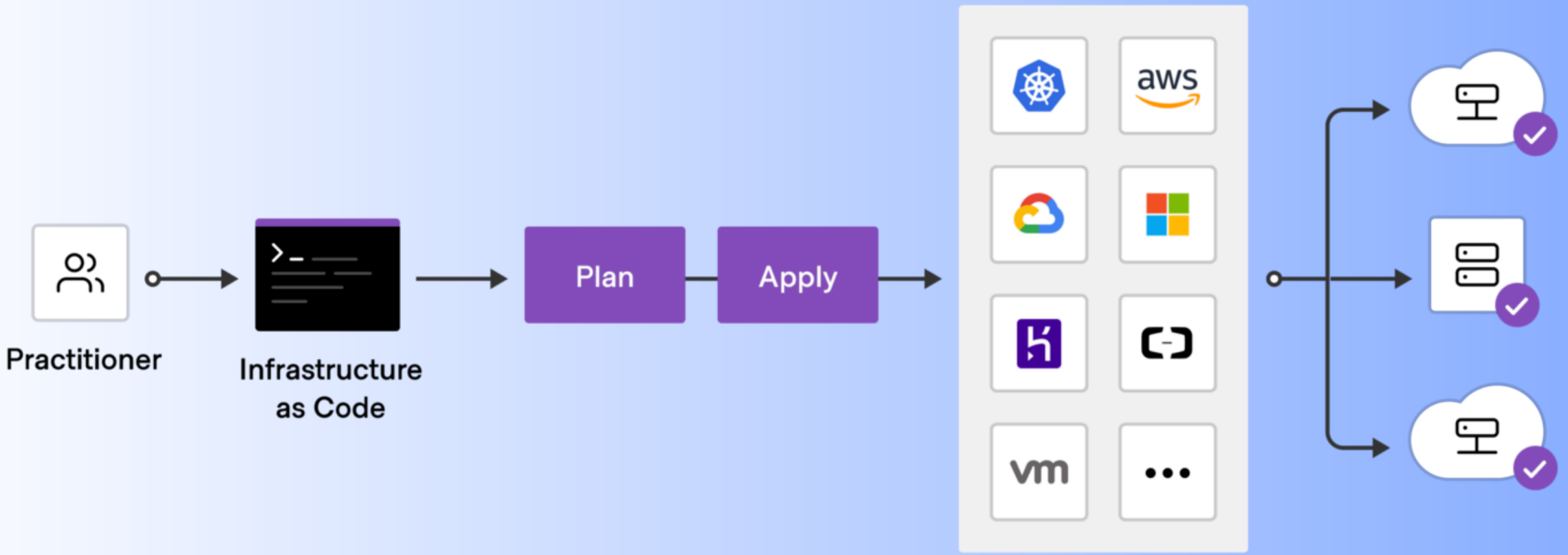
- IaC allows infrastructure provisioning using code rather than manual processes.
- It ensures consistent environments and enables scaling.

Team 1

TOOLS :

- **Terraform**
  - Open-source tool to define and provision infrastructure using a declarative configuration language.

Team 1



Team 1

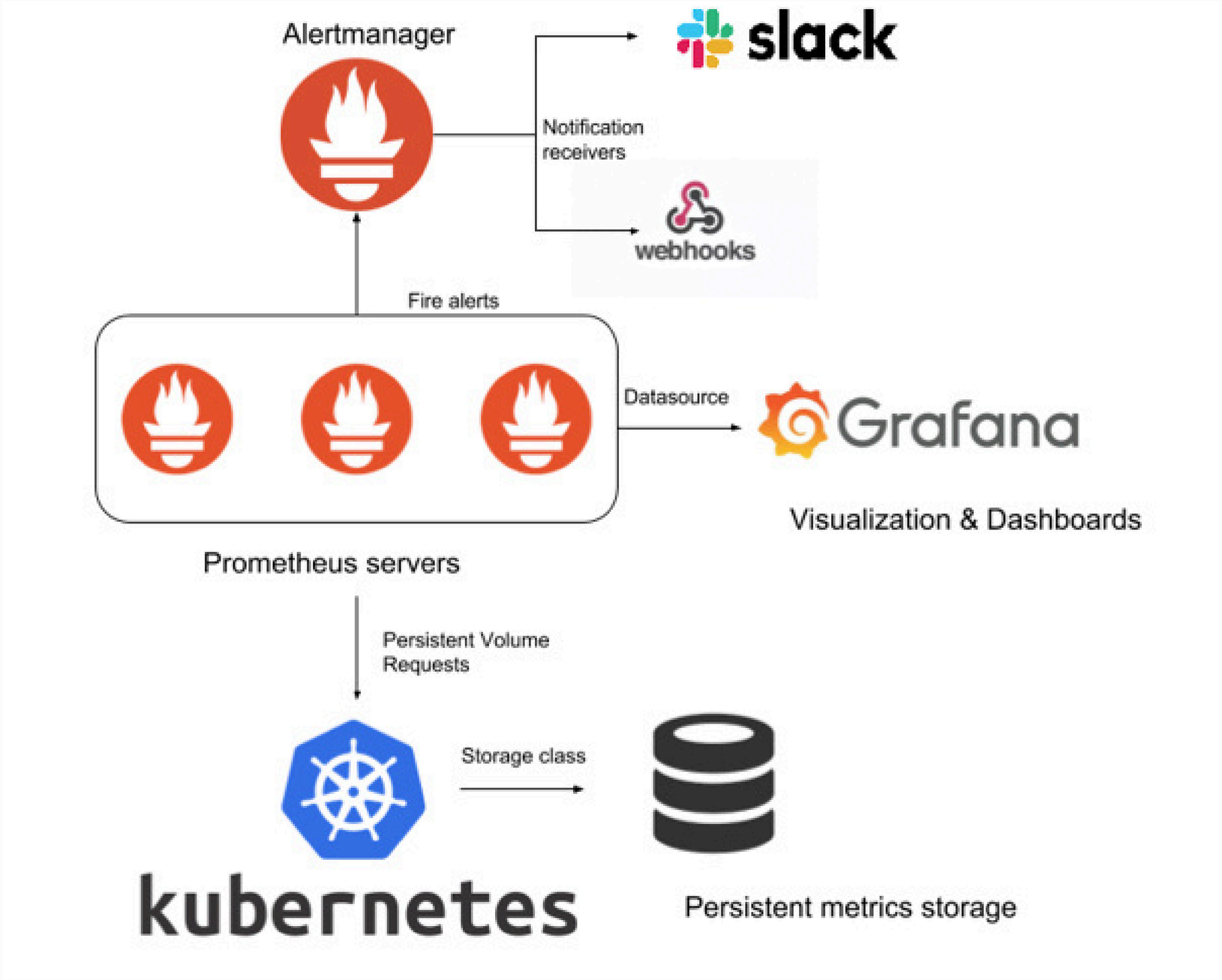
6-MONITORING AND OBSERVABILITY

- **Monitoring tracks application and infrastructure performance, while observability helps debug and understand complex systems.**

Team 1

PROMETHEUS

- **Features**:
  - Collects metrics from configured targets at regular intervals.
  - Stores data as a time-series: metrics + timestamp.
  - Includes an alerting system that integrates with tools like **Alertmanager**.
- Best For Monitoring system performance, application metrics, and infrastructure


Alertmanager
slack
Notification
receivers
webhooks
Fire alerts
Datasource
Grafana
Visualization & Dashboards
Prometheus servers
Persistent Volume
Requests
Storage class
kubernetes
Persistent metrics storage

Team 1

GRAFANA

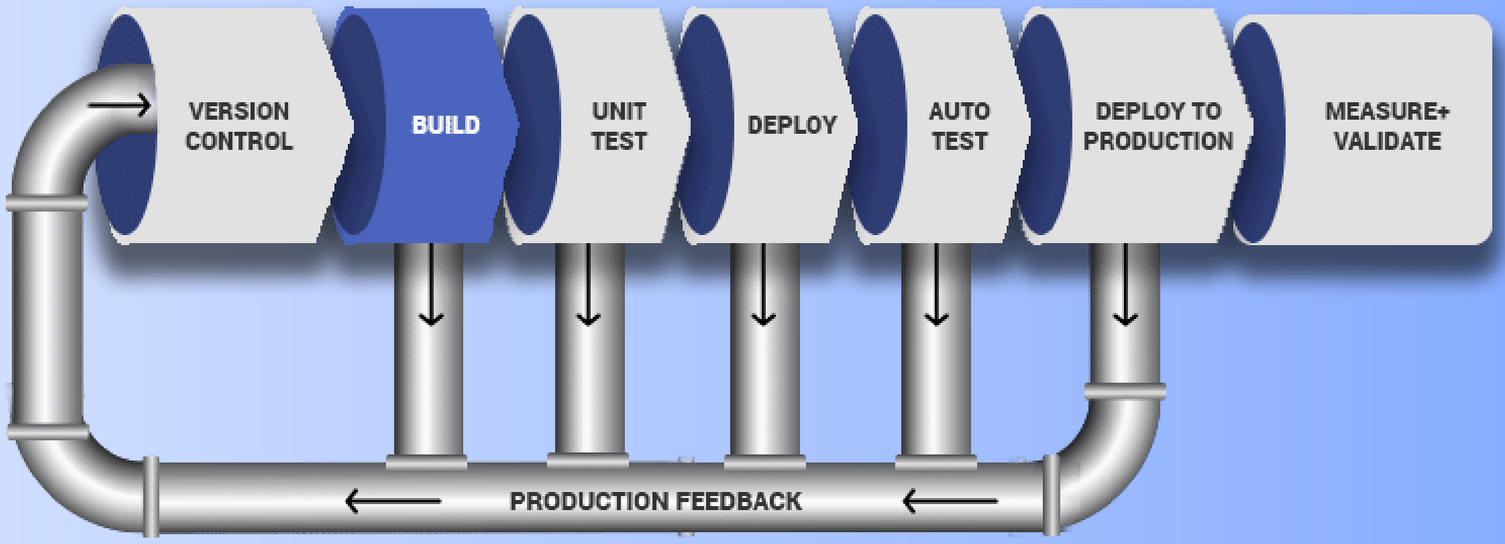
- **Features**:
  - Integrates with Prometheus and other data sources.
  - Builds interactive, customizable dashboards.
- Best For Visualizing Prometheus data and creating real-time monitoring dashboards.

Team 1

GRAFANA DASHBOARD

Team 1

OVERVIEW OF THE DEVOPS PIPELINE



Team 1

STEPS IN A TYPICAL DEVOPS WORKFLOW

1.Plan

Teams collaborate to define requirements, goals, and timelines for the project.

2.Code

Developers write, review, and commit code using version control systems

3.Build

The committed code is compiled and packaged into deployable artifacts (e.g., containers or binaries).

Team 1

CONT. STEPS IN A TYPICAL DEVOPS WORKFLOW

4.Test

Automated tests are executed to validate the functionality, security, and performance of the code.

5.Release

The tested build is prepared for deployment to production or pre-production environments.

6.Deploy

The release is deployed to the production environment using automated processes.

Team 1

CONT. STEPS IN A TYPICAL DEVOPS WORKFLOW

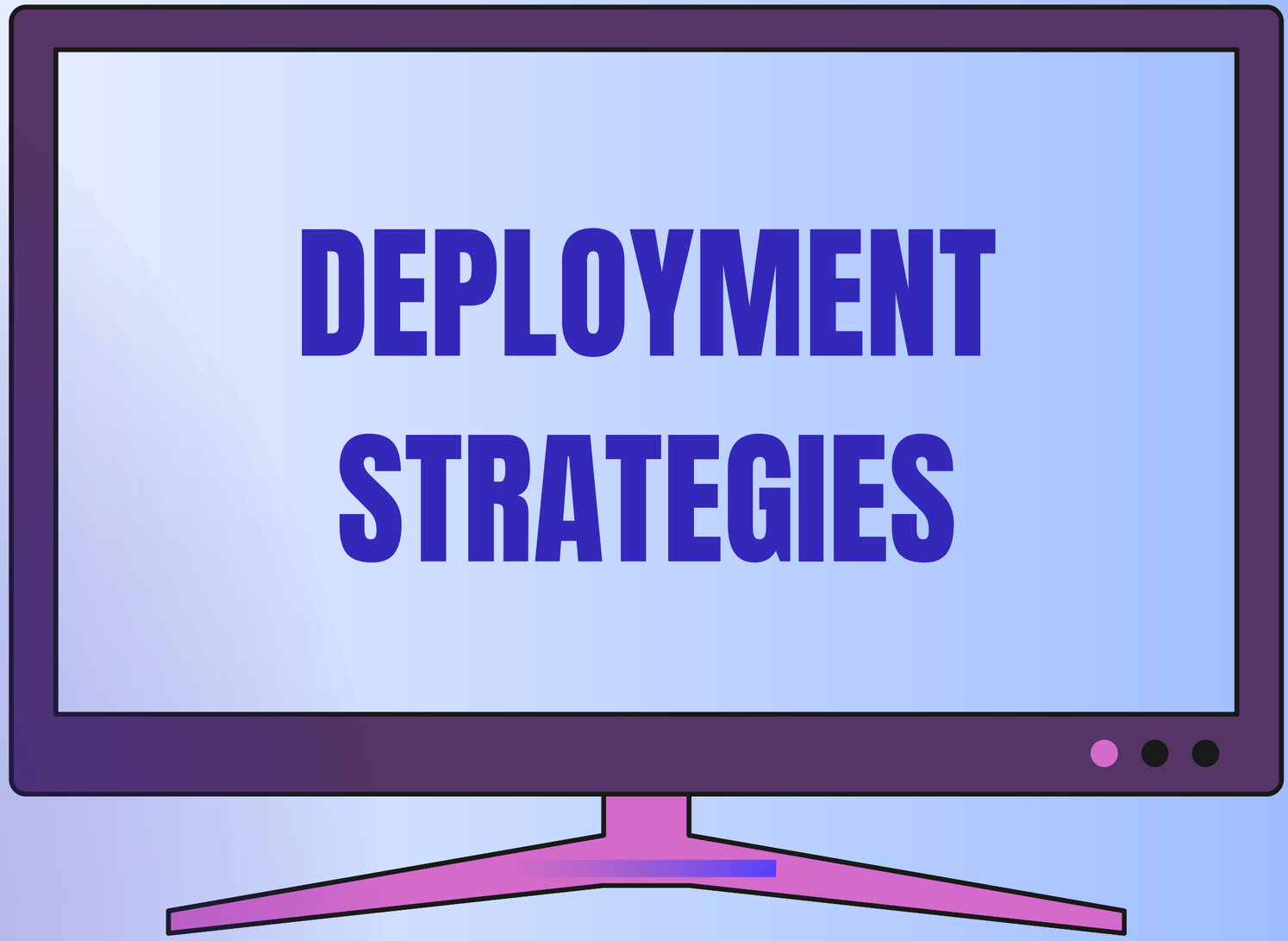
7.Operate

Once deployed, the application is monitored to ensure its health, availability, and performance.

8.Monitor and feedback

Continuous monitoring provides real-time insights, and user feedback is collected for improvements.

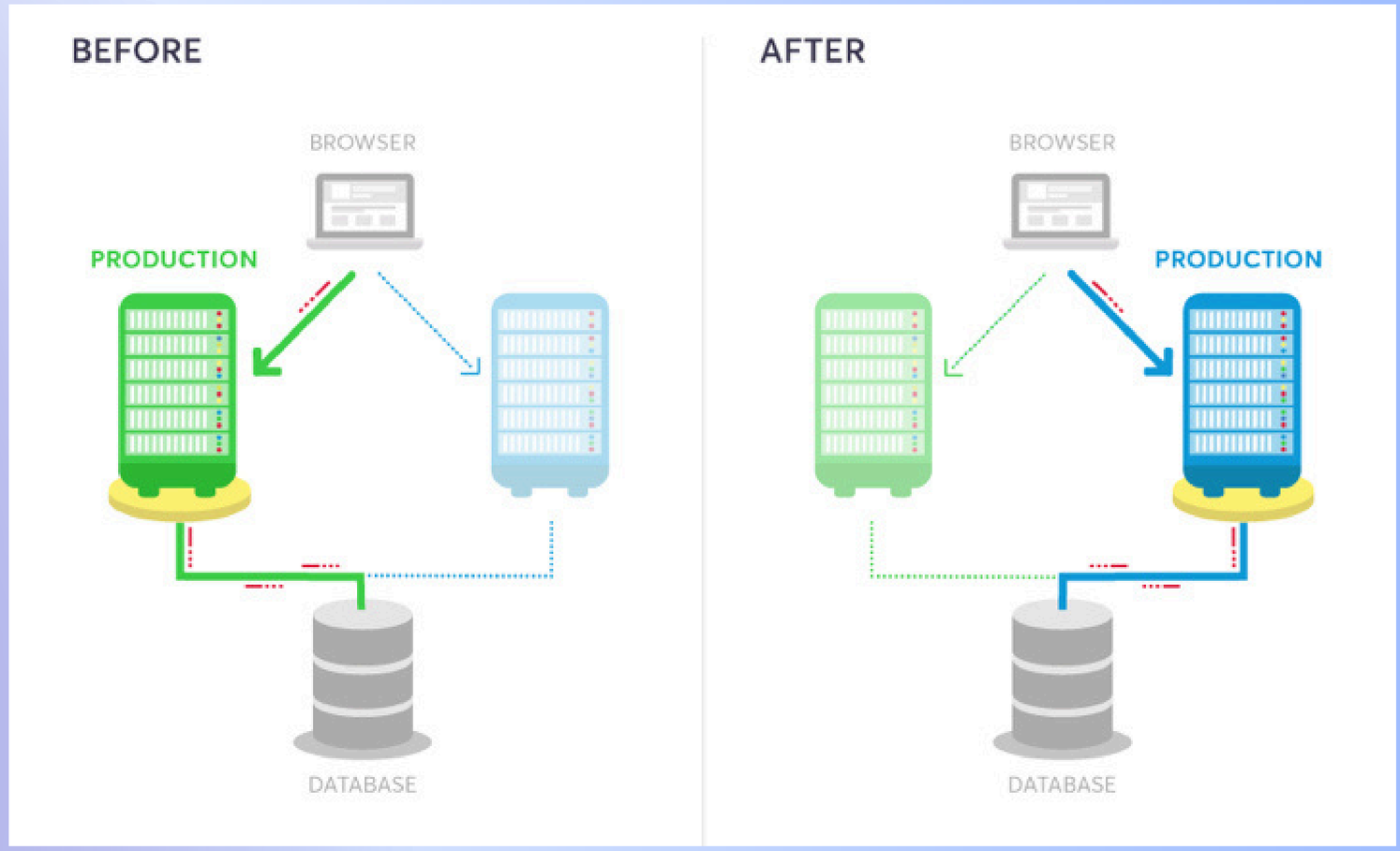
Team 1



Team 1

DEPLOYMENT STRATEGIES

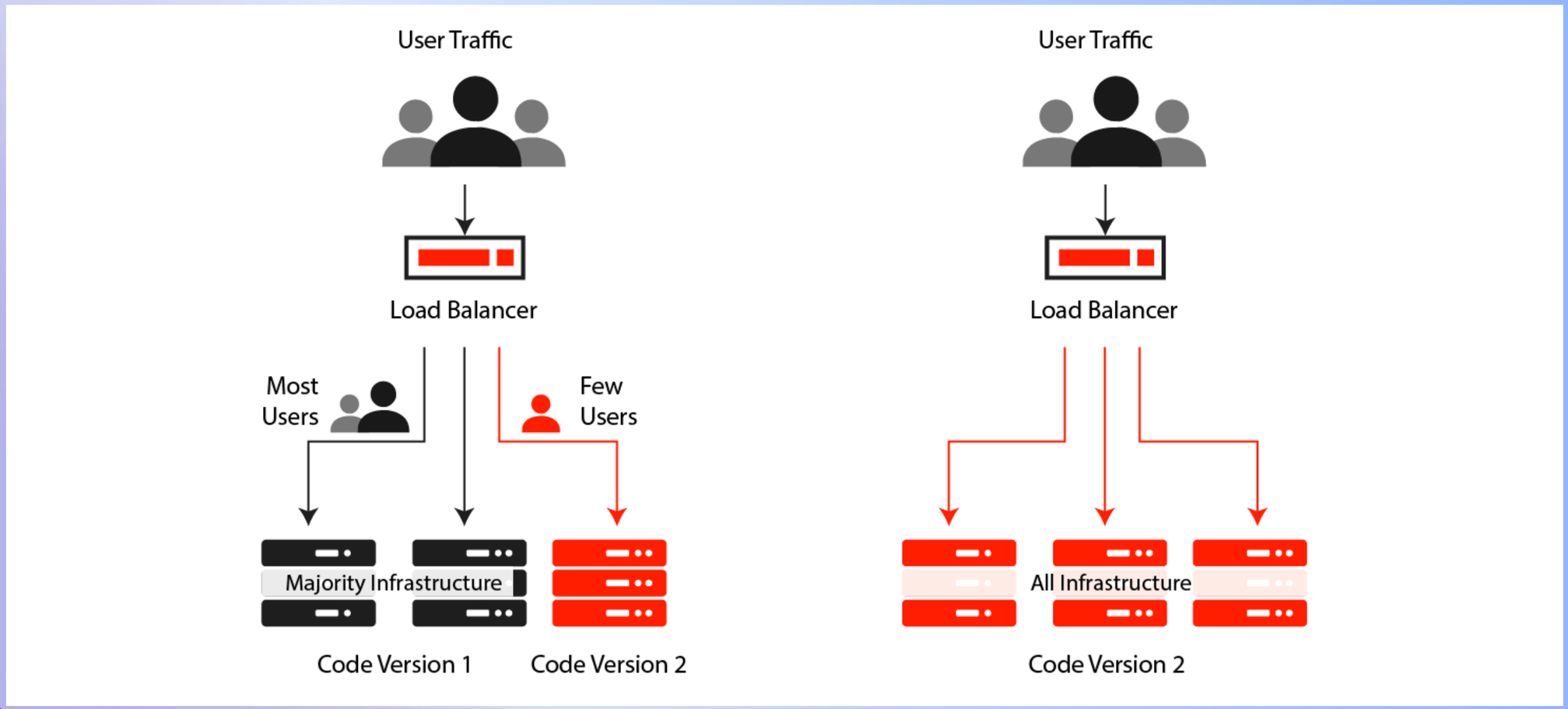
1.Blue-Green Deployment



Team 1

DEPLOYMENT STRATEGIES

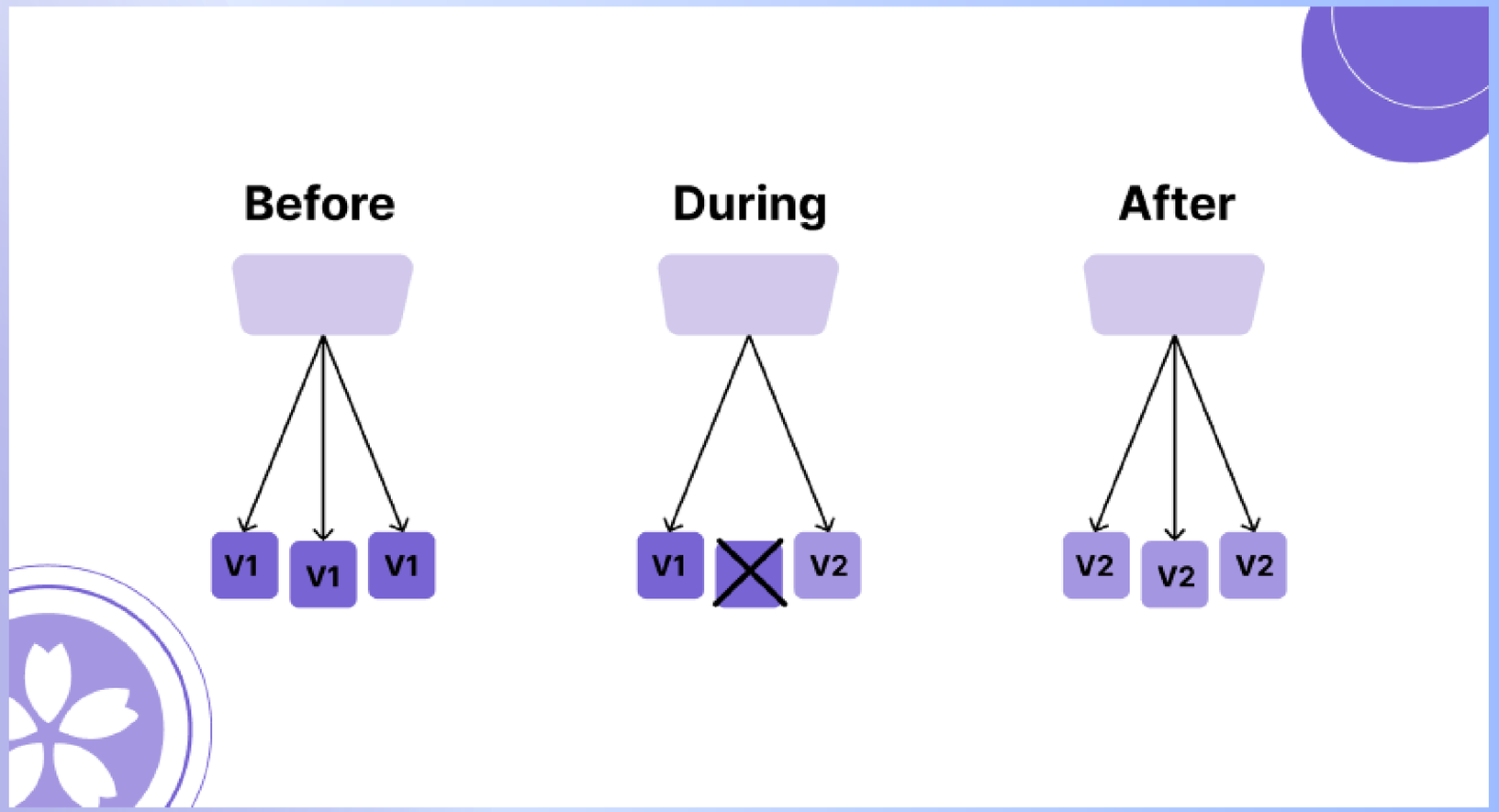
2.Canary Deployment



Team 1

DEPLOYMENT STRATEGIES

3.Rolling Deployment



Team 1

DEPLOYMENT STRATEGIES

4.A/B Testing

Team 1

COMMON CHALLENGES IN DEVOPS ADOPTION

1. Cultural Resistance
2. Integration of Legacy Systems
3. Lack of Skilled Personnel
4. Tools Integration Issues
5. Security Concerns

Team 1

SOLUTIONS TO OVERCOME THESE CHALLENGES

1. Fostering a DevOps Culture.
2. Gradual Integration of Legacy Systems
3. Building a Skilled Workforce
4. Choosing the Right Tools
5. Implementing DevSecOps

Team 1

CHANGE MANAGEMENT IN DEVOPS

1. Challenges in Change Management
2. Resistance to new tools and practices.
3. Lack of clear communication and understanding between teams.
4. Employees' reluctance to shift from manual processes to automated ones.
5. Effective Change Management Strategies
6. Leadership Involvement: Senior leadership must drive change and demonstrate its value.

Team 1

ORGANIZATIONAL CULTURE AND DEVOPS

- Organizational Culture in DevOps: Foster collaboration, shared responsibility, and continuous improvement.
- Aligning Culture with DevOps: Promote transparency, encourage a growth mindset, and reward collaboration.

Team 1

THE FUTURE OF DEVOPS

Team 1

EMERGING TRENDS IN DEVOPS

Microservices & Containers:
Modular, cloud-native application architectures enabling agile development and deployment.

AI/ML Integration:
Leveraging artificial intelligence and machine learning to automate and optimize DevOps processes.

Serverless Computing:
Event-driven, scalable infrastructures reducing operational overhead and enhancing flexibility.

Team 1

DEVOPS AND CLOUD-NATIVE APPLICATIONS

- **Rapid Scaling :**
  Cloud-native apps scale effortlessly to meet dynamic user demands.
- **Continuous Deployment:**
  Streamlined CI/CD pipelines enable frequent, reliable software releases.
- **Resilient Architecture:**
  Microservices and containers provide failover mechanisms for high availability.

Team 1

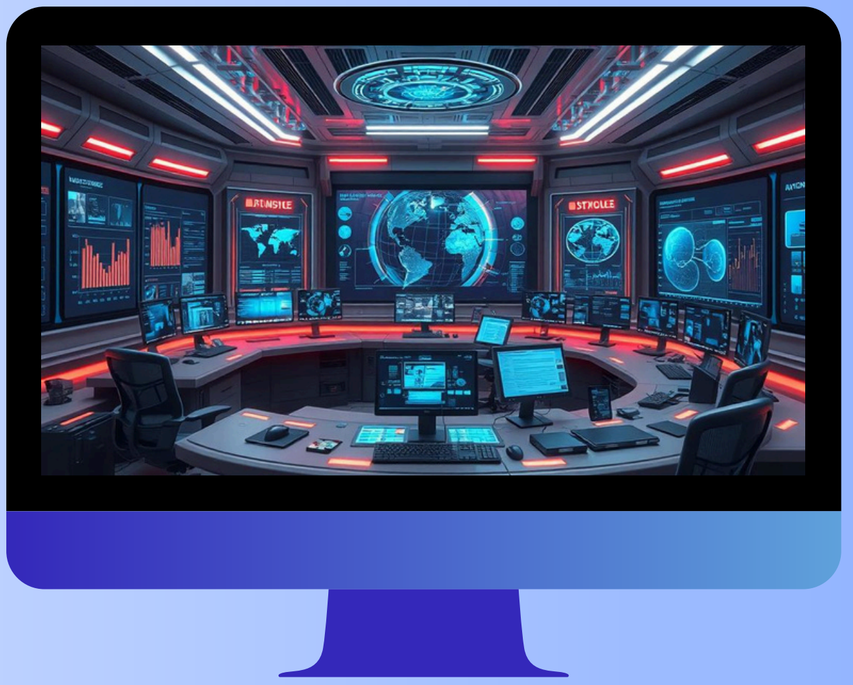
AI AND AUTOMATION IN THE FUTURE OF DEVOPS

1.Automated Infrastructure

2.Predictive Maintenance

3. Autonomous Testing

4.Intelligent Monitoring



Team 1

FINAL THOUGHTS ON DEVOPS EVOLUTION

Team 1

EVOLVING TOOLS AND TECHNOLOGIES IN THE DEVOPS LANDSCAPE

Cloud Platforms :
AWS, Azure, Google Cloud, and others

Containerization:
Docker and related orchestration tools

CI/CD Pipelines:
GitHub Actions

Team 1

KEY TRENDS IN THE EVOLUTION OF DEVOPS

1.Continuous Evolution

DevOps will adapt to emerging technologies and user demands.

2.Increased Automation:

AI and ML will streamline and optimize DevOps workflows.

3. Secure by Design:

Security and compliance will be integral to DevOps practices from the start.

4.Collaborative Ecosystems:

DevOps will foster cross-functional teams and encourage open innovation.

CONCLUSION

DevOps is evolving with AI-powered automation tools improving workflows and collaboration, while technologies like GitOps and Kubernetes shape the future. The demand for skilled DevOps professionals will rise, driving transformations across industries like manufacturing, healthcare, and finance.

ANY QUESTIONS ?

Team 1

THANK YOU!

Team Members:

Hagar Mahmoud Elmadany

Ahmed Mahmoud Elgendy

Esraa Tarek Fouda

Anas Nashat Ahmed

Amal Mohsen Mohamed

Rafat Gomaa Rafat