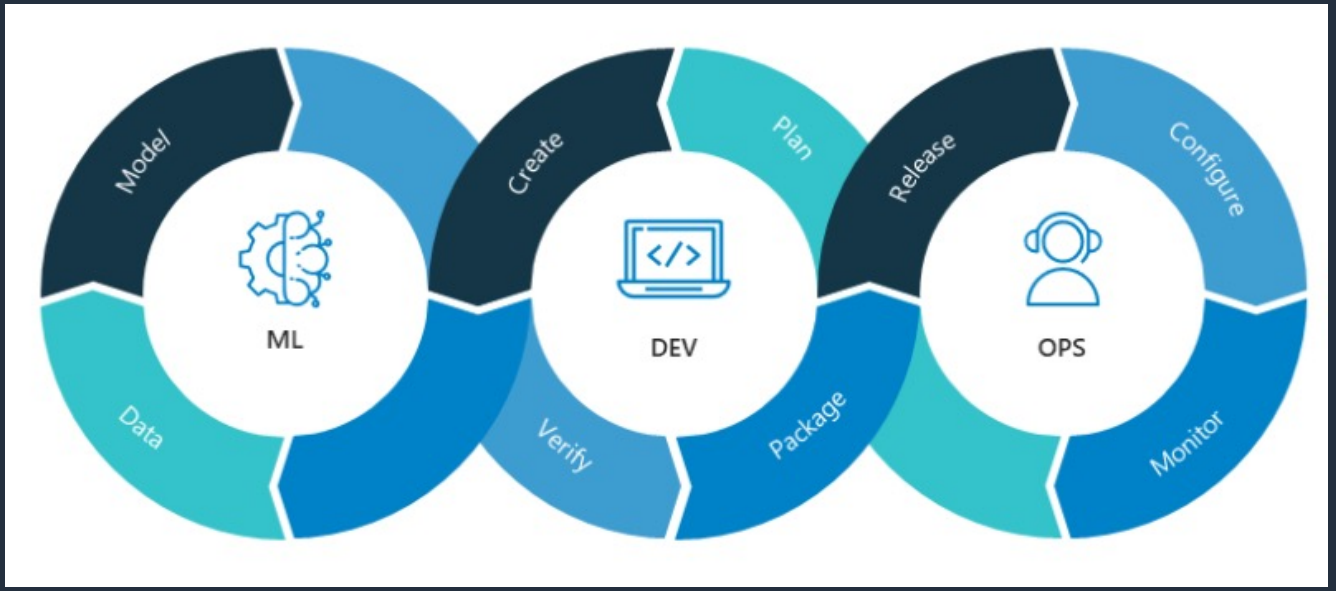
Practical MLOps for Data Scientists & DevOps Engineers

About the course



Why this course is different ?

We thank you for choosing us!!

Curated by Industry Experts

Created with the approach to take the practitioner from Academia to Realtime business use case

Best Practices in Production Usage

Notebooks and Scripts to use it on your client projects with little modifications

Gain the most sought-after skillset in Data Science

About Instructor

- **Murthy** is a Data Science Manager working in a Top Consulting Firm and sharing his knowledge through Manifold AI Learning. His Day-to-Day work includes:
  - Leading a Team of 25 People and is responsible for End-to-End Implementation of Data Science, Machine Learning & Deep Learning Projects
  - Manage the End-to-End Pipeline starting from Experimentation to Operationalization of Machine Learning Models in the Production Environment, including the maintenance of Machine Learning Models.
  - Worked on Various domains like – Retail, Healthcare, Banking & Insurance.

- **Impacts Made:**
  - Successfully Led & delivered multiple Data Science Projects from POC to Production with overall revenue close to 100m$
  - Leader on MLOps Approach in my Organization
  - Author of 10 Top selling e-learning content on Data Science
  - Trained more than 50k learners through Live Bootcamps
  - Top Instructor on Data Science with Manifold AI Learning

Share your ML Projects & Your Experimentation by Tagging us on LinkedIn

Learn with Expert Live

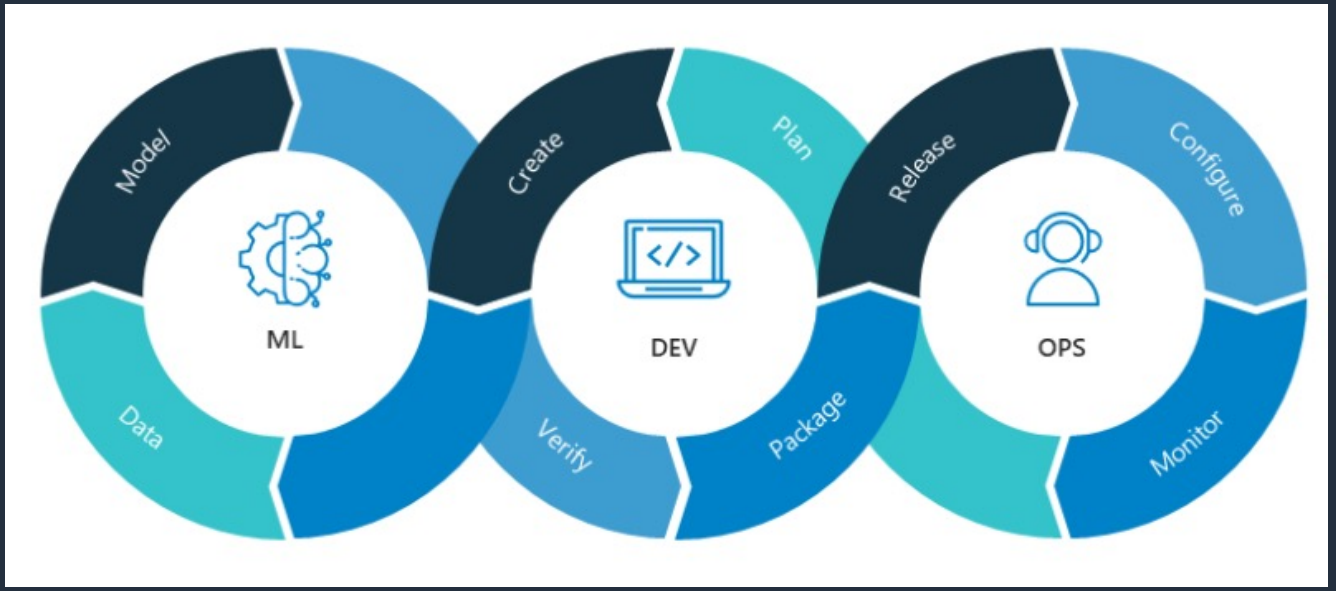
Bootcamp for Limited Learners

Link - https://www.manifoldailearning.in/courses/MLOps---LLMOps-AIOps-Bootcamp---A-Private-Community-65c63808e4b09cf855132de4

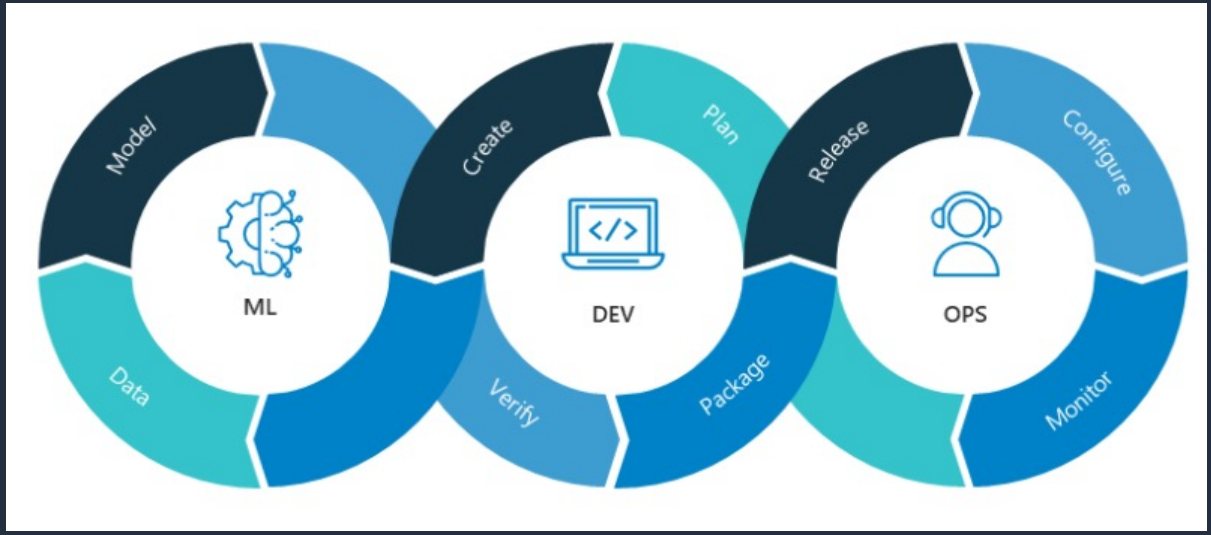
We are Excited to see you on our class!!

Practical MLOps for Data Scientists & DevOps Engineers

What & Why MLOps ?



What is MLOps ?



MLOps stands for Machine Learning Operations

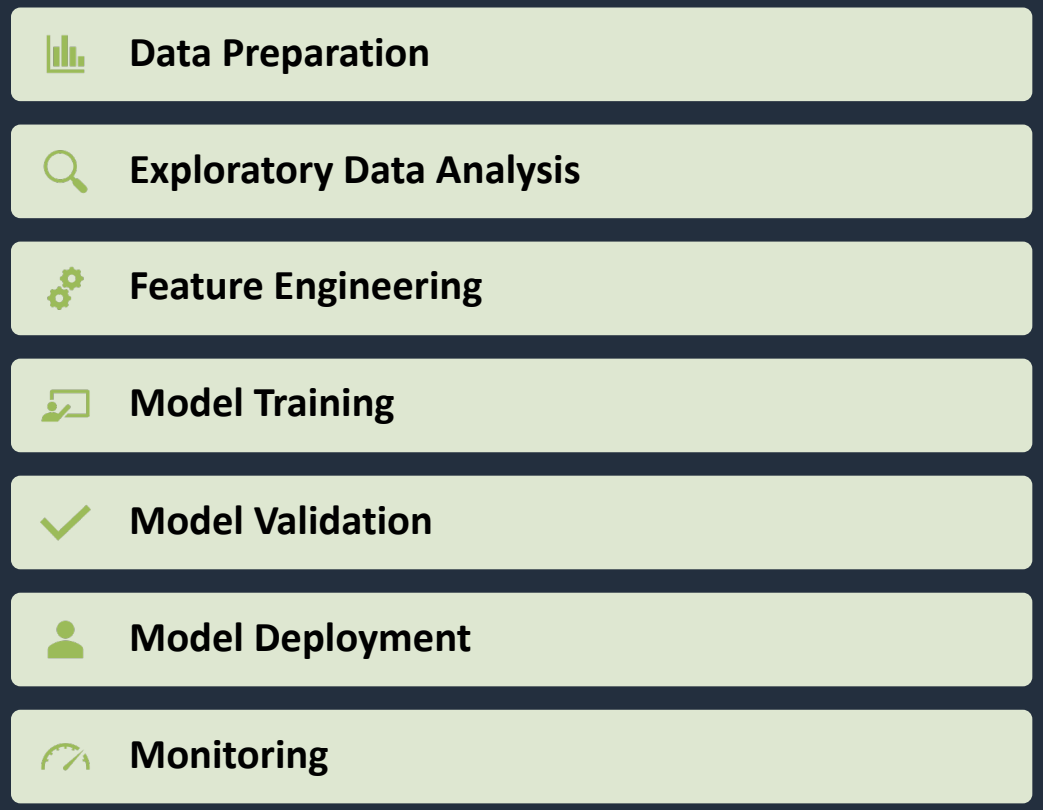
MLOps is a core function of Machine Learning engineering, focused on streamlining the process of taking machine learning models to production, and then maintaining and monitoring them

MLOps is a collaborative function, often comprising data scientists, devops engineers, and IT

Who are all Key People in a Machine Learning Projects ?

- Data Engineer
- Data Scientist
- ML Engineer
- Business Stakeholder
- Data Governance Officer

Process of Machine Learning



Why MLOps are the Need Right Now ?

- Building the Models for Personal Project is different than the one built for Organization
- Take care of Dependencies
- Communication between Multiple Teams
- Managing the Model Risks
- Manage the Infrastructure for Scale

Solution ?

A Streamlined approach for Machine Learning Projects → MLOps

Some Guiding Principles

- Always Keep Business Goals in mind
- Have a Data Centric Approach
- Modular Approach in Implementation
- Mature in a way such that Process should guide automation

Next Video :

Fundamentals of MLOps

Goal of MLOps on Use Case

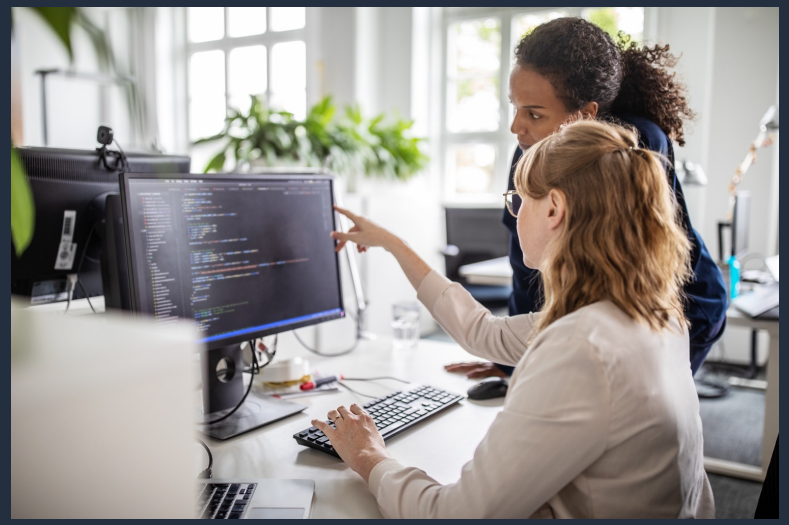
Take care of End To End Process of Data Science from Data Preprocessing to Deployment

MLOps Fundamentals

Environments & Assets in Machine Learning Projects

Environments

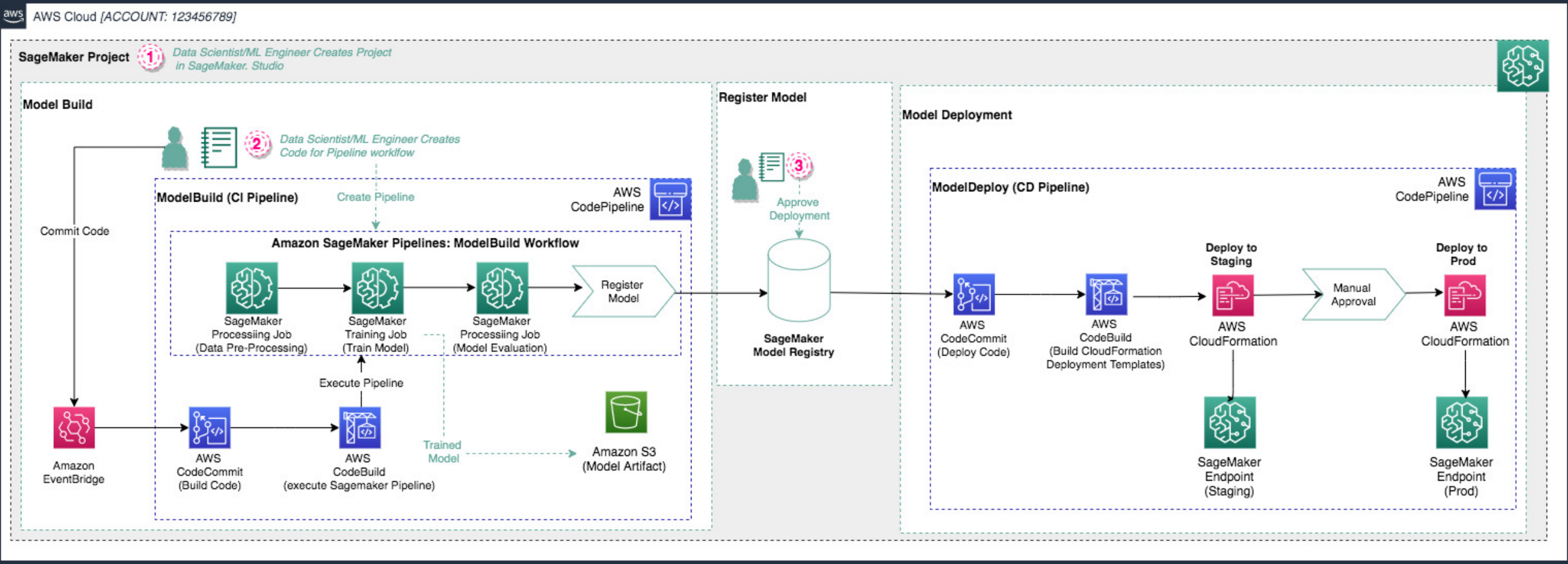
1. Dev - can be accessed by people -no guarantee of quality - low risk
2. Stage - can be accessed by subset of people - test of dev code - quality is matched with prod - treated as a live product
3. Prod - Very least number of people has access (tight control on access) - Live Product -highest quality in assets - business critical



Assets

- **ML Project Code -** Stored in Version Control Repository (Git)
- **Models -** Model & Model Artifacts with Model Registry
- **Data -** Separate environments for storage.
  - Dev – Temporary
  - Stage – Almost same as Prod – updates on frequent basis (Monthly, Quarterly, etc.)
  - Prod – Reliable and Fresh

CI CD Pipeline



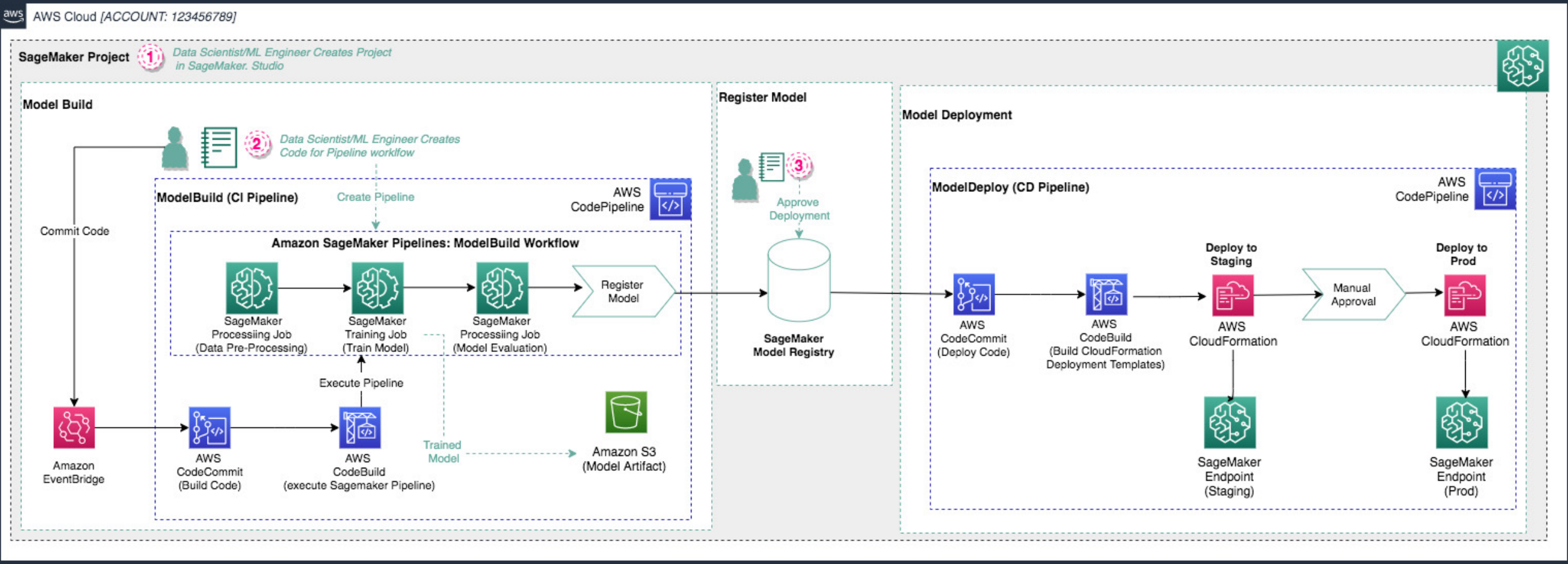
Next Video :

Deep Dive into MLOps

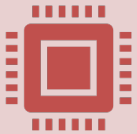
MLOps Deep Dive

Understanding Process

CI CD Pipeline



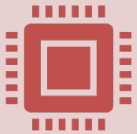
MLOps



Data : Read-only access to production data and read-write access to a separate dev storage environment to develop and experiment with new features and other data tables.

EDA : Exploration of data, prepare the data for the Applying on model

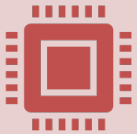
MLOps - Contd.



Repository : A Place to store all the source code with version enabled

Feature Table : Reading the raw data table and feature tables and writes to tables in Feature Store.

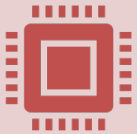
MLOps - Contd.



Model Training : Create Model Training Pipeline, involving – Training & Tuning, Evaluation and Model Output

Commit the Repo: Update the Dev Branch with updated code

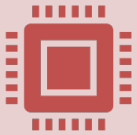
MLOps - Contd.



Dev Environment: Perform Integration Test

Merge the Code : Merge with Stage Branch

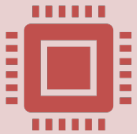
MLOps - Contd.



Stage Environment: Perform Integration Test (CI)

Merge the Code : Run Integration testing with latest data

Deployment



Deployment Training : Create Model Training Pipeline, involving – Training & Tuning, Evaluation and Model Output

Commit the Repo: Update the Dev Branch with updated code

Continuous Deployment

- Compliance Checks
- Comparison between Stage Vs Prod
- Deploy Model to Prod

Deployment Strategies

- Online Serving (REST APIs)
- Batch/Streaming inference

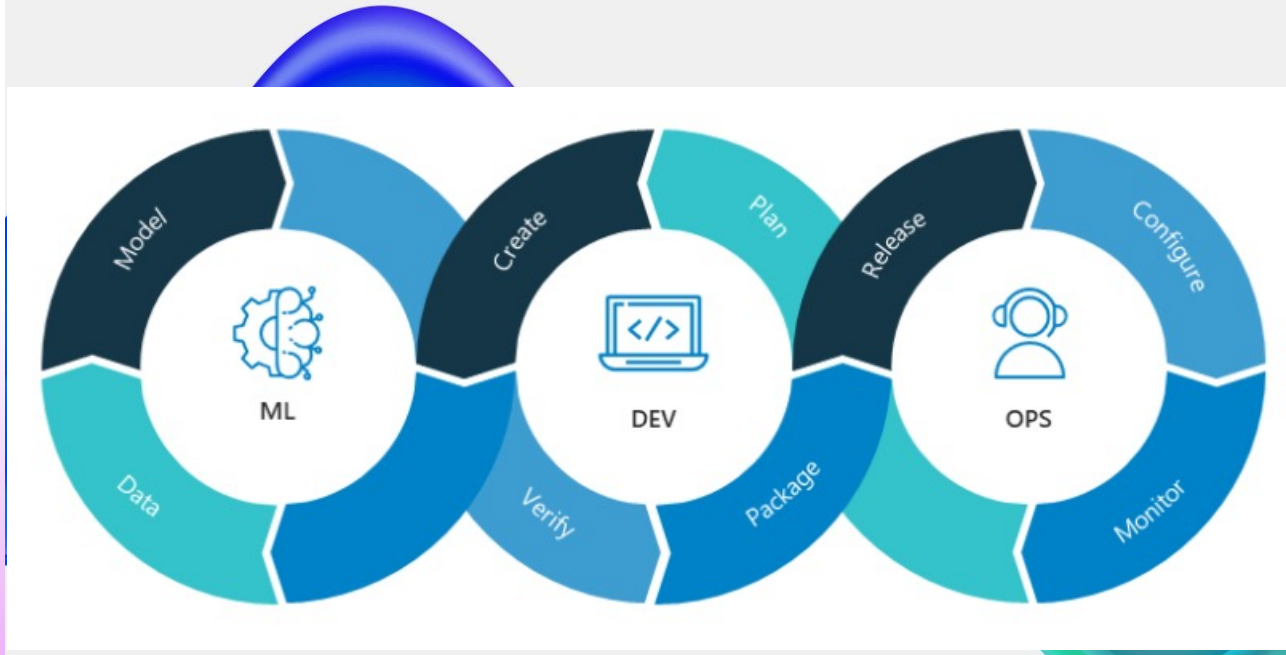
Monitoring

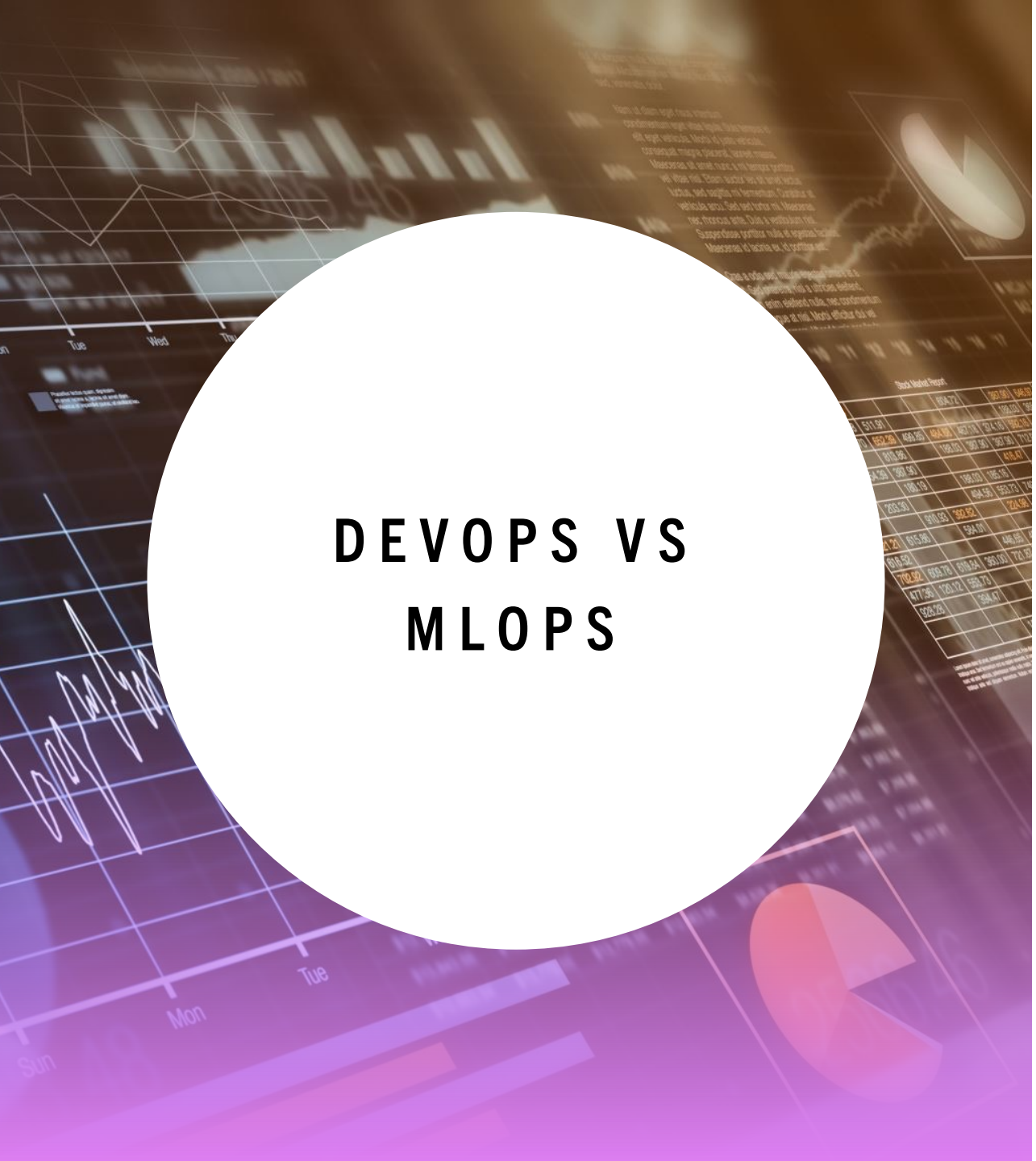
- Data Ingestion
- Accuracy & Data Drift
- Monitor Metrics
- Retrigger Model Training

Retrain of Models

- Scheduled
- Trigger based on Events

WHY DEVOPS IS NOT SAME AS MLOPS





DEVOPS VS MLOPS

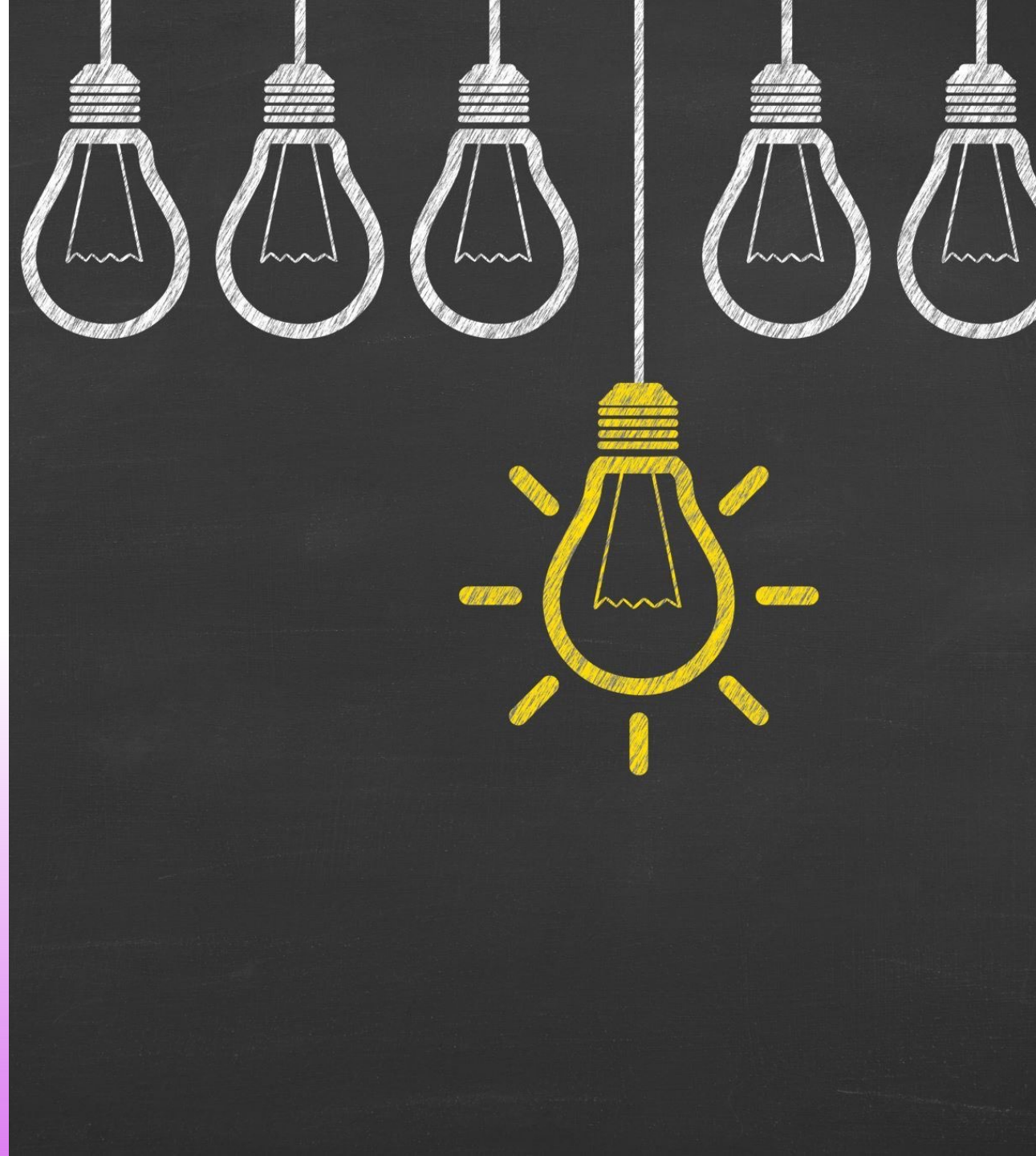
- Although DevOps and MLOps are well-known terms among data and analytics professionals, many business executives do not fully grasp the magnitude of the Machine Learning revolution sweeping across industries, nor do they understand the significance of these terms for their organizations in effectively managing ML.
- As a result, many organizations struggle to operationalize their ML efforts because they mistakenly apply their well-established DevOps principles to ML without recognizing the differences and the role these practices play within a business strategy.

QUICK RECAP ON DEVOPS

- In recent decades, DevOps has gained significant popularity as an approach to effectively managing software development.
- The term "DevOps" was introduced by Patrick Debois, a renowned software developer and consultant, in 2009.
- The origins of DevOps can be traced back to the early 2000s, when companies started realizing the importance of enhanced collaboration and communication between software developers and IT operations teams.

WHY DEVOPS IS NOT SUITABLE FOR MACHINE LEARNING

- While DevOps may be suitable for managing conventional software development, it is not the most optimal approach for the development and operationalization of ML models.
- ML models possess the capability to learn from data and make predictions or classifications without explicit programming, leading to significant advancements in fields like image recognition, natural language processing, and predictive analytics.
- The development and deployment of ML models necessitate specialized infrastructure, efficient data management, robust processing capabilities, and expertise in specialized skills and knowledge to ensure effective implementation.

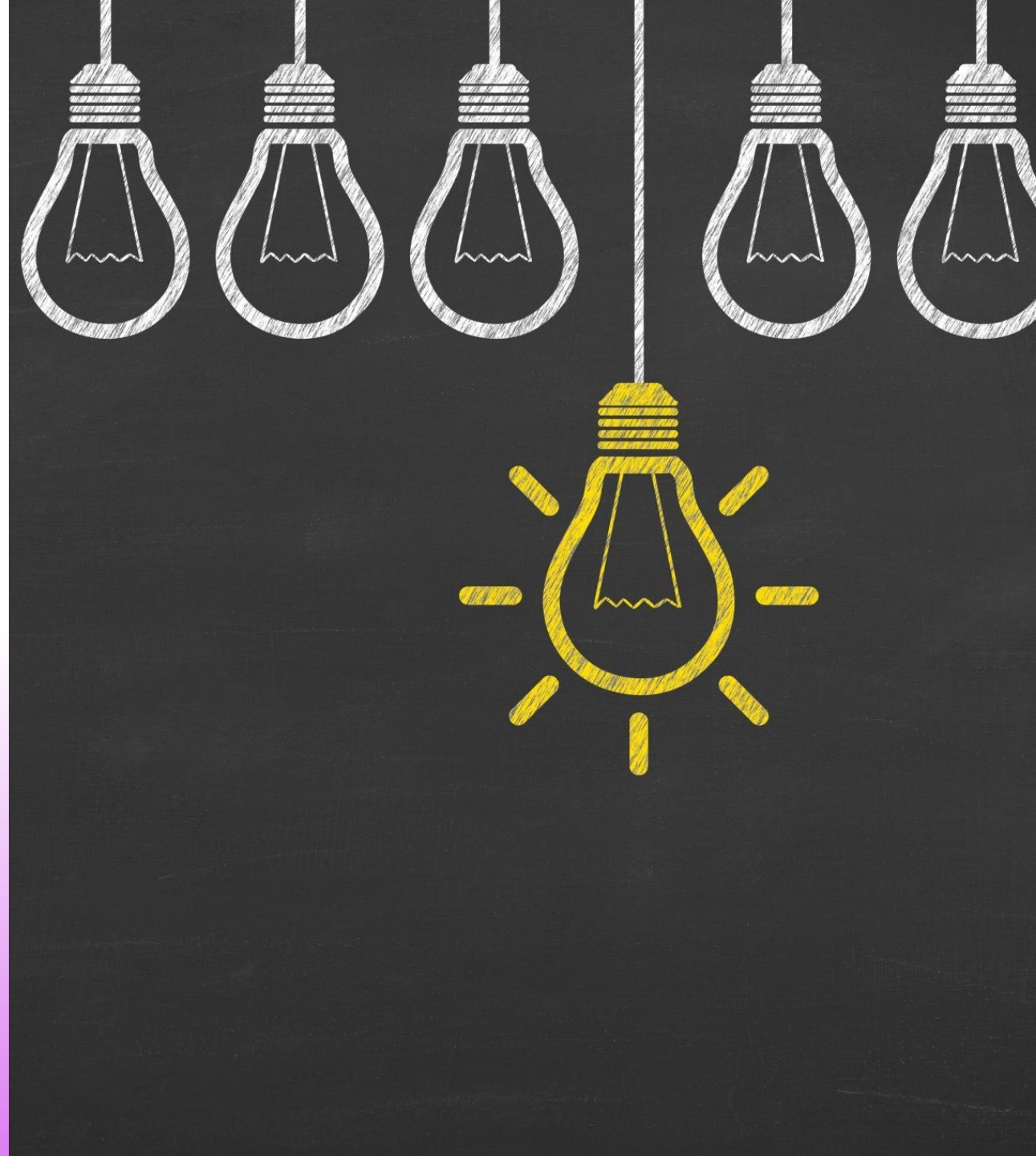


UNIQUE CHALLENGES OF MACHINE LEARNING

- ML development is characterized by its iterative and experimental nature, distinguishing it from traditional software development practices. and knowledge: ML requires specialized skills and knowledge in areas such as data engineering, data science, statistics, and math - and how to apply these in the context of a business use case
- ML necessitates specialized skills and expertise in areas such as data engineering, data science, statistics, and mathematics, and their application within specific business use cases.
- ML models often demand specialized infrastructure and resources, including GPUs, TPUs, distributed computing systems, and tools for data storage and management.
- Once a model is developed, it must be effectively deployed and managed within a production environment.

THE UNIQUE CHALLENGES OF MACHINE LEARNING

- Effective data management and governance are crucial for the success of ML development because ML models heavily depend on extensive data for training and making accurate predictions. Therefore, ensuring proper data management and governance becomes paramount in the ML development process.



FINAL

DevOps - this is the way for Traditional Software development

MLOps - this is the highway - Needs more process oriented approach to overcome the challenges

THANK YOU

WHAT & WHY AWS

OVERVIEW OF AMAZON WEB SERVICES

- Amazon Web Services offers a broad set of global cloud-based products including compute, storage, databases, analytics, networking, mobile, developer tools, management tools, IoT, security, and enterprise applications: on-demand, available in seconds, with pay-as-you-go pricing
- This allows enterprises, start-ups, small and medium-sized businesses, and customers in the public sector to access the building blocks they need to respond quickly to changing business requirements

INTRODUCTION TO AWS

- In 2006, Amazon Web Services began offering IT infrastructure services to businesses as web services—now commonly known as cloud computing
- One of the key benefits of cloud computing is the opportunity to replace upfront capital infrastructure expenses with low variable costs that scale with your business
- With the cloud, businesses no longer need to plan for and procure servers and other IT infrastructure weeks or months in advance

- Cloud computing is the on-demand delivery of compute power, database, storage, applications, and other IT resources through a cloud services platform via the internet with pay-as-you-go pricing
- Cloud computing provides a simple way to access servers, storage, databases and a broad set of application services over the internet
- A cloud services platform such as Amazon Web Services owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application

SIX ADVANTAGES OF CLOUD COMPUTING

- Trade fixed expense for variable expense – Instead of having to invest heavily in data centers and servers before you know how you're going to use them, you can pay only when you consume computing resources, and pay only for how much you consume
- Benefit from massive economies of scale – By using cloud computing, you can achieve a lower variable cost than you can get on your own
- Stop guessing capacity – Eliminate guessing on your infrastructure capacity needs
- Increase speed and agility – In a cloud computing environment, new IT resources are only a click away, which means that you reduce the time to make those resources available to your developers from weeks to just minutes

SIX ADVANTAGES OF CLOUD COMPUTING CONTD.

- Stop spending money running and maintaining data centers – Focus on projects that differentiate your business, not the infrastructure
- Go global in minutes – Easily deploy your application in multiple regions around the world with just a few clicks

TYPES OF CLOUD COMPUTING

- Cloud computing provides developers and IT departments with the ability to focus on what matters most and avoid undifferentiated work such as procurement, maintenance, and capacity planning
- As cloud computing has grown in popularity, several different models and deployment strategies have emerged to help meet specific needs of different users
- Understanding the differences between Infrastructure as a Service, Platform as a Service, and Software as a Service, as well as what deployment strategies you can use, can help you decide what set of services is right for your needs

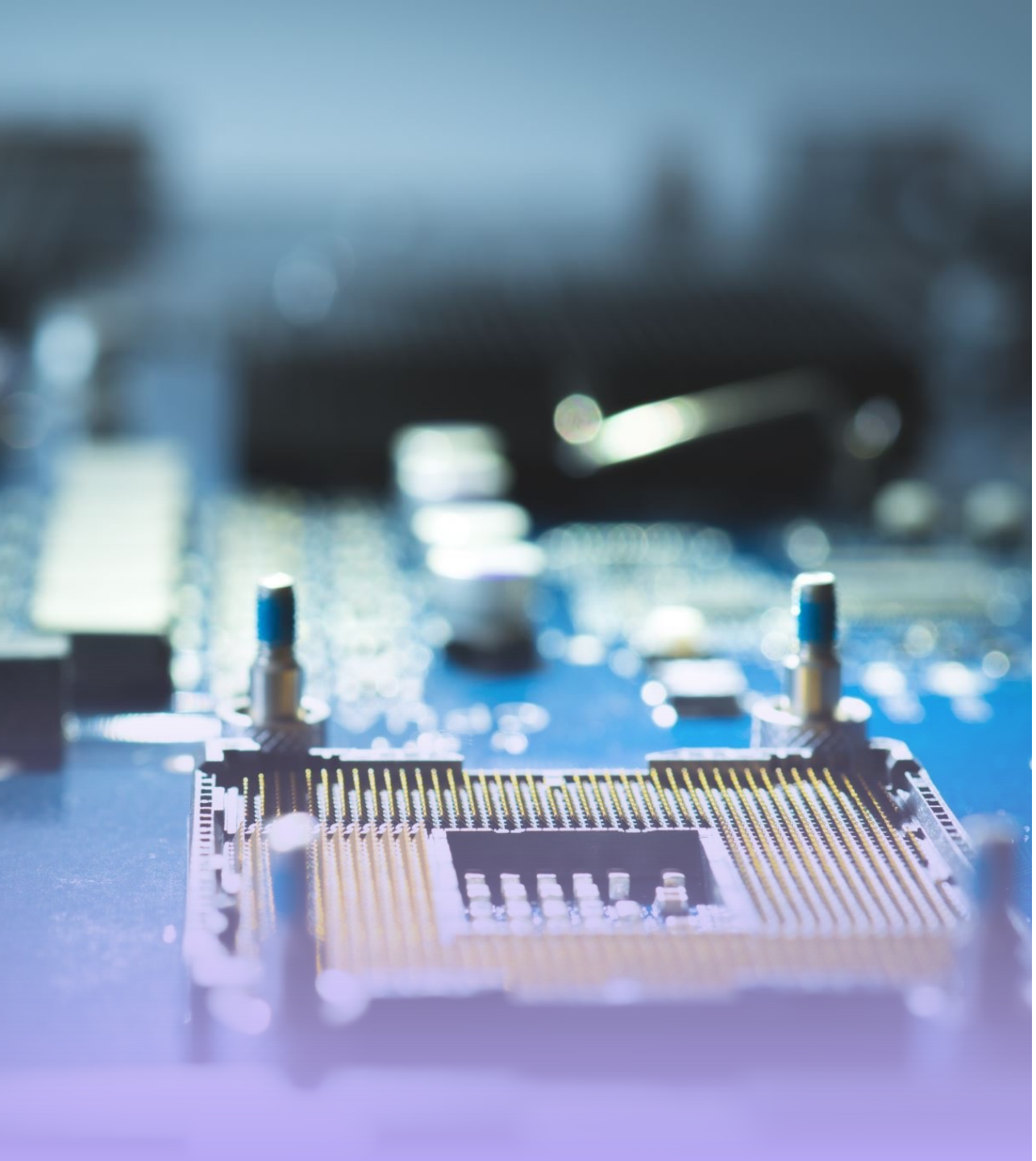
CLOUD COMPUTING MODELS

INFRASTRUCTURE AS A SERVICE

- Infrastructure as a Service contains the basic building blocks for cloud IT and typically provides access to networking features, computers , and data storage space
- IaaS provides you with the highest level of flexibility and management control over your IT resources and is most similar to existing IT resources that many IT departments and developers are familiar with today

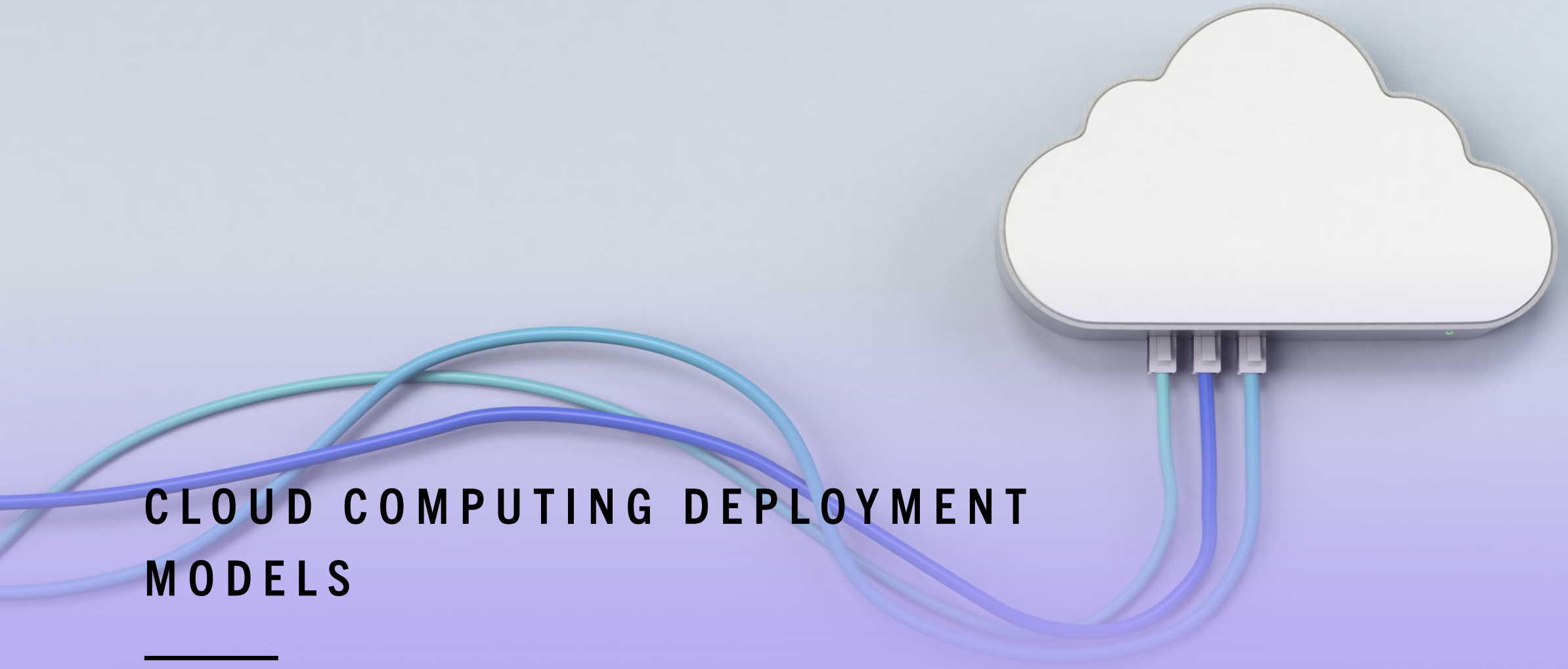
PLATFORM AS A SERVICE

- Platform as a Service removes the need for your organization to manage the underlying infrastructure and allows you to focus on the deployment and management of your applications
- This helps you be more efficient as you don’t need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application



SOFTWARE AS A SERVICE

- Software as a Service provides you with a completed product that is run and managed by the service provider
- With a SaaS offering you do not have to think about how the service is maintained or how the underlying infrastructure is managed; you only need to think about how you will use that particular piece of software
- A common example of a SaaS application is web-based email which you can use to send and receive email without having to manage feature additions to the email product or maintain the servers and operating systems that the email program is running on



CLOUD COMPUTING DEPLOYMENT MODELS

CLOUD

HYBRID

- A hybrid deployment is a way to connect infrastructure and applications between cloud-based resources and existing resources that are not located in the cloud
- The most common method of hybrid deployment is between the cloud and existing on-premises infrastructure to extend, and grow, an organization's infrastructure into the cloud while connecting cloud resources to the internal system
- For more information on how AWS can help you with your hybrid deployment, visit our Hybrid Cloud with AWS page

ON-PREMISES

- The deployment of resources on-premises, using virtualization and resource management tools, is sometimes called the “private cloud.”
- On-premises deployment doesn’t provide many of the benefits of cloud computing but is sometimes sought for its ability to provide dedicated resources
- In most cases this deployment model is the same as legacy IT infrastructure while using application management and virtualization technologies to try and increase resource utilization

GLOBAL INFRASTRUCTURE OF AWS

GLOBAL INFRASTRUCTURE

- The AWS Cloud infrastructure is built around AWS Regions and Availability Zones
- Availability Zones consist of one or more discrete data centers, each with redundant power, networking, and connectivity, housed in separate facilities
- These Availability Zones offer you the ability to operate production applications and databases that are more highly available, fault tolerant, and scalable than would be possible from a single data center

BENEFITS OF AWS SECURITY

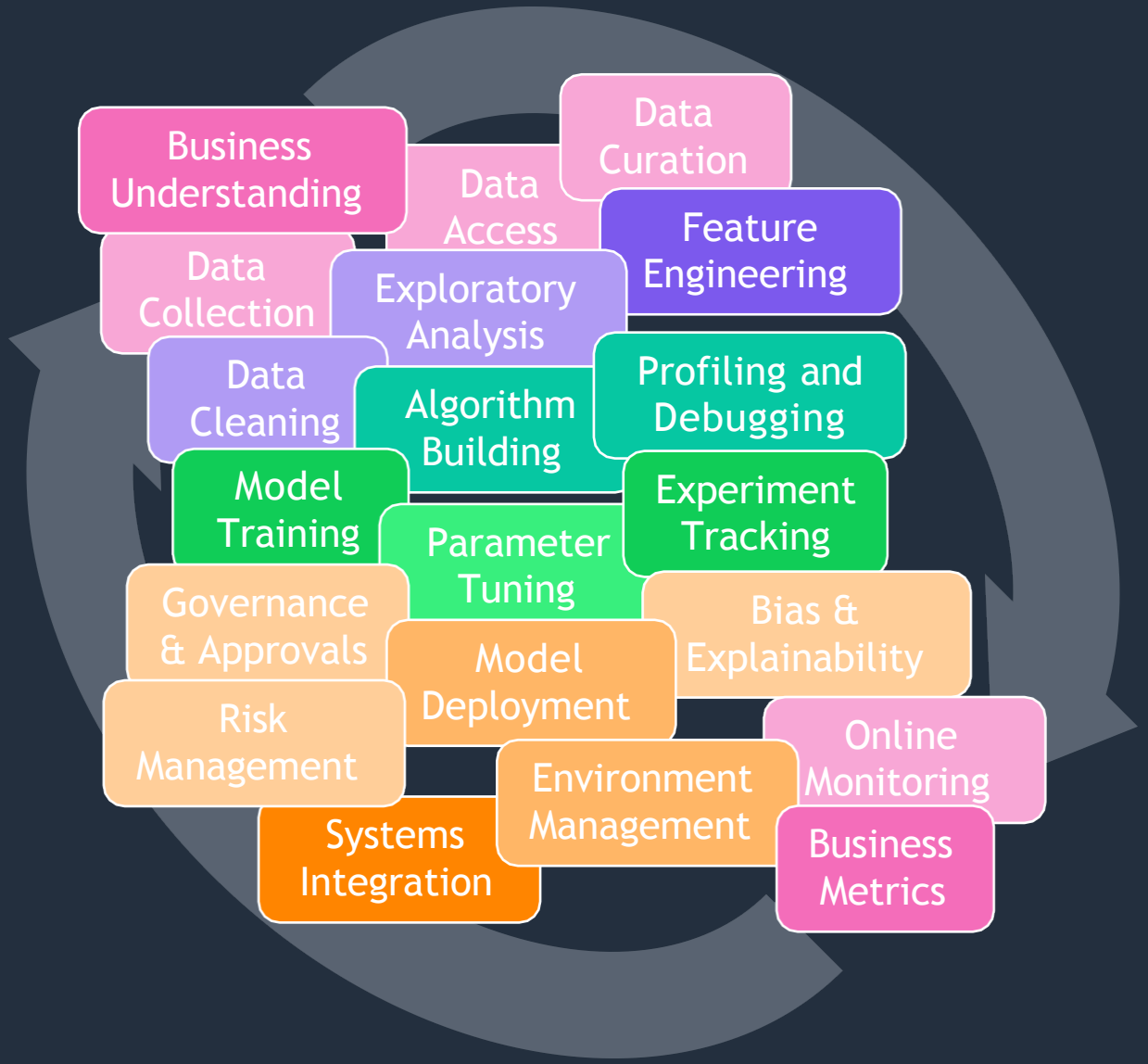
- Keep Your data safe — The AWS infrastructure puts strong safeguards in place to help protect your privacy
- Meet compliance requirements — AWS manages dozens of compliance programs in its infrastructure
- Save money: —Cut costs by using AWS data centers
- Scale quickly — Security scales with your AWS Cloud usage

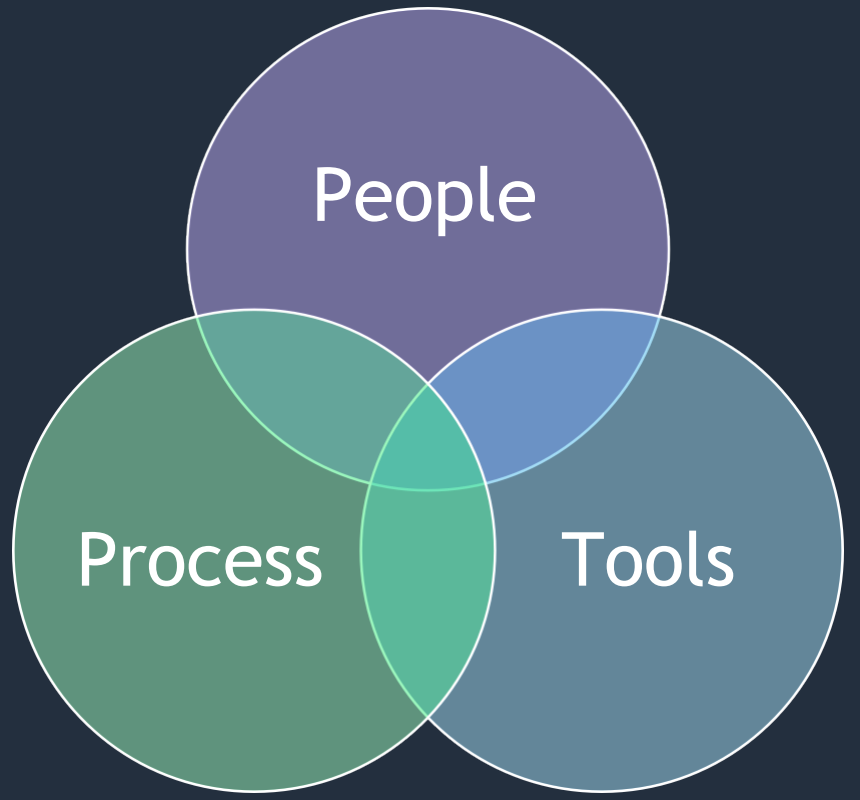
Technical Stack of AWS for MLOps & Machine Learning

AWS

Technical Stack of AWS for MLOps & Machine Learning

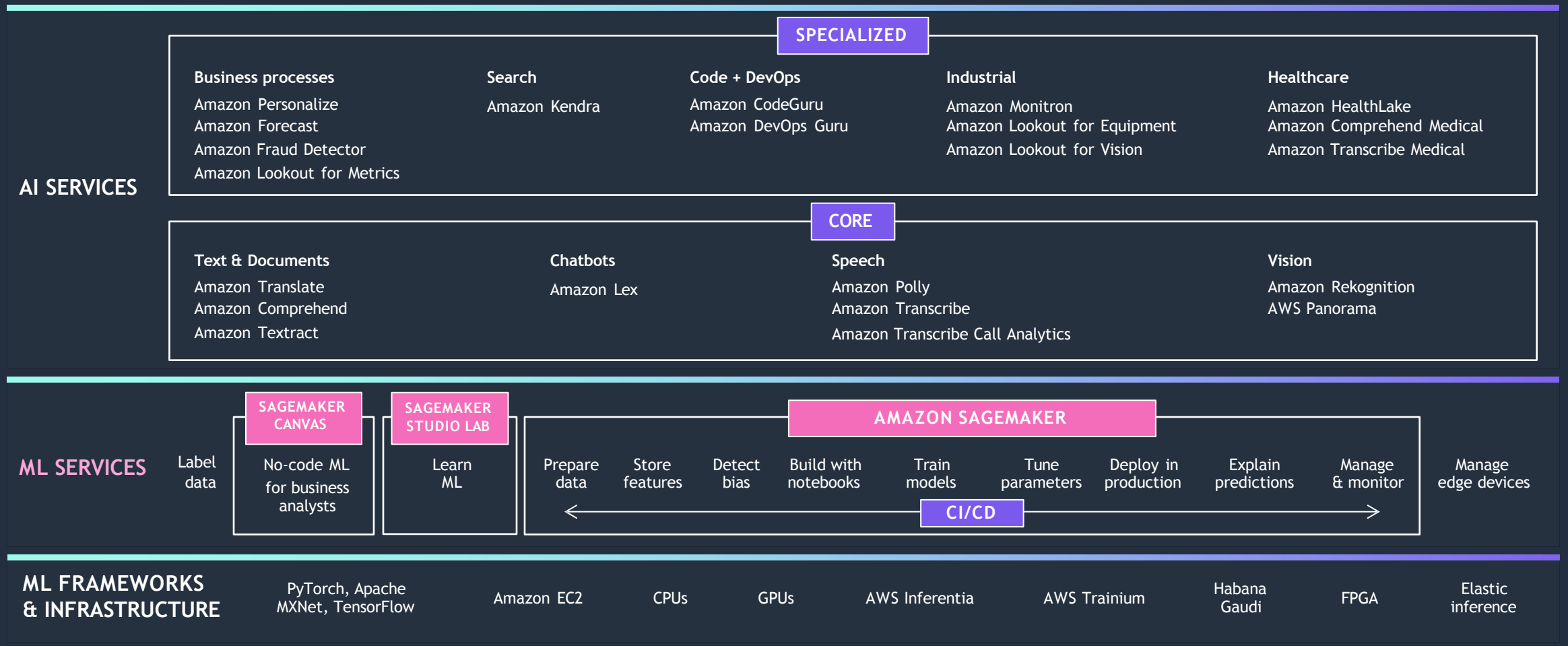
ML projects are multi-disciplinary and iterative





…In other words, “difficult”!

Amazon SageMaker and the AWS AI/ML Stack



Modular features for the end-to-end ML lifecycle

PREPARE

Geospatial
Visualize geospatial data

Ground Truth
Create high quality datasets for ML

Data Wrangler
Aggregate and prepare data for ML

Processing
Built-in Python, BYO R/Spark

Feature Store
Store, catalog, search, and reuse features

Clarify
Detect bias and understand model predictions

BUILD

Studio Notebooks & Notebook Instances
Fully managed Jupyter notebooks with elastic compute

Studio Lab
Free ML development environment

Autopilot
Automatically create ML models with full visibility

Built-in Algorithms
Integrated tabular, NLP, and vision algorithms

Bring Your Own
Bring your own container and algorithms

JumpStart
UI based discovery, training, and deployment of models, solutions, and examples

Local Mode
Test and prototype on your local machine

TRAIN & TUNE

Fully Managed Training
Broad hardware options, easy to setup and scale

Automatic Model Tuning
Hyperparameter optimization

Customization Support
Integrate with popular open source frameworks and libraries

Experiments
Track, visualize, and share model artifacts across teams

Managed Spot Training
Reduce training cost by up to 90%

Debugger and Profiler
Debug and profile training runs

Distributed Training Libraries
High performance training for large datasets and models

Training Compiler
Faster deep learning model training

DEPLOY & MANAGE

Fully Managed Deployment
Ultra low latency, high throughput inference

Real-Time & Serverless Inference
For steady and intermittent traffic patterns

Asynchronous Inference
For large payloads or long processing times

Batch Transform
For offline inference on batches of large datasets

Multi-Model Endpoints
Reduce cost by hosting multiple models per instance

Multi-Container Endpoints
Reduce cost by hosting multiple containers per instance

Shadow Testing
Validate model performance in production

Inference Recommender
Automatically select compute instance and configuration

Model Monitor
Maintain accuracy of deployed models

Kubernetes Operators & Components
Manage and monitor models on edge devices

Edge Manager
Manage and monitor models on edge devices

MLOps: Pipelines | Projects | Model Registry
Workflow automation, CI/CD for ML, central model catalog

Canvas
Generate accurate machine learning predictions—no code required

Studio | RStudio
Integrated development environment (IDE) for ML

Governance
Model Cards | Dashboard | Permissions

MLOps

How to make the most of the course !!

Key Recommendations

Recommended to learn in sequence

Feel free to Jump into specific section to make the most for your use-case

Once a module is completed, recommended to review and think about its learning & its implementation for your personal & client work

Keep in mind – Some Services are chargeable and not part of Free-tier on AWS – Ensure to stop services once the learning activity is completed

Practice on your personal project (No Alternate for this)

Understanding Software Development Life Cycle: Its Benefits & Challenges

Introduction to SDLC

Software Development Life Cycle (SDLC) is a process used by software development teams to plan, design, develop, test, and deploy applications. Understanding the SDLC process is essential for successful software development projects.

Phases Of SDLC

- Planning

  Identifying the scope of the project and setting goals and objectives.

- Coding Phase

  System Design, static code analysis and code review.

- Building Phase

  Building the Software

- Testing

  Testing the application to ensure it meets requirements.

- Release Phase

  Team Packaging, managing & deploying releases across various environments.

- Deployment

  Deploying the application to production environment.

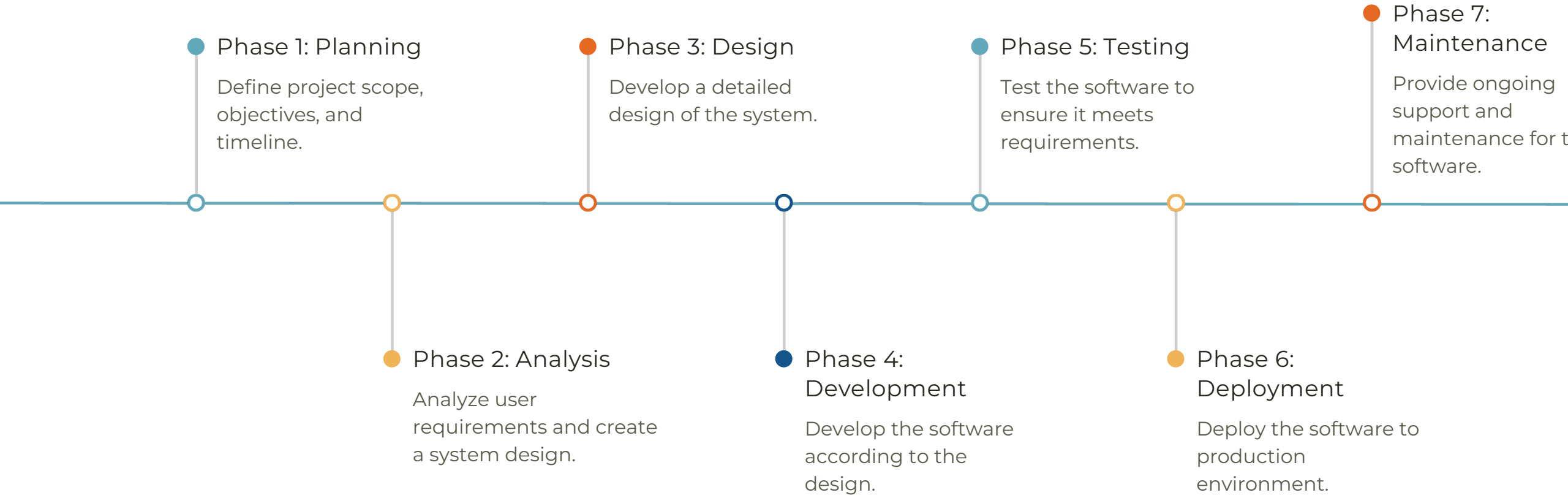
- Operate

  Use of Software in the production Environment

- Monitor

  Software, System, User Experience & Security are monitored

Processes Involved in SDLC



“**The Benefits** of SDLC are that it provides a structured approach to software development, ensuring that all necessary steps are taken and that the end product is of the highest quality.”

The Advantages of Using the SDLC

Improved Quality

SDLC helps to ensure that the product meets customer requirements and is of high quality.

Reduced Risk

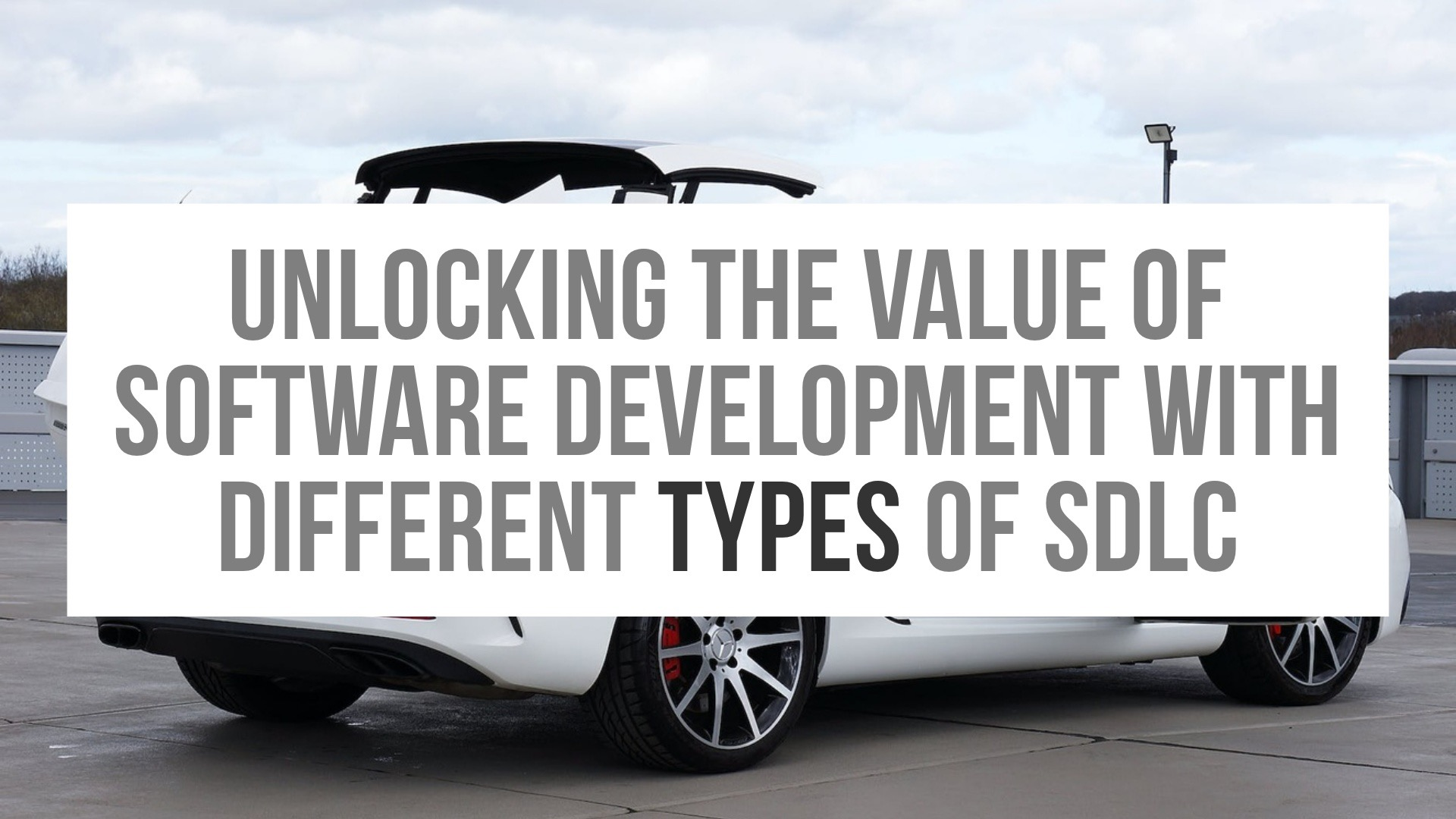
SDLC helps to identify and mitigate risks early in the development process.

Increased Efficiency

SDLC helps to streamline the development process, resulting in faster delivery times.

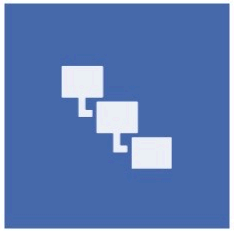
The use of SDLC can help organizations to improve the quality of their products, reduce risk, and increase efficiency.


00:00:12
Next Video :
Types of SDLC



UNLOCKING THE VALUE OF SOFTWARE DEVELOPMENT WITH DIFFERENT TYPES OF SDLC

THE DIFFERENT TYPES OF SDLC



Waterfall Model

The Waterfall Model is a linear approach to software development, with each phase of the process completed before the next begins.

Agile Model

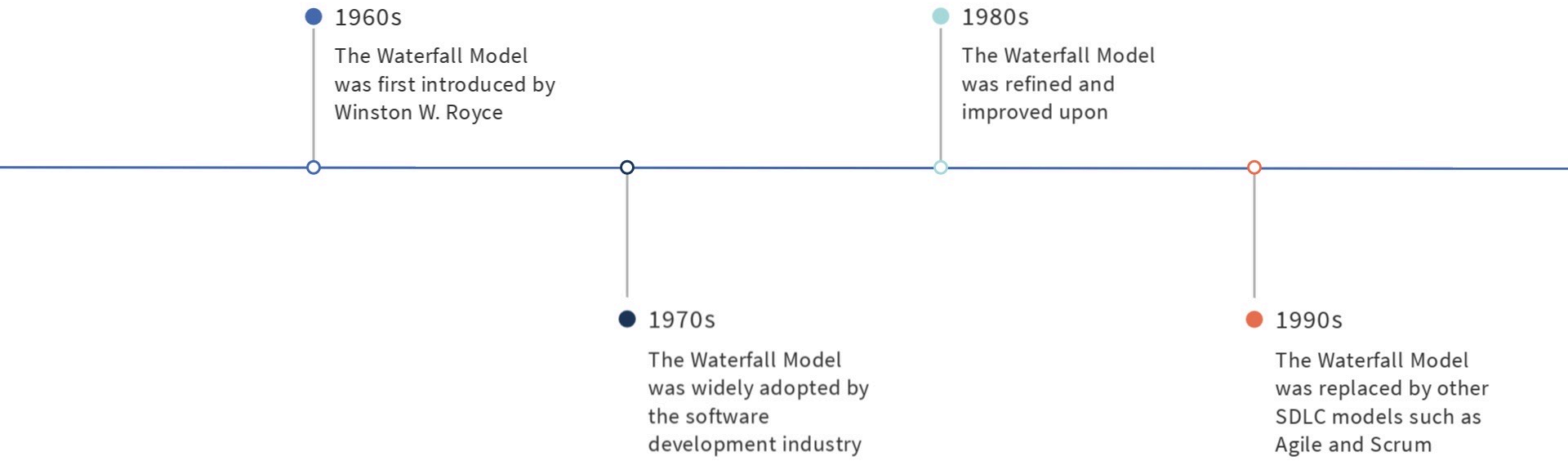
The Agile Model is an iterative approach to software development, with each phase of the process overlapping with the next.

V-Model

The V-Model is a variation of the Waterfall Model, with each phase of the process linked to its corresponding testing phase.

These are just a few of the different types of SDLC models available for unlocking the value of software development. Each model has its own advantages and disadvantages, so it is important to choose the right one for your project.

WATERFALL MODEL



BENEFITS OF AGILE METHODOLOGY

Definition of Agile Methodology

Agile methodology is an iterative approach to project management that focuses on delivering incremental value to customers quickly and efficiently.

Benefits of Agile Methodology

Agile methodology allows teams to respond quickly to changes in customer needs, prioritize tasks, and deliver results faster.

Implementation of Agile Methodology

Implementing agile methodology requires teams to break down projects into smaller tasks, collaborate closely, and use feedback loops to continuously improve.

Agile methodology is a powerful tool for teams to deliver value quickly and efficiently. By breaking down projects into smaller tasks, collaborating closely, and using feedback loops, teams can successfully implement agile methodology.

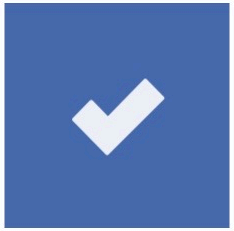
“"THE V-MODEL IS A TYPE OF SDLC THAT UNLOCKS THE VALUE OF SOFTWARE DEVELOPMENT BY PROVIDING A STRUCTURED APPROACH TO THE PROCESS." ,”

:

THE LEAN MODEL

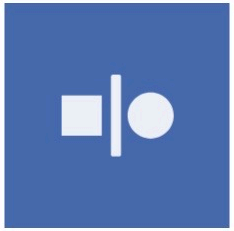
What is the Lean Model?

The Lean Model is an iterative approach to software development that focuses on reducing waste and increasing efficiency.



Benefits of the Lean Model

The Lean Model helps to reduce costs, increase customer satisfaction, and improve product quality.



Comparing the Lean Model to Other Types of SDLC

The Lean Model is different from other types of SDLC in that it emphasizes continuous improvement and customer feedback.

The Lean Model is a powerful tool for unlocking the value of software development, and can be used in combination with other types of SDLC to maximize efficiency and customer satisfaction.

Waterfall Vs Agile Vs DevOps

SDLC

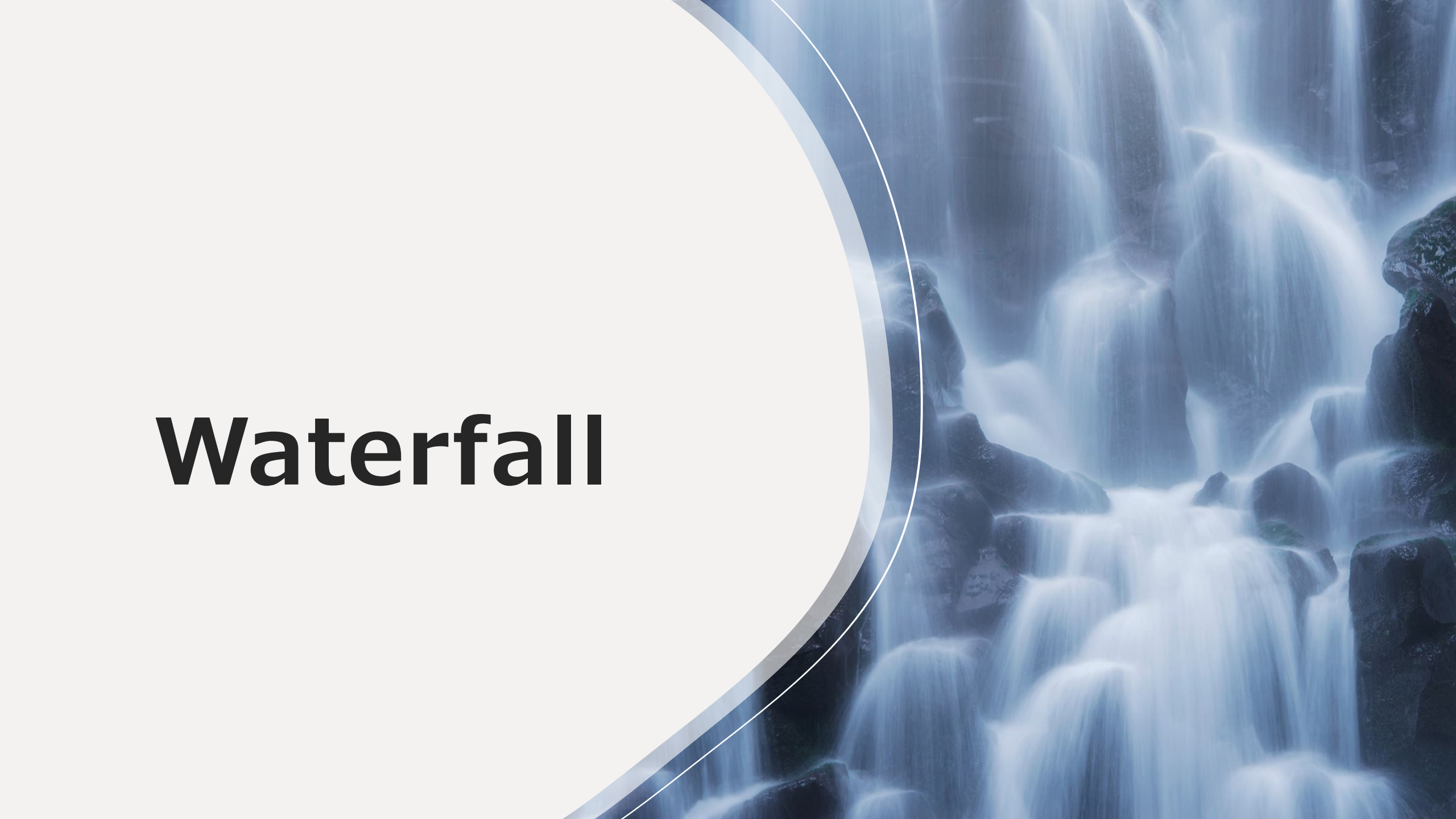
In the face of changing times, organizations are prompted to reassess their current production methodologies.

While each organization has its own distinctive approach to work, they typically rely on three highly effective methodologies: Waterfall, Agile, and DevOps.

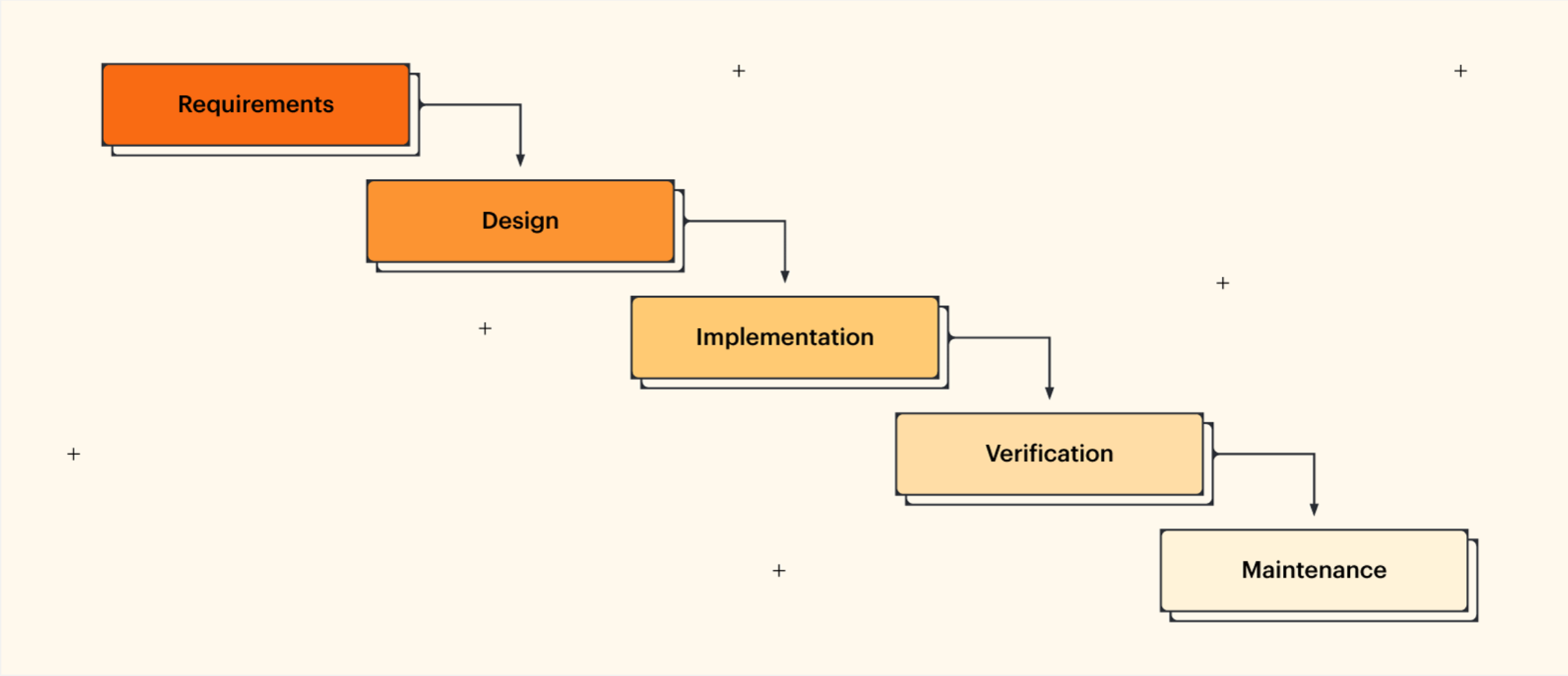
These three methodologies have been thoroughly tested and proven to deliver favorable outcomes for organizations throughout their existence.

Comparing Waterfall vs Agile vs Devops

Waterfall



Waterfall Approach



Advantages of Waterfall

The presence of a partially established framework enables the subsequent team to make modifications or corrections to the project.

This sequential process ensures that the client receives a product that is the result of collaborative team efforts.

What gives Waterfall an advantage over the other two methodologies is that it does not require any certification for individuals to participate in the Waterfall production.

Cons of Waterfall

Waterfall operates as a production line for software development.

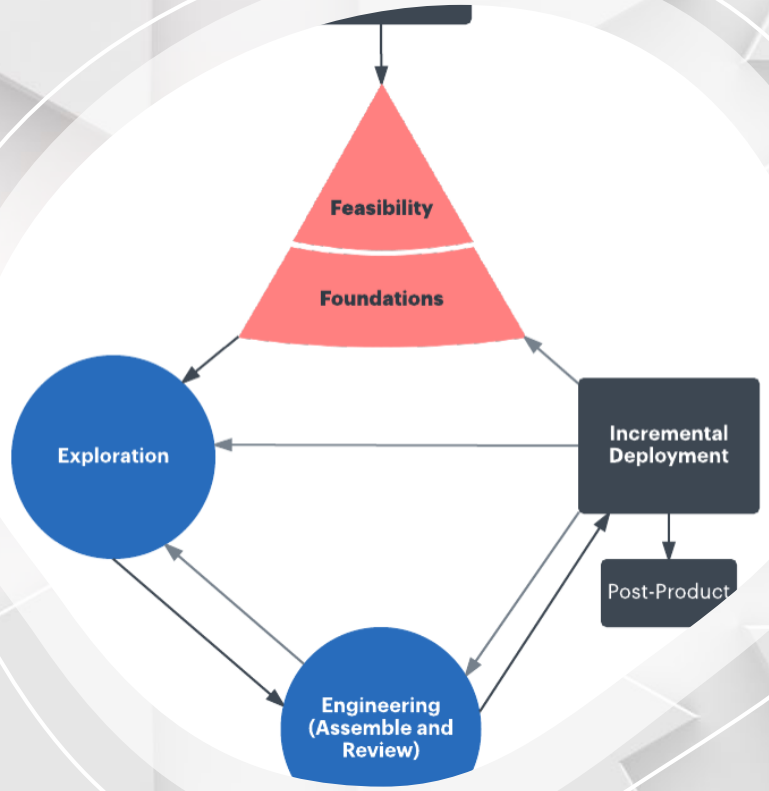
This approach can lead to delays in development and require the development team to invest additional resources in exploring alternative methods, as Waterfall restricts changes once a stage is completed.

It is crucial to establish the end goals from the beginning, as this enables effective planning of the various stages involved.

Favored for Small scale projects

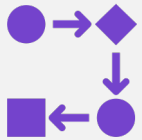
Agile

`


Feasibility
Foundations
Exploration
Incremental Deployment
Post-Product
Engineering (Assemble and Review)

Agile Methodology

Agile methodology is a flexible and iterative approach to software development and project management. It emphasizes adaptability, collaboration, and continuous improvement throughout the development process.



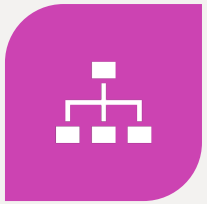
In agile, the development process is divided into short iterations called sprints, typically lasting one to four weeks.

Principles of Agile Methodology

CUSTOMER COLLABORATION

ITERATIVE DEVELOPMENT

ADAPTIVE PLANNING



SELF-ORGANIZING TEAMS

CONTINUOUS IMPROVEMENT

Advantages of Agile

- While the Waterfall approach yields reliable outcomes, Agile methodology alleviates the burden on the production team by facilitating error handling and failure mitigation through enhanced collaboration.
- The adoption of Agile leads to a considerable reduction in development-to-deployment time as the development team can focus on releasing completed components.
- Consequently, the testing team no longer needs to wait for the entire package to be available before conducting bug testing.

Cons of Agile

Software developed using Agile methodology does not have a fixed endpoint or final version.

As long as there is demand, the company is obligated to continuously release updates and improvements.

Apple, for example, has consistently introduced new versions of iOS since the initial release of the iPhone.

Organizations must be prepared to address this ongoing demand.

While this is a notable advantage of Agile, it can also be a disadvantage as each team, working within their own sprint, develops their own set of deliverables.

DevOps

DevOps stands apart from its counterparts by amalgamating the most valuable aspects of the Agile process with IT production teams.

In contrast to the fragmented nature of working independently, where miscommunication and delayed rollouts were prevalent due to the need for coordination among numerous team members, DevOps provides a revolutionary solution by merging these teams together.

AGILE VS DEVOPS

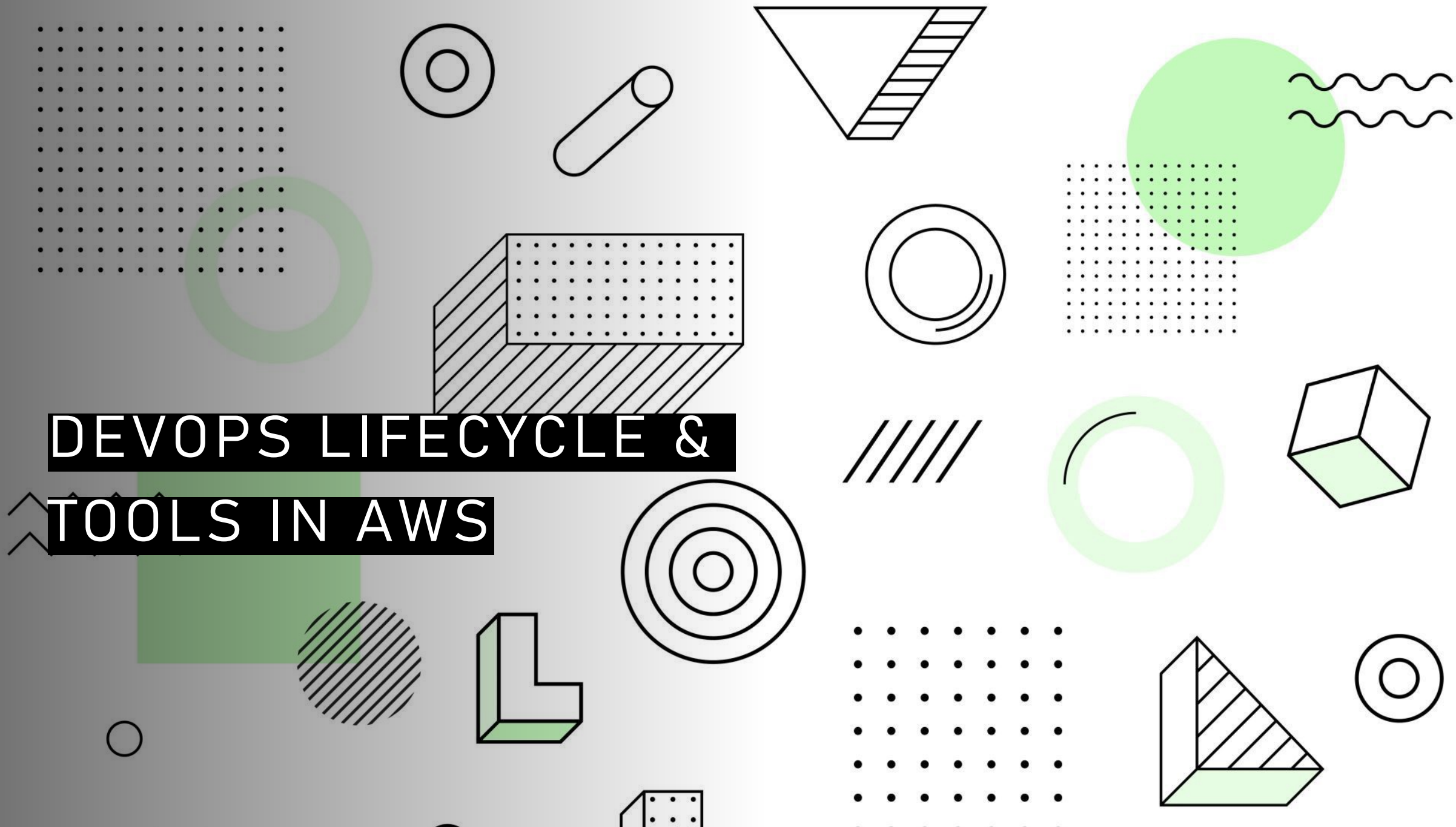
While Agile methodology focused on flexibility, enabling team members to rectify mistakes, DevOps emphasizes automation to save time and effort.

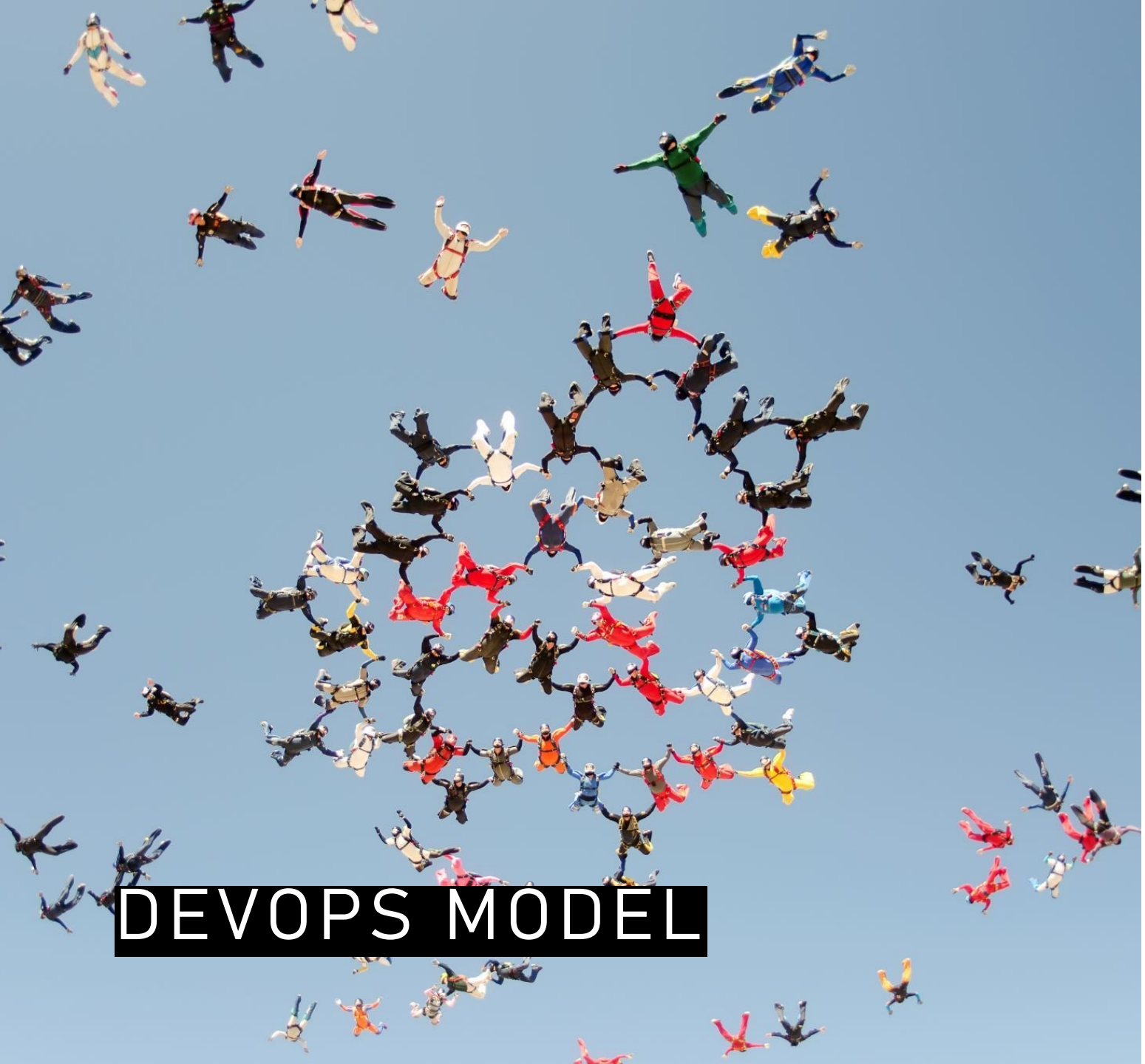
Since DevOps is derived from Agile, the principle of completing work in iterations remains. By adopting this approach, the pressure on DevOps team members is alleviated, as they can develop, test, and deploy components in a more coordinated manner through incremental updates.

Next Video

DevOps Lifecycle and Tools in AWS

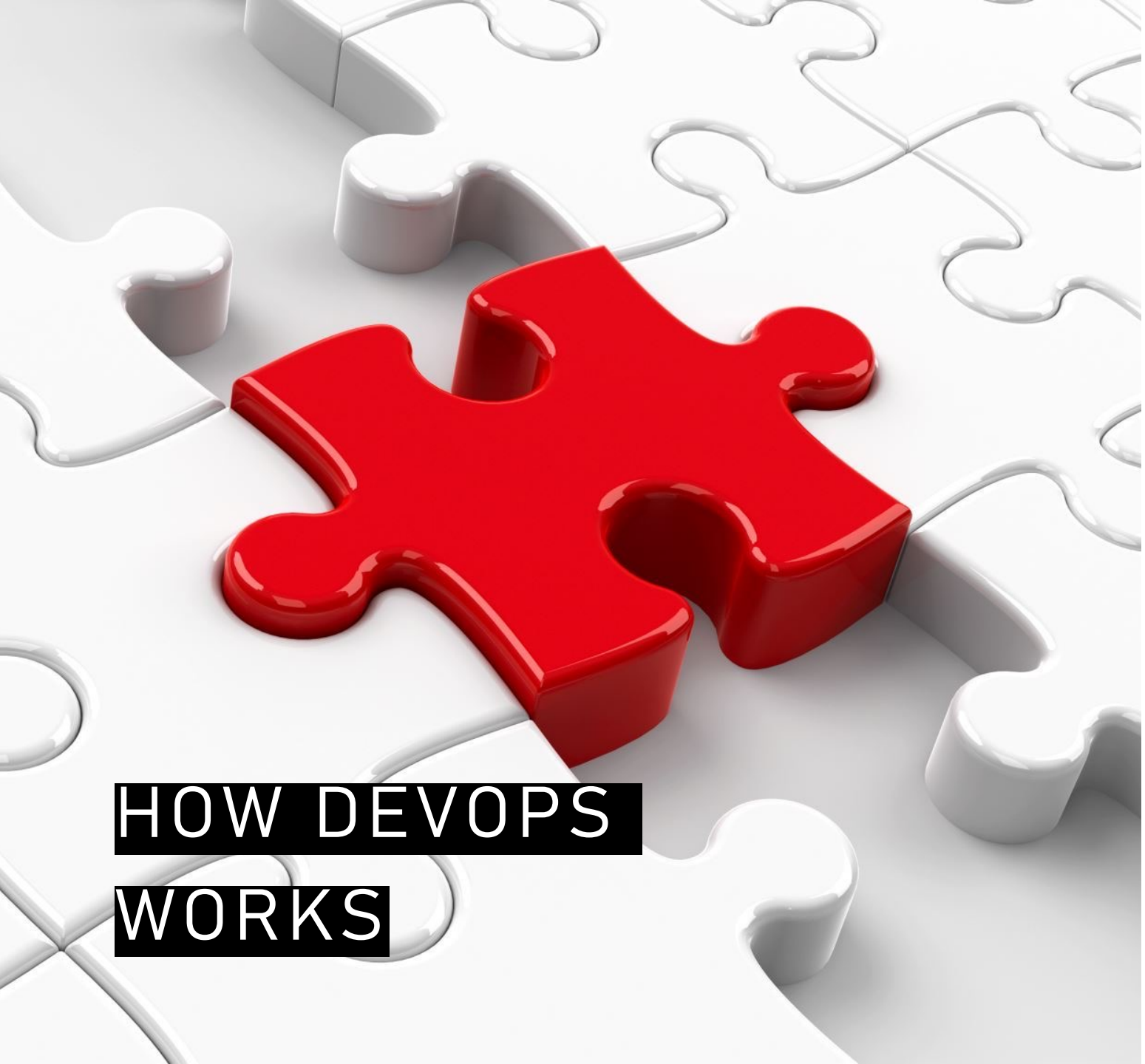
DEVOPS LIFECYCLE & TOOLS IN AWS





DEVOPS MODEL

- DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes
- This speed enables organizations to better serve their customers and compete more effectively in the market



HOW DEVOPS WORKS

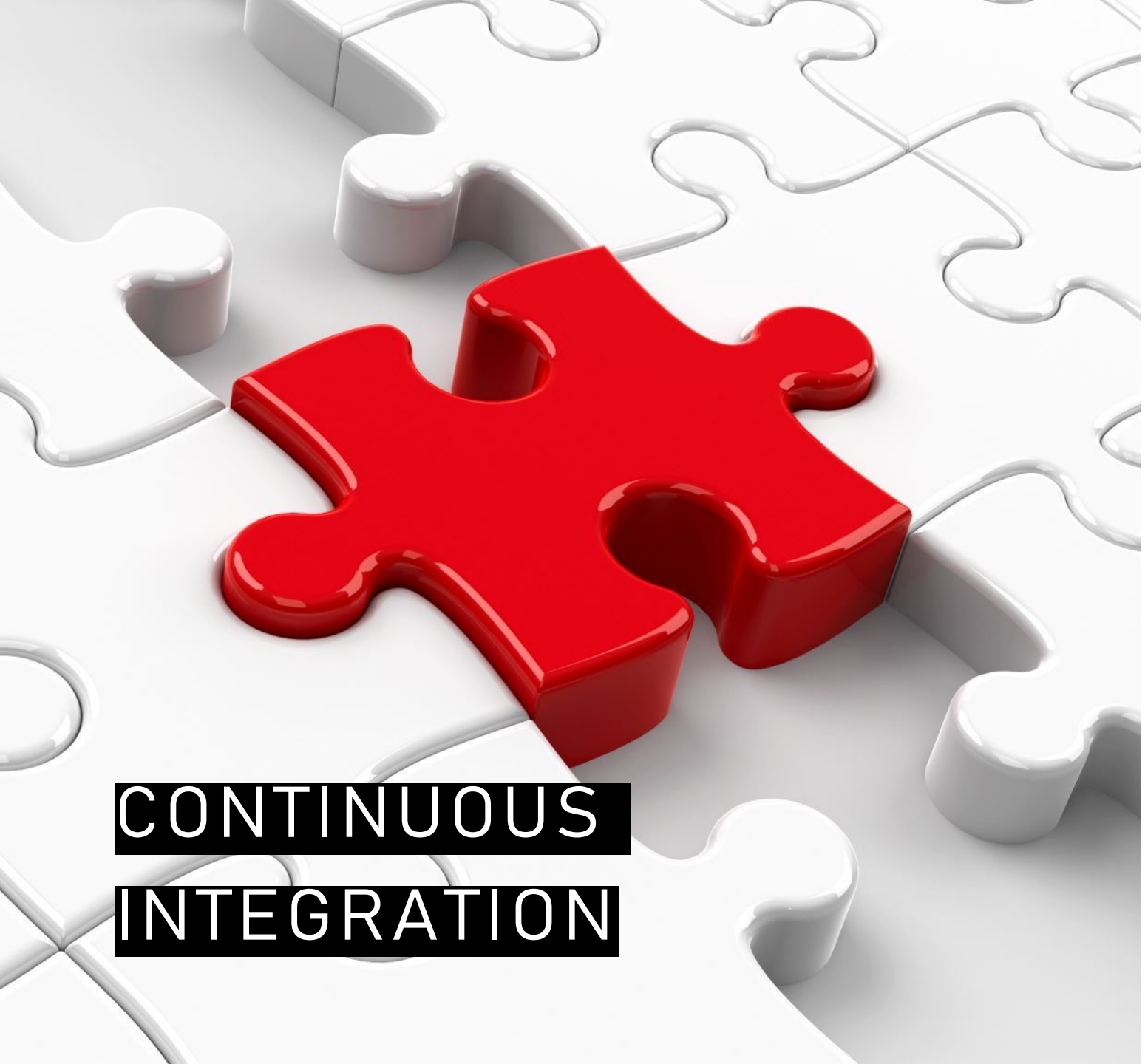
- Under a DevOps model, development and operations teams are no longer “siloed.”
- In some DevOps models, quality assurance and security teams may also become more tightly integrated with development and operations and throughout the application lifecycle
- These tools also help engineers independently accomplish tasks that normally would have required help from other teams, and this further increases a team’s velocity

Source : AWS

DEVOPS PRACTICES

- Continuous Integration
- Continuous Delivery
- Microservices
- Infrastructure as Code
- Monitoring and Logging
- Communication and Collaboration

Source : AWS



CONTINUOUS INTEGRATION

- Continuous integration is a software development practice where developers regularly merge their code changes into a central repository, after which automated builds and tests are run
- The key goals of continuous integration are to find and address bugs quicker, improve software quality, and reduce the time it takes to validate and release new software updates

Source : AWS

CONTINUOUS DELIVERY

- Continuous delivery is a software development practice where code changes are automatically built, tested, and prepared for a release to production
- It expands upon continuous integration by deploying all code changes to a testing environment and/or a production environment after the build stage
- When continuous delivery is implemented properly, developers will always have a deployment-ready build artifact that has passed through a standardized test process

Source : AWS

MICROSERVICES

- The microservices architecture is a design approach to build a single application as a set of small services
- Each service runs in its own process and communicates with other services through a well-defined interface using a lightweight mechanism, typically an HTTP-based application programming interface
- Microservices are built around business capabilities; each service is scoped to a single purpose

Source : AWS

INFRASTRUCTURE AS CODE

- Infrastructure as code is a practice in which infrastructure is provisioned and managed using code and software development techniques, such as version control and continuous integration
- The cloud's API-driven model enables developers and system administrators to interact with infrastructure programmatically, and at scale, instead of needing to manually set up and configure resources
- Thus, engineers can interface with infrastructure using code-based tools and treat infrastructure in a manner similar to how they treat application code

Source : AWS

CONFIGURATION MANAGEMENT

- Developers and system administrators use code to automate operating system and host configuration, operational tasks, and more
- It frees developers and systems administrators from manually configuring operating systems, system applications, or server software

Source : AWS

POLICY AS CODE

- With infrastructure and its configuration codified with the cloud, organizations can monitor and enforce compliance dynamically and at scale
- This makes it easier for organizations to govern changes over resources and ensure that security measures are properly enforced in a distributed manner
- This allows teams within an organization to move at higher velocity since non-compliant resources can be automatically flagged for further investigation or even automatically brought back into compliance

Source : AWS

MONITORING AND LOGGING

- Organizations monitor metrics and logs to see how application and infrastructure performance impacts the experience of their product's end user
- By capturing, categorizing, and then analyzing data and logs generated by applications and infrastructure, organizations understand how changes or updates impact users, shedding insights into the root causes of problems or unexpected changes
- Active monitoring becomes increasingly important as services must be available 24/7 and as application and infrastructure update frequency increases

Source : AWS

COMMUNICATION AND COLLABORATION

- Increased communication and collaboration in an organization is one of the key cultural aspects of DevOps
- The use of DevOps tooling and automation of the software delivery process establishes collaboration by physically bringing together the workflows and responsibilities of development and operations
- AWS provides a set of flexible services designed to enable companies to more rapidly and reliably build and deliver products using AWS and DevOps practices

Source : AWS

AWS TOOLS FOR DEVOPS

Software Release Workflows

AWS CodePipeline

AWS CodePipeline is a continuous integration and continuous delivery service for fast and reliable application and infrastructure updates. CodePipeline builds, tests, and deploys your code every time there is a code change, based on the release process models you define. This enables you to rapidly and reliably deliver features and updates.

Learn more »

Build and Test Code

AWS CodeBuild

AWS CodeBuild is a fully managed build service that compiles source code, runs tests, and produces software packages that are ready to deploy. With CodeBuild, you don't need to provision, manage, and scale your own build servers. CodeBuild scales continuously and processes multiple builds concurrently, so your builds are not left waiting in a queue.

Learn more »

Deployment Automation

AWS CodeDeploy

AWS CodeDeploy automates code deployments to any instance, including Amazon EC2 instances and on-premises servers. AWS CodeDeploy makes it easier for you to rapidly release new features, helps you avoid downtime during application deployment, and handles the complexity of updating your applications.

Learn more »

Unified CI/CD Projects

AWS CodeStar

AWS CodeStar enables you to quickly develop, build, and deploy applications on AWS. AWS CodeStar provides a unified user interface, enabling you to easily manage your software development activities in one place. With AWS CodeStar, you can set up your entire continuous delivery toolchain in minutes, allowing you to start releasing code faster.

Learn more »

Source : AWS

NEXT VIDEO

Getting Started with Amazon Web Services – For Beginners

Amazon Web Services

Getting Familiar with Basics of AWS

Agenda of the Section

- Creation of AWS Account
- Setting Up of Account (IAM User and MFA)
- Configure CLI
- Hands On - IAM
- IAM Policy Generator
- S3 Bucket (Creation, Versioning, Lifecycle Management)
- EC2 Instances (Launch & SSH)

Amazon Web Services

S3 - Simple Storage Service

Agenda

What is S3 Bucket?

S3 Storage Class

Types of S3 Storage Classes

What is S3 Bucket?

Amazon Simple Storage Service is a storage service that can be maintained and accessed over the Internet

Using the web service, we can store and retrieve an unlimited amount of data

S3 in Amazon has two entities called **buckets and objects**

AWS S3 Bucket Benefits

- 99.99% durability
- 5GB of Amazon S3 standard storage
- S3 provides Encryption to store
- Multiple copies are maintained to enable the restoring the data in case of data corruption
- We can S3 is Highly Scalable since it automatically scales your storage according to your requirement
- Pay per use Model

Amazon S3 Storage Classes

Amazon S3 offers a wide range of storage classes for various use cases

Gives power to the user for the storage of data appropriately as per the user’s need.

All Amazon S3 storage classes have a high level of reliability and support SSL data encryption during transmission, only cost is different depending on storage classes

Types of S3 Bucket Storage Classes

S3 Standard
S3 Standard-IA
S3 Intelligent-Tiering
S3 One Zone-IA
S3 Glacier
S3 Glacier Deep Archive
S3 Outposts

S3 Standard

High Availability and low latency

Data is stored in multiple locations

The durability of 99.999999999% and availability of 99.99% availability over a given year

Most expensive storage class among all others

S3 Intelligent-Tiering

Low latency and high throughput performance

Automatically moves the data between two access tiers. (Infrequent Access and Frequent Access)

The durability of 99.999999999% and availability of 99.99% availability over a given year

Small monthly monitoring and auto-tiering fee

S3 One Zone-IA

Low Latency and High throughput performance
The durability of 99.999999999% and availability of 99.5% availability over a given year
Data will be lost if the Availability Zone where the data is stored is destroyed
Suitable for larger objects greater than 128 KB kept for at least 30 days

S3 Glacier

Low-cost design for long-term archiving

Data will be available in case of entire Availability Zone destruction

The durability of 99.999999999% and availability of 99.9% availability over a given year

It has a minimum storage duration period of 90 days

S3 Glacier Deep Archive

Lowest cost storage option in S3

The durability of 99.999999999% and availability of 99.9% availability over a given year

Retrieval costs can be reduced by using bulk retrieval

It has a minimum storage duration period of 180 days

S3 Outposts

provides object storage to our on-premises AWS outposts environment.

easy to store, retrieve, secure, control access, tag, and report on the data.

S3 Object compatibility and bucket management is through S3 SDK

For durable and redundant storage of data on Outposts

S3 on Outposts will give users 48TB or 96TB of S3 storage capacity, with up 100 buckets on each Outpost

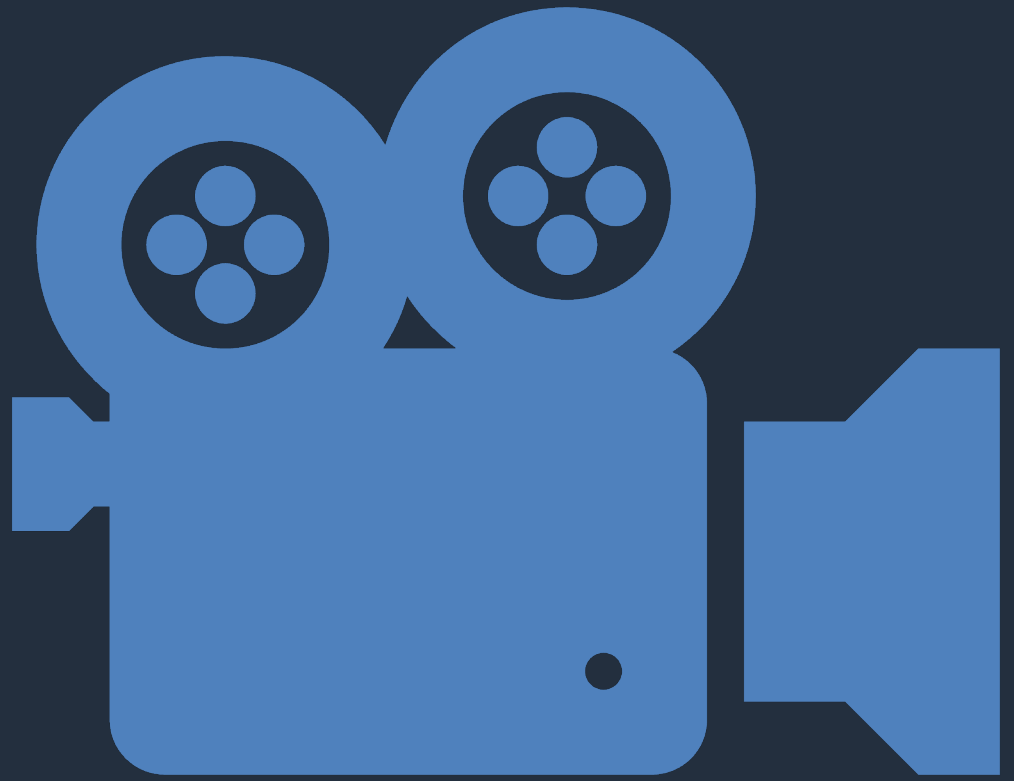
S3 Standard-IA

S3 Standard-Infrequent Access is optimized for long-lived and less frequently accessed data but requires rapid access whenever required

Similar to S3 Standard, it also offers high durability, low latency, and high throughput but has a low per GB storage price and per GB retrieval fee

This is ideal for backups, long-term storage, and as a data store for disaster recovery

Next Video



S3 BUCKET HANDS ON

Linux Operating System

DevOps & Data Scientists

Agenda of this Section:

This section is intended to help the practitioners to become aware of the Linux Operating System which is a basic for working on Command Line Interface on Servers

Infrastructure used in Hands-On Videos

EC2 Instance with Amazon Linux 2 AMI

Next Video:

Basics of Linux

Basics of Linux

DevOps & Data Scientists

Agenda

- What is Linux
- Components of Linux
- Why Linux ?
- Linux Distro
- Bash in Linux
- Package Manager in Linux

What is Linux ?

Linux is an operating system similar to Windows, iOS, and Mac OS.

It is worth noting that Android, a highly popular platform worldwide, operates on the Linux operating system.

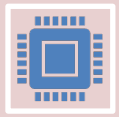
An operating system is a software that oversees the hardware resources of your computer. In essence, it facilitates the communication between your software and hardware components.

Without the operating system (OS), the software would be unable to operate.

Components of Linux Operating System:

- Bootloader - This software manages the computer's boot process
- Kernel - The kernel, known as "Linux," is the core component of the system
- Init system - This subsystem initiates the user space and controls daemons
- Daemons - These are background services like printing, sound, and scheduling that either start during boot or after logging into the desktop
- Graphical server - Also referred to as the X server or X, this subsystem handles graphics display on the monitor
- Desktop environment - This is the part of the operating system that users directly interact with
- Applications - While desktop environments have limited applications, Linux, like Windows and macOS, provides a vast selection of high-quality software titles

Why use Linux?



Open-source nature: Linux is an open-source operating system, which means its source code is freely available to the public.

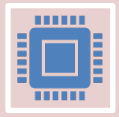
Stability and reliability: Linux is known for its stability and reliability.

Security: Linux has a strong focus on security. Due to its open-source nature, security vulnerabilities can be quickly identified and fixed by the community.

Why use Linux? (Contd.)

Flexibility and customization: Users have the freedom to choose from various desktop environments, customize their system's appearance, and configure it to their preferences.

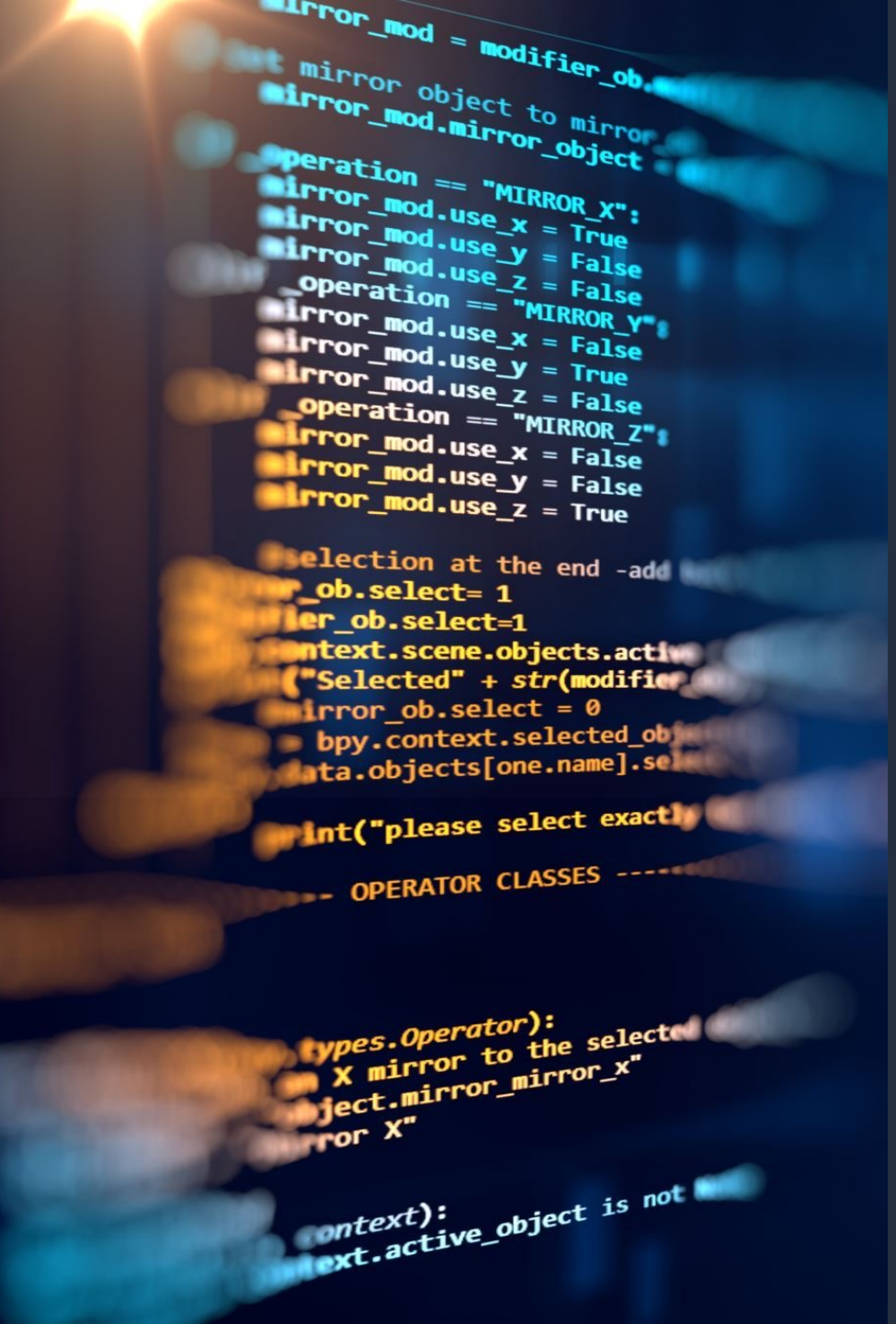
Software and package management: Linux distributions provide centralized software repositories and package management systems.



Community and support: Linux has a large and vibrant community of users and developers worldwide.

Why use Linux? (Contd)

Cost-effective: Linux is often considered a cost-effective option compared to proprietary operating systems like Windows or macOS.



What is a “distribution?”

A Linux distribution, also known as a distro, is a complete operating system package that consists of the Linux kernel, various software applications, libraries, and configuration files
Different distributions may have different default software, desktop environments, package management systems, and overall design philosophies
They are created and maintained by different organizations or communities, each with its own goals and target audience

Popular Linux Distributions

Ubuntu: One of the most widely used and beginner-friendly distributions. It focuses on ease of use, stability, and a large user community.

Debian: Known for its stability, reliability, and adherence to open-source principles. Debian serves as the base for several other distributions, including Ubuntu.

Fedora: Developed by the Fedora Project and sponsored by Red Hat. It emphasizes the use of cutting-edge technologies and serves as a testing ground for new features that may eventually make their way into Red Hat Enterprise Linux (RHEL).

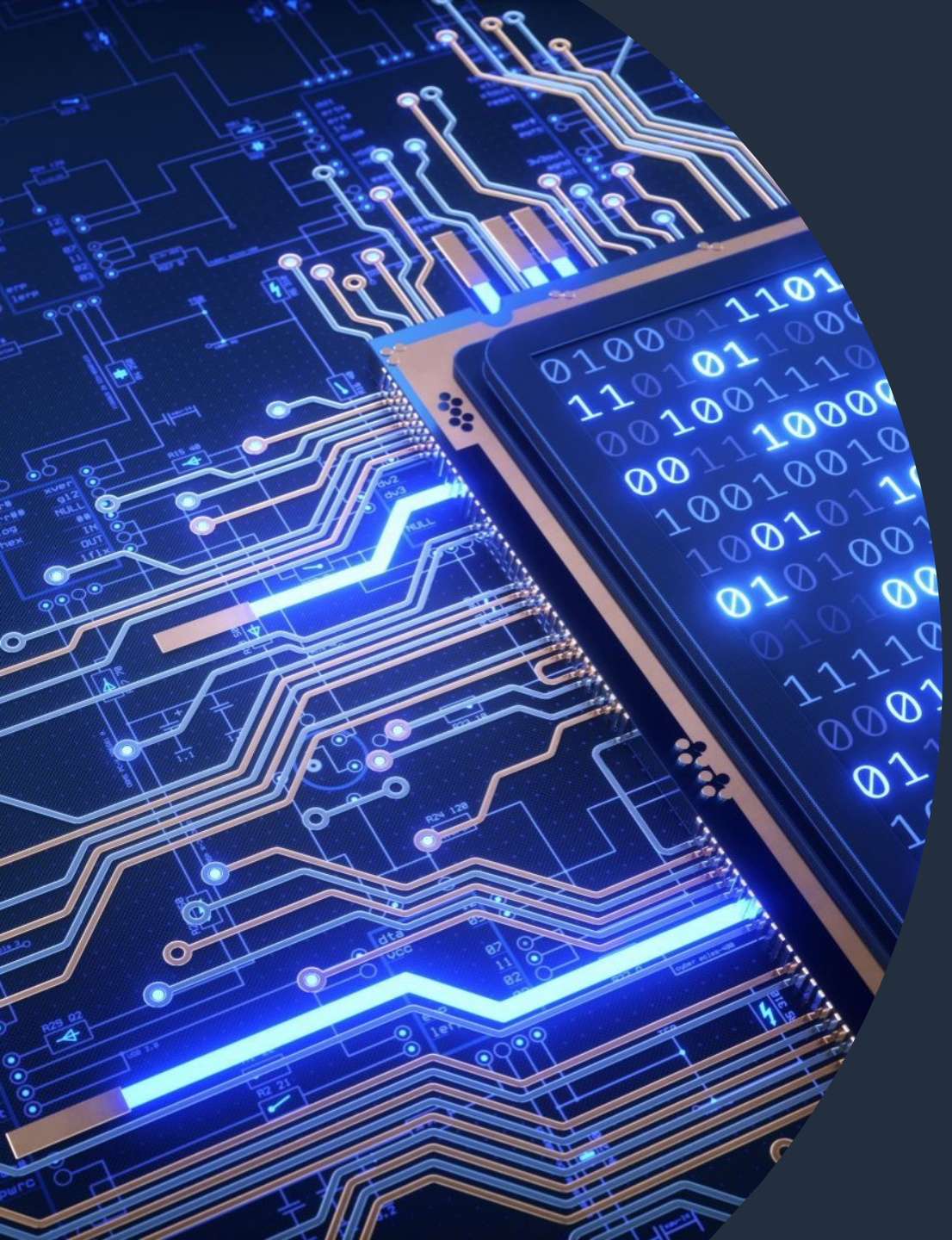
CentOS: Based on the source code of Red Hat Enterprise Linux, CentOS aims to provide a free, community-supported alternative to RHEL. It is popular for server deployments.

Arch Linux: Designed for advanced users who prefer a DIY (do-it-yourself) approach. Arch Linux focuses on simplicity, minimalism, and user-centric customization.

openSUSE: A community-driven distribution known for its stability, user-friendly configuration tools, and professional-grade features

Linux Mint: Built on top of Ubuntu, Linux Mint provides a polished and user-friendly desktop environment, making it an attractive choice for beginners transitioning from Windows.

Manjaro: Based on Arch Linux, Manjaro is known for its user-friendly approach and pre-installed software packages. It offers a balance between cutting-edge software and stability.

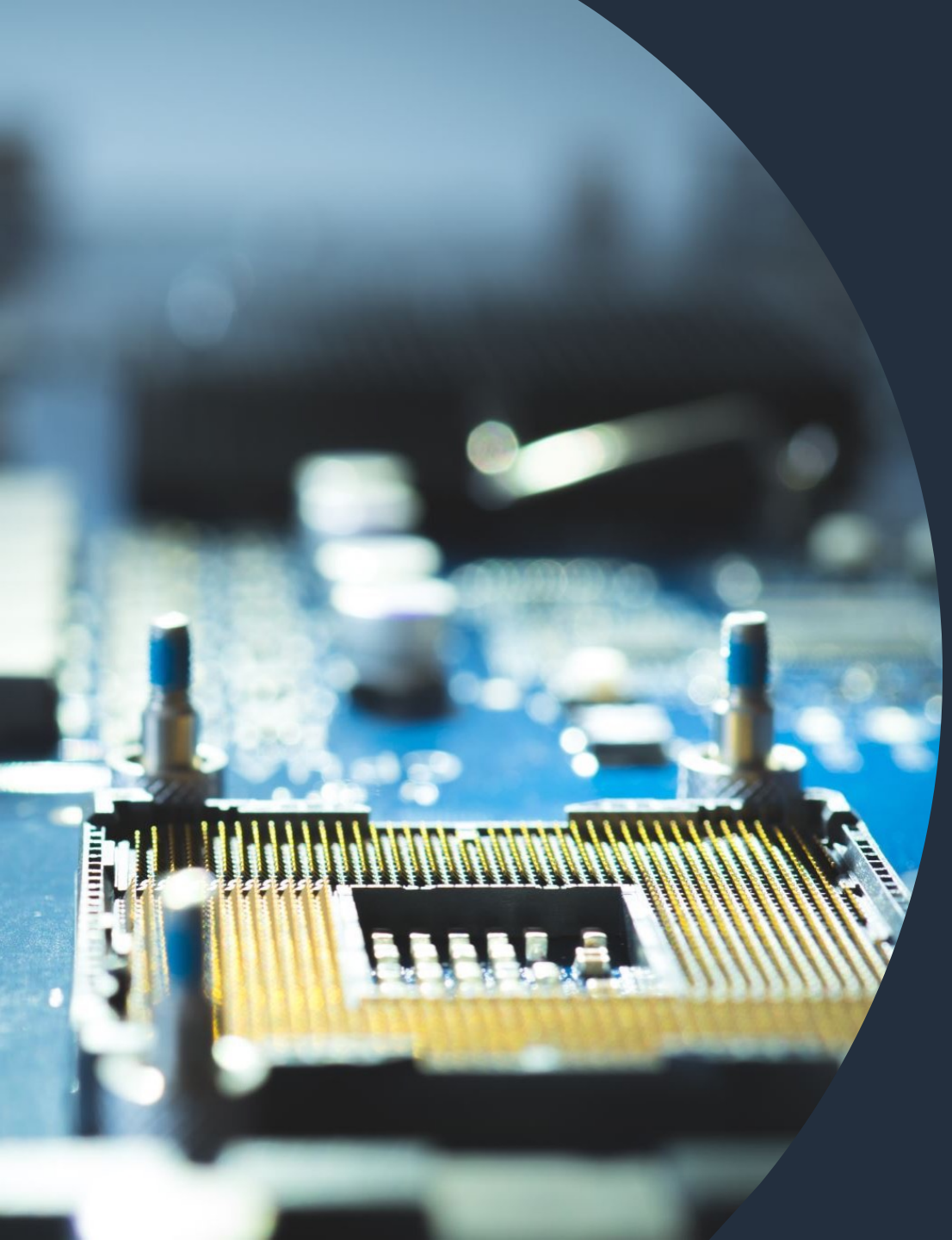


Bash in Linux

Bash, short for "Bourne Again Shell," is a command-line interpreter and scripting language that is widely used in Linux and Unix-based operating systems

Bash provides a textual interface for users to interact with the operating system by executing commands

Bash offers powerful features and capabilities, including command-line completion, command history, variables, conditionals, loops, functions, and input/output redirection



Package Managers in Linux

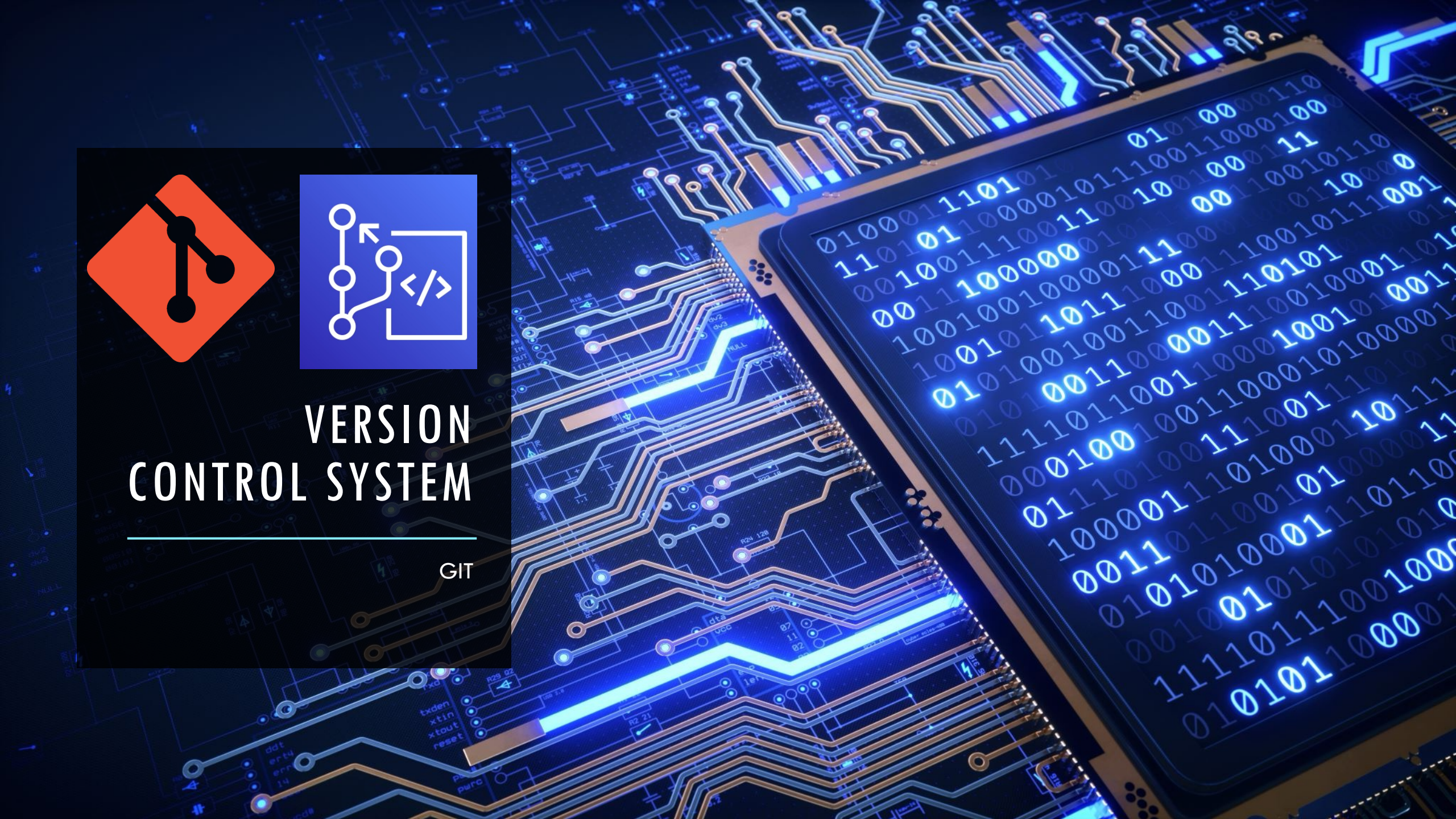
Package managers in Linux are software tools that handle the installation, management, and removal of software packages on a Linux distribution

Package managers simplify the process of installing and updating software, making it more efficient and convenient for users

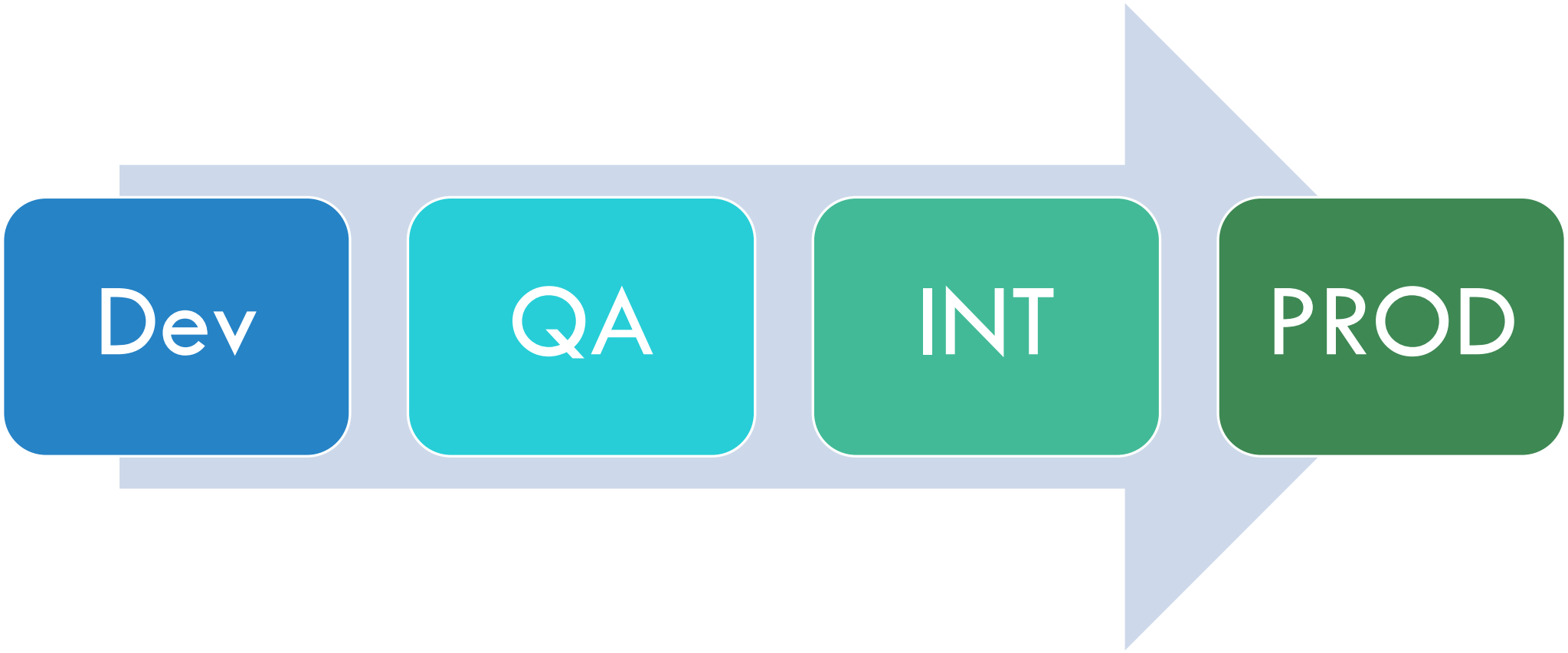
Package managers simplify the software management process in Linux, making it convenient for users to keep their systems up to date and install new software with ease

Common Package Managers in Linux

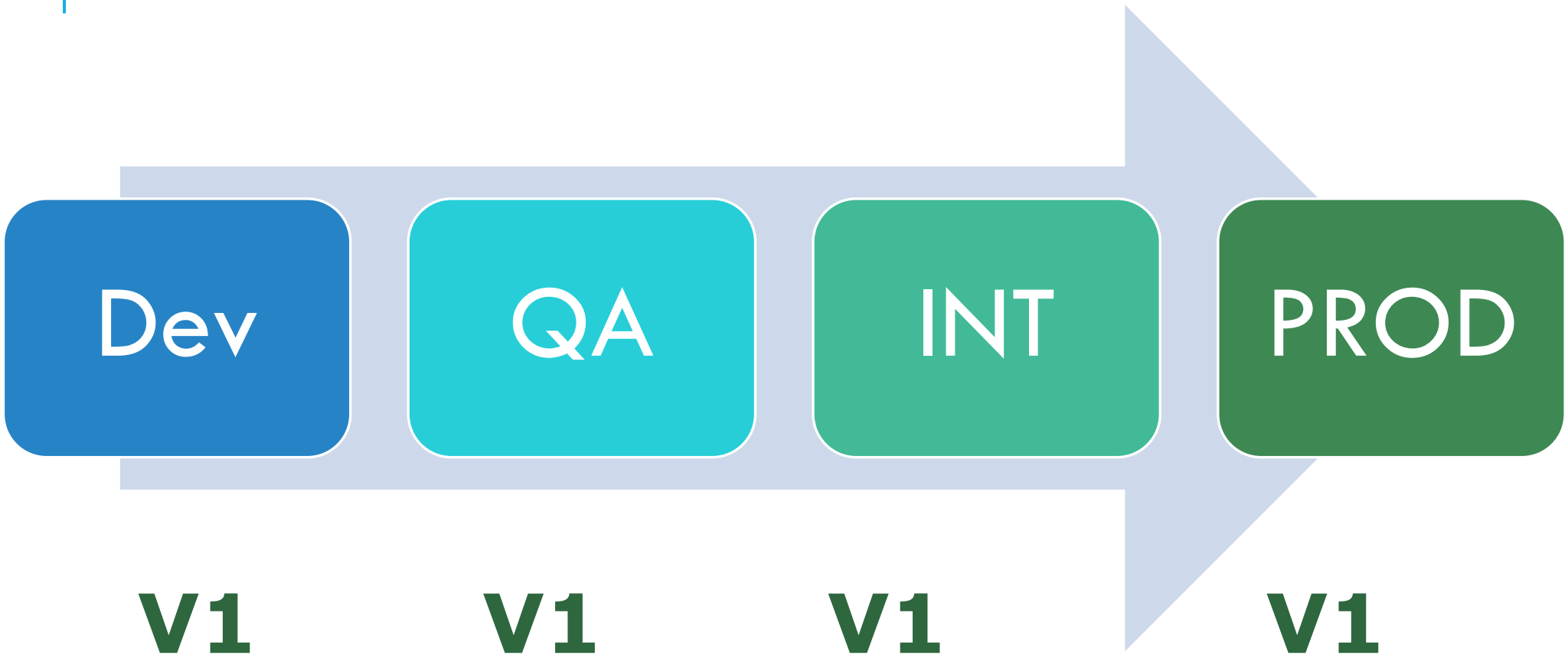
- APT
- YUM
- DNF
- Pacman
- Zypper


VERSION
CONTROL SYSTEM
GIT

ENVIRONMENTS FOR APPLICATION



CODE DEPLOYMENT….



ISSUES.....WITH V2

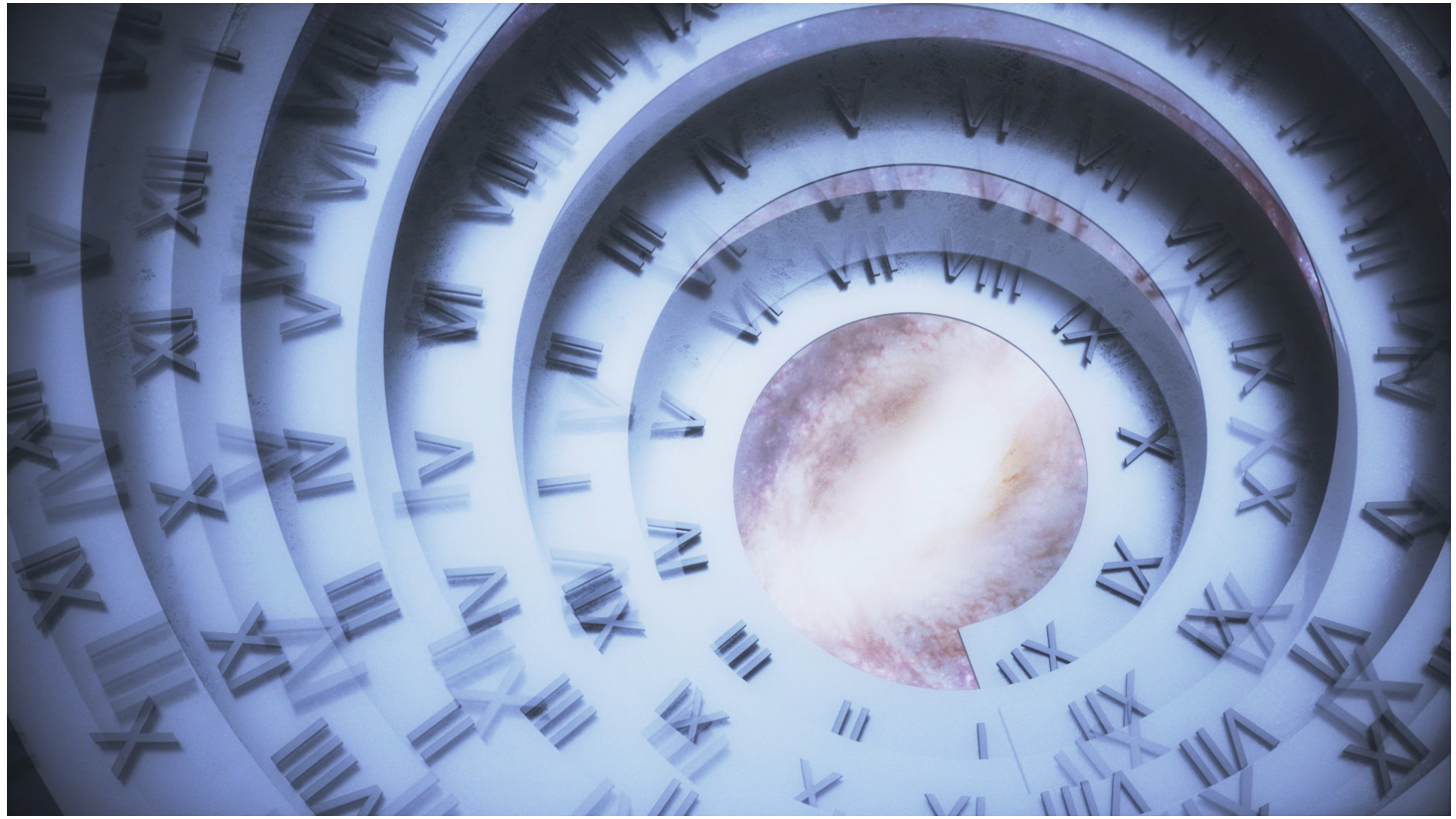
ROLLBACK ????

WITHOUT VERSION CONTROL SYSTEM

- Once Saved, All the changes are permanent and can't revert.
- No tracking of information
- Release Failures – need to redo
- Downtime of Application

WITH VCS

Time Travel and bring back the working version of code. ☺



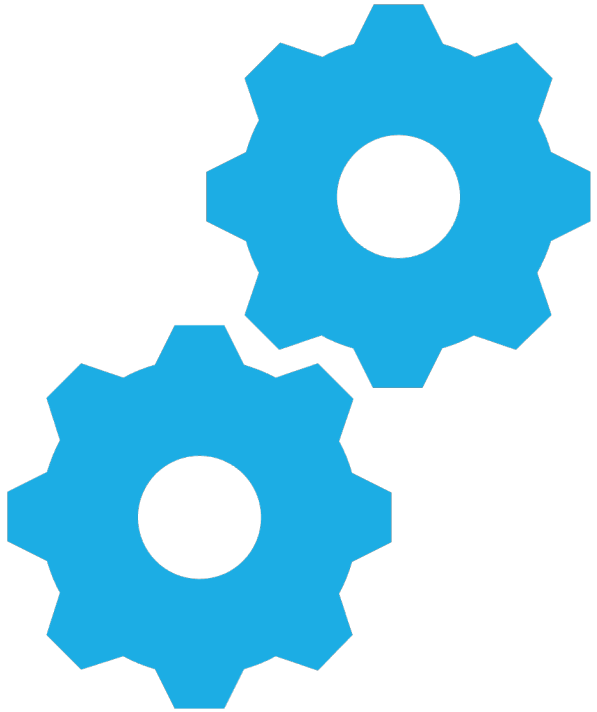
WHAT IS VERSION CONTROL?

Version control, also known as source control, is the practice of tracking and managing changes to software code.

WHAT IS VERSION CONTROL SYSTEMS ?

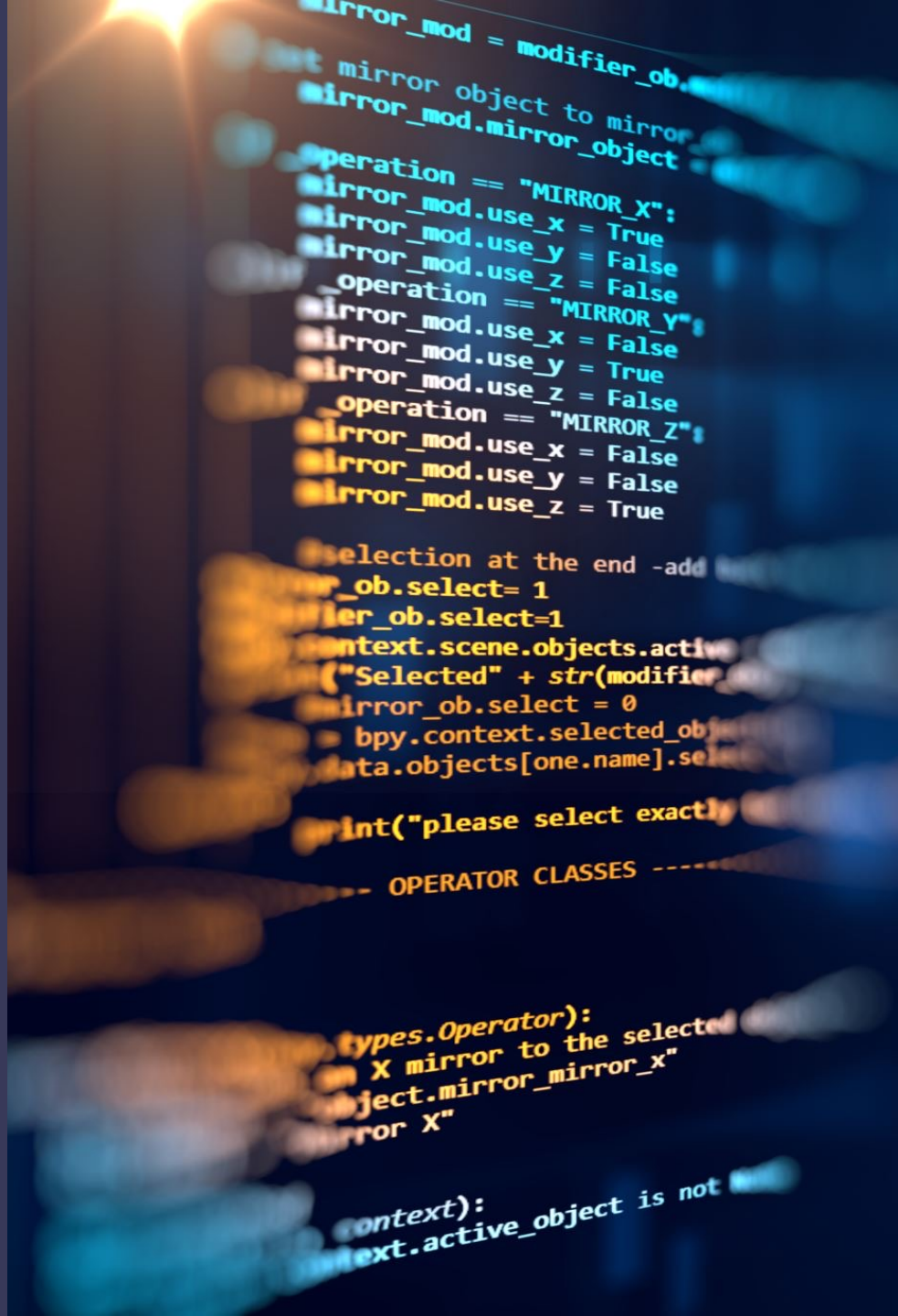
Version control systems are software tools that help a software team manage changes to source code over time.

Version control software keeps track of every modification to the code in a special kind of database. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members.



VCS ON LOCAL SYSTEM

- The practice of having the CODE saved in the local PC
- Local DB keeps a record of the changes.
- SCCS (Source Code Control System) & RCS (Revision Control System)



CHALLENGES!!

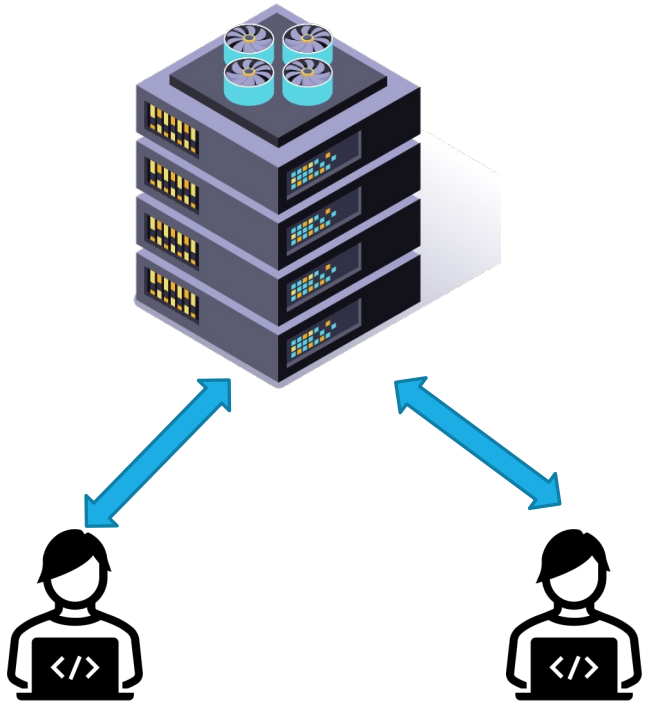
Multiple developers cannot work on same project.

Solution :

Centralized VCS

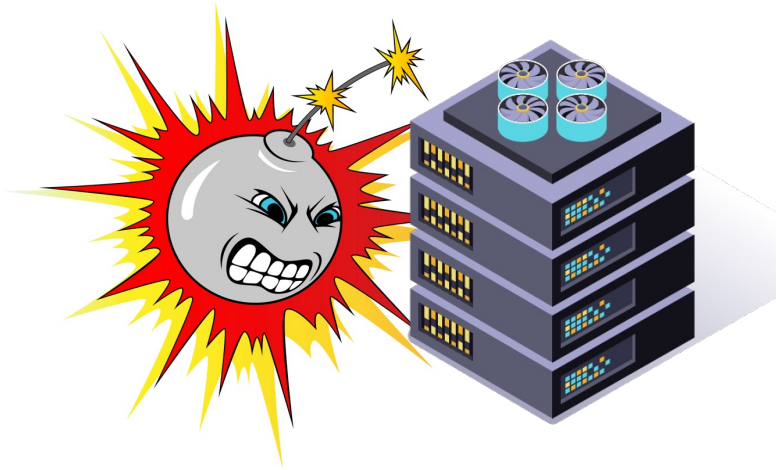
CENTRALIZED VCS

- In CVC, A Central Repo is maintained where all the versioned files are kept.
- Developers can checkout and check-in changes from their own computers.



CHALLENGES!!

If server becomes un-responsive, then entire project is lost/ cannot proceed

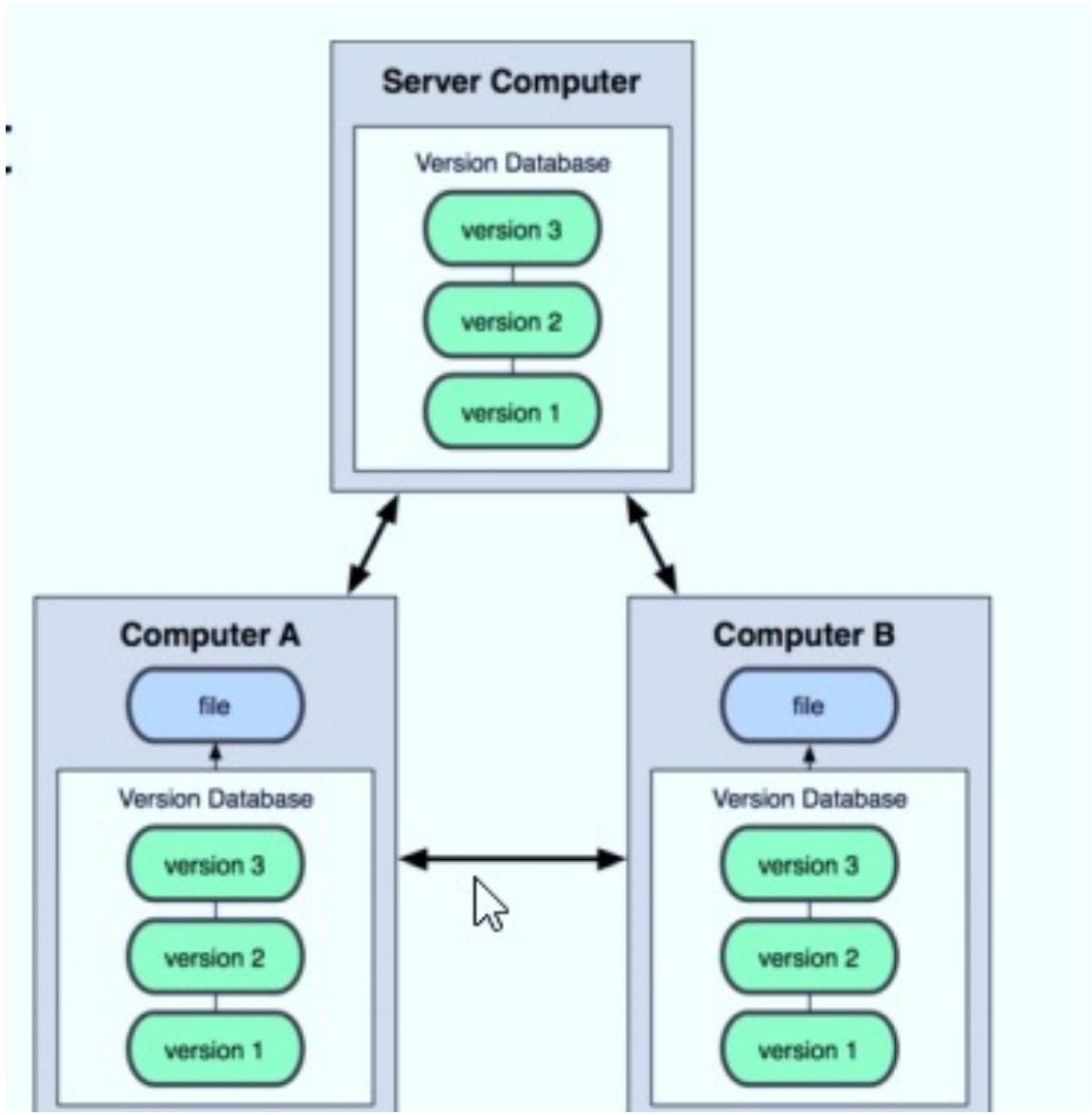


Solution:

Distributed Version Control

DISTRIBUTED VCS

- Version database (CODE) is stored on every Developer's local system and remote server
- Changes are made is Local and then PUSH to server
- Highly Available



GIT

Git is a mature, actively maintained open-source project originally developed in 2005 by Linus Torvalds, the famous creator of the Linux operating system kernel.

Having a distributed architecture, Git is an example of a DVCS (hence Distributed Version Control System).

AWS CODE COMMIT

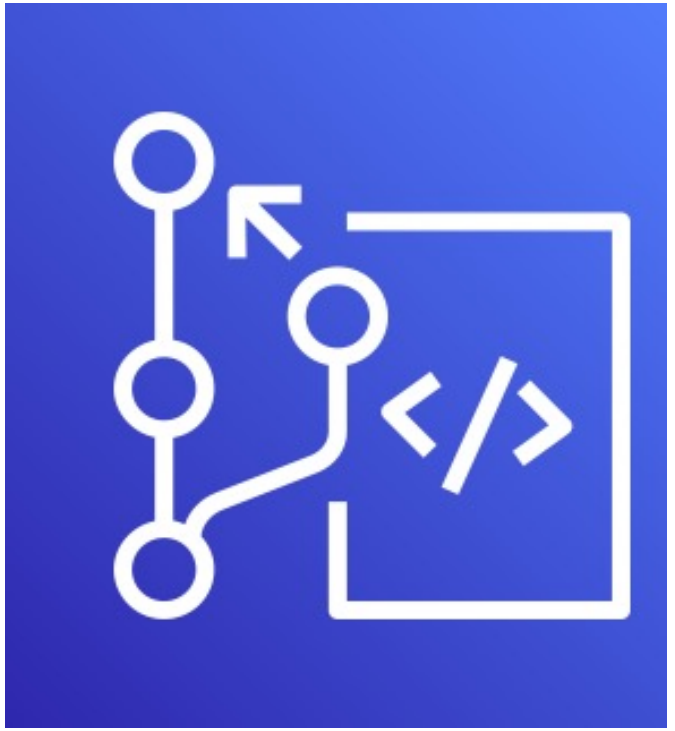
AWS CodeCommit is a secure, highly scalable, managed source control service that hosts private Git repositories.

It makes it easy for teams to securely collaborate on code with contributions encrypted in transit and at rest.

CodeCommit eliminates the need for you to manage your own source control system or worry about scaling its infrastructure.

We can use CodeCommit to store anything from code to binaries.

Supports the standard functionality of Git, so it works seamlessly with your existing Git-based tools.



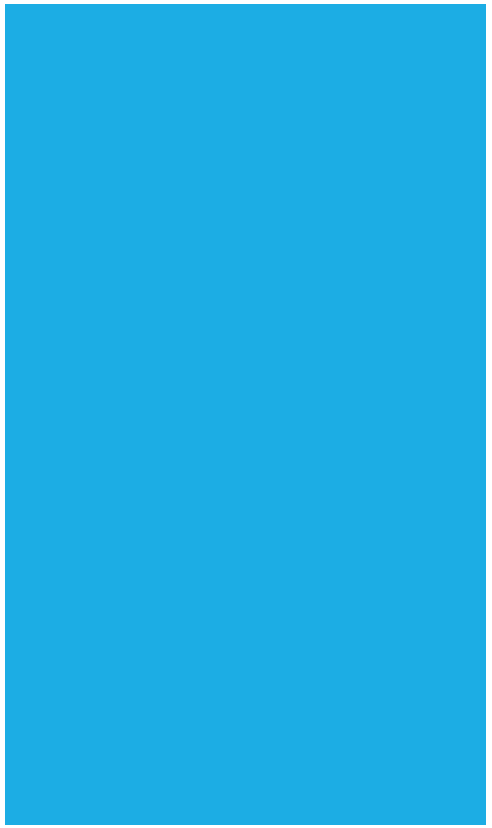
GETTING THE SYSTEM READY

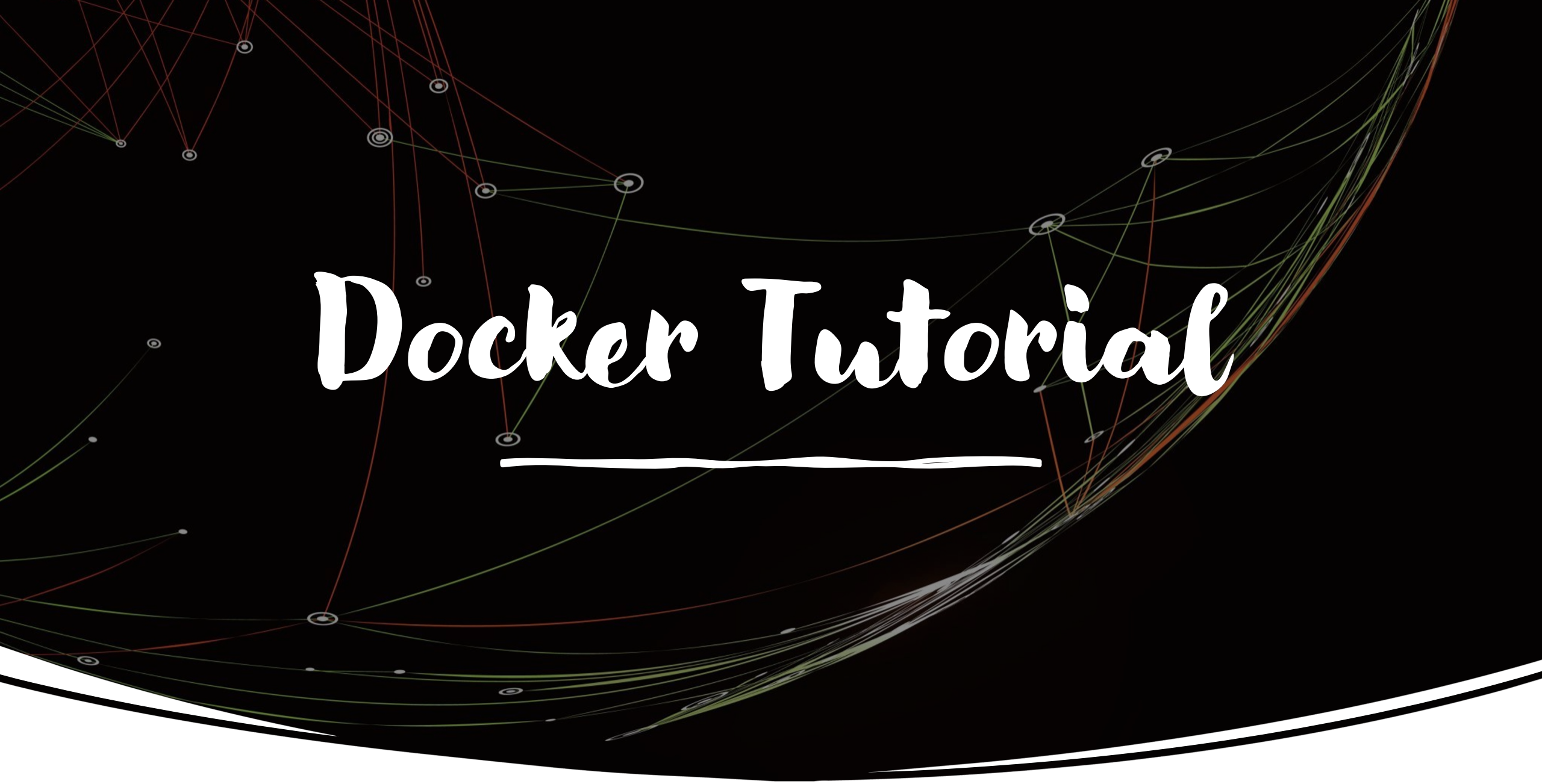
git --version → check if its already installed

apt-get update (ubuntu) or yum update (rhel , centos)

apt-get install git or yum install git

https://git-scm.com/downloads




Docker Tutorial

Introduction to Docker

- Docker is an open-source containerization platform that allows you to package applications and their dependencies into lightweight, portable containers
- Containerization technology enables applications to run consistently across different environments, from development to production, without worrying about differences in operating systems or dependencies
- Docker provides an efficient and scalable solution for deploying and managing applications, making it a popular choice among developers and DevOps teams

Introduction to Docker

- Some key benefits of Docker include
  - Efficiency: Docker containers are lightweight and share the host operating system, reducing resource usage and enabling faster deployment times
  - Consistency: Containers encapsulate the application and its dependencies, ensuring consistent behavior across different environments
  - Isolation: Each container operates in its own isolated environment, providing security and preventing conflicts between applications
- Docker has gained widespread adoption in the industry and has a vibrant ecosystem with a large community of contributors and ready-to-use images

Places where Docker can be Used ?

- Application Deployment: Docker is commonly used to package and deploy applications
- Microservices Architecture: Docker is well-suited for building and deploying microservices-based architectures
- Continuous Integration and Continuous Deployment : Docker plays a crucial role in CI/CD pipelines
- Development Environments: Docker provides a consistent and reproducible environment for developers
- Scalable Infrastructure: Docker can be used to build and manage scalable infrastructure
- Testing and QA: Docker simplifies the process of testing applications across different environments

What is Containerization Technology

- Containerization technology refers to the practice of encapsulating an application and its dependencies into a self-contained unit called a container
- Containerization technology enables the bundling of an application with its dependencies, libraries, and runtime environment, ensuring that it can be executed reliably on any system that supports containers, regardless of the underlying infrastructure or operating system
- Containers leverage operating system-level virtualization, where multiple containers share the same host operating system kernel but are isolated from one another

Docker Architecture

- The Docker architecture is composed of several components working together to provide a containerization platform
- Understanding the Docker architecture is essential for effectively utilizing Docker
- It provides a consistent and portable environment for applications, simplifies deployment workflows, and enhances scalability and resource efficiency

Host OS with Docker Engine

- The host operating system is the underlying operating system on which Docker runs
- Docker Engine, also known as the Docker daemon, is the core component of Docker that interacts with the host OS to manage containers
- Docker Engine is responsible for container runtime and provides tools for building, running, and managing containers
- Compatibility: Docker Engine is designed to work on various operating systems, including Linux, Windows, and macOS
- Linux-based Host OS: Docker originated on Linux and has native support for Linux containers
- Windows and macOS Host OS: On non-Linux systems like Windows and macOS, Docker relies on virtualization technologies to provide containerization

Host OS with Docker Engine

- Container Runtime: Docker Engine manages the container lifecycle, including container creation, starting, stopping, and deletion
- Interaction with Docker CLI: Docker CLI serves as the interface for users to interact with Docker Engine
- Interacting with the Host OS: Containers run on top of the host OS, leveraging the underlying kernel for system resources

Docker CLI for Managing Containers

- The Docker Command-Line Interface is a powerful tool for interacting with Docker Engine and managing containers
- Docker CLI provides a wide range of commands to build, run, manage, and troubleshoot containers effectively
- docker run
    - Creates and runs a new container from an image
    - Example: docker run -d --name my-container my-image
- docker ps
    - Lists running containers
    - Example: docker ps

Docker CLI for Managing Containers

docker stop/start/restart

Stops, starts, or restarts a container

Example: docker stop my-container

docker rm

Removes one or more containers

Example: docker rm my-container

docker logs

Displays the logs of a container

Example: docker logs my-container

Docker CLI for Managing Containers

- docker exec
  - Runs a command inside a running container
  - Example: docker exec -it my-container bash
- docker inspect
  - Provides detailed information about a container
  - Example: docker inspect my-container
- docker cp
  - Copies files/folders between the container and the host
  - Example: docker cp my-container:/path/to/file /host/path

Docker CLI for Managing Containers

- docker stats
  - Displays real-time resource usage statistics of running containers
  - Example: docker stats my-container

Images Stored in the Docker Registry

Containers Isolated from Each Other

Working with Docker

- Certainly!
- Here's an example slide on the topic "Working with Docker": Slide X: Working with Docker

Building Docker Images

- Building Docker images enables developers to package their applications along with their dependencies, ensuring consistent and reproducible environments for deployment
- Feel free to customize the slide content and design as per your presentation style and requirements

Running Docker Containers

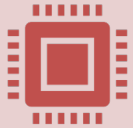
- Let's consider a simple example where we want to containerize a web application using Docker
- # Expose port 80 for web traffic # Start the nginx server This command builds the Docker image and tags it with the name 'my-web-app'
- This command creates a container from the 'my-web-app' image, runs it in detached mode , and maps port 8080 of the host machine to port 80 of the container

Amazon Web Services

Introduction to Amazon Sagemaker

Intended for Learners of AWS MLOps Course by
Manifold AI Learning

Learning Pathway of MLOps :

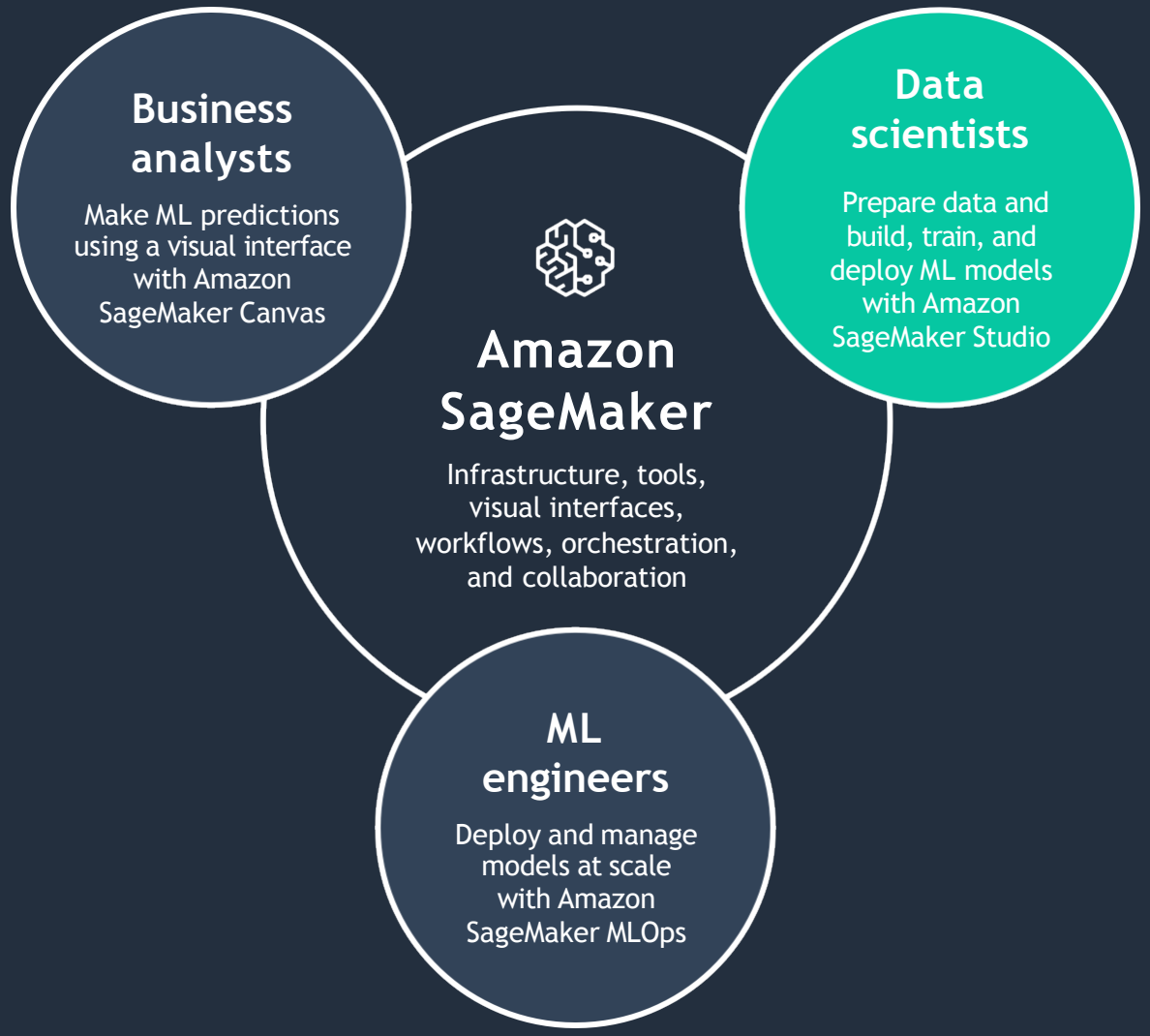


Building, training, and deploying models from SageMaker Studio

SageMaker infrastructure fundamentals

Pre-built and custom algorithms

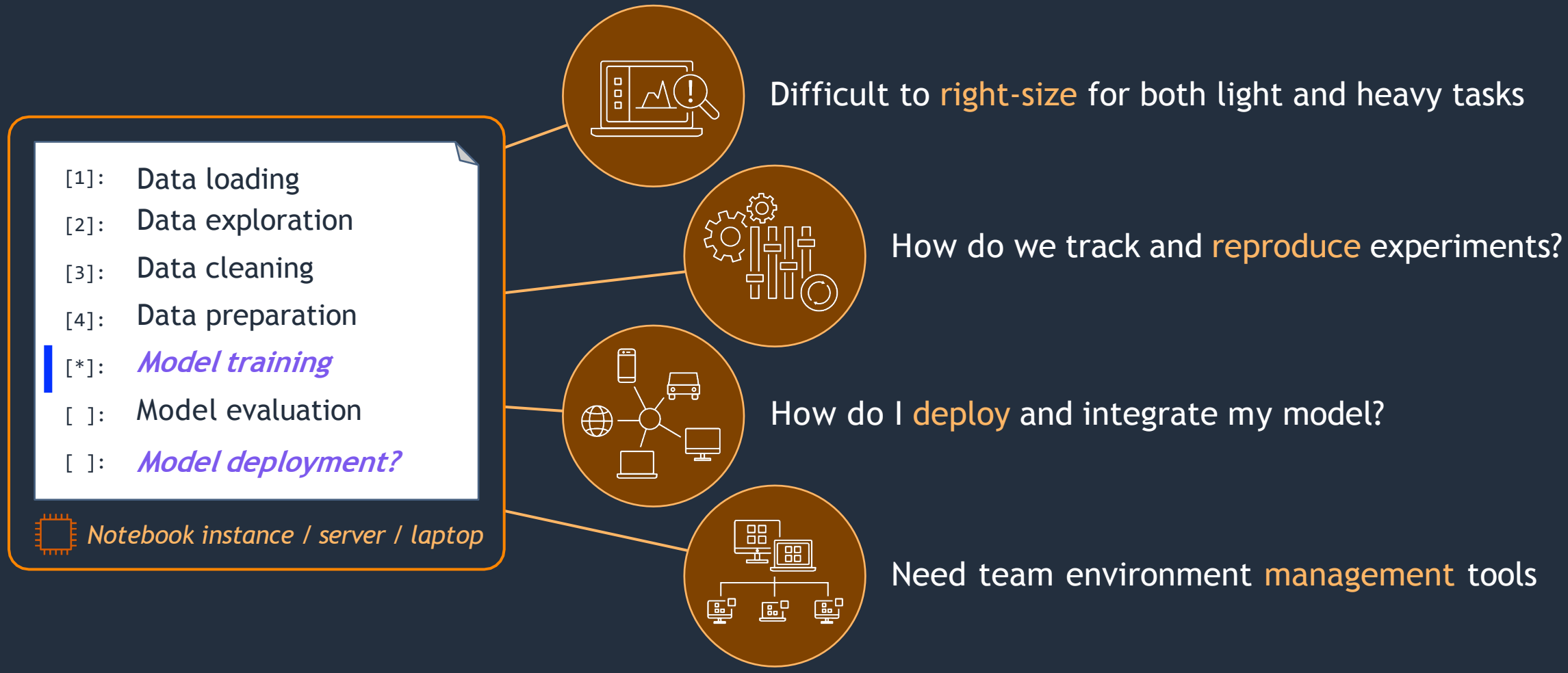
Sagemaker :


Business analysts
Make ML predictions using a visual interface with Amazon SageMaker Canvas
Data scientists
Prepare data and build, train, and deploy ML models with Amazon SageMaker Studio
Amazon SageMaker
Infrastructure, tools, visual interfaces, workflows, orchestration, and collaboration
ML engineers
Deploy and manage models at scale with Amazon SageMaker MLOps

Adjusting to SageMaker workflows

“Work from the notebook, but not on the notebook ”

Notebook-based experiments: Familiar but limiting



What is AWS Sagemaker ?

Amazon SageMaker is a fully managed machine learning service.

- Easily build and train machine learning models
- deploy into a production-ready hosted environment.
- Provides Jupyter authoring notebook instance for easy access to your data sources for exploration and analysis

References :

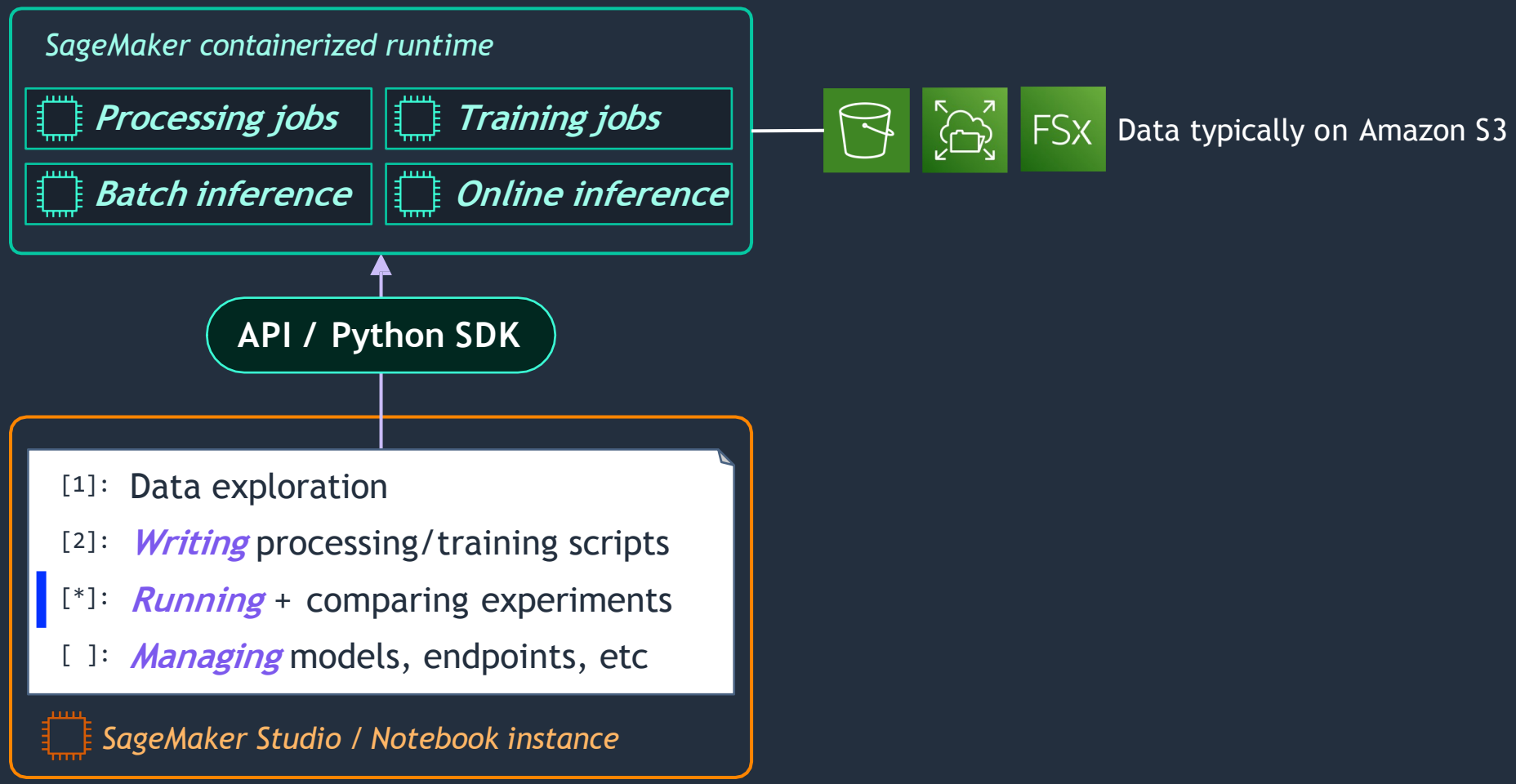
https://docs.aws.amazon.com/sagemaker/latest/dg/whatis-features-alpha.html

Amazon Web Services

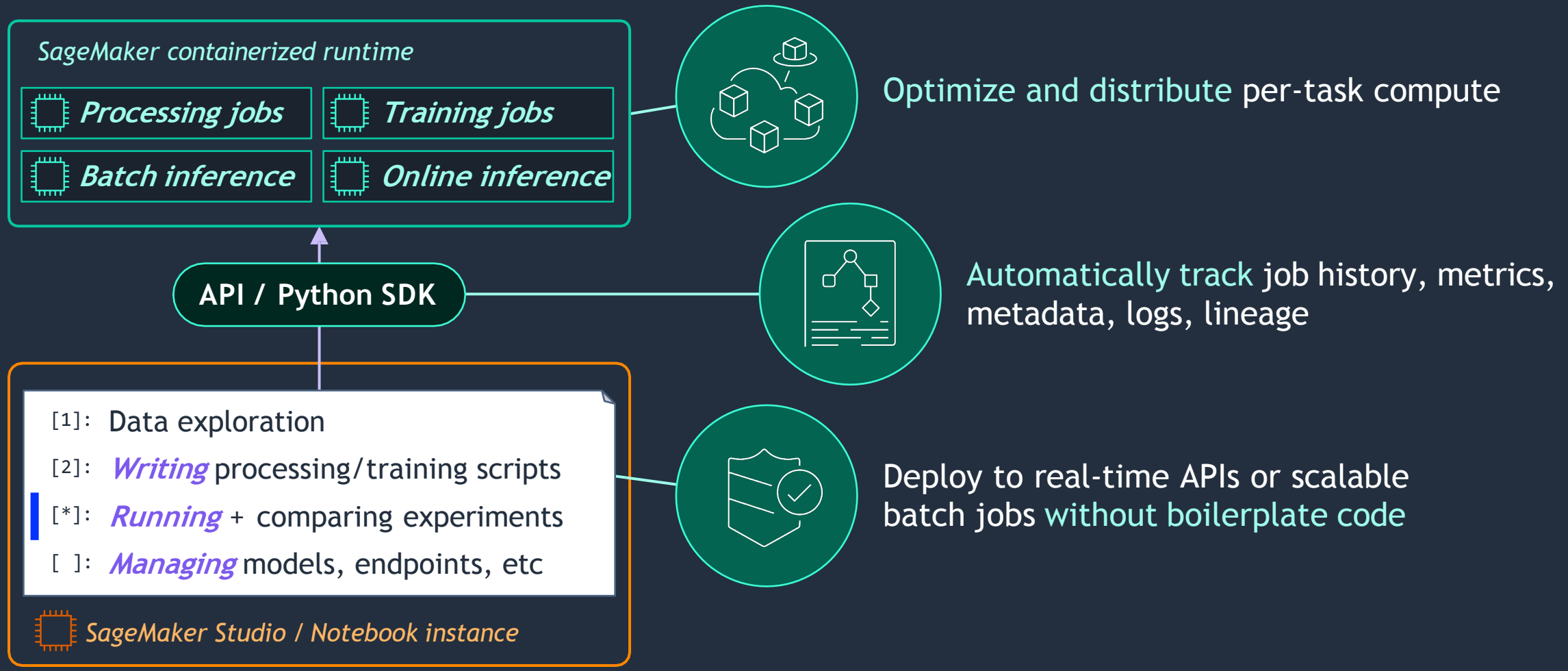
Why Amazon Sagemaker ?

Intended for Learners of AWS MLOps Course by
Manifold AI Learning

SageMaker separates notebook from job infrastructure



SageMaker separates notebook from job infrastructure



Next Journey :

Get started with Amazon Sagemaker

Setting Up Sagemaker

Sagemaker Studio Setup