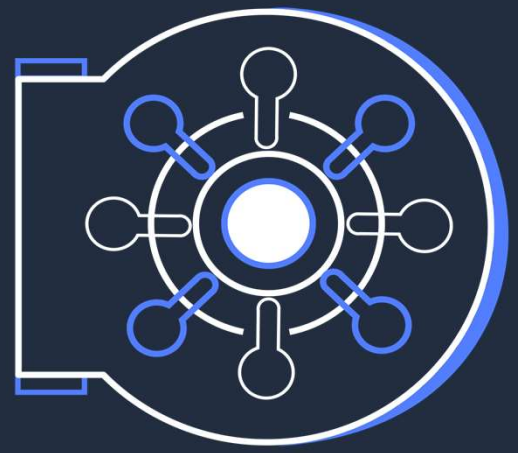
AWS TECHSHIFT

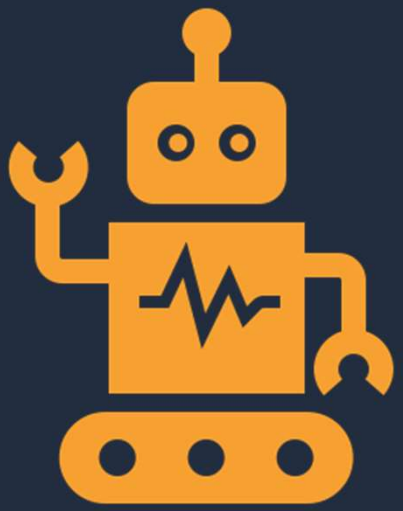
EMBARK



CI / CD

AWS TECHSHIFT Why CI / CD?

Speed



Risk

AWS TECHSHIFT What is CI?

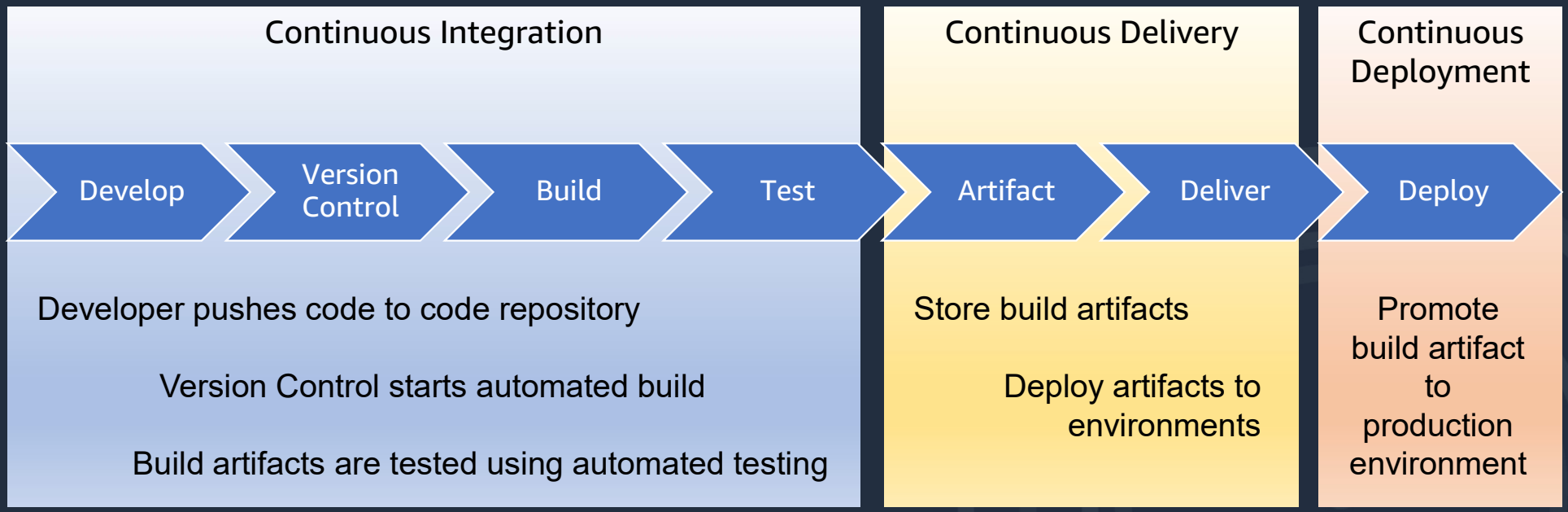
Continuous integration (**CI**) is the practice of merging all developers' working copies to a shared mainline several times a day

What is CD?

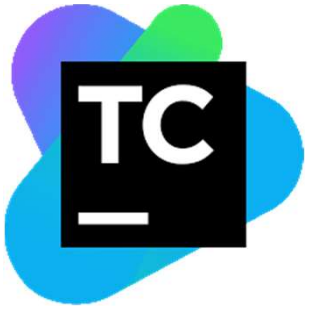
Continuous delivery (CD) is a software engineering approach in which teams produce software in short cycles, ensuring that the software can be reliably released at any time and, when releasing the software, doing so manually.

Continuous deployment (CD) is a software engineering approach in which software functionalities are delivered frequently through automated deployments.

AWS TECHSHIFT The CI/CD flow



AWS TECHSHIFT CI/CD Tool chains



AWS TECHSHIFT AWS CI/CD Services

AWS CodeCommit

Version Control : Host secure and highly scalable private Git repositories

AWS CodeStar

Unified CI/CD Projects : A unified user interface, enabling you to easily manage your software development activities

AWS TECHSHIFT AWS CI/CD Services

AWS CodePipeline

Software Release Workflows : Builds, tests, and deploys code every time there is a code change, based on the release process models

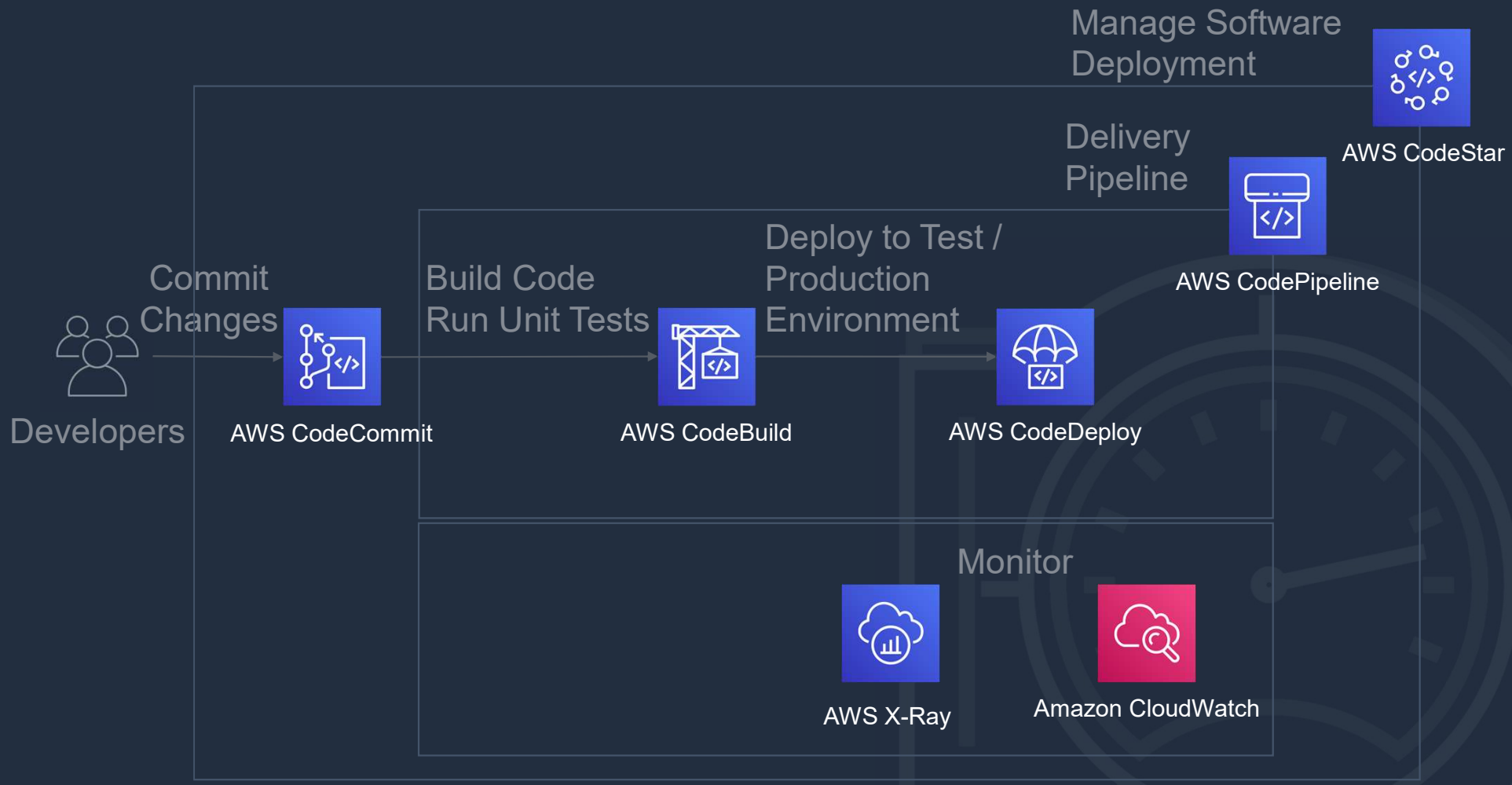
AWS CodeBuild

Build & Test Code : Compiles source code, runs tests, and produces software packages that are ready to deploy

AWS CodeDeploy

Deployment Automation : Automate code deployment to any instance

AWS TECHSHIFT AWS CI/CD Flow



Pipeline Stages

Source

Build

Test

Approval

Deploy

Invoke

AWS Code Commit

Amazon ECR

Amazon S3

GitHub

Pipeline Stages

Source

Build

Test

Approval

Deploy

Invoke

AWS Code Build

Jenkins

Pipeline Stages

Source
Build
Test
Approval
Deploy
Invoke

AWS Code Build
AWS Device Farm
Jenkins
BlazeMeter
Ghost Inspector UI Testing
Runscope API Monitoring

Pipeline Stages

Source

Build

Test

Approval

Deploy

Invoke

Manual approval

Pipeline Stages

Source

Build

Test

Approval

Deploy

Invoke

AWS CloudFormation

AWS Code Deploy

AWS Elastic Beanstalk

AWS service Catalog

Amazon ECS

Amazon ECS (Blue/Green)

Amazon S3

Pipeline Stages

Source

Build

Test

Approval

Deploy

Invoke

AWS Lambda

- Specify Instance and OS
  - Amazon Linux 2
  - Ubuntu
- Set environment variables
- Add runtimes
  - Android, Docker, DotNET, GoLang, NodeJS, Java, PHP, Python, Ruby
- BuildSpec.yml controls build process in phases

AWS CodeBuild

```
version: 0.2
phases:
  install:
    runtime-versions:
      docker: 18
      nodejs: 10
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - $(aws ecr get-login --no-include-email --region $AWS_REGION)
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - echo Build number set to $BUILD_NUM
      - echo Build hash set to $CODEBUILD_RESOLVED_SOURCE_VERSION
      - echo $BUILD_NUM > buildnum.txt
      - echo $CODEBUILD_RESOLVED_SOURCE_VERSION > buildhash.txt
      - docker build -t $AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/tsa/gallery:$BUILD_NUM .
      - docker tag $AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/tsa/gallery:$BUILD_NUM
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/tsa/gallery:latest
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - docker push $AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/tsa/gallery:$BUILD_NUM
      - docker push $AWS_ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/tsa/gallery:latest
```

End