AWS CodeCommit

- CodeCommit is a **fully managed version control service** that hosts private Git repositories in the AWS cloud (**similar to Github**).
- You can create your own code repository and **use Git commands** to interact with your own repository and other repositories.
- The AWS CodeCommit Console lets you **visualize your code**, **pull requests**, **commits, branches** and other settings.

CodeCommit Concepts

- **Repository :** A repository is the fundamental version control object in CodeCommit used to securely store code and files for your project.
- **File** : A file is a version-controlled, program code, a piece of information available to users of the repository and branch where it is stored.
- **Pull Request** : A **pull request(PR)** allows you and other users to review, comment on, and merge code changes from one branch to another. (**Remote Merge Operation**)
- **Approval Rule** : An approval rule is used to designate a number of users who will approve a pull request before it is merged into your branch. This is similar to Branch Protection Rule in Github. (minimum 2 approvers)

CodeCommit Concepts

- **Commits** : A commit is a snapshot of the contents and changes to the contents of your repository. It Includes information like:
    - Who committed the change.
    - The date and time of the commit.
    - Changes made as part of the commit.
- **Branches:** In Git, branches are **simply pointers or references to a commit**.
    - Branches are used to **separate work** on a new or different version of files **without impacting work in other branches**.
    - You can use branches to **develop new features**, **store a specific version** of your project from a particular commit, etc.
- **Active User**: An **active user** is any unique AWS identity (IAM user/role) that accesses AWS CodeCommit repositories during the month.
- AWS identities that are created through your use of other AWS Services, such as AWS CodeBuild and AWS CodePipeline, as well as servers accessing CodeCommit using a unique AWS identity, count as active users.

CodeCommit Authentication

- Repository in CodeCommit is region specific.
- IAM credentials are required for CodeCommit Authentication.
- IAM supports CodeCommit with three types of credentials:
  - **HTTPS Git credentials**, an IAM-generated user name and password pair you can use to communicate with CodeCommit repositories over HTTPS.
    - Here, Clone Repository Url should be https url
    - git clone **https://git-codecommit.us-east-1.amazonaws.com/v1/repos/my-test-repo**
  - **SSH keys**, a locally generated public-private key pair that you can associate with your IAM user to communicate with CodeCommit repositories over SSH.
    - Here, Clone Repository Url should be ssh url
    - git clone **ssh://git-codecommit.us-east-1.amazonaws.com/v1/repos/my-test-repo**

Use Git credentials and HTTPS with CodeCommit (recommended)

- With Git credentials, a **static user name** and **password** pair for your IAM user is generated that can be used as credentials for HTTPS connections.
- Because these credentials are universal for all supported OS and compatible with most credential management systems, development environments, and other software development tools, this is the recommended method.
  - You can reset the password for Git credentials at any time.
  - You can also make the credentials inactive or delete them if you no longer need them.
- Steps to generate **https** credentials for CodeCommit.
  - Navigate to **IAM > Users > Select/Create IAM User > Security Credentials > HTTPS Git credentials for AWS CodeCommit > Generate.**

Use SSH keys and SSH with CodeCommit

- With SSH connections, create public (**id_rsa.pub**) and private key (**id_rsa**) files (using **ssh-keygen**) on your local machine that Git and CodeCommit uses for SSH authentication.
- You associate the public key with your IAM user and store the private key on your local machine.
- Steps to setup Public Key in IAM User for ssh authentication for CodeCommit.
  - Navigate to **IAM > Users > Select/Create IAM User > Security Credentials > Upload SSH public key > Paste the contents of your SSH public key > Upload SSH public key.**
  - Copy or save the information in **SSH Key ID** (for example, *APKAEIBAERJR2EXAMPLE*)

Use SSH keys and SSH with CodeCommit

- On your local machine, create a **config** file in the **~/.ssh** directory, and then add the following lines to the file, where the value for User is the SSH key ID you copied earlier.
- Execute command : **chmod 600 config**
- Validate SSH Connection using : **ssh git-codecommit.us-east-1.amazonaws.com**

```
Host git-codecommit.*.amazonaws.com
  User APKAEIBAERJR2EXAMPLE
  IdentityFile ~/.ssh/codecommit_rsa
```

IAM Policy for CodeCommit

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "codecommit:GitPush",
                "codecommit:DeleteBranch",
                "codecommit:PutFile",
                "codecommit:MergeBranchesByFastForward",
                "codecommit:MergeBranchesByThreeWay",
                "codecommit:MergePullRequestByFastForward",
            ],
            "Resource": "arn:aws:codecommit:us-east-2:<AWS_ACCOUNT_ID>:<REOPO_NAME",
            "Condition": {
                "StringEqualsIfExists": {
                    "codecommit:References": [
                        "refs/heads/main"
                    ]
                },
                "Null": {
                    "codecommit:References": false
                }
            }
        }
    ]
}
```

- Assign CodeCommit Policy to IAM User or IAM Group.
- IAM Managed Policy **AWSCodeCommitPowerUser** or another managed policy for CodeCommit access.
- As a best practice, no IAM User should be allowed to direct push changes into **master/main** branch, hence below IAM policy can be attached to IAM Users.

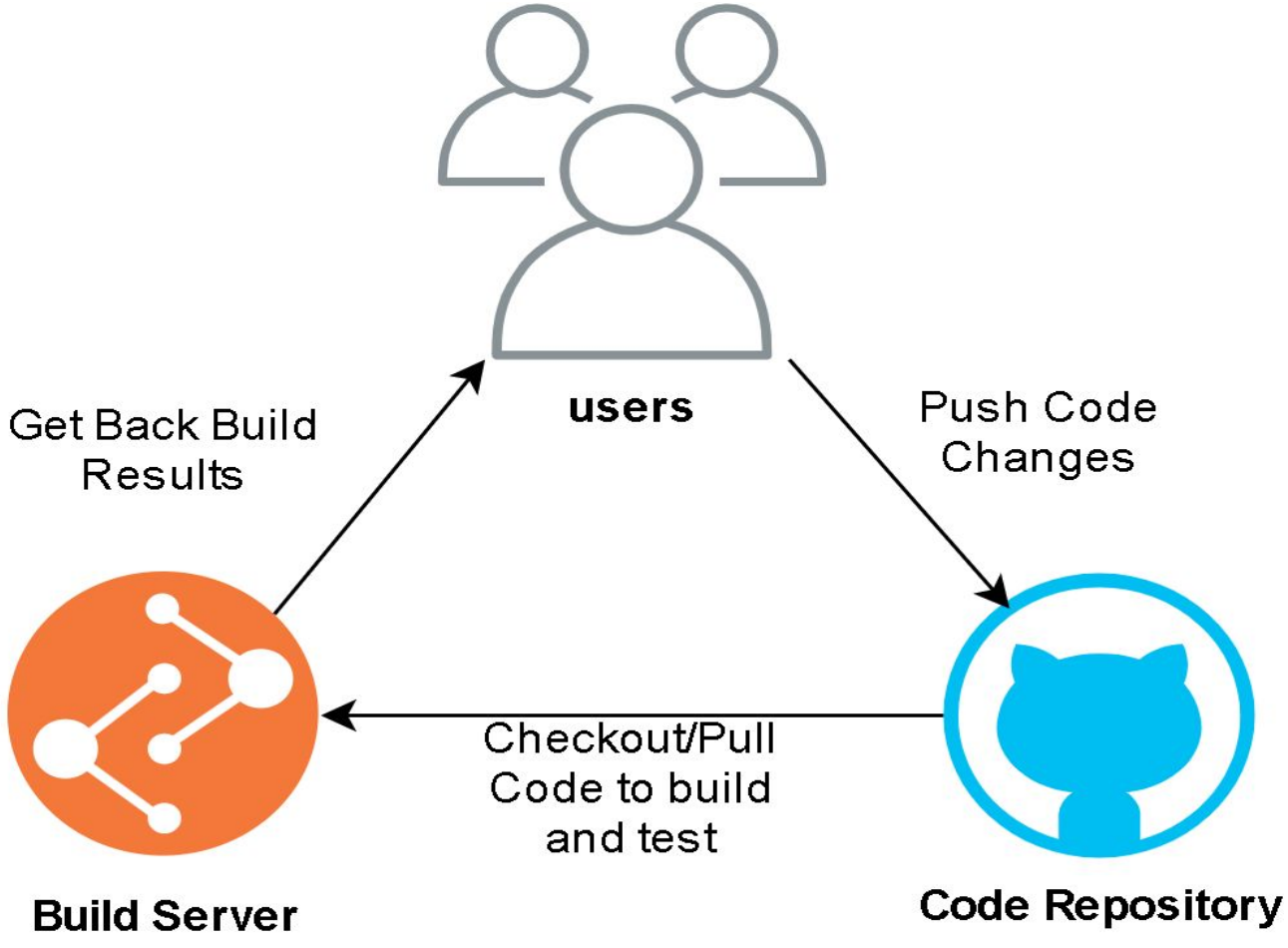
CodeCommit Pricing

- The **first 5 active** users per month are free of charge. You also get to have unlimited repositories, with 50 GB-month total worth of storage, and 10,000 Git requests/month at no cost.
- You are billed for each active user beyond the first 5 per month. You also get an additional 10GB-month of storage per active user, and an additional 2,000 Git requests per active user.
- More details on Pricing here.

Continuous Integration?

- Developers push the code to a **code repository** often (**GitHub / CodeCommit / Bitbucket / Gitlab etc**) - all of these DVCS (git commands are same)
- A **testing / build server** checks/pulls the code from *Code Repo* as soon as it's pushed (**CodeBuild / Jenkins CI , Travis CI , Circle CI** etc)
- The developer gets feedback about the tests and checks that have passed / failed.
- Deliver faster as the code is tested
- Deploy often
- Find bugs early, fix bugs

Continuous Integration



What is AWS CodeBuild?

- AWS CodeBuild is a **fully managed build service**.
- It is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy.
- Execute instructions for Infrastructure or any aws cli commands.

Concepts

- A **build project** defines how CodeBuild will run a build. It includes information such as where to get the source code, which build environment to use, the build commands to run, and where to store the build output.
- A **build environment** is the combination of ***operating system, programming language runtime****, and tools* used by CodeBuild to run a build.
- The **build specification (required)** is a YAML file that lets you choose the commands to run at each phase of the build and other settings.
- Without a build spec, CodeBuild cannot successfully convert your build input into build output or locate the build output artifact in the build environment to upload to your output bucket.
- If you include a build spec as part of the source code, by default, the build spec file must be named **buildspec.yml** and placed in the root of your source directory.
- A collection of input files is called **build input artifacts** or **build input** and a deployable version of a source code is called **build output artifact or build output.**

Features

- AWS CodeBuild runs your builds in preconfigured build environments that contain the operating system, programming language runtime, and build tools (such as Apache Maven, Gradle, npm) required to complete the task.
- You just specify your source codes location and select settings for your build, such as the build environment to use and the build commands to run during a build.
- AWS CodeBuild builds your code and stores the artifacts into an Amazon S3 bucket, or you can use a build command to upload them to an artifact repository.
- AWS CodeBuild provides build environments for:

  Java, Python, Node.js, Ruby, Go, Android, .NET Core for Linux, Docker

Features

Build environment compute types

You can choose from three levels of compute capacity that vary by the amount of CPU and memory to best suit to your development needs

Environment computeType	Memory	vCPUs	Disk Space (GB)
BUILD_GENERAL1_SMALL ***Free Tier : 100 build minutes*** *of build.general1.small per month*	3 GB	2	64
BUILD_GENERAL1_MEDIUM	7 GB	4	128
BUILD_GENERAL1_LARGE	15 GB	8	128
BUILD_GENERAL1_2XLARGE	145 GB	72	824 (SSD)

Features

- You can define the specific commands that you want AWS CodeBuild to perform, such as installing build tool packages, running unit tests, and packaging your code. You can choose from three levels of compute capacity that vary by the amount of CPU and memory to best suit to your development needs
- You can integrate CodeBuild into existing CI/CD workflows using its source integrations, build commands, or Jenkins integration.
- CodeBuild can connect to AWS CodeCommit, S3, GitHub, and GitHub Enterprise and Bitbucket to pull source code for builds.
- You can access your past build results through the console, CloudWatch, or the API.

Steps in a Build Process

1. CodeBuild will create a **temporary compute container** of the class defined in the build project (**Ubuntu**, Amazon Linux, Windows Server)
2. CodeBuild loads it with the specified runtime environment.
3. CodeBuild downloads the source code (Github/Codecommit/S3)
4. CodeBuild executes the commands configured in the project (**buildspec.yml file**)
5. CodeBuild uploads the generated artifact to an S3 bucket.
6. Then it destroys the compute container.
7. All CodeBuild Execution Logs are store in CloudWatch Logs.

Note : Build Duration is calculated in minutes, from the time you submit your build until your build is terminated, rounded up to the nearest minute.

Monitoring and Security

- You can specify a key stored in the AWS Key Management Service to encrypt your artifacts.
- CodeBuild provides security and separation at the infrastructure and execution levels.
- You can use Amazon CloudWatch to watch your builds, report when something is wrong, and take automatic actions when appropriate.
- You can monitor your builds at two levels:
  - At the project level: These metrics are for all builds in the specified project only.
  - At the AWS account level: These metrics are for all builds in one account
- ProjectName is the only AWS CodeBuild metrics dimension. If it is specified, then the metrics are for that project. If it is not specified, then the metrics are for the current AWS account.

AWS CodeBuild Pricing

- You are charged for **compute resources based on the duration it takes for your build to execute**. The per-minute rate depends on the compute type you use.
- For more information click here : **AWS CodeBuild pricing.**

CodeBuild Features

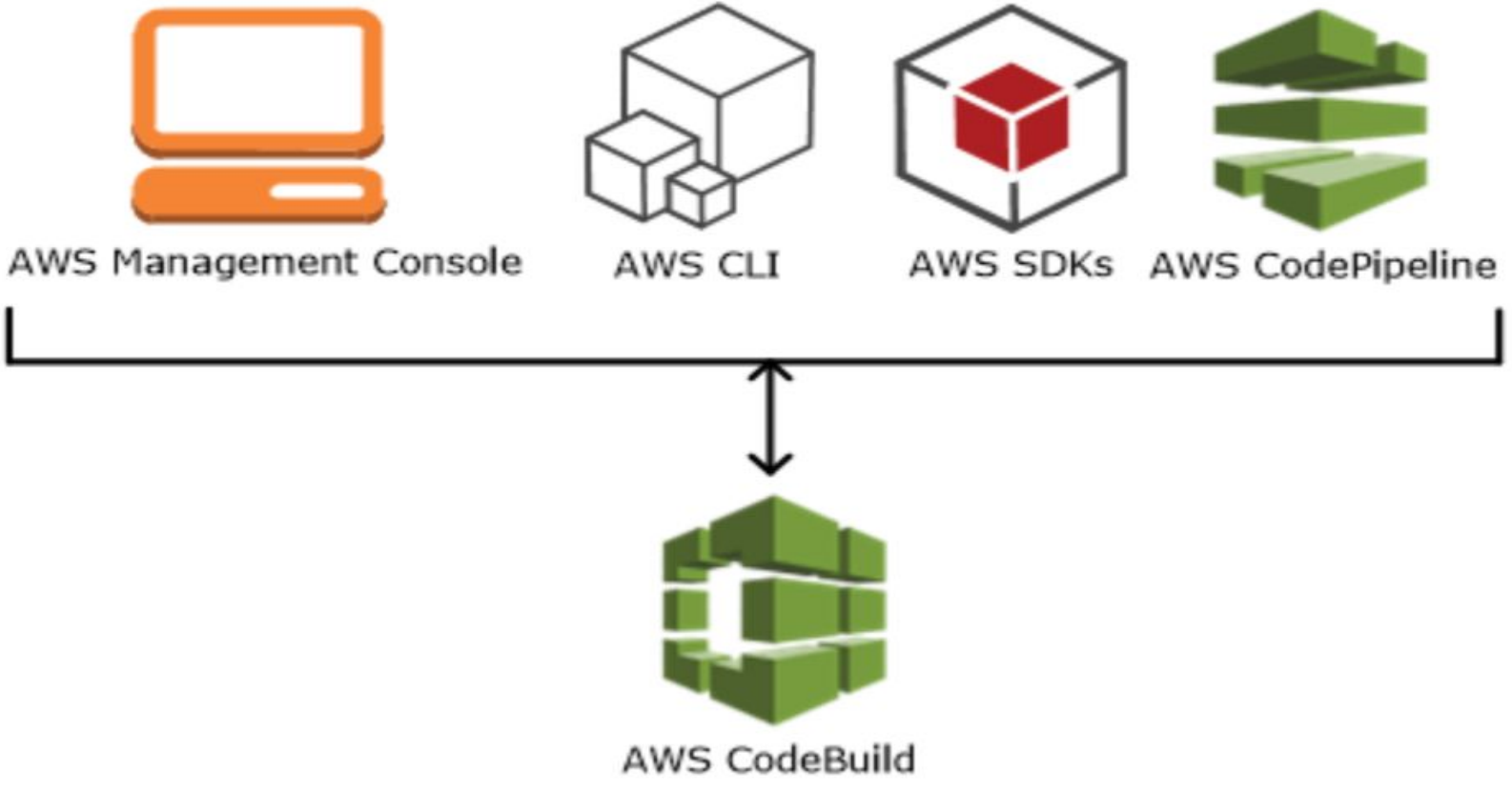
- Compiles your source code, runs unit tests, and produces artifacts that are ready to deploy.
- Eliminates the need to provision, manage, and scale your own build servers.
- It provides prepackaged build environments for the most popular programming languages and build tools such as Apache Maven, Gradle, and more.
- We can also customize build environments in CodeBuild to use our own build tools.
- Scales automatically to meet peak build requests.

CodeBuild Pricing and Usage

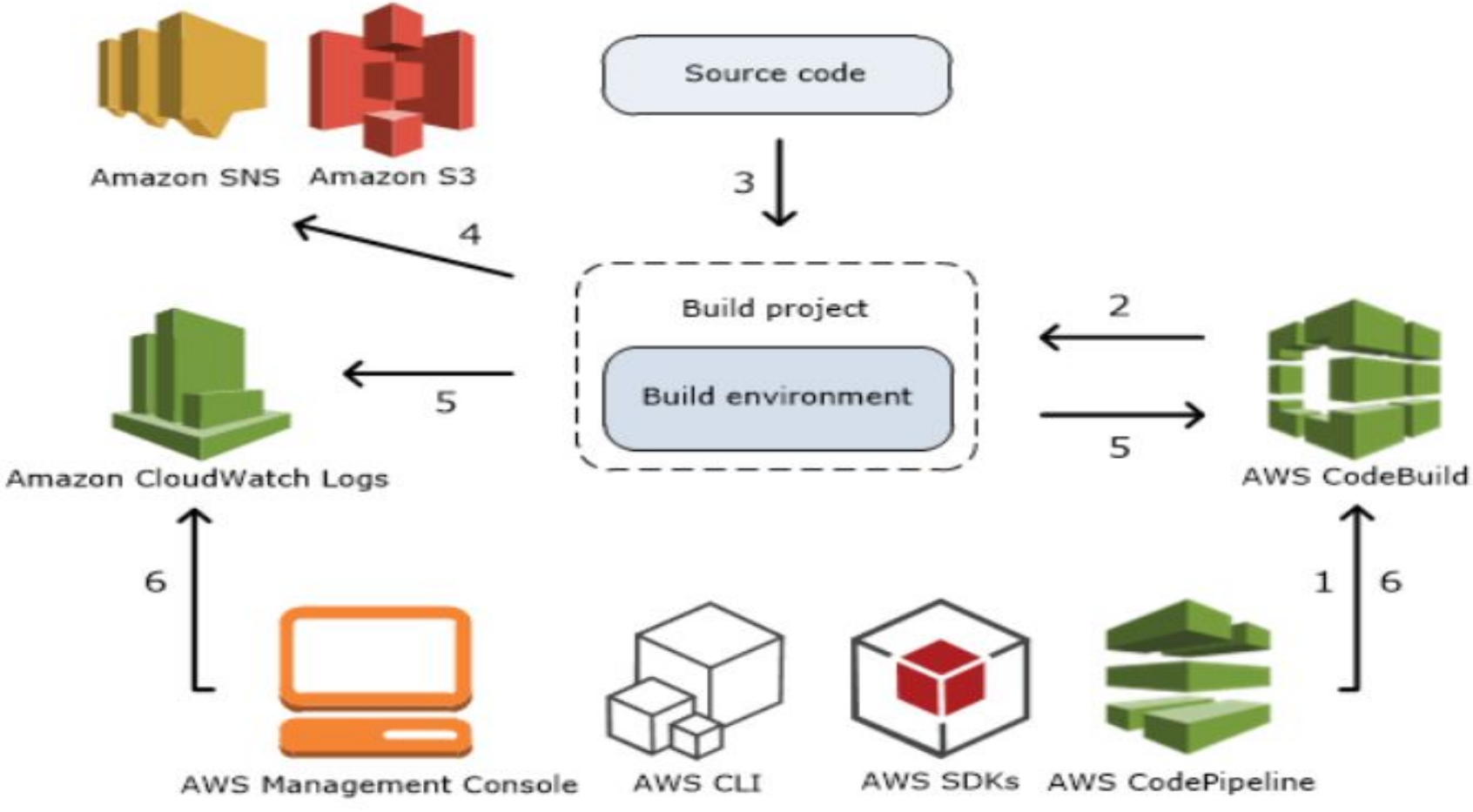
- **Pay for usage**: the time it takes to complete the builds
- **Docker** : CodeBuild leverages Docker under the hood for reproducible builds.
- **Secure:** CodeBuild Uses IAM Service as Build Permissions, Integration with KMS for encryption of build artifacts, and VPC for network security, CloudTrail for API calls logging.

- Source Code from **S3/ GitHub / CodeCommit / CodePipeline.**
- Build instructions are defined in code (***buildspec.yml file***)
- CodeBuild Outputs the Execution logs to Amazon S3 & AWS CloudWatch Logs Metrics to monitor CodeBuild statistics.
- Use CloudWatch Alarms to notify if you need “thresholds” for failures
- SNS notifications
- Use CloudWatch Events to detect failed builds and trigger notifications
- Alternative to other build tools such as Jenkins.

How to run CodeBuild



How CodeBuild Works



Build Environment Reference for CodeBuild

- A build environment contains a **Docker image.**
- When you provide information to CodeBuild about the build environment, you specify the identifier of a Docker image in a supported repository type that can be :

1. *CodeBuild Docker image repository*
2. *Publicly available images in Docker Hub*
3. *Amazon Elastic Container Registry (Amazon ECR) repositories that your AWS account has permissions to access*

Official AWS Github CodeBuild repository for managed Docker images.

https://github.com/aws/aws-codebuild-docker-images

Environment Variables in Build Environment

- AWS CodeBuild provides several environment variables that you can use in your build commands:
- Below are few of them:
  - **AWS_REGION**: The AWS Region where the build is running (for example, us-east-1).
  - **CODEBUILD_BUILD_NUMBER:** The current build number for the project. *This number is useful for semantic versioning of Build Versions.*
  - **CODEBUILD_RESOLVED_SOURCE_VERSION:** An identifier for the version of a build's source code. Its format depends on the source code repository:
  - The URL to the input artifact or source code repository.

    For S3, this is s3:// followed by the bucket name and path to the input artifact.

    For CodeCommit and GitHub, this is the repository's clone URL.

    For CodePipeline, then this might be empty.
  - **CODEBUILD_SRC_DIR**: The directory path that CodeBuild uses for the build (for example, /codebuild/output/src241206719/src/git-codecommit.us-east-1.amazonaws.com/v1/repos/codebuild-cf-stack-test).

Source Version Sample with CodeBuild

- To list all of the available environment variables in a build environment, you can run the **printenv** command (for Linux Based Images) in **buildspec.yml** file
- To specify a GitHub/CodeCommit repository version with a commit ID and reference
- In Source version, enter **refs/heads/master^{046e8b67481d53bdc86c3f6affdd5d1afae6d369}.**

This is the same commit ID and a reference to a branch in the format **refs/heads/branchname^{full-commit-SHA}.**

Create a Notification Rule

- You can use notification rules to notify users when important changes, such as build successes and failures, occur. Notification rules specify both the events and the Amazon SNS topic that is used to send notifications.
- On the build project page, choose **Notify**, and then choose **Create notification rule**.
- In **Events that trigger notifications**, select the events for which we want to send notifications.

Category	Events
Build state	Failed Succeeded In-progress Stopped
Build phase	Failure Success