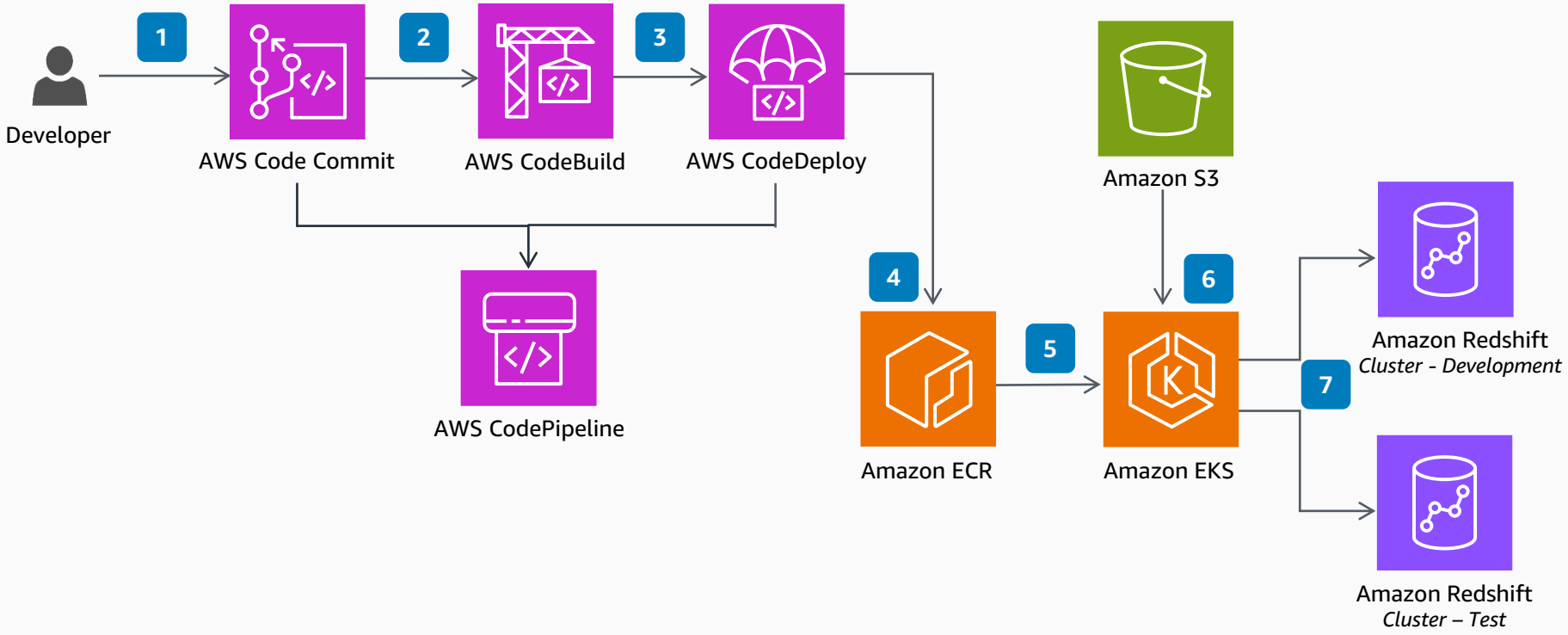
Guidance for DevOps on Amazon Redshift

This architecture displays best practices for running development and operations (DevOps) on Amazon Redshift.



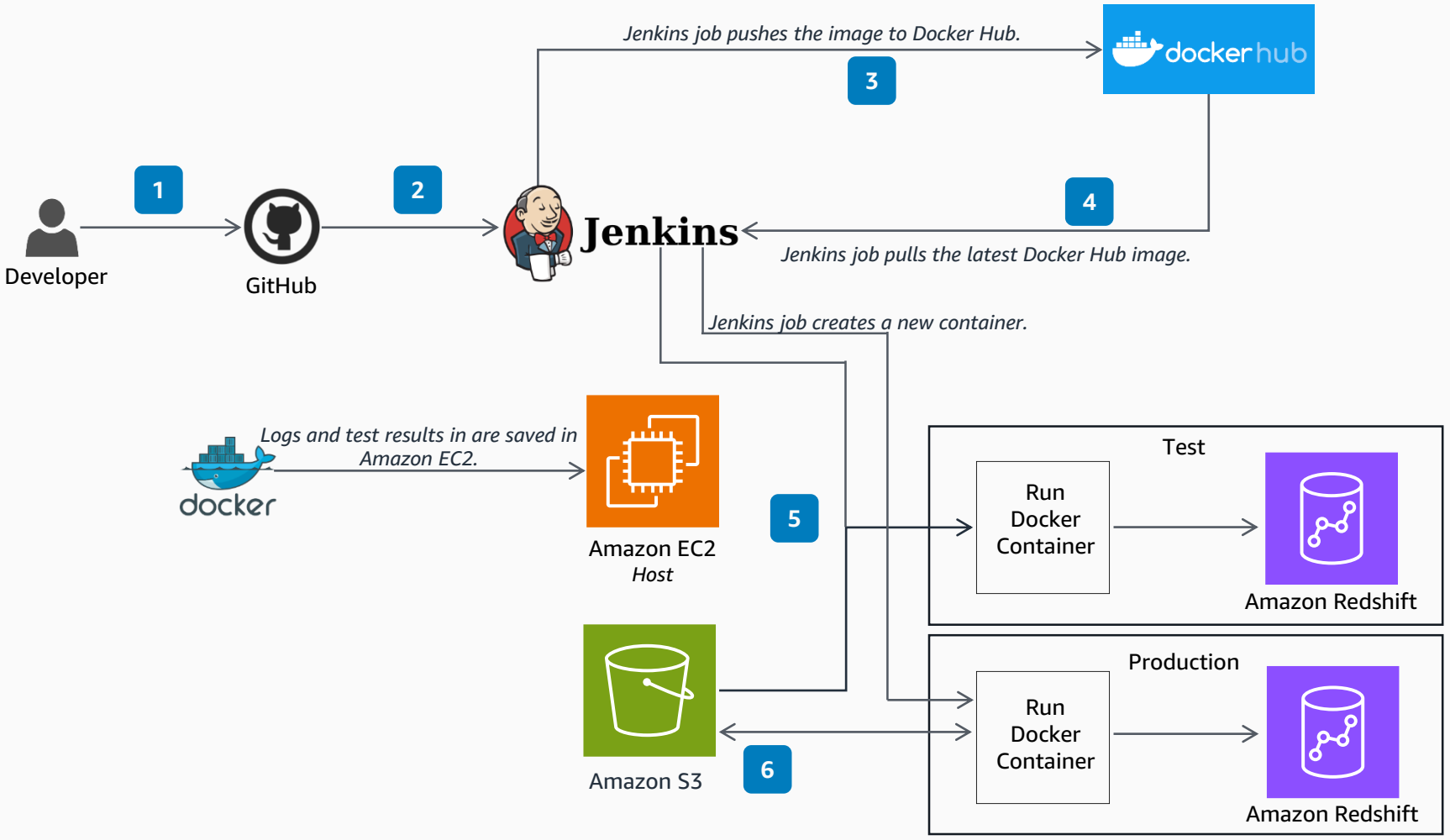
1. Add or modify Data Definition Language or Data Manipulation Language (DDL/DML) scripts in the configuration files. Commit changes into **AWS CodeCommit**.
2. **CodeCommit** invokes a code build using configuration specified in the buildspec.yml file. The build is based on pre_build, build, and post_build commands.
3. **AWS CodeBuild** builds the container image and pushes it to **AWS CodeDeploy**.
4. **CodeDeploy** pushes the container image to the **Amazon Elastic Container Registry** (Amazon ECR) repository.
5. **Amazon Elastic Kubernetes Service** (Amazon EKS) cluster extracts the latest image from **Amazon ECR** and deploys it as a new cluster. A task is defined to run application code like a service.
6. **Amazon EKS** cluster picks up the container image, reads configuration in the **Amazon Simple Storage Service** (Amazon S3) bucket to determine the invoking step, connects with the **Amazon Redshift** cluster, and invokes the DDL/DML code.
7. Code is deployed in the **Amazon Redshift** cluster environment, the task completes, and the deployment service waits for another change to the DDL/DML script.

Reviewed for technical accuracy June 19, 2023

AWS Reference Architecture

Guidance for DevOps on Amazon Redshift

This architecture displays DevOps best practices to build and run open source software on Amazon Redshift.



1. Add or modify DDL/DML scripts in configuration files. Commit changes into GitHub for deployment.
2. GitHub uses a webhook to start the build process on Jenkins. A declarative pipeline job in Jenkins is kicked to start the build process.
3. Jenkins builds a Docker container image using the configuration specified in the Docker file. Once an image is built, the Jenkins job pushes the image to Docker Hub.
4. Jenkins job pulls the latest Docker Hub image, and creates a new container, invoking the DDL/DML scripts specified in the configuration file.
5. A Docker container with the latest image is invoked to deploy the changes in multiple environments. Logs and test results in the Docker container are saved to the mapped **Amazon Elastic Compute Cloud** (Amazon EC2) directory.
6. The code performed in the Docker container checks each performed step from config files, and saves it in an **Amazon S3** bucket. In the event of container restart, the last saved checkpoint is used.

AWS Reference Architecture