



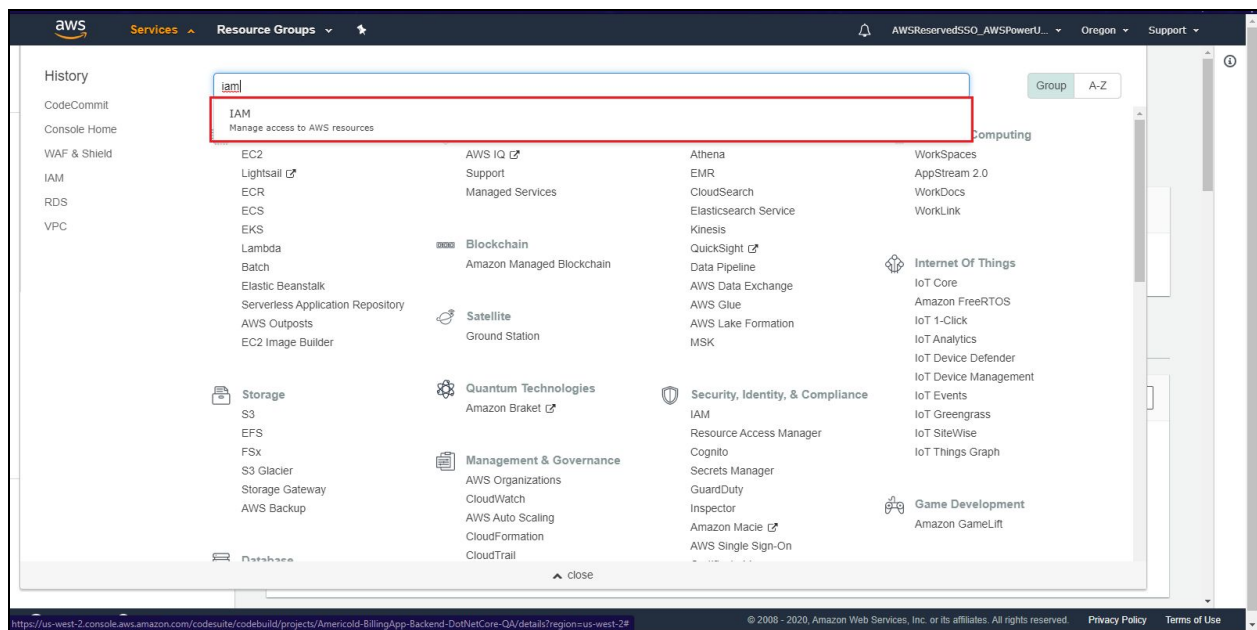
AWS Code Commit

Agenda:

AWS code commit is a fully-managed source control service that hosts secure git-based repositories. It makes it easy for teams to collaborate on code in a secure and highly scalable ecosystem. code commit eliminates the need to operate your own source control system or worry about scaling its infrastructure. You can use code commit to securely store anything from source code to binaries, and it works seamlessly with your existing Git tools.

By the end of the tutorial, we learn how to how to provide access to the developers for generating git credentials, access the repositories, create a repository, and how to create branches, manage the pull requests and code merging from branch to branch.

First, we need to login to the AWS console and search for the **IAM** service.



Following is the dashboard of the IAM service.

Identity and Access Management (IAM)

Dashboard

Access management

- Groups
- Users
- Roles
- Policies
- Identity providers
- Account settings

Access reports

- Access analyzer
- Archive rules
- Analyzer details

Credential report

Organization activity

Service control policies (SCPs)

Search IAM

Welcome to Identity and Access Management

IAM users sign-in link:
<https://646632670602.signin.aws.amazon.com/console> | Customize

IAM Resources

Users: 11 Roles: 149
Groups: 1 Identity Providers: 1
Customer Managed Policies: 126

Security Status 4 out of 4 complete.

- ✓ Activate MFA on your root account
- ✓ Create individual IAM users
- ✓ Use groups to assign permissions
- ✓ Apply an IAM password policy

Feature Spotlight

Introduction to AWS IAM

Additional Information

- [IAM best practices](#)
- [IAM documentation](#)
- [Web Identity Federation Playground](#)
- [Policy Simulator](#)
- [Videos, IAM release history and additional resources](#)

Feedback English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Goto users section in the dashboard and select an IAM user

Identity and Access Management (IAM)

Dashboard

Access management

- Groups
- Users**
- Roles
- Policies
- Identity providers
- Account settings

Access reports

- Access analyzer
- Archive rules
- Analyzer details

Credential report

Organization activity

Service control policies (SCPs)

Search IAM

AWS account ID: 646632670602

Add user Delete user

Find users by username or access key

Showing 11 results

User name	Groups	Access key age	Password age	Last activity	MFA
<input type="checkbox"/> akathuria	Developers	✓ 29 days	None	29 days	Not enabled
<input type="checkbox"/> asagi	Developers	None	65 days	61 days	Not enabled
<input type="checkbox"/> brepalle	Developers	None	76 days	76 days	Not enabled
<input type="checkbox"/> cnuguri	Developers	None	71 days	70 days	Not enabled
<input type="checkbox"/> MBhasuri	Developers	✓ 57 days	71 days	3 days	Not enabled
<input type="checkbox"/> nkenguva	Developers	None	76 days	71 days	Not enabled
<input type="checkbox"/> sa-codeaccess	Developers	✓ 79 days	None	Today	Not enabled
<input type="checkbox"/> sa-elasticsearch	Developers	✓ 76 days	None	None	Not enabled
<input type="checkbox"/> SPitta	None	None	43 days	None	Not enabled
<input type="checkbox"/> spotmuru	Developers	None	70 days	65 days	Not enabled
<input type="checkbox"/> Turbonomic-Connector	None	✓ 47 days	None	Yesterday	Not enabled

Feedback English (US)

© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Now click on add permissions to provide necessary permissions to the IAM user to generate security credentials for git to access git repositories from development environment.

The screenshot shows the AWS IAM console for a user named 'cnuguri'. The 'Summary' tab is active, and the 'Permissions' sub-tab is selected. A red box highlights the list of policies attached to the user. The policies are:

Policy name	Policy type
Attached directly	
AWSCodeCommitFullAccess	AWS managed policy
IAMUserChangePassword	AWS managed policy
Attached from group	
IAMSelfManageServiceSpecificCredentials	AWS managed policy from group Developers
AmazonS3FullAccess	AWS managed policy from group Developers
IAMReadOnlyAccess	AWS managed policy from group Developers

Here we have given permissions for

- **IAMReadOnlyAccess** will provide visibility for IAM dashboard.
- **CodeCommitFull access** to access repositories and can able to create repos and branches for repositories.
- **SelfManageServiceSpecificCredentials** to generate security credentials to manage the repositories from development environment.

To generate security credentials, click on security credentials tab in IAM User dashboard.

The screenshot shows the AWS IAM console for a user named 'cnuguri'. The 'Security credentials' tab is selected. The 'Sign-in credentials' section is visible, showing the console sign-in link, console password, assigned MFA device, and signing certificates. The 'Access keys' section is also visible, showing a table with columns for Access key ID, Created, Last used, and Status. The 'Create access key' button is visible above the table.

Click on generate credentials for HTTPS Git credentials for AWS CodeCommit and download the credentials. These credentials helps us to manage the repositories from development environment.

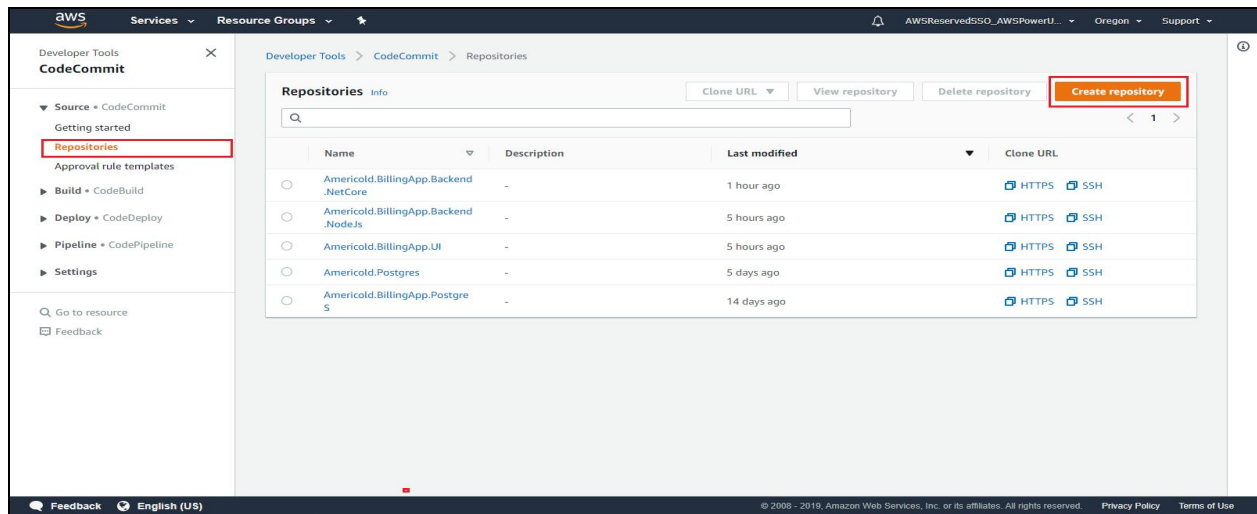
The screenshot shows the AWS IAM console 'Users' page. The left sidebar contains navigation links for Identity and Access Management (IAM), including Dashboard, Access management, Groups, Users, Roles, Policies, Identity providers, Account settings, Access reports, Access analyzer, Archive rules, Analyzer details, Credential report, Organization activity, and Service control policies (SCPs). The main content area has sections for 'Create access key', 'SSH keys for AWS CodeCommit', and 'HTTPS Git credentials for AWS CodeCommit'. The 'Generate credentials' button for the user 'cnuguri-at-646632670602' is highlighted with a red box. The table below shows the user's status and creation date.

User name	Status	Created
cnuguri-at-646632670602	Active	2019-10-30 20:25 UTC+0530

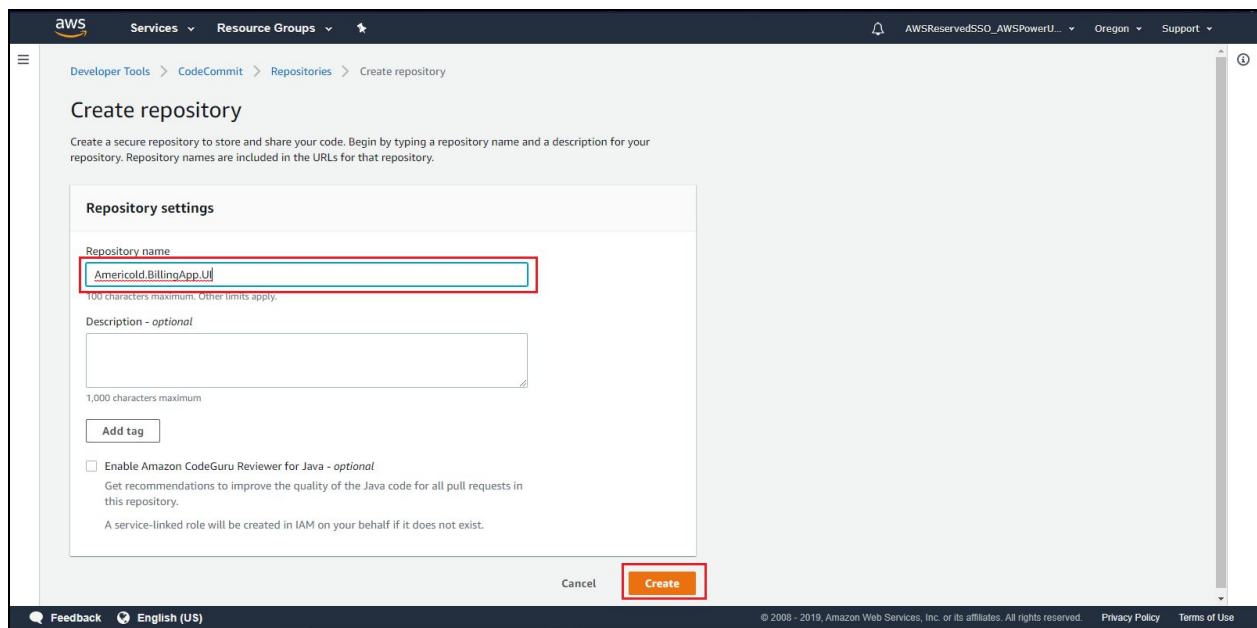
Now search for **CodeCommit Service**.

The screenshot shows the AWS Services search page. The search bar contains the text 'code'. The search results list various AWS services, including Amazon CodeGuru, CodeBuild, CodeCommit, CodeDeploy, CodePipeline, CodeStar, Elastic Transcoder, Cloud9, EFS, FSx, S3 Glacier, Storage Gateway, AWS Backup, Management & Governance, AWS Organizations, CloudWatch, AWS Auto Scaling, CloudFormation, CloudTrail, Resource Access Manager, Cognito, Secrets Manager, GuardDuty, Inspector, Amazon Macie, AWS Single Sign-On, Amazon GameLift, and Amazon GameLift. The 'CodeCommit' service is highlighted with a red box.

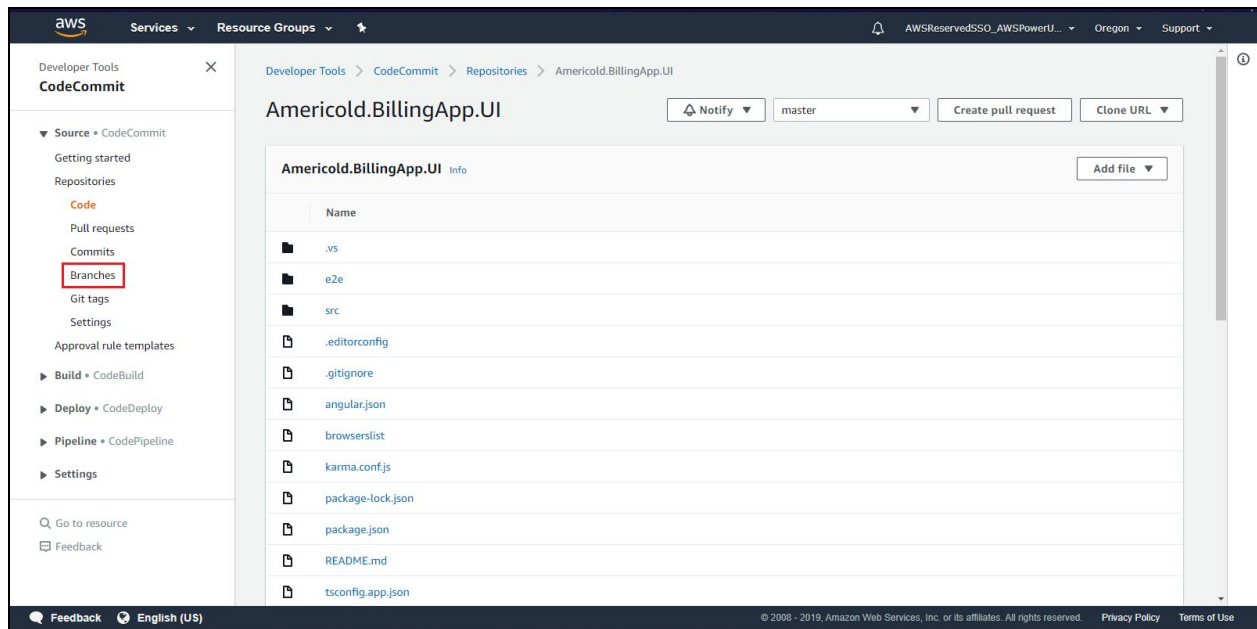
Now goto repositories and Click on **Create repository** to create a new repository.



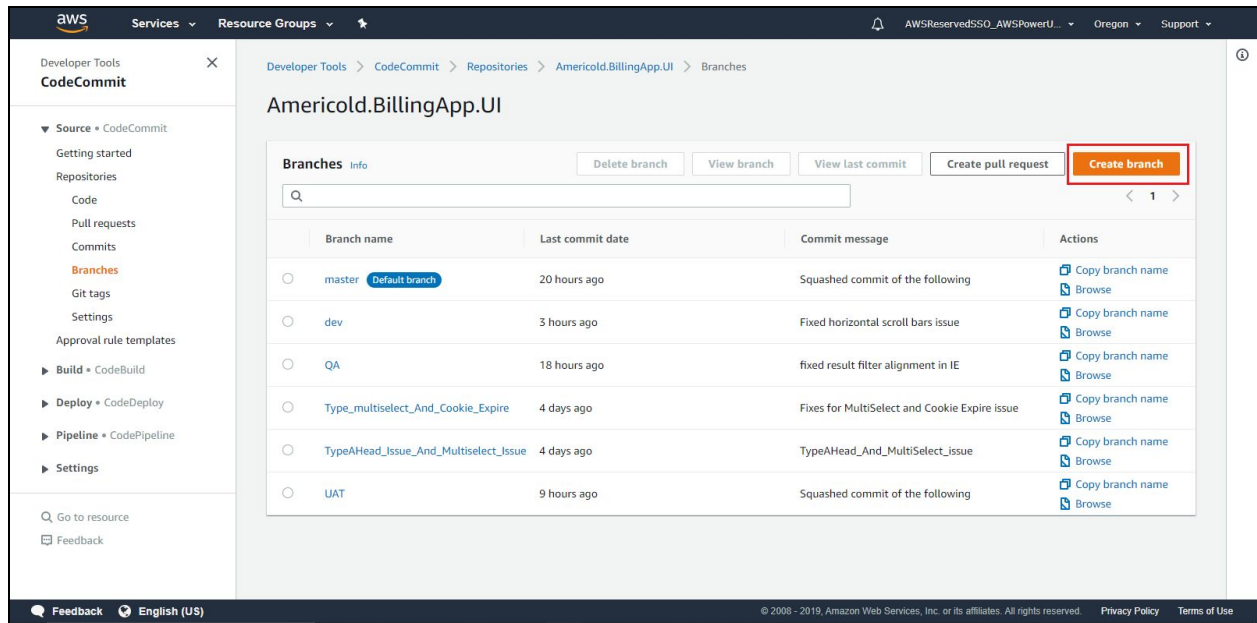
Enter the name of the repository as **Americold_BillingApp_UI** and Click on **Create**.



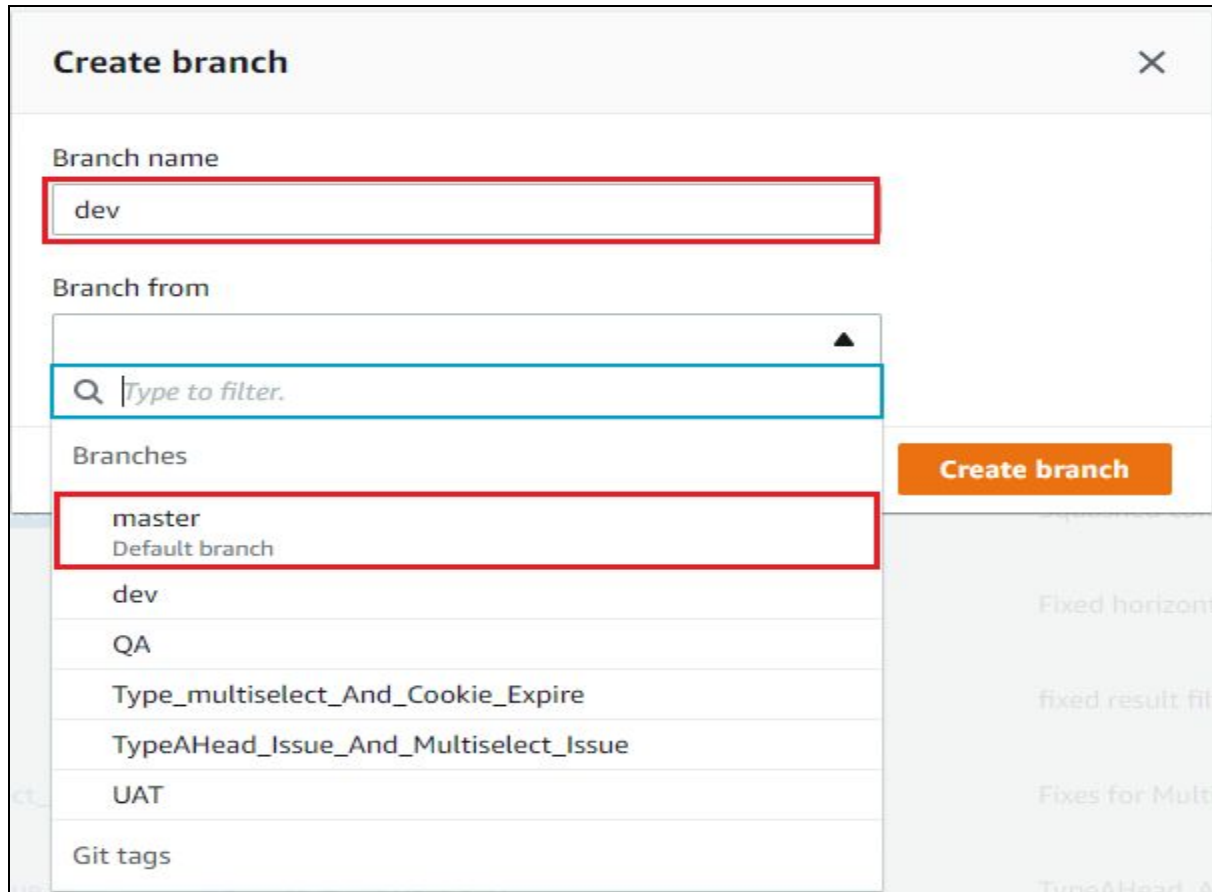
Once the repository got created, go to **branches** section which is shown in the left pane of the dashboard to manage branches for the repository.



Now, Click on create branch to create a new branch in the repository.



Enter the Branch name as **dev** and Select the **master branch** as a parent branch.



Create branch [X]

Branch name
dev

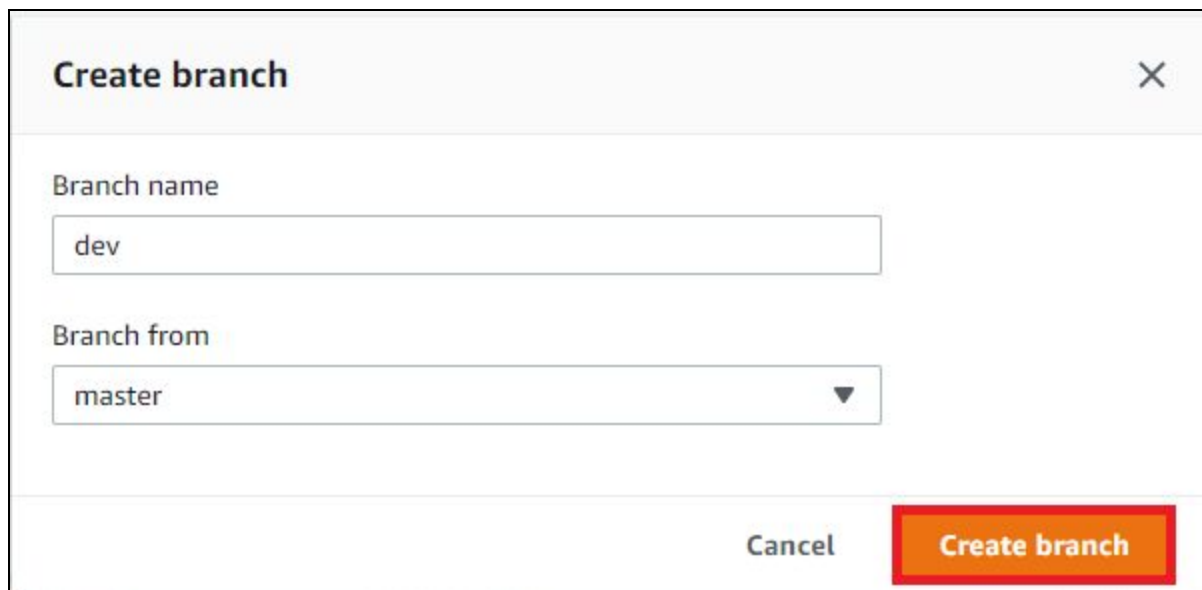
Branch from
[Type to filter.]

Branches

- master (Default branch)
- dev
- QA
- Type_multiselect_And_Cookie_Expire
- TypeAhead_Issue_And_Multiselect_Issue
- UAT
- Git tags

Create branch

Next, Click on **Create branch**.



Create branch [X]

Branch name
dev

Branch from
master

Cancel Create branch

We can also see the Commit history of the repository in commits section.

The screenshot shows the AWS CodeCommit interface for the repository 'Americold.BillingApp.UI'. The left sidebar contains a navigation menu with 'Commits' highlighted. The main content area shows the 'Commits' tab selected, displaying a list of commits. The table below represents the data shown in the screenshot:

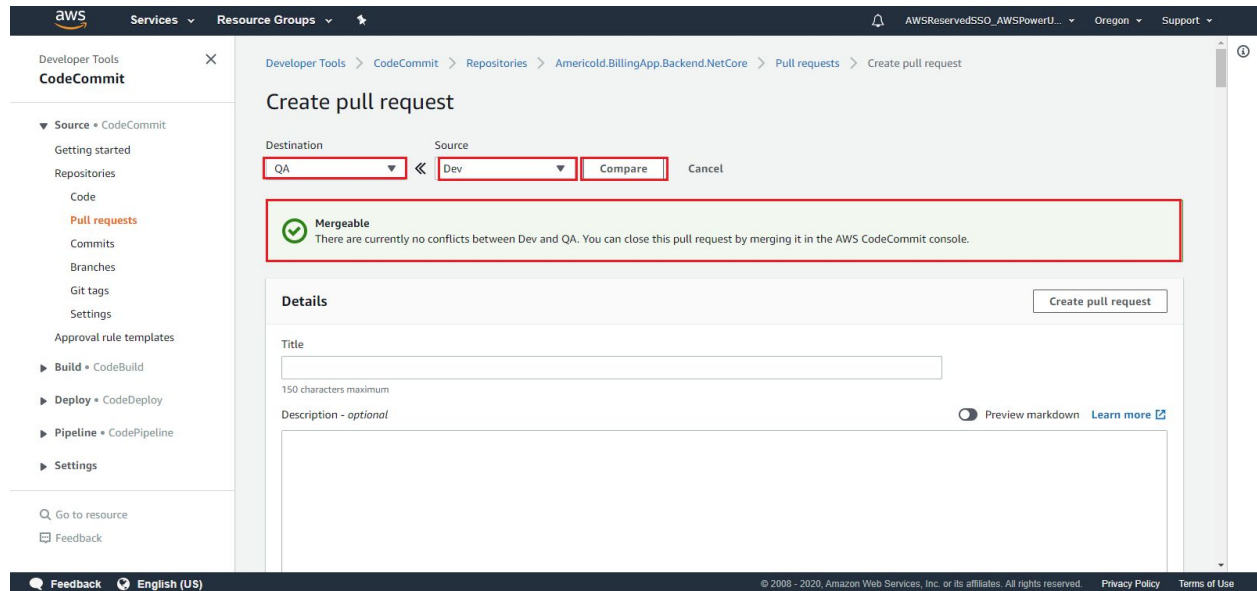
Commit ID	Commit message	Commit date	Author	Actions
27f242db	Squashed commit of the following	20 hours ago	Chandra Nuguri	Copy ID , Browse
fc5de2ea	Conflicts resolved in the console.	2 days ago	AWSReservedSSO_AWSPowerUserAccess_250b81458cac5df9/cnuguri@AMERICOLD.COM	Copy ID , Browse
57f1fc84	Conflicts resolved in the console.	2 days ago	AWSReservedSSO_AWSPowerUserAccess_250b81458cac5df9/cnuguri@AMERICOLD.COM	Copy ID , Browse
e2c16897	Resolving merge conflict issue	2 days ago	Chandra Nuguri	Copy ID , Browse
2cce1fb1	[Chandra Nuguri]: Updated the logic in consolidation search criteria w.r.t multi...	2 days ago	Chandra Nuguri	Copy ID , Browse

Merging code from one branch to another branch:

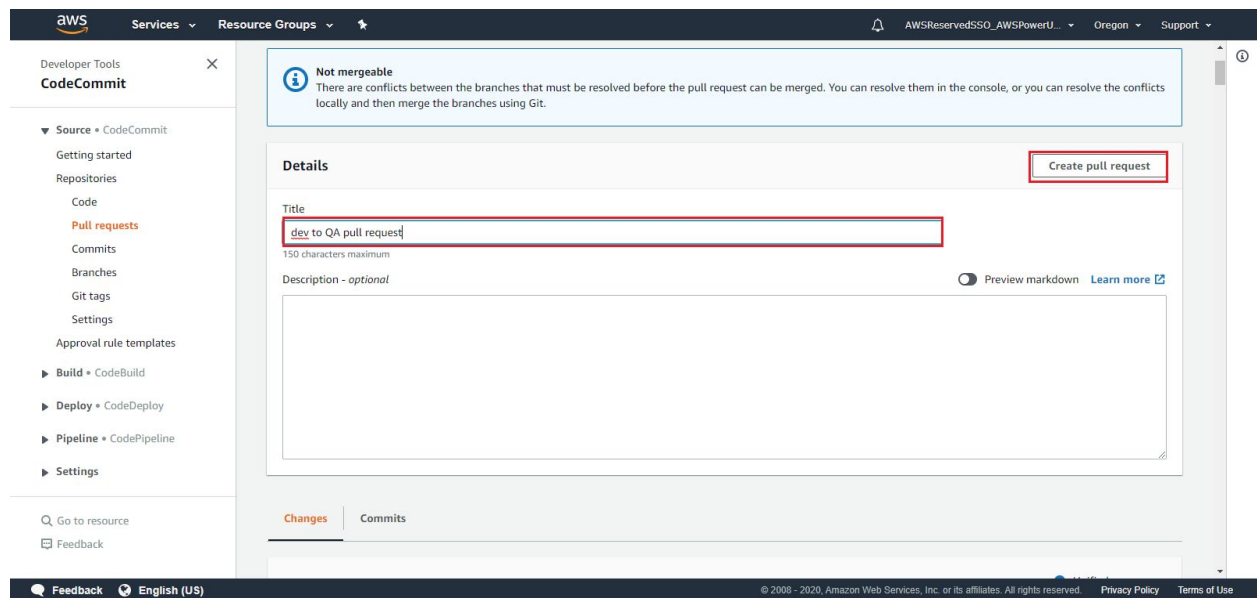
Currently we are in dev branch. To merge code from dev branch to QA we need to create a pull request. Now click on create pull request.

The screenshot shows the AWS CodeCommit interface for the repository 'Americold.BillingApp.UI'. The left sidebar contains a navigation menu with 'Code' highlighted. The main content area shows the 'Code' tab selected, displaying a file explorer view of the repository. The file explorer shows a directory structure with files like .vs, e2e, src, .editorconfig, .gitignore, angular.json, browserslist, karma.conf.js, package-lock.json, package.json, and README.md. The 'Create pull request' button is highlighted in the top right corner.

Select the source and destination branches and click on compare. If mergeable it will show the status as mergeable.



If it is mergeable give a title to the pull request and click on create pull request.



Now click on merge to merge the code from dev to QA

The screenshot shows the AWS CodeCommit console interface. On the left is a navigation pane with 'Developer Tools' and 'CodeCommit' selected. Under 'CodeCommit', there's a list of options: 'Getting started', 'Repositories', 'Code', 'Pull requests' (highlighted in orange), 'Commits', 'Branches', 'Git tags', 'Settings', and 'Approval rule templates'. Below these are 'Build' (CodeBuild), 'Deploy' (CodeDeploy), 'Pipeline' (CodePipeline), and 'Settings'. The main content area shows the details of a pull request titled '143: dev to QA pull request'. At the top right of this area are buttons for 'Close pull request' and 'Merge'. Below the title, there are status indicators: 'Open', 'No approval rules', and 'No merge conflicts'. It also shows 'Destination: QA', 'Source: Dev', 'Author: asagi@AMERICOLD.COM', and 'Approvals: 0'. There are tabs for 'Details', 'Activity', 'Changes', 'Commits', and 'Approvals'. The 'Details' tab is active, showing a message: 'This pull request does not have a description.' and an 'Edit details' button. The footer of the console shows the URL 'https://us-west-2.console.aws.amazon.com/codesuite/codecommit/repositories/Americold.BillingApp.Backend.NetCore/pull-requests/143/merge?region=us-west-2' and copyright information.

Once we click on merge in above step it redirects us to the following dashboard. Here we need to select a merge strategy for merging. Here we are selecting squash and merge.

The screenshot shows the 'Merge pull request 143: dev to QA pull request' dashboard in the AWS CodeCommit console. The left navigation pane is the same as in the previous screenshot. The main content area is titled 'Merge pull request 143: dev to QA pull request'. It shows 'Pull request: 143 dev to QA pull request' and 'Destination: QA', 'Source: Dev'. Under 'Merge strategy info', it states: 'Determines the way in which the current pull request will be merged into the destination branch'. There are three strategy options: 'Fast forward merge' (selected with a radio button), 'Squash and merge' (highlighted with a red box and selected with a radio button), and '3-way merge' (unselected with a radio button). Each option includes a brief description and a diagram. Below the strategies, there's a section for 'Commit message - optional' with a 'Preview markdown' toggle. It shows a 'Squashed commit of the following' with details: 'commit 059da405', 'Author: Nagabhushan <nkenguva@americold.com>', and 'Date: Mon Jan 13 2020 18:29:43 GMT+0530 (India Standard Time)'. The footer shows 'Feedback', 'English (US)', and copyright information.

Now give the name and email address of the author who is merging the code. By default delete source branch will be selected, we need to deselect the checkbox.

The screenshot shows the AWS CodeCommit 'Merge pull request' interface. On the left, a sidebar lists navigation options: Developer Tools, CodeCommit, Source, Getting started, Repositories, Code, Pull requests (highlighted), Commits, Branches, Git tags, Settings, Approval rule templates, Build, Deploy, Pipeline, and Settings. The main area displays three merge strategies: 'git merge --ff-only', 'git merge --squash' (selected), and 'git merge --no-ff'. Below these, the 'Commit message - optional' section shows a 'Squashed commit of the following' with commit details: 'commit 059da405', 'Author: Nagabhushan <nkenguva@americold.com>', and 'Date: Mon Jan 13 2020 18:29:43 GMT+0530 (India Standard Time)'. It also lists 'Changes wrt dynamic date range' with 'commit 4173f829'. The 'Author name' field is 'Anjaneya Sagi' and the 'Email address' field is 'anjaneya.sagi@americold.com'. The checkbox 'Delete source branch Dev after merging?' is checked. At the bottom right, there are 'Cancel' and 'Merge pull request' buttons.

After deselecting the checkbox click on merge pull request, then the QA pipeline will automatically triggers and deployment to QA environment will be done.

This screenshot shows the same AWS CodeCommit 'Merge pull request' interface as the previous one, but with the 'Delete source branch Dev after merging?' checkbox unchecked. All other details, including the author information and commit messages, remain the same. The 'Merge pull request' button is still visible at the bottom right.