



CREATE THE PRODUCT

What are we going to do?

We have provisioned a detective control to look for AWS RDS Instances that have don't have encryption enabled. We can do better, and create an AWS Service Catalog product that meets the encryption requirement by default using service catalog tools. When users create a new RDS instance using this product, encryption at rest is enabled by default and no further configuration is required.

We are going to perform the following steps:

- define a product with a version and a portfolio
- add the source code for our product
- share that portfolio with a spoke account

Step by step guide

Here are the steps you need to follow to “Create the product”

Define a product with a version and a portfolio

- Navigate to the [ServiceCatalogFactory CodeCommit repository](#) again
- Click on *portfolios*

The screenshot shows the AWS Developer Tools console. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and a star icon. The left sidebar is titled 'Developer Tools' and shows 'CodeCommit' as the selected tool. Under 'CodeCommit', there are links for 'Getting started', 'Repositories', 'Code' (highlighted in orange), 'Pull requests', 'Commits', 'Branches', 'Git tags', and 'Settings'. Below these are links for 'Build' (CodeBuild), 'Deploy' (CodeDeploy), and 'Pipeline' (CodePipeline). At the bottom of the sidebar are 'Go to resource' and 'Feedback' links. The main content area shows the breadcrumb 'Developer Tools > CodeCommit > Repositories > ServiceCatalogFactory'. The title 'ServiceCatalogFactory' is displayed. Below the title are two buttons: 'master' (with a dropdown arrow) and 'Create pull request'. To the right of these is a 'Clone URL' button (with a dropdown arrow). Below this is a section titled 'ServiceCatalogFactory' with an 'Info' link and an 'Add file' button (with a dropdown arrow). A table below this section has a header 'Name' and one row with a folder icon and the text 'portfolios'.

- Click on *reinvent.yaml*

The screenshot shows the AWS CodeCommit console. The left sidebar is titled "Developer Tools" and "CodeCommit". It lists various actions: "Source" (selected), "Getting started", "Repositories", "Code" (highlighted in orange), "Pull requests", "Commits", "Branches", "Git tags", "Settings", "Build" (CodeBuild), "Deploy" (CodeDeploy), and "Pipeline" (CodePipeline). Below these is a search bar "Go to resource" and a "Feedback" link. The main content area shows the breadcrumb "Developer Tools > CodeCommit > Repositories > ServiceCatalogFactory". The repository name "ServiceCatalogFactory" is displayed at the top. Below it, there is a dropdown menu showing "master", a "Create pull request" button, and a "Clone URL" button. A table lists the files in the repository: a directory icon for ".." and a file icon for "reinvent.yaml". An "Add file" button is visible in the top right of the file list.

- Click *Edit*

This screenshot shows the same AWS CodeCommit console as the previous one, but with the "reinvent.yaml" file selected. The breadcrumb now includes the file name: "Developer Tools > CodeCommit > Repositories > ServiceCatalogFactory". The file list shows "reinvent.yaml" with a file icon. An "Edit" button is now visible next to the file name. The file content is displayed in a text area, showing line numbers 1 through 6.

- Add the following to the products section:

```

- Name: "rds-instance"
  Owner: "data-governance@example.com"
  Description: "A compliant RDS Instance you can use that meets
data governance standards"
  Distributor: "cloud-engineering"
  SupportDescription: "Speak to data-governance@example.com about
exceptions and speak to cloud-engineering@example.com about
implementation issues"
  SupportEmail: "cloud-engineering@example.com"
  SupportUrl: "https://wiki.example.com/cloud-engineering/data-
governance/rds-instance"
  Options:
    ShouldCFNNag: True
  Tags:
    - Key: "type"
      Value: "governance"
    - Key: "creator"
      Value: "cloud-engineering"
  Versions:
    - Name: "v1"
      Description: "v1 of rds-instance"
      Active: True
      Source:
        Provider: "CodeCommit"
        Configuration:
          RepositoryName: "rds-instance"
          BranchName: "master"
  Portfolios:
    - "cloud-engineering-self-service"

```

- Add the following to the portfolios section:

```

- DisplayName: "cloud-engineering-self-service"
  Description: "Portfolio containing products that you can use
to ensure you meet the governance guidelines"
  ProviderName: "cloud-engineering"
  Associations:
    - "arn:aws:iam::${AWS::AccountId}:role/TeamRole"
  Tags:
    - Key: "type"
      Value: "governance"
    - Key: "creator"
      Value: "cloud-engineering"

```

- Once completed it should look like this:

```

Schema: factory-2019-04-01
Products:

```

```
- Name: "aws-config-desired-instance-types"
  Owner: "budget-and-cost-governance@example.com"
  Description: "Enables AWS Config rule - desired-instance-type
with our RIs"
  Distributor: "cloud-engineering"
  SupportDescription: "Speak to budget-and-cost-
governance@example.com about exceptions and speak to cloud-
engineering@example.com about implementation issues"
  SupportEmail: "cloud-engineering@example.com"
  SupportUrl: "https://wiki.example.com/cloud-
engineering/budget-and-cost-governance/aws-config-desired-
instance-types"
  Tags:
    - Key: "type"
      Value: "governance"
    - Key: "creator"
      Value: "cloud-engineering"
    - Key: "cost-center"
      Value: "governance"
  Versions:
    - Name: "v1"
      Description: "v1 of aws-config-desired-instance-types"
      Active: True
      Source:
        Provider: "CodeCommit"
        Configuration:
          RepositoryName: "aws-config-desired-instance-types"
          BranchName: "master"
  Portfolios:
    - "cloud-engineering-governance"

- Name: "aws-config-rds-storage-encrypted"
  Owner: "data-governance@example.com"
  Description: "Enables AWS Config rule - aws-config-rds-
storage-encrypted"
  Distributor: "cloud-engineering"
  SupportDescription: "Speak to data-governance@example.com
about exceptions and speak to cloud-engineering@example.com about
implementation issues"
  SupportEmail: "cloud-engineering@example.com"
  SupportUrl: "https://wiki.example.com/cloud-engineering/data-
governance/aws-config-rds-storage-encrypted"
  Tags:
    - Key: "type"
      Value: "governance"
    - Key: "creator"
      Value: "cloud-engineering"
    - Key: "cost-center"
      Value: "governance"
  Versions:
    - Name: "v1"
      Description: "v1 of aws-config-rds-storage-encrypted"
      Active: True
      Source:
        Provider: "CodeCommit"
        Configuration:
          RepositoryName: "aws-config-rds-storage-encrypted"
```

```
        BranchName: "master"
Portfolios:
  - "cloud-engineering-governance"

  - Name: "rds-instance"
    Owner: "data-governance@example.com"
    Description: "A compliant RDS Instance you can use that meets
data governance standards"
    Distributor: "cloud-engineering"
    SupportDescription: "Speak to data-governance@example.com
about exceptions and speak to cloud-engineering@example.com about
implementation issues"
    SupportEmail: "cloud-engineering@example.com"
    SupportUrl: "https://wiki.example.com/cloud-engineering/data-
governance/rds-instance"
    Options:
      ShouldCFNNag: True
    Tags:
      - Key: "type"
        Value: "governance"
      - Key: "creator"
        Value: "cloud-engineering"
    Versions:
      - Name: "v1"
        Description: "v1 of rds-instance"
        Active: True
        Source:
          Provider: "CodeCommit"
          Configuration:
            RepositoryName: "rds-instance"
            BranchName: "master"
Portfolios:
  - "cloud-engineering-self-service"

Portfolios:
  - DisplayName: "cloud-engineering-governance"
    Description: "Portfolio containing the products needed to
govern AWS accounts"
    ProviderName: "cloud-engineering"
    Associations:
      - "arn:aws:iam::${AWS::AccountId}:role/TeamRole"
    Tags:
      - Key: "type"
        Value: "governance"
      - Key: "creator"
        Value: "cloud-engineering"
      - Key: "cost-center"
        Value: "governance"

  - DisplayName: "cloud-engineering-self-service"
    Description: "Portfolio containing products that you can use
to ensure you meet the governance guidelines"
    ProviderName: "cloud-engineering"
    Associations:
      - "arn:aws:iam::${AWS::AccountId}:role/TeamRole"
    Tags:
      - Key: "type"
```

```
Value: "governance"  
- Key: "creator"  
Value: "cloud-engineering"
```

Note

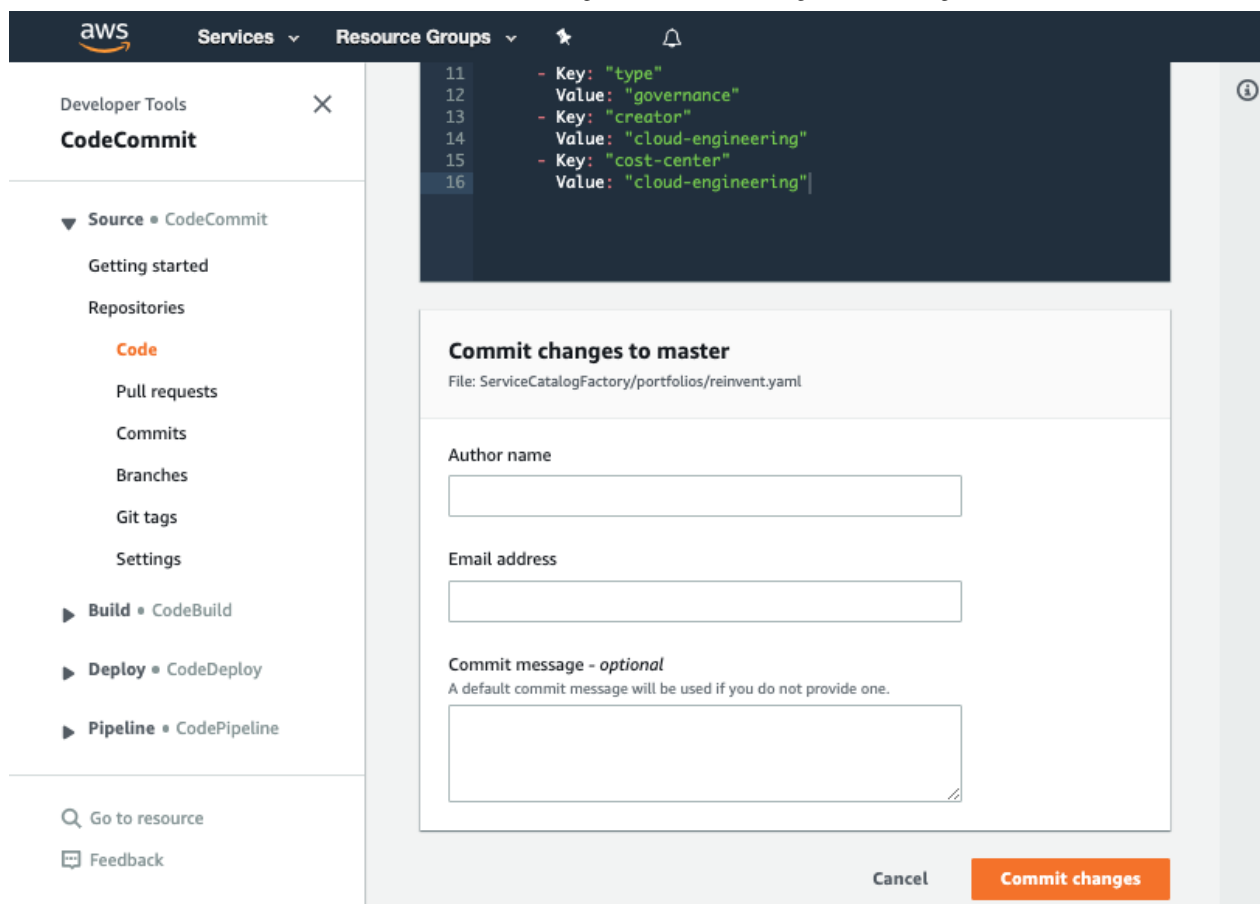
Have a look at the highlighted lines. We are using this to turn on *cfn-nag*, an open source tool by Stelligent that looks for insecure configuration of resources. This will add an extra layer of governance ensuring the AWS CloudFormation templates we are using meets the quality bar set by *cfn-nag*.

- Set your *Author name*
- Set your *Email address*
- Set your *Commit message*

Tip

Using a good / unique commit message will help you understand what is going on later.

- Click the *Commit changes* button:



What did we just do?

The YAML we pasted in the previous step told the framework to perform several actions:

- create a product named *rds-instance*
- add a *v1* of our product
- create a portfolio named *cloud-engineering-self-service*
- add the product: *rds-instance* to the portfolio: *cloud-engineering-self-service*

Verify the change worked

Once you have made your changes the [ServiceCatalogFactory Pipeline](#) should have run. If you were very quick, the pipeline may still be running. If it has not yet started feel free to hit the *Release change* button.

Once it has completed it should show the *Source* and *Build* stages in green to indicate they have completed successfully:

The screenshot shows the AWS Service Catalog console interface. On the left, a sidebar contains navigation links: Developer Tools, CodePipeline, Source (CodeCommit), Build (CodeBuild), Deploy (CodeDeploy), Pipeline (CodePipeline), Getting started, Pipelines, Pipeline (highlighted), History, and Settings. Below these are search and feedback options. The main content area displays the 'Source' stage of a CodePipeline pipeline, which is 'Succeeded' and completed 1 minute ago. Below the Source stage is a 'Build' stage, also 'Succeeded' and completed 'Just now'. A 'Disable transition' button is visible between the two stages. Both stages show the commit ID 'ac2c822c' and the provider 'CodeCommit: dfsfd'. A 'View current revisions' button is present for each stage. On the right side of the console, there are two green checkmark icons in a vertical column.

Note

If this is failing please raise your hand for some assistance

Add the source code for our product

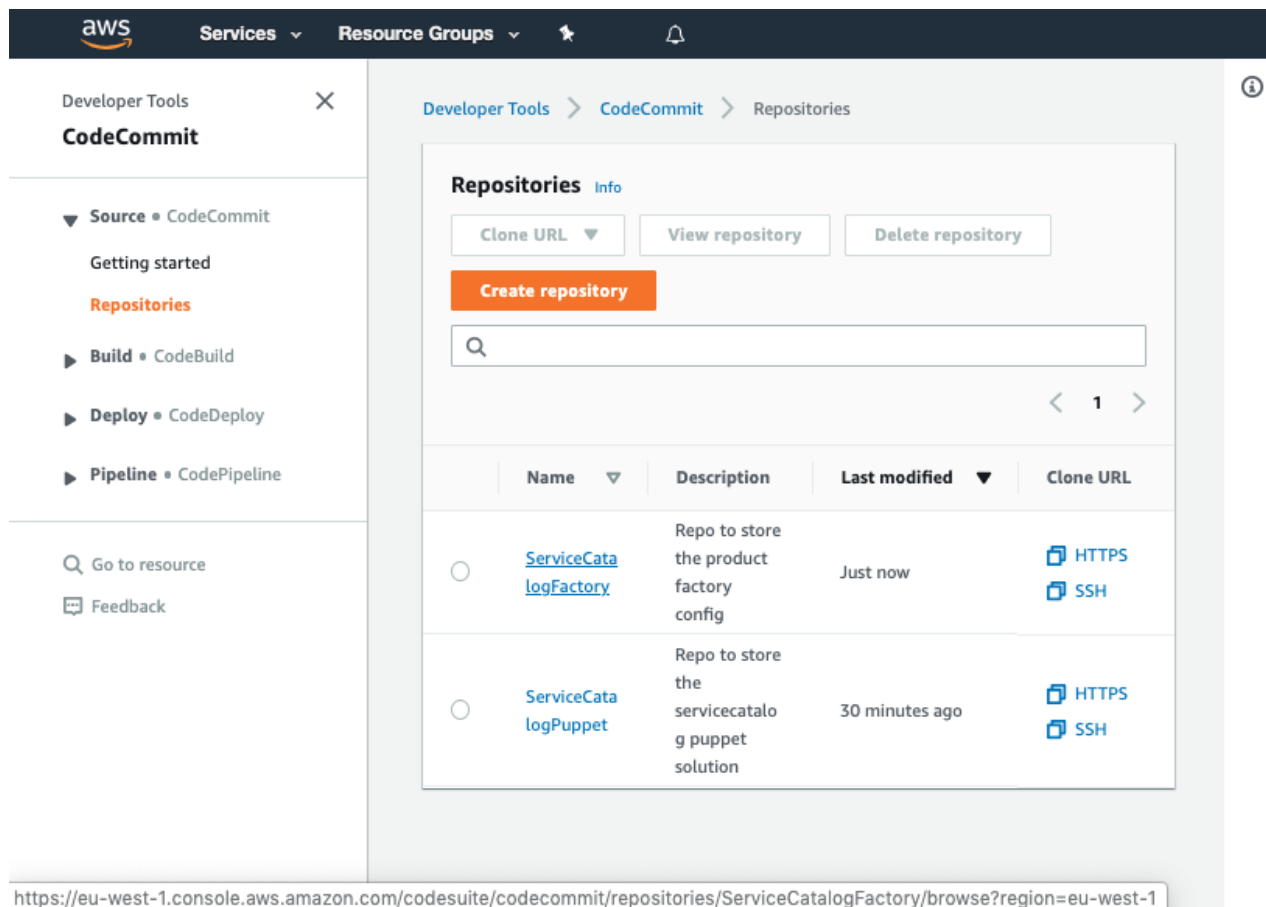
When you configured your product version, you specified the following version:

```
Versions:
- Name: "v1"
  Description: "v1 of rds-instance"
  Active: True
  Source:
    Provider: "CodeCommit"
    Configuration:
      RepositoryName: "rds-instance"
      BranchName: "master"
```

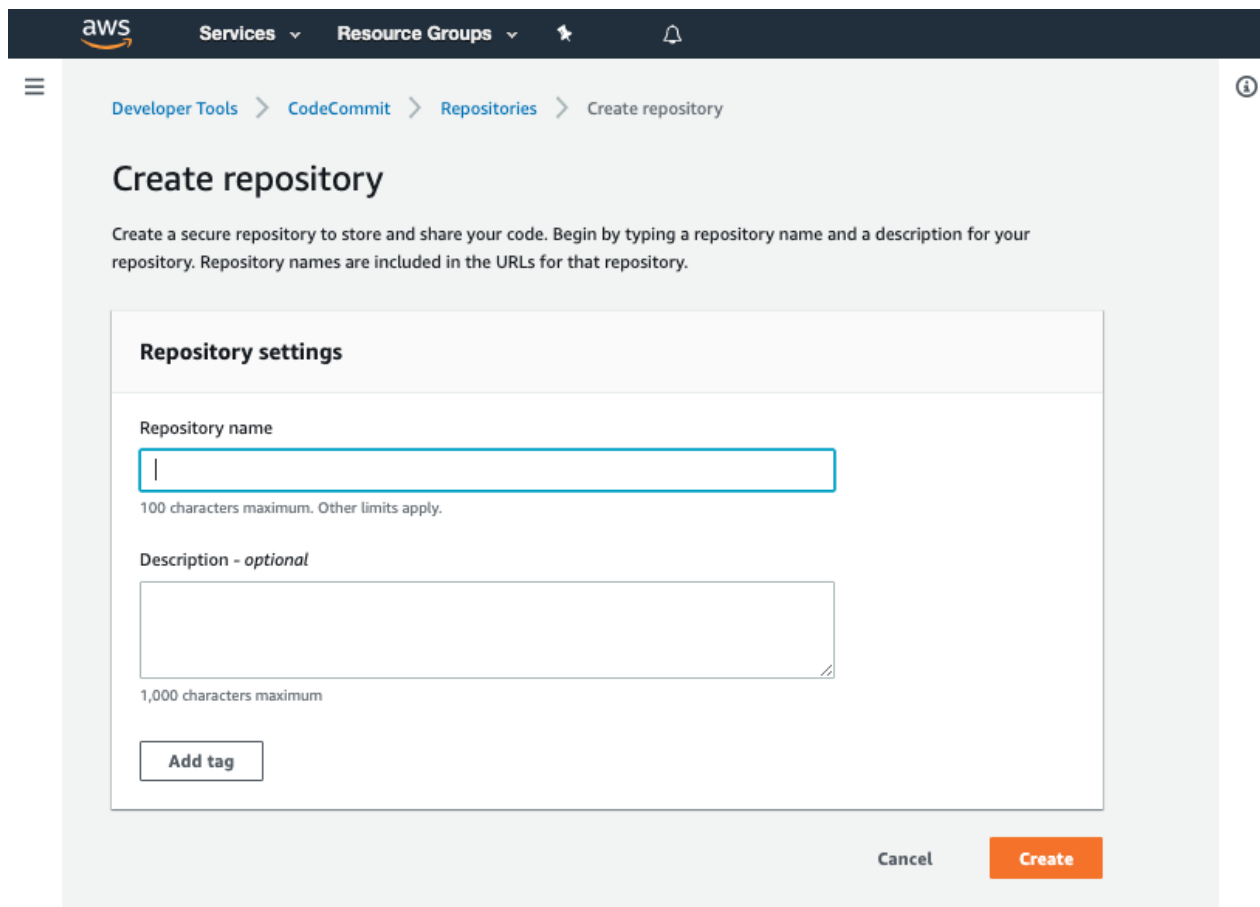
This tells the framework the source code for the product comes from the *master* branch of a *CodeCommit* repository of the name *rds-instance*.

We now need to create the CodeCommit repository and add the AWS CloudFormation template we are going to use for our product.

- Navigate to [AWS CodeCommit](#)
- Click *Create repository*



- Input the name `rds-instance`



The screenshot shows the AWS IAM console interface for creating a new repository. At the top, the navigation bar includes the AWS logo, 'Services', 'Resource Groups', and a search icon. The breadcrumb trail reads: 'Developer Tools > CodeCommit > Repositories > Create repository'. The main heading is 'Create repository'. Below it, a paragraph states: 'Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.' The 'Repository settings' section contains a 'Repository name' text box with a note '100 characters maximum. Other limits apply.' and an optional 'Description' text box with a note '1,000 characters maximum'. An 'Add tag' button is located below the description box. At the bottom right, there are 'Cancel' and 'Create' buttons.

aws Services Resource Groups

Developer Tools > CodeCommit > Repositories > Create repository

Create repository

Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.

Repository settings

Repository name

100 characters maximum. Other limits apply.

Description - *optional*

1,000 characters maximum

Add tag

Cancel Create

- Click *Create*

aws Services Resource Groups

Developer Tools > CodeCommit > Repositories > Create repository

Create repository

Create a secure repository to store and share your code. Begin by typing a repository name and a description for your repository. Repository names are included in the URLs for that repository.

Repository settings

Repository name

100 characters maximum. Other limits apply.

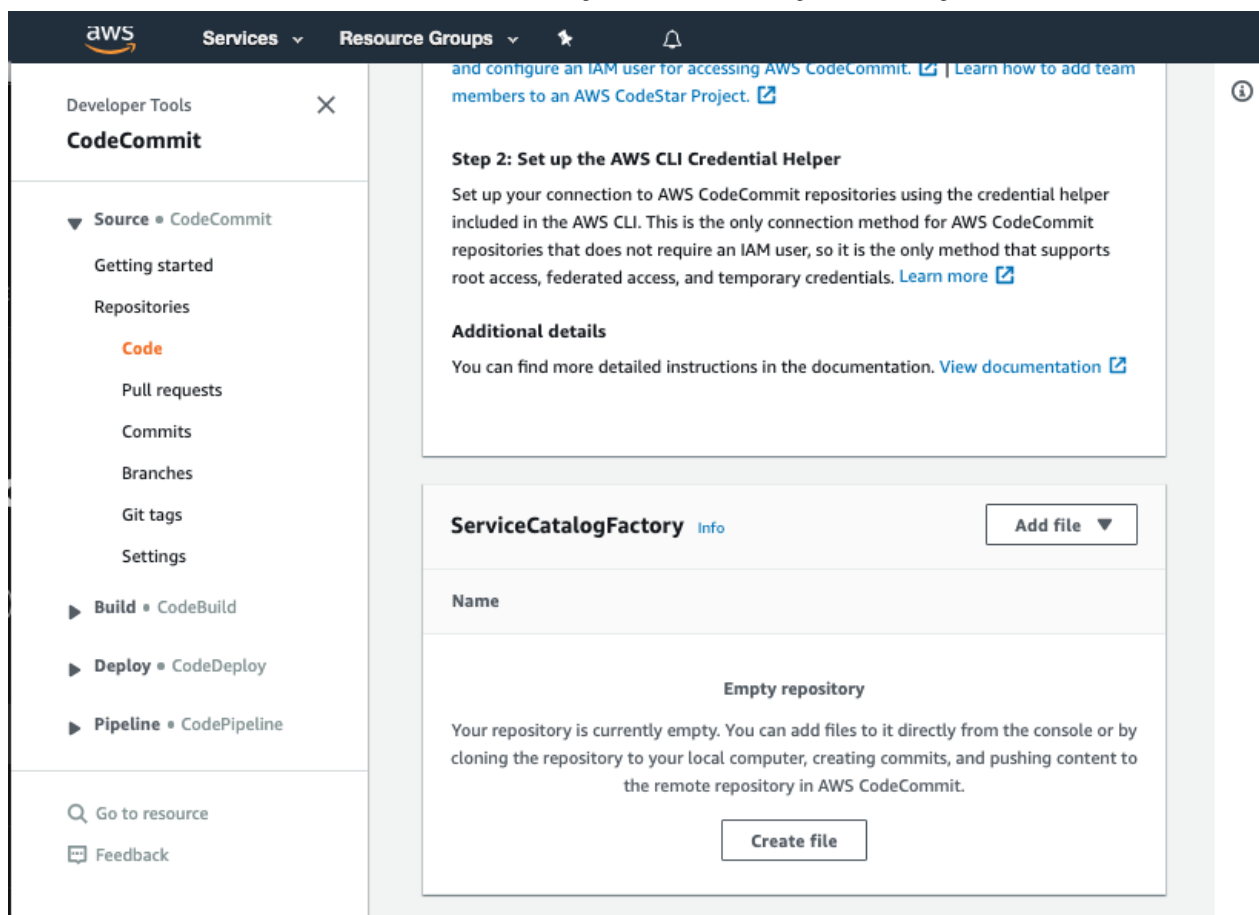
Description - *optional*

1,000 characters maximum

Add tag

Cancel Create

- Scroll down to the bottom of the page and hit the *Create file* button



- Copy the following snippet into the main input field:

```
AWSTemplateFormatVersion: 2010-09-09
Description: "RDS Storage Encrypted"
```

```
Parameters:
```

```
  RdsDbMasterUsername:
    Description: RdsDbMasterUsername
    Type: String
    Default: someuser
```

```
  RdsDbMasterUserPassword:
    Description: RdsDbMasterUserPassword
    Type: String
    NoEcho: true
```

```
  RdsDbDatabaseName:
    Description: DbDatabaseName
    Type: String
    Default: mysql57_database
```

```
Resources:
```

```
  VPC:
    Type: AWS::EC2::VPC
    Properties:
```

```
CidrBlock: 10.0.0.0/16
EnableDnsSupport: 'false'
EnableDnsHostnames: 'false'

Subnet1:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId:
      Ref: VPC
    CidrBlock: 10.0.0.0/24
    AvailabilityZone: !Select [0, !GetAZs {Ref: 'AWS::Region'}]

Subnet2:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId:
      Ref: VPC
    CidrBlock: 10.0.1.0/24
    AvailabilityZone: !Select [1, !GetAZs {Ref: 'AWS::Region'}]

RdsDbSubnetGroup:
  Type: AWS::RDS::DBSubnetGroup
  Properties:
    DBSubnetGroupDescription: Database subnets for RDS
    SubnetIds:
      - !Ref Subnet1
      - !Ref Subnet2

RdsSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Description: Used to grant access to and from the VPC
  Properties:
    VpcId: !Ref VPC
    GroupDescription: Allow MySQL (TCP3306) access to and from
the VPC
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 3306
        ToPort: 3306
        CidrIp: 10.0.0.0/32
    SecurityGroupEgress:
      - IpProtocol: tcp
        FromPort: 3306
        ToPort: 3306
        CidrIp: 10.0.0.0/32

RdsDbClusterParameterGroup:
  Type: AWS::RDS::DBClusterParameterGroup
  Properties:
    Description: CloudFormation Aurora Cluster Parameter Group
    Family: aurora-mysql5.7
    Parameters:
      server_audit_logging: 0
      server_audit_events:
'CONNECT,QUERY,QUERY_DCL,QUERY_DDL,QUERY_DML,TABLE'

RdsDbCluster:
```

```
Type: AWS::RDS::DBCluster
Properties:
  DBSubnetGroupName: !Ref RdsDbSubnetGroup
  MasterUsername: !Ref RdsDbMasterUsername
  MasterUserPassword: !Ref RdsDbMasterUserPassword
  DatabaseName: !Ref RdsDbDatabaseName
  Engine: aurora-mysql
  VpcSecurityGroupIds:
    - !Ref RdsSecurityGroup
  DBClusterIdentifier: !Sub '${AWS::StackName}-dbcluster'
  DBClusterParameterGroupName: !Ref RdsDbClusterParameterGroup
  PreferredBackupWindow: 18:05-18:35

RdsDbParameterGroup:
  Type: AWS::RDS::DBParameterGroup
  Properties:
    Description: CloudFormation Aurora Parameter Group
    Family: aurora-mysql5.7
    Parameters:
      aurora_lab_mode: 0
      general_log: 1
      slow_query_log: 1
      long_query_time: 10

RdsDbInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSubnetGroupName: !Ref RdsDbSubnetGroup
    DBParameterGroupName: !Ref RdsDbParameterGroup
    Engine: aurora-mysql
    DBClusterIdentifier: !Ref RdsDbCluster
    AutoMinorVersionUpgrade: 'true'
    PubliclyAccessible: 'false'
    PreferredMaintenanceWindow: Thu:19:05-Thu:19:35
    AvailabilityZone: !Select [0, !GetAZs {Ref: 'AWS::Region'}]
    DBInstanceClass: 'db.t2.small'
```

- Set the *File name* to `product.template.yaml`
- Set your *Author name*
- Set your *Email address*
- Set your *Commit message*
- Click *Commit changes*

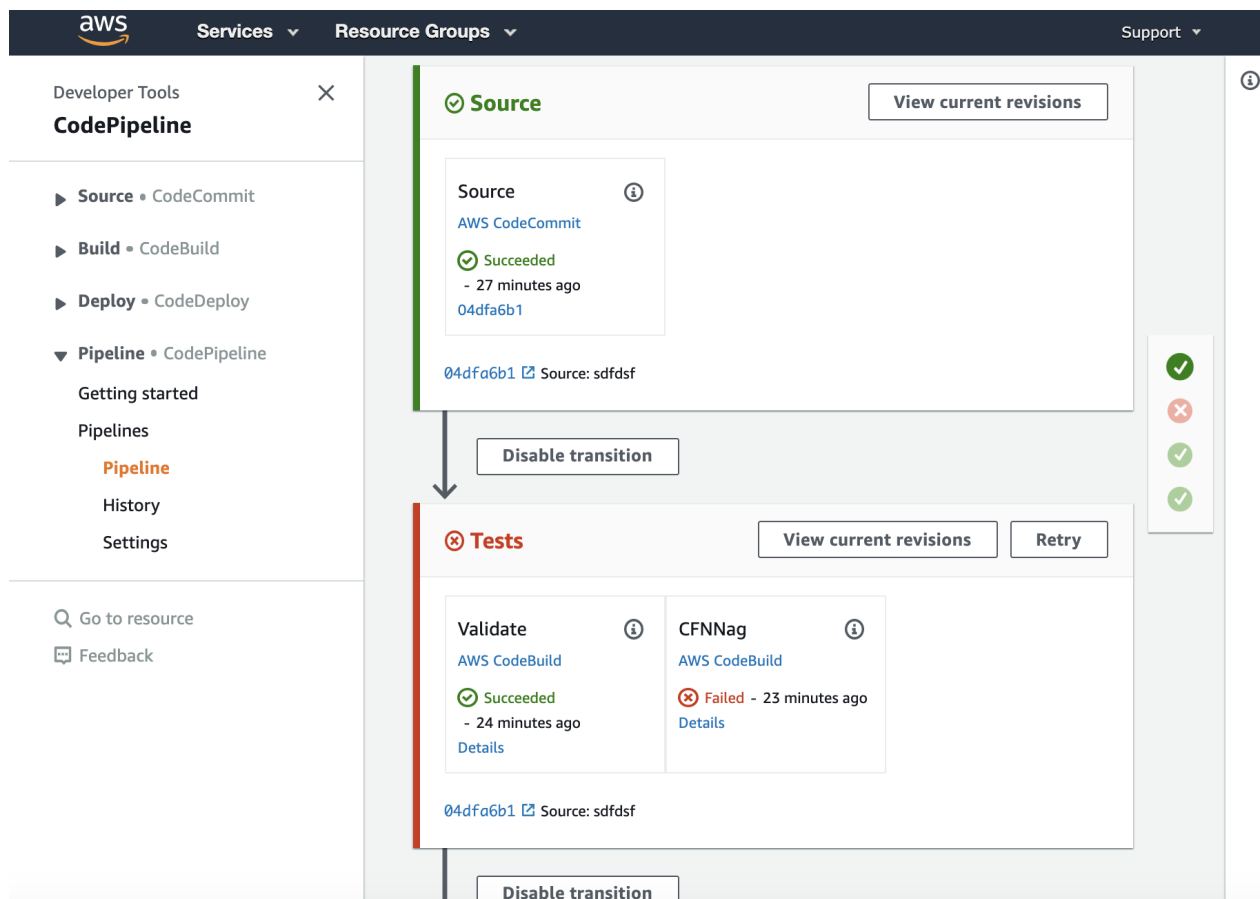


Tip

Using a good / unique commit message will help you understand what is going on later.

Creating that file should trigger your [rds-instance-v1-pipeline](#).

Once the pipeline has completed it should show the *Source* stage in green to indicate it has completed successfully but it should show the CFNNAg action within the Tests stage as failing:



Clicking the *Details* link within the CFNNAg box will bring you to the AWS CodeBuild project. When you scroll near to the bottom of that page you should see an error:

```

· [0;31;49m| FAIL F26· [0m
· [0;31;49m| · [0m
· [0;31;49m| Resources: ["RdsDbCluster"]· [0m
· [0;31;49m| Line Numbers: [84]· [0m
· [0;31;49m| · [0m
· [0;31;49m| RDS DBCluster should have StorageEncrypted enabled· [0m

```

CFNNAg has determined you are not applying encryption to your DBCluster. This is a violation of the data governance guidelines and so we need to fix it.

- Go to [AWS CodeCommit](#)
- Click on the *rds-instance* repository
- Click on *product.template.yaml*
- Click on edit
- Replace the contents with this:

```
AWSTemplateFormatVersion: 2010-09-09
Description: "RDS Storage Encrypted"

Parameters:
  RdsDbMasterUsername:
    Description: RdsDbMasterUsername
    Type: String
    Default: someuser

  RdsDbMasterUserPassword:
    Description: RdsDbMasterUserPassword
    Type: String
    NoEcho: true

  RdsDbDatabaseName:
    Description: DbDatabaseName
    Type: String
    Default: mysql57_database

Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: 10.0.0.0/16
      EnableDnsSupport: 'false'
      EnableDnsHostnames: 'false'

  Subnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId:
        Ref: VPC
      CidrBlock: 10.0.0.0/24
      AvailabilityZone: !Select [0, !GetAZs {Ref: 'AWS::Region'}]

  Subnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId:
        Ref: VPC
      CidrBlock: 10.0.1.0/24
      AvailabilityZone: !Select [1, !GetAZs {Ref: 'AWS::Region'}]
```

```

RdsDbSubnetGroup:
  Type: AWS::RDS::DBSubnetGroup
  Properties:
    DBSubnetGroupDescription: Database subnets for RDS
    SubnetIds:
      - !Ref Subnet1
      - !Ref Subnet2

RdsSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Description: Used to grant access to and from the VPC
  Properties:
    VpcId: !Ref VPC
    GroupDescription: Allow MySQL (TCP3306) access to and from
the VPC
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 3306
        ToPort: 3306
        CidrIp: 10.0.0.0/32
    SecurityGroupEgress:
      - IpProtocol: tcp
        FromPort: 3306
        ToPort: 3306
        CidrIp: 10.0.0.0/32

RdsDbClusterParameterGroup:
  Type: AWS::RDS::DBClusterParameterGroup
  Properties:
    Description: CloudFormation Aurora Cluster Parameter Group
    Family: aurora-mysql5.7
    Parameters:
      server_audit_logging: 0
      server_audit_events:
'CONNECT,QUERY,QUERY_DCL,QUERY_DDL,QUERY_DML,TABLE'

RdsDbCluster:
  Type: AWS::RDS::DBCluster
  Properties:
    DBSubnetGroupName: !Ref RdsDbSubnetGroup
    MasterUsername: !Ref RdsDbMasterUsername
    MasterUserPassword: !Ref RdsDbMasterUserPassword
    DatabaseName: !Ref RdsDbDatabaseName
    Engine: aurora-mysql
    VpcSecurityGroupIds:
      - !Ref RdsSecurityGroup
    DBClusterIdentifier: !Sub '${AWS::StackName}-dbcluster'
    DBClusterParameterGroupName: !Ref RdsDbClusterParameterGroup
    PreferredBackupWindow: 18:05-18:35
    StorageEncrypted: True

RdsDbParameterGroup:
  Type: AWS::RDS::DBParameterGroup
  Properties:
    Description: CloudFormation Aurora Parameter Group
    Family: aurora-mysql5.7
    Parameters:

```

```
aurora_lab_mode: 0
general_log: 1
slow_query_log: 1
long_query_time: 10

RdsDbInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBSubnetGroupName: !Ref RdsDbSubnetGroup
    DBParameterGroupName: !Ref RdsDbParameterGroup
    Engine: aurora-mysql
    DBClusterIdentifier: !Ref RdsDbCluster
    AutoMinorVersionUpgrade: 'true'
    PubliclyAccessible: 'false'
    PreferredMaintenanceWindow: Thu:19:05–Thu:19:35
    AvailabilityZone: !Select [0, !GetAZs {Ref: 'AWS::Region'}]
    DBInstanceClass: 'db.t2.small'
    StorageEncrypted: True
```

Please observe the highlighted lines showing where we have made a change. We have added:

StorageEncrypted: True

- Set your *Author name*
- Set your *Email address*
- Set your *Commit message*
- Click *Commit changes*



Tip

Using a good / unique commit message will help you understand what is going on later.

Creating that file should trigger your [rds-instance-v1-pipeline](#).

Once the pipeline has completed it should show the *Source* and *Tests* stages in green to indicate they have completed successfully:

The screenshot displays the AWS CodePipeline console interface. On the left, a sidebar shows the navigation menu with 'Developer Tools' and 'CodePipeline' selected. The main area is titled 'Source' and shows a successful build status with a green checkmark and the text 'Succeeded - 8 minutes ago'. Below this, a 'Tests' section shows two successful build status items: 'Validate' and 'CFNNag', both with green checkmarks and 'Succeeded' status. The interface includes buttons for 'View current revisions' and 'Disable transition'.

Tip

You should see your commit message on this screen, it will help you know which version of ServiceCatalogFactory repository the pipeline is processing.

Note

If this is failing please raise your hand for some assistance

Once you have verified the pipeline has run you can go to [Service Catalog products](#) to view your newly created version.

You should see the product you created listed:

The screenshot displays the AWS Service Catalog administrator console. At the top, there's a navigation bar with the AWS logo, 'Services', 'Resource Groups', and a notification bell. Below this, the 'aws service catalog' header is visible. A left-hand navigation menu lists various sections: Products, Provisioned Products, Administration (expanded), Products (highlighted), Portfolios, TagOptions Library, Service Actions, Preferences, and Your Marketplace Software. The main content area is titled 'Admin - Product List' with an 'Info' link. It features a 'Products (1)' header with a refresh button and an 'Actions' dropdown. An orange 'Upload new product' button is also present. Below this is a search bar labeled 'Search product'. A pagination bar shows '< 1 >' and a settings gear icon. A table with headers 'Product name', 'Created', 'Vendor', and 'Description' is shown, but it is currently empty. A message 'Your product should appear here' is displayed within the table area. A blue notification banner at the top of the main content area informs about a redesign of the administrator console and the launch of the 'Budget Visibility' feature. The footer contains a 'Feedback' link, 'English (US)' language selection, copyright information for 2008-2019, and links to 'Privacy Policy' and 'Terms of Use'.

Click on the product and verify v1 is there

The screenshot shows the AWS Service Catalog console. The left sidebar contains navigation links: Products, Provisioned Products, Administration, Products (highlighted), Portfolios, TagOptions Library, Service Actions, Preferences, and Your Marketplace Software. The top navigation bar includes the AWS logo, Services, Resource Groups, and a notification bell. The main content area is titled 'Support Details' and shows 'Product version(s) (2)'. Below this, there are buttons for 'Create new version' and 'Actions', and a search bar. A table lists the product versions:

Name	Status	Created time	Description
v1	Active	Wed, Oct 16, 2019, 10:43:19 AM GMT+1	
-	Active	Wed, Oct 16, 2019, 10:33:32 AM GMT+1	Placeholder version, do not provision

The footer includes a Feedback button, English (US) language selection, and copyright information: © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy and Terms of Use.

Note

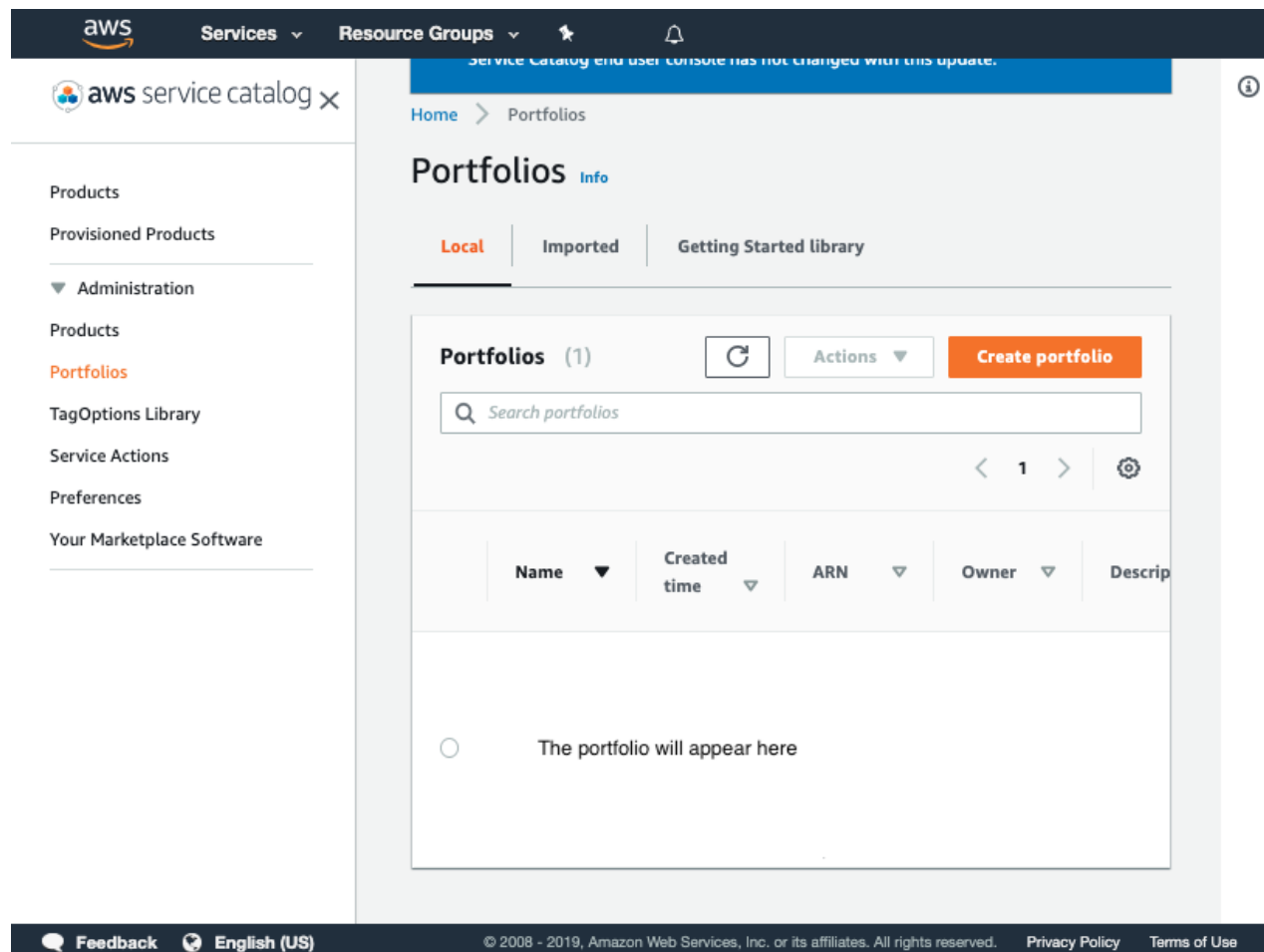
If you cannot see your version please raise your hand for some assistance

You have now successfully created a version for your product!

Verify that the product was added to the portfolio

Now that you have verified the pipeline has run you can go to [Service Catalog portfolios](#) to view your portfolio.

- Click on *cloud-engineering-self-service*



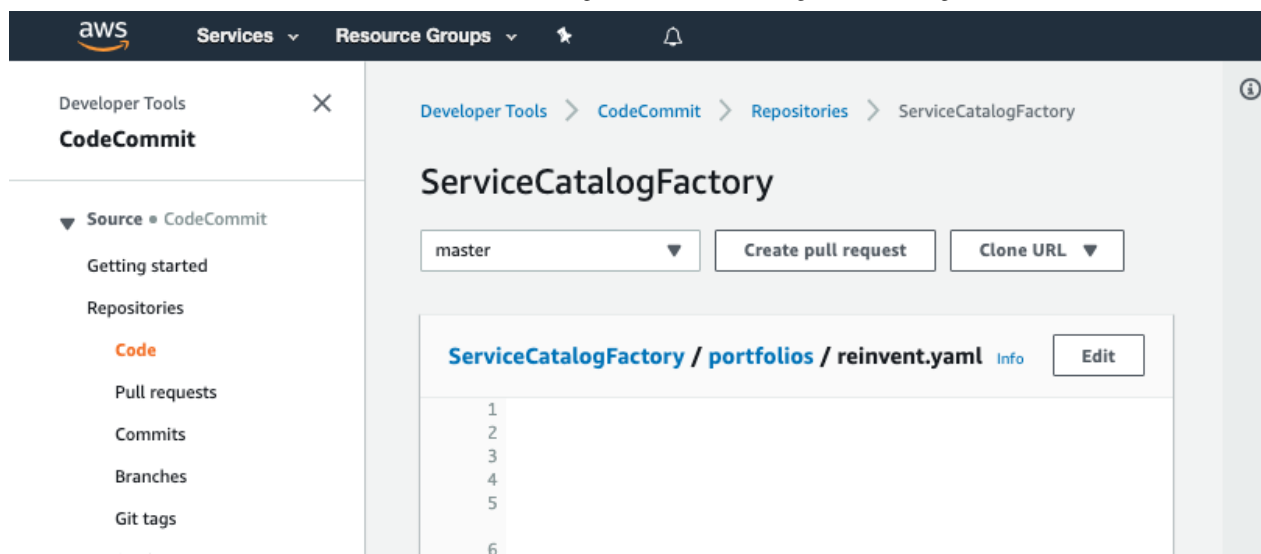
- Click on the product *rds-instance*
- Click on the version *v1*

The screenshot shows the AWS Service Catalog console. The left sidebar contains navigation links: Products, Provisioned Products, Administration, Products (highlighted), Portfolios, TagOptions Library, Service Actions, Preferences, and Your Marketplace Software. The main content area is titled 'Support Details' and shows 'Versions (2)', 'Portfolios (1)', 'Tags (5)', and 'TagOptions (0)'. The 'Product version(s) (2)' section includes a 'Create new version' button and an 'Actions' dropdown. Below this is a search bar and a table of product versions.

	Name	Status	Created time	Description
<input type="radio"/>	-	Active	Wed, Oct 16, 2019, 10:33:32 AM GMT+1	Placeholder version, do not provision
<input type="radio"/>	v1	Active	Wed, Oct 16, 2019, 10:43:19 AM GMT+1	

Share portfolio with a spoke account

- Navigate to the `ServiceCatalogPuppet` `CodeCommit` repository again
- Click on `manifest.yaml`
- Click `Edit`



- Append the following snippet to the YAML document in the main input field (be careful with your indentation):

```
spoke-local-portfolios:
  cloud-engineering-self-service:
    portfolio: "reinvent-cloud-engineering-self-service"
    deploy_to:
      tags:
        - tag: "type:prod"
          regions: "default_region"
```

- The main input field should look like this:

```
accounts:
  - account_id: "<YOUR_ACCOUNT_ID_WITHOUT_HYPHENS>"
    name: "puppet-account"
    default_region: "eu-west-1"
    regions_enabled:
      - "eu-west-1"
      - "eu-west-2"
    tags:
      - "type:prod"
      - "partition:eu"

launches:
  aws-config-desired-instance-types:
    portfolio: "reinvent-cloud-engineering-governance"
    product: "aws-config-desired-instance-types"
    version: "v1"
    parameters:
      InstanceType:
```

```
    default: "t2.medium, t2.large, t2.xlarge"
  deploy_to:
    tags:
      - tag: "type:prod"
        regions: "default_region"
aws-config-rds-storage-encrypted:
  portfolio: "reinvent-cloud-engineering-governance"
  product: "aws-config-rds-storage-encrypted"
  version: "v1"
  deploy_to:
    tags:
      - tag: "type:prod"
        regions: "default_region"

spoke-local-portfolios:
  cloud-engineering-self-service:
    portfolio: "reinvent-cloud-engineering-self-service"
    deploy_to:
      tags:
        - tag: "type:prod"
          regions: "default_region"
```

Committing the manifest file

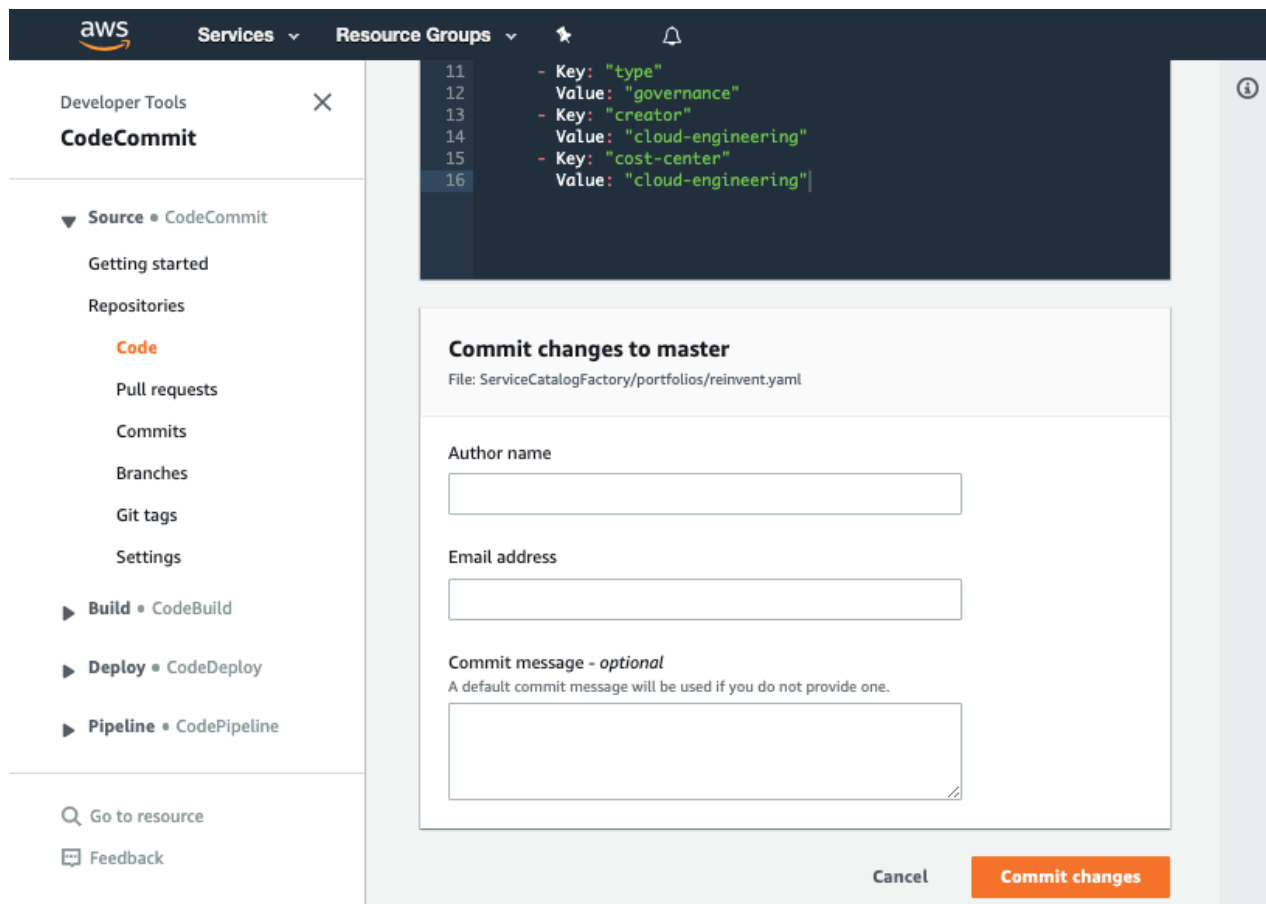
- Set your *Author name*
- Set your *Email address*
- Set your *Commit message*



Tip

Using a good / unique commit message will help you understand what is going on later.

- Click the *Commit changes* button:



Verifying the sharing

Once you have made your changes the [ServiceCatalogPuppet Pipeline](#) should have run. If you were quick may still be running. If it has not yet started feel free to hit the *Release change* button.

Once it has completed it should show the *Source* and *Build* stages in green to indicate they have completed successfully:

The screenshot displays the AWS CodePipeline console interface. On the left, a sidebar shows the navigation menu with options like 'Developer Tools', 'CodePipeline', 'Source', 'Build', 'Deploy', and 'Pipeline'. The main area shows a pipeline named 'CodeCommit' with a status of 'Succeeded - 1 day ago'. Below this, there are three stages: 'Generate', 'Deploy', and 'Deploy'. Each stage has a 'View current revisions' button. The 'Generate' stage shows a job 'GenerateShares' with a status of 'Succeeded - 1 day ago'. The 'Deploy' stage shows a job 'Deploy' with a status of 'Succeeded - 1 day ago'. A 'Disable transition' button is visible between the stages. The top of the console shows the AWS logo, 'Services', 'Resource Groups', and user information.

Note

If this is failing please raise your hand for some assistance

Once you have verified the pipeline has run you can go to [Service Catalog portfolios](#) to view your shared product.

When you share a portfolio the framework will decide if it should share the portfolio. If the target account is the same as the factory account it will not share the portfolio as it is not needed.

Note

If you cannot see your product please raise your hand for some assistance

