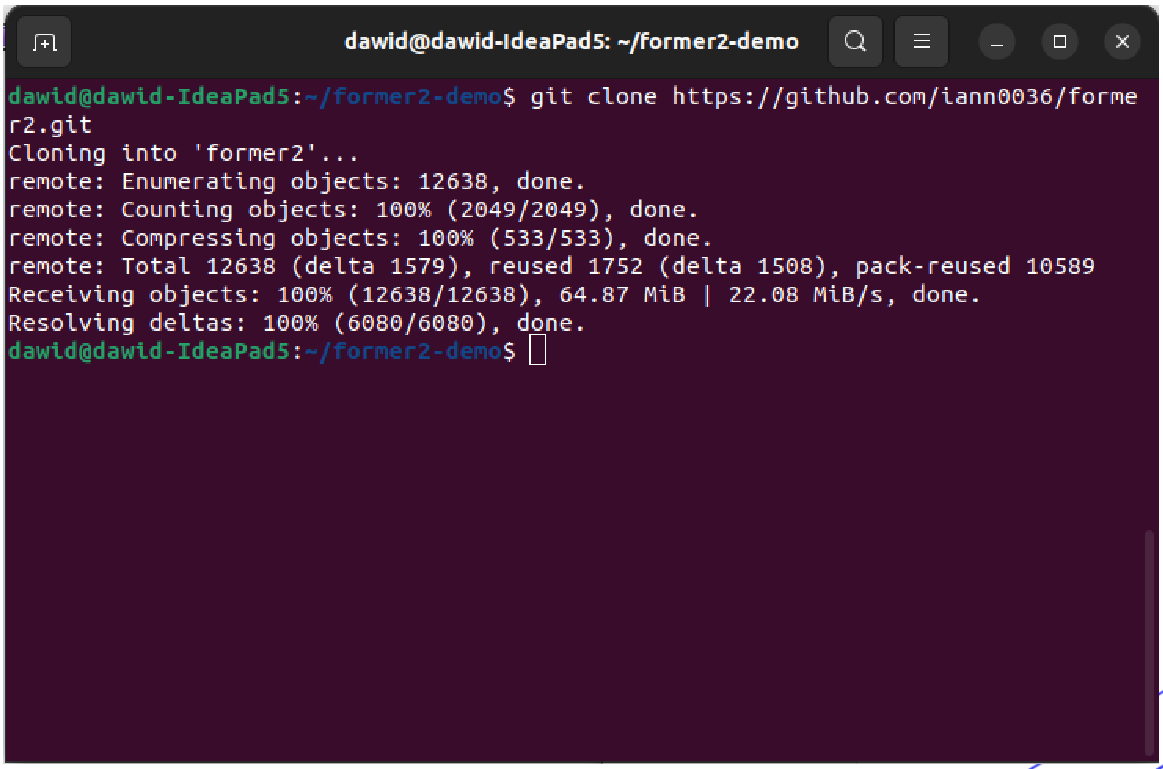
AWS CloudFormation backup using Former2

Former2 is a powerful tool that streamlines the process of generating CloudFormation templates from existing AWS resources. By reverse-engineering your deployed infrastructure, Former2 generates accurate and comprehensive CloudFormation templates, saving you considerable time and effort. This tool is especially beneficial when creating a disaster recovery plan or migrating to infrastructure-as-code, as it accelerates the transition by automatically generating templates that reflect your existing AWS setup. With Former2, users can quickly capture their AWS resources' configurations in CloudFormation syntax, facilitating consistent and reproducible infrastructure deployments.

The official website is https://former2.com/, but it is recommended to deploy it locally by cloning the GitHub repository, using Docker Compose to deploy the stack, and accessing it from the browser using Incognito mode. The tool uses IAM user credentials that are stored in the browser's cache. The user should have ReadOnly permission.

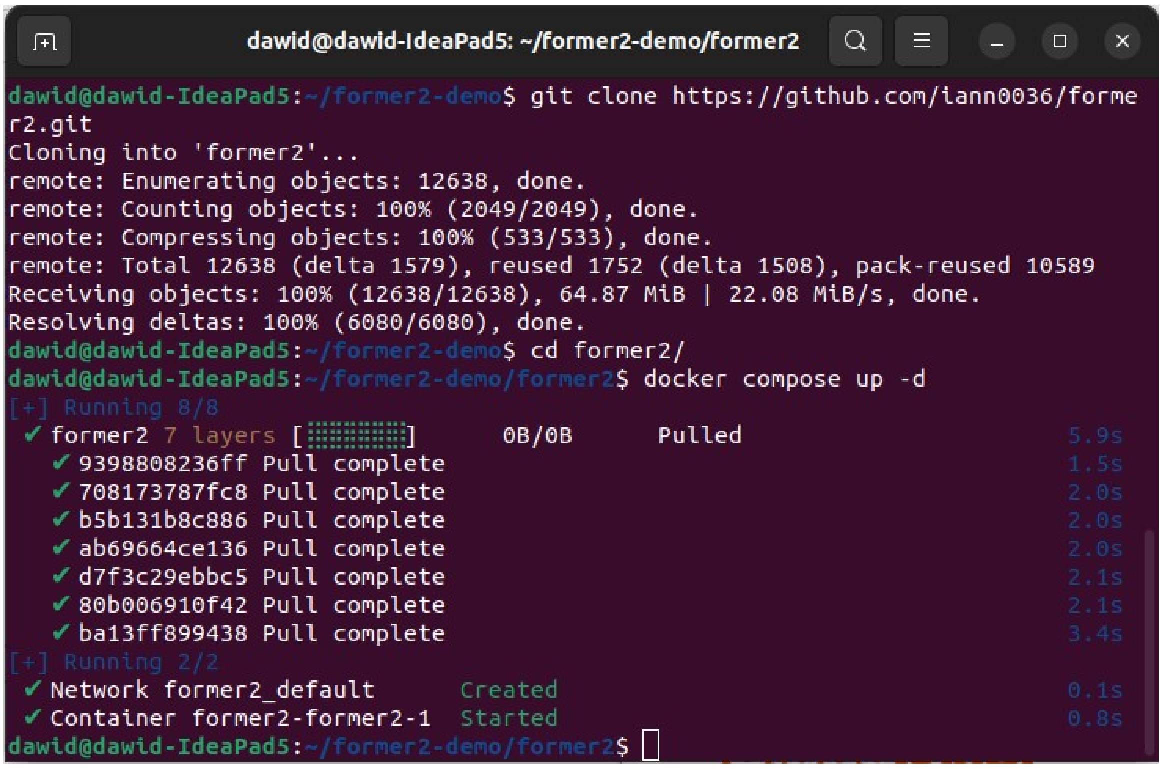
Deployment

1. Open the terminal and clone the Github repository from https://github.com/iann0036/former2 to Your local drive.



```
dawid@dawid-IdeaPad5:~/former2-demo$ git clone https://github.com/iann0036/forme
r2.git
Cloning into 'former2'...
remote: Enumerating objects: 12638, done.
remote: Counting objects: 100% (2049/2049), done.
remote: Compressing objects: 100% (533/533), done.
remote: Total 12638 (delta 1579), reused 1752 (delta 1508), pack-reused 10589
Receiving objects: 100% (12638/12638), 64.87 MiB | 22.08 MiB/s, done.
Resolving deltas: 100% (6080/6080), done.
dawid@dawid-IdeaPad5:~/former2-demo$
```

2. Go to the newly created folder former2 and run *docker compose up -d*.

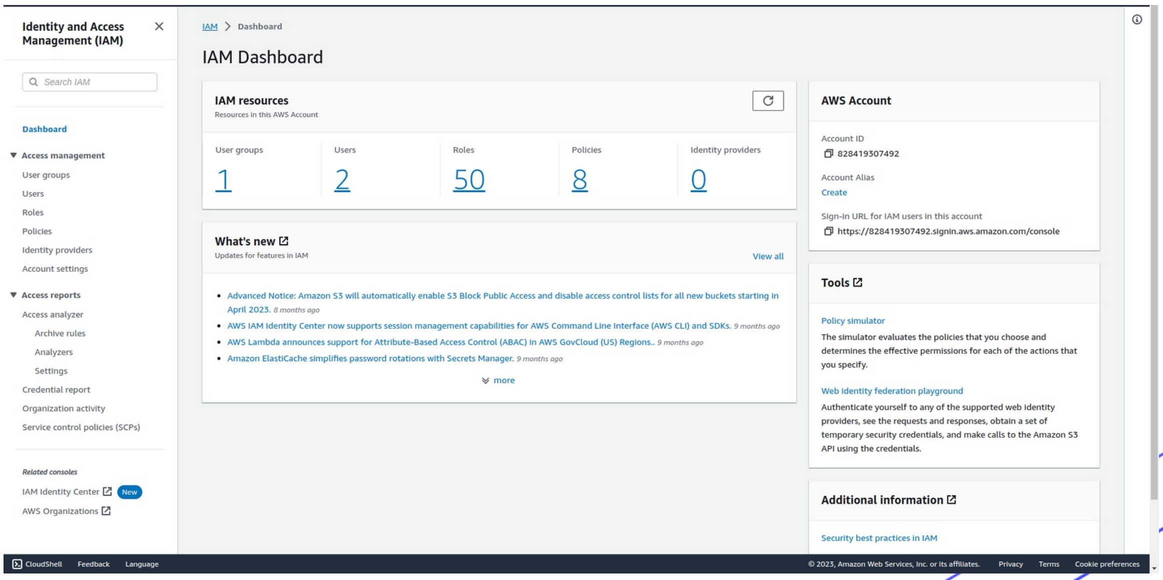


```
dawid@dawid-IdeaPad5: ~/former2-demo/former2
dawid@dawid-IdeaPad5:~/former2-demo$ git clone https://github.com/iann0036/forme
r2.git
Cloning into 'former2'...
remote: Enumerating objects: 12638, done.
remote: Counting objects: 100% (2049/2049), done.
remote: Compressing objects: 100% (533/533), done.
remote: Total 12638 (delta 1579), reused 1752 (delta 1508), pack-reused 10589
Receiving objects: 100% (12638/12638), 64.87 MiB | 22.08 MiB/s, done.
Resolving deltas: 100% (6080/6080), done.
dawid@dawid-IdeaPad5:~/former2-demo$ cd former2/
dawid@dawid-IdeaPad5:~/former2-demo/former2$ docker compose up -d
[+] Running 8/8
 ✔ former2 7 layers [⣿⣿⣿⣿⣿⣿⣿]      0B/0B      Pulled                       5.9s
   ✔ 9398808236ff Pull complete                                             1.5s
   ✔ 708173787fc8 Pull complete                                             2.0s
   ✔ b5b131b8c886 Pull complete                                             2.0s
   ✔ ab69664ce136 Pull complete                                             2.0s
   ✔ d7f3c29ebbc5 Pull complete                                             2.1s
   ✔ 80b006910f42 Pull complete                                             2.1s
   ✔ ba13ff899438 Pull complete                                             3.4s
[+] Running 2/2
 ✔ Network former2_default      Created                                     0.1s
 ✔ Container former2-former2-1  Started                                     0.8s
dawid@dawid-IdeaPad5:~/former2-demo/former2$
```

3. The stack has been locally deployed. It can be accessed from the browser on ***localhost***.

IAM User creation

1. Log in to Your AWS console and go to the IAM service.



Identity and Access Management (IAM)

Search IAM

Dashboard

▼ Access management
- User groups
- Users
- Roles
- Policies
- Identity providers
- Account settings

▼ Access reports
- Access analyzer
  - Archive rules
  - Analyzers
  - Settings
- Credential report
- Organization activity
- Service control policies (SCPs)

Related consoles
- IAM Identity Center New
- AWS Organizations

IAM > Dashboard

IAM Dashboard

IAM resources
Resources in this AWS Account

User groups	Users	Roles	Policies	Identity providers
1	2	50	8	0

What's new
Updates for features in IAM
View all

- Advanced Notice: Amazon S3 will automatically enable S3 Block Public Access and disable access control lists for all new buckets starting in April 2023. *8 months ago*
- AWS IAM Identity Center now supports session management capabilities for AWS Command Line Interface (AWS CLI) and SDKs. *9 months ago*
- AWS Lambda announces support for Attribute-Based Access Control (ABAC) in AWS GovCloud (US) Regions.. *9 months ago*
- Amazon ElastiCache simplifies password rotations with Secrets Manager. *9 months ago*

more

AWS Account
Account ID
828419307492
Account Alias
Create
Sign-in URL for IAM users in this account
https://828419307492.signin.aws.amazon.com/console

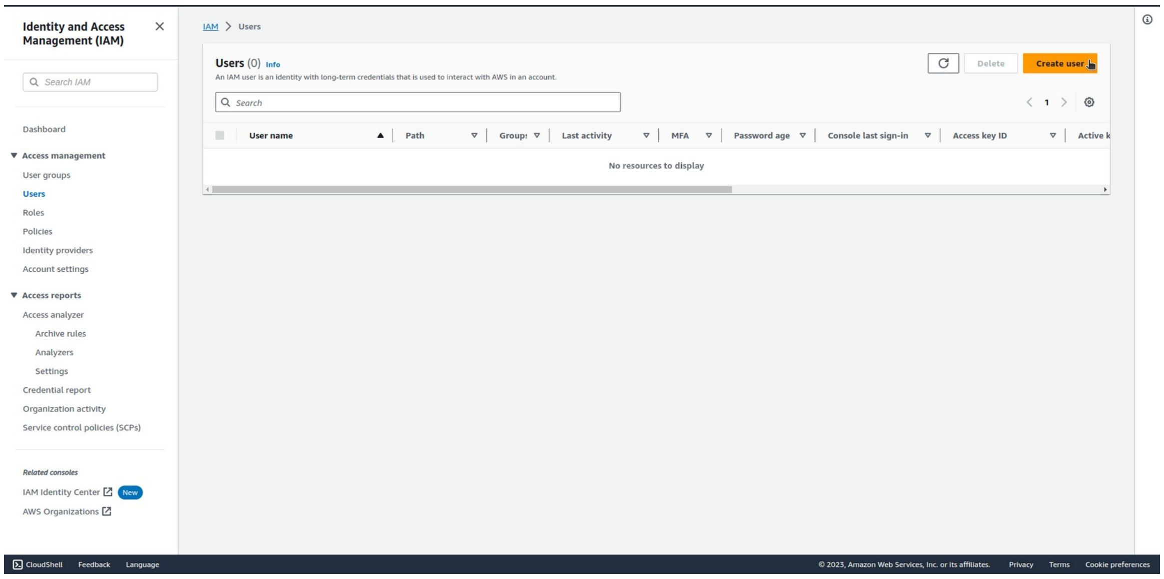
Tools
Policy simulator
The simulator evaluates the policies that you choose and determines the effective permissions for each of the actions that you specify.

Web identity federation playground
Authenticate yourself to any of the supported web identity providers, see the requests and responses, obtain a set of temporary security credentials, and make calls to the Amazon S3 API using the credentials.

Additional information
Security best practices in IAM

CloudShell Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

2. Select ***Users*** and click on ***Create user*** in the upper right corner.



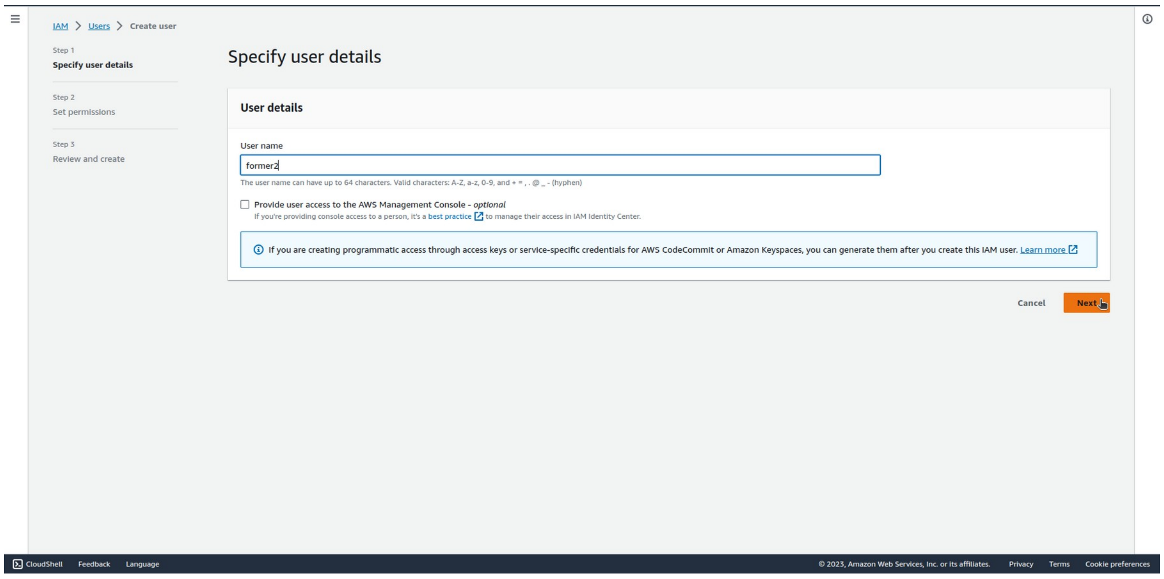
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Groups	Last activity	MFA	Password age	Console last sign-in	Access key ID	Active k

No resources to display

© 2023, Amazon Web Services, Inc. or its affiliates.

3. Give the new user a name and click on ***Next***.


IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Cancel
Next
CloudShell Feedback Language
Privacy Terms Cookie preferences

Specify user details

User details

User name: former2

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)

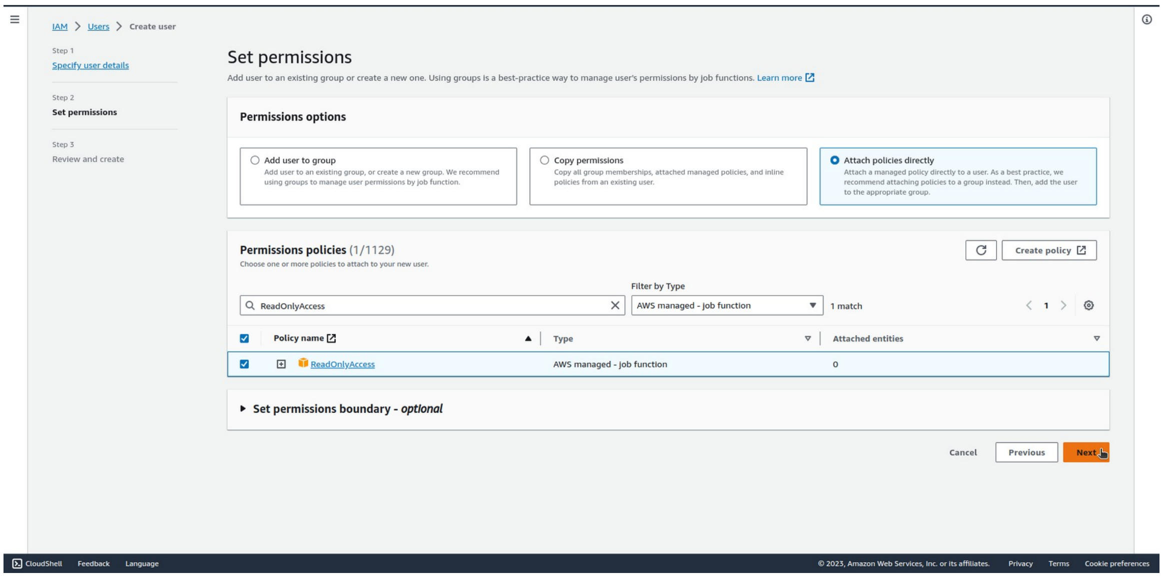
☐ Provide user access to the AWS Management Console - *optional*

If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.

If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more

© 2023, Amazon Web Services, Inc. or its affiliates.

4. Choose ***Attach policies directly*** tab, search for ***ReadOnlyAccess*** policy and filter by type AWS managed – job function. Select the checkbox next to the policy and click on ***Next***.


IAM > Users > Create user

Step 1
Specify user details

Step 2
Set permissions

Step 3
Review and create

Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more

Permissions options

- ○ Add user to group — Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- ○ Copy permissions — Copy all group memberships, attached managed policies, and inline policies from an existing user.
- ● Attach policies directly — Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1129)

Choose one or more policies to attach to your new user.

Search: ReadOnlyAccess

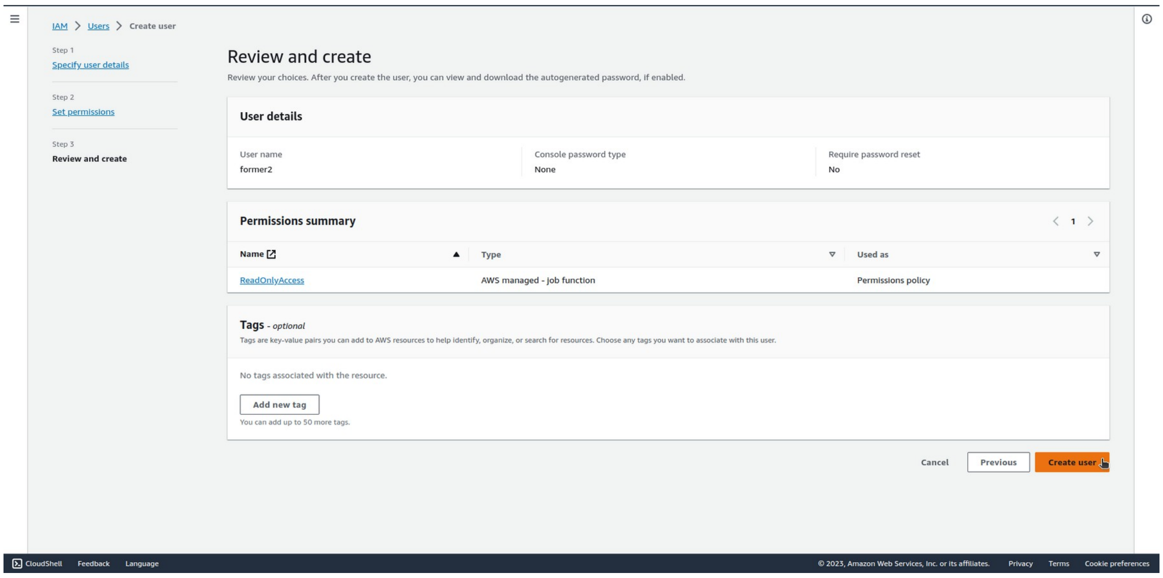
Filter by Type: AWS managed - job function

1 match

✓	Policy name	Type	Attached entities
✓	ReadOnlyAccess	AWS managed - job function	0

▸ Set permissions boundary - *optional*

5. Review the new user settings and click on ***Create user***.


IAM > Users > Create user

Step 1
Specify user details

Step 2
Set permissions

Step 3
Review and create

Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name	Console password type	Require password reset
former2	None	No

Permissions summary

Name	Type	Used as
ReadOnlyAccess	AWS managed - job function	Permissions policy

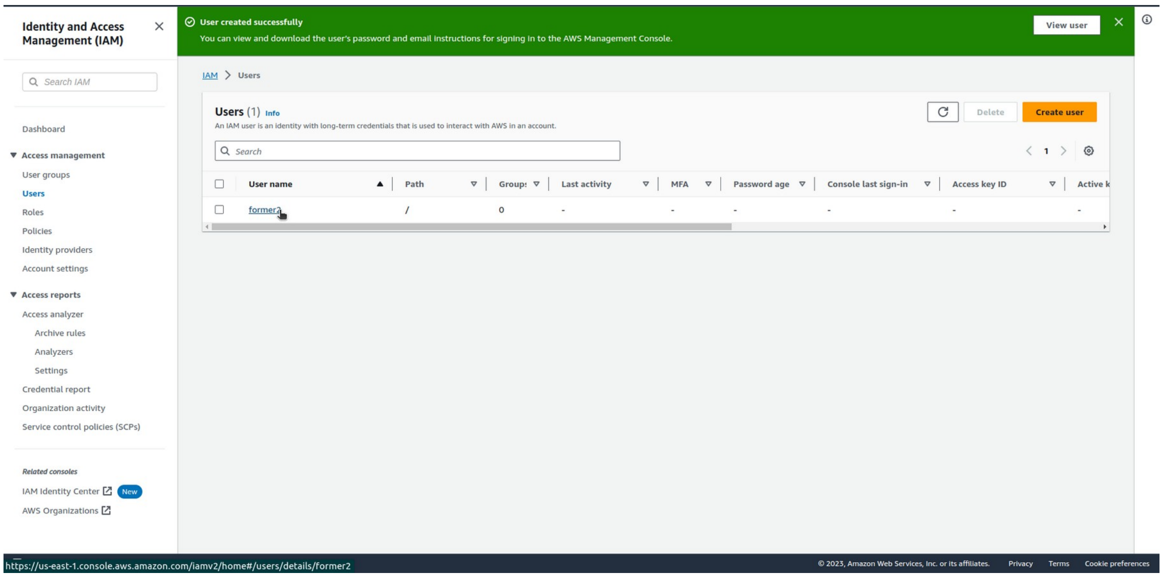
Tags - *optional*

Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

You can add up to 50 more tags.

6. Click on the name of the newly created user.



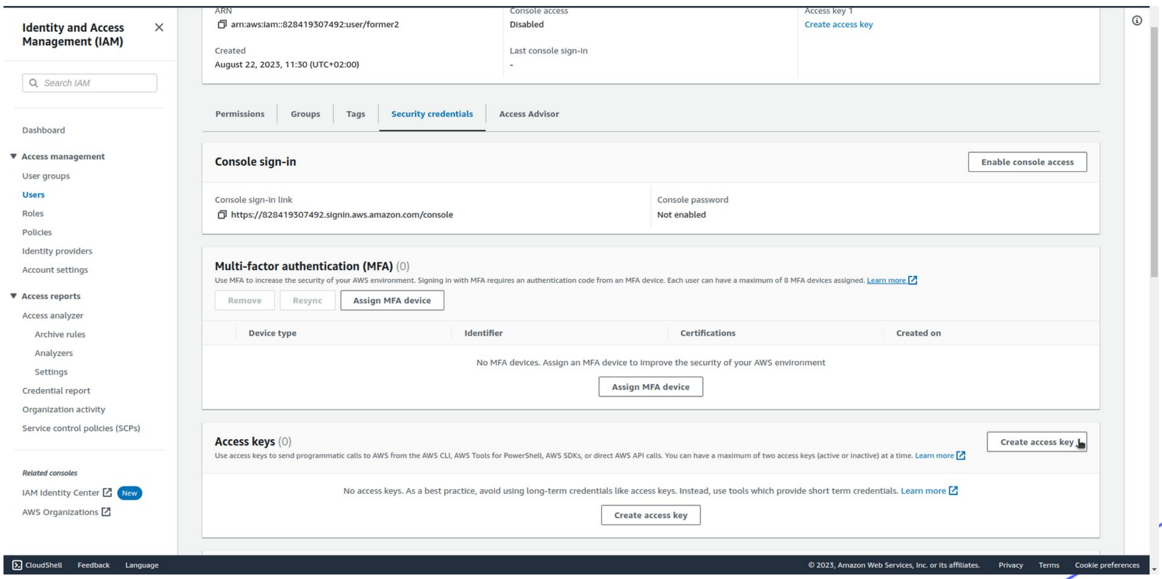
User created successfully
You can view and download the user's password and email instructions for signing in to the AWS Management Console.

Users (1)
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Group:	Last activity	MFA	Password age	Console last sign-in	Access key ID	Active k
former2	/	0	-	-	-	-	-	-

https://us-east-1.console.aws.amazon.com/iamv2/home#/users/details/former2

7. Select ***Security credentials*** tab and click on ***Create access key*** under the ***Access keys*** section.



ARN
arn:aws:iam::828419307492:user/former2

Created
August 22, 2023, 11:30 (UTC+02:00)

Console access
Disabled

Last console sign-in
-

Access key 1
Create access key

Console sign-in

Console sign-in link
https://828419307492.signin.aws.amazon.com/console

Console password
Not enabled

Multi-factor authentication (MFA) (0)
Use MFA to increase the security of your AWS environment. Signing in with MFA requires an authentication code from an MFA device. Each user can have a maximum of 8 MFA devices assigned. Learn more

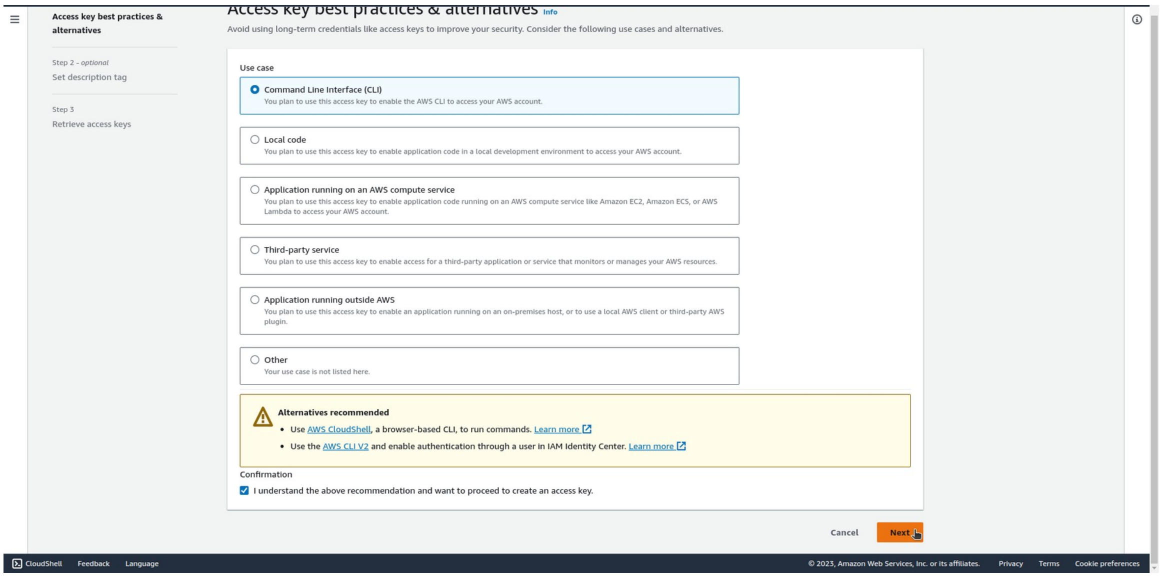
Device type	Identifier	Certifications	Created on

No MFA devices. Assign an MFA device to improve the security of your AWS environment

Access keys (0)
Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. Learn more

No access keys. As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. Learn more

8. Select ***Command Line Interface (CLI)*** option and select the checkbox on the bottom of the page. Click on ***Next***.


Access key best practices & alternatives
Step 2 - *optional*
Set description tag
Step 3
Retrieve access keys

Access key best practices & alternatives Info
Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case
[✓ Command Line Interface (CLI)]
You plan to use this access key to enable the AWS CLI to access your AWS account.

Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

Application running outside AWS
You plan to use this access key to enable an application running on an on-premises host, or to use a local AWS client or third-party AWS plugin.

Other
Your use case is not listed here.

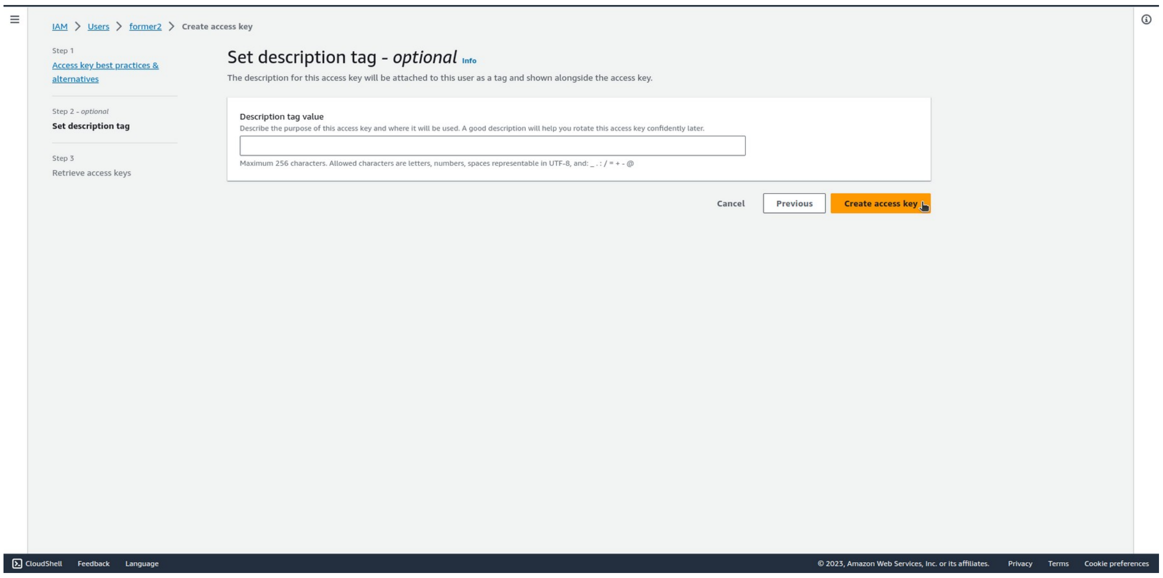
Alternatives recommended
- Use AWS CloudShell, a browser-based CLI, to run commands. Learn more
- Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center. Learn more

Confirmation
[✓ I understand the above recommendation and want to proceed to create an access key.]

Cancel Next

CloudShell Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

9. (Optional) Describe the purpose of this access key and click on ***Create access key***.


IAM > Users > former2 > Create access key

Step 1
Access key best practices & alternatives
Step 2 - *optional*
Set description tag
Step 3
Retrieve access keys

Set description tag - *optional* Info
The description for this access key will be attached to this user as a tag and shown alongside the access key.

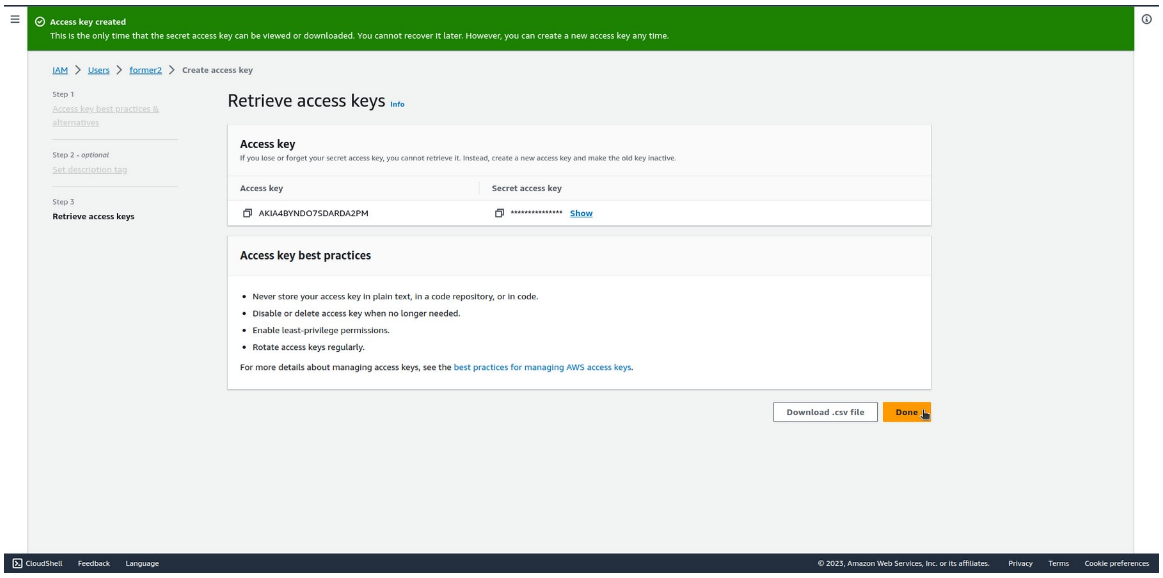
Description tag value
Describe the purpose of this access key and where it will be used. A good description will help you rotate this access key confidently later.

Maximum 256 characters. Allowed characters are letters, numbers, spaces representable in UTF-8, and: _ . : / = + - @

Cancel Previous Create access key

CloudShell Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

10. Save the Access key and Secret access key for later use with Former2 tool and click on ***Done***.



Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

IAM > Users > former2 > Create access key

Step 1
Access key best practices & alternatives

Step 2 - *optional*
Set description tag

Step 3
Retrieve access keys

Retrieve access keys Info

Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIA4BYNDO7SDARDA2PM	*************** Show

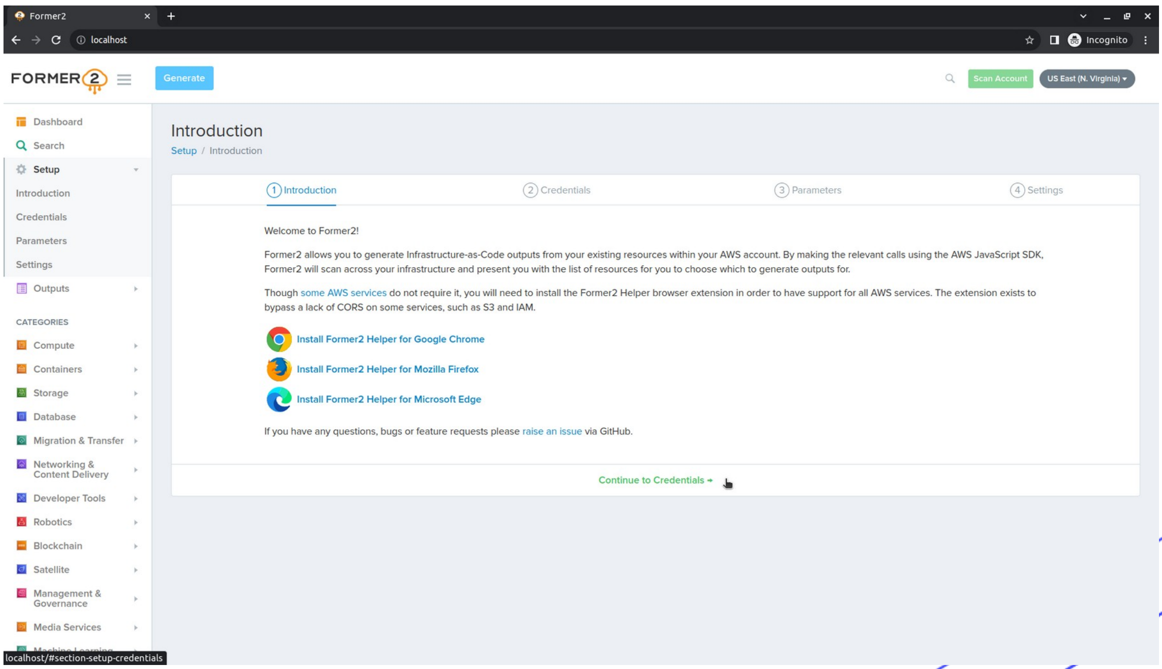
Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the best practices for managing AWS access keys.

Former2 tool usage

1. Open Your browser in Incognito mode and type ***localhost*** in the URL. You should see the welcome page of Former2. Click on ***Continue to credentials***.



Introduction
Setup / Introduction

1 Introduction · 2 Credentials · 3 Parameters · 4 Settings

Welcome to Former2!

Former2 allows you to generate Infrastructure-as-Code outputs from your existing resources within your AWS account. By making the relevant calls using the AWS JavaScript SDK, Former2 will scan across your infrastructure and present you with the list of resources for you to choose which to generate outputs for.

Though some AWS services do not require it, you will need to install the Former2 Helper browser extension in order to have support for all AWS services. The extension exists to bypass a lack of CORS on some services, such as S3 and IAM.

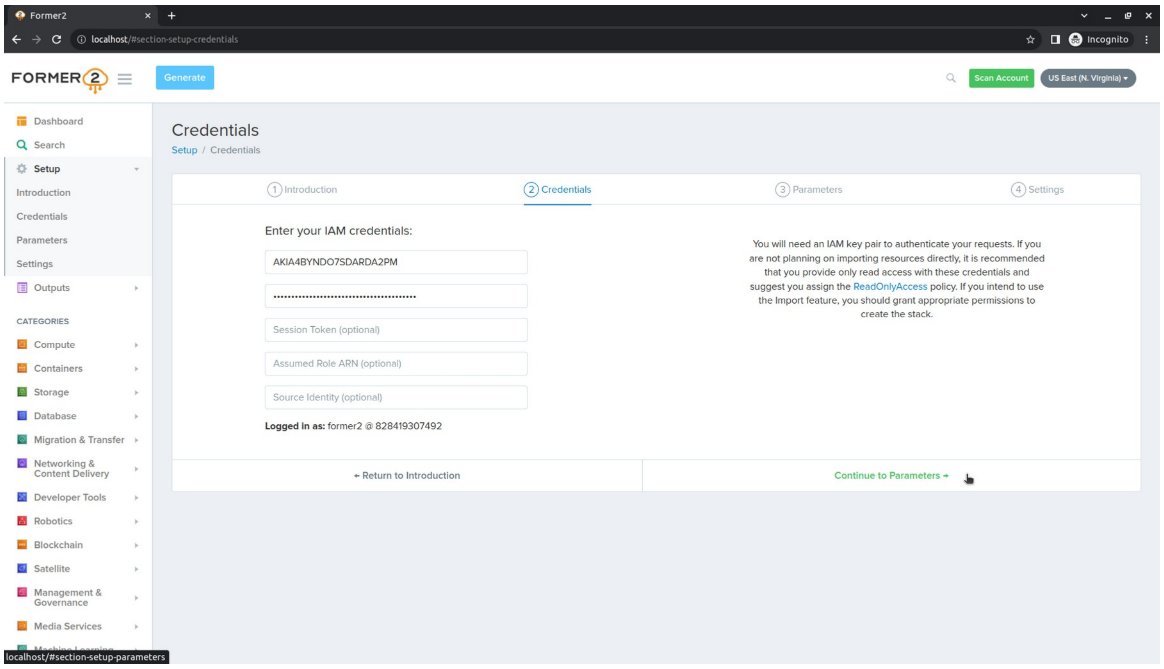
Install Former2 Helper for Google Chrome
Install Former2 Helper for Mozilla Firefox
Install Former2 Helper for Microsoft Edge

If you have any questions, bugs or feature requests please raise an issue via GitHub.

Continue to Credentials →

localhost/#section-setup-credentials

2. Enter the previously created Access key and Secret Access Key and click on ***Continue to Parameters***.



Enter your IAM credentials:

AKIA4BYNDO7SDARDA2PM

••

Session Token (optional)

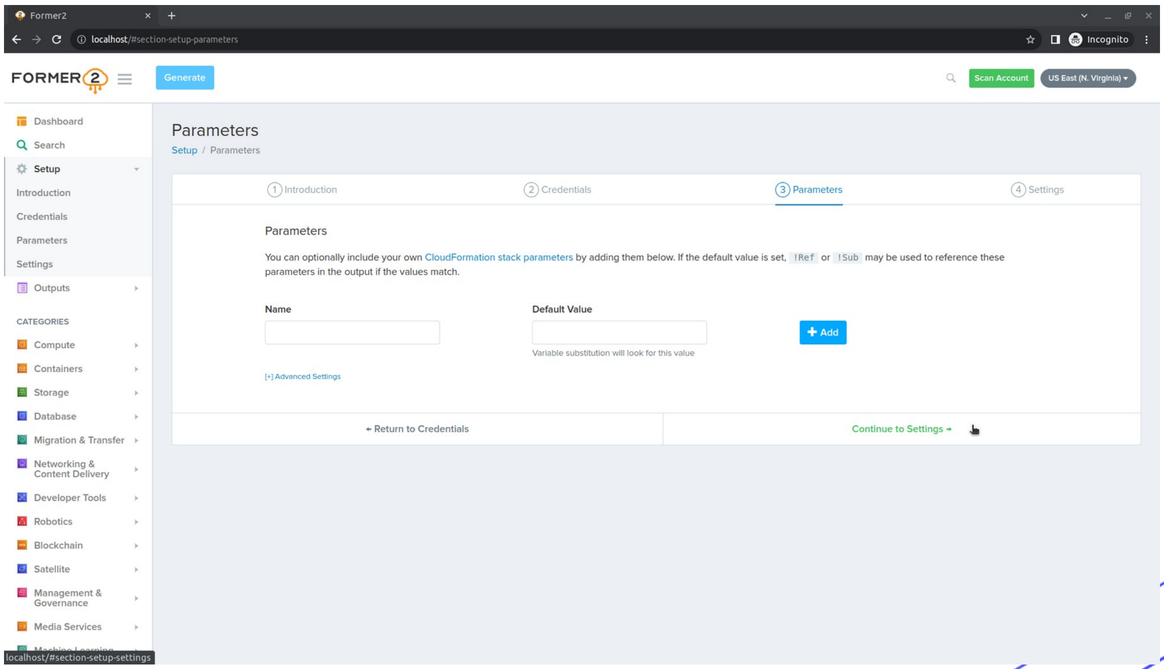
Assumed Role ARN (optional)

Source Identity (optional)

Logged in as: former2 @ 828419307492

You will need an IAM key pair to authenticate your requests. If you are not planning on importing resources directly, it is recommended that you provide only read access with these credentials and suggest you assign the ReadOnlyAccess policy. If you intend to use the Import feature, you should grant appropriate permissions to create the stack.

3. (Optional) You can set Your own CloudFormation stack parameters. Click on ***Continue to Settings***.



Parameters

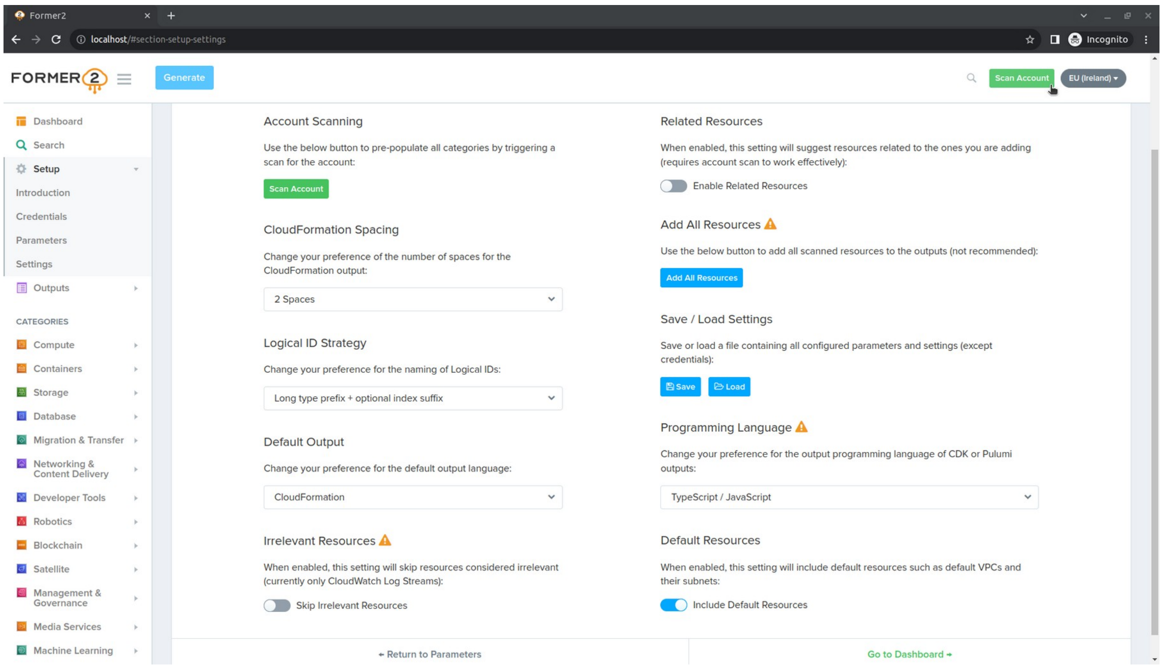
You can optionally include your own CloudFormation stack parameters by adding them below. If the default value is set, `!Ref` or `!Sub` may be used to reference these parameters in the output if the values match.

Name

Default Value

Variable substitution will look for this value

4. Now You can set the default settings like number of spaces in output YAML files, default output language, programming language, etc. Select Your region in the upper right corner and click on ***Scan Account***.



Account Scanning

Use the below button to pre-populate all categories by triggering a scan for the account:

Scan Account

CloudFormation Spacing

Change your preference of the number of spaces for the CloudFormation output:

2 Spaces

Logical ID Strategy

Change your preference for the naming of Logical IDs:

Long type prefix + optional index suffix

Default Output

Change your preference for the default output language:

CloudFormation

Irrelevant Resources

When enabled, this setting will skip resources considered irrelevant (currently only CloudWatch Log Streams):

Skip Irrelevant Resources

Related Resources

When enabled, this setting will suggest resources related to the ones you are adding (requires account scan to work effectively):

Enable Related Resources

Add All Resources

Use the below button to add all scanned resources to the outputs (not recommended):

Add All Resources

Save / Load Settings

Save or load a file containing all configured parameters and settings (except credentials):

Save Load

Programming Language

Change your preference for the output programming language of CDK or Pulumi outputs:

TypeScript / JavaScript

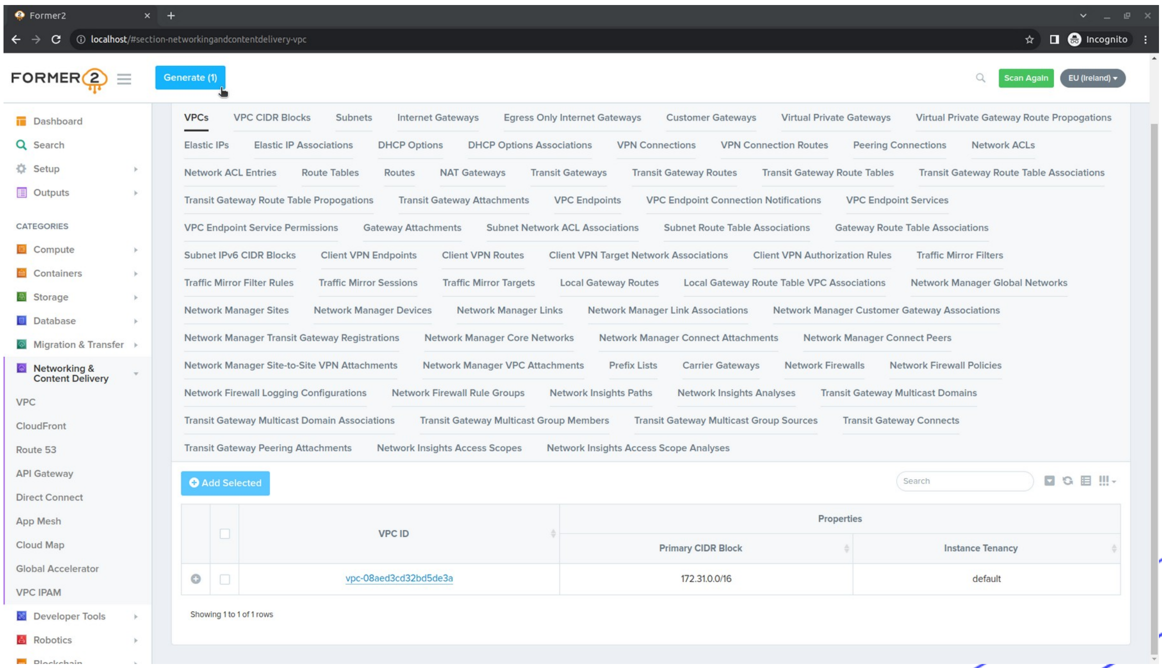
Default Resources

When enabled, this setting will include default resources such as default VPCs and their subnets:

Include Default Resources

← Return to Parameters Go to Dashboard →

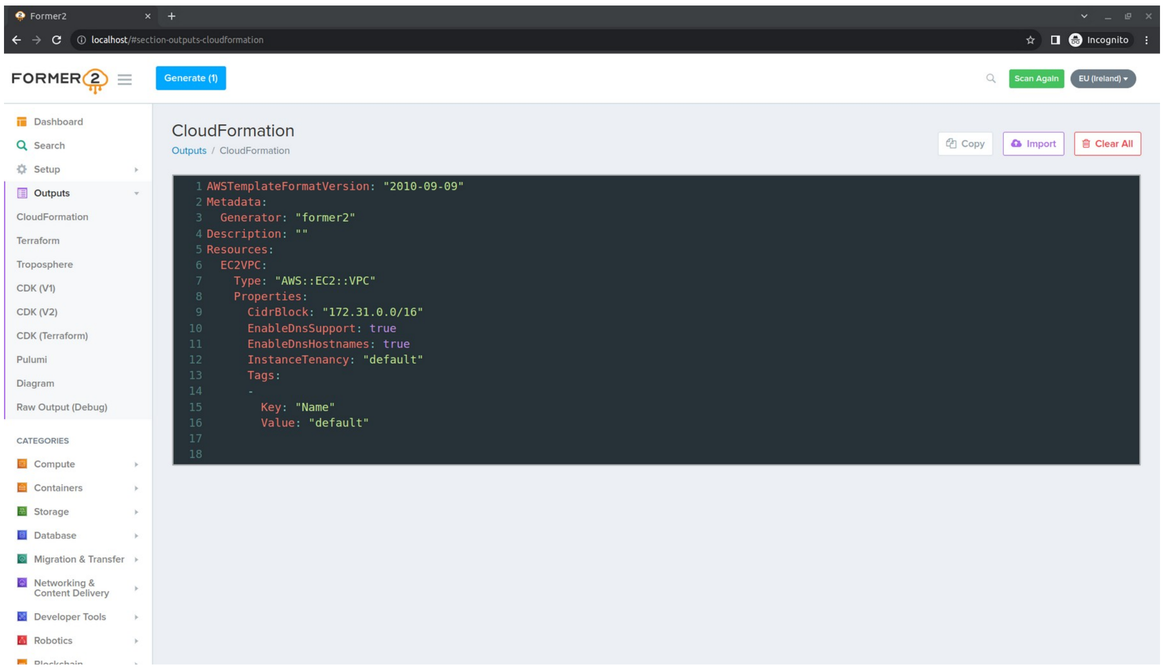
5. After the scan will be completed go to ***Dashboard***, select the service that You want to get template of, select the checkbox next to the resource, click on ***Add Selected*** and then on ***Generate*** on the top of the page.


Former2
localhost/#section-networkingandcontentdelivery-vpc
Incognito
FORMER2
Generate (1)
Scan Again
EU (Ireland)
Dashboard
Search
Setup
Outputs
CATEGORIES
Compute
Containers
Storage
Database
Migration & Transfer
Networking & Content Delivery
VPC
CloudFront
Route 53
API Gateway
Direct Connect
App Mesh
Cloud Map
Global Accelerator
VPC IPAM
Developer Tools
Robotics
VPCs
VPC CIDR Blocks
Subnets
Internet Gateways
Egress Only Internet Gateways
Customer Gateways
Virtual Private Gateways
Virtual Private Gateway Route Propogations
Elastic IPs
Elastic IP Associations
DHCP Options
DHCP Options Associations
VPN Connections
VPN Connection Routes
Peering Connections
Network ACLs
Network ACL Entries
Route Tables
Routes
NAT Gateways
Transit Gateways
Transit Gateway Routes
Transit Gateway Route Tables
Transit Gateway Route Table Associations
Transit Gateway Route Table Propogations
Transit Gateway Attachments
VPC Endpoints
VPC Endpoint Connection Notifications
VPC Endpoint Services
VPC Endpoint Service Permissions
Gateway Attachments
Subnet Network ACL Associations
Subnet Route Table Associations
Gateway Route Table Associations
Subnet IPv6 CIDR Blocks
Client VPN Endpoints
Client VPN Routes
Client VPN Target Network Associations
Client VPN Authorization Rules
Traffic Mirror Filters
Traffic Mirror Filter Rules
Traffic Mirror Sessions
Traffic Mirror Targets
Local Gateway Routes
Local Gateway Route Table VPC Associations
Network Manager Global Networks
Network Manager Sites
Network Manager Devices
Network Manager Links
Network Manager Link Associations
Network Manager Customer Gateway Associations
Network Manager Transit Gateway Registrations
Network Manager Core Networks
Network Manager Connect Attachments
Network Manager Connect Peers
Network Manager Site-to-Site VPN Attachments
Network Manager VPC Attachments
Prefix Lists
Carrier Gateways
Network Firewalls
Network Firewall Policies
Network Firewall Logging Configurations
Network Firewall Rule Groups
Network Insights Paths
Network Insights Analyses
Transit Gateway Multicast Domains
Transit Gateway Multicast Domain Associations
Transit Gateway Multicast Group Members
Transit Gateway Multicast Group Sources
Transit Gateway Connects
Transit Gateway Peering Attachments
Network Insights Access Scopes
Network Insights Access Scope Analyses
Add Selected
Search

		VPC ID	Properties	
			Primary CIDR Block	Instance Tenancy
⊕	☐	vpc-08aed3cd32bd5de3a	172.31.0.0/16	default

Showing 1 to 1 of 1 rows

6. The template is ready, You can now copy it to the file.



```
AWSTemplateFormatVersion: "2010-09-09"
Metadata:
  Generator: "former2"
Description: ""
Resources:
  EC2VPC:
    Type: "AWS::EC2::VPC"
    Properties:
      CidrBlock: "172.31.0.0/16"
      EnableDnsSupport: true
      EnableDnsHostnames: true
      InstanceTenancy: "default"
      Tags:
      -
        Key: "Name"
        Value: "default"


```

You can add as many resources as You want. Take under consideration that those templates always have to be double checked if everything was imported correctly. The potential variables in most of the time will be hardcoded and not all the dependencies could be imported.

Cleanup

After the work is done use ***docker compose down*** to delete the stack.