Service Workbench on AWS Installation Guide

Table of Contents

Overview

The Service Workbench on AWS cloud-based solution provides secure access to data, tooling, and compute power. With Service Workbench, researchers can perform research in a securely pre-configured environment. Service Workbench enables the creation of baseline research templates, simplifying data access and providing cost transparency.

Use the Service Workbench on AWS Installation Guide to deploy the solution with AWS Cloud9 or an EC2 instance.

Installation options

Use case for installation using AWS Cloud9

AWS Cloud9 offers a more straight-forward installation process to get users started more quickly.

Use case for installation using EC2

For users already using EC2 instances, this method may be best for their use case.

Installation using AWS Cloud9

Create the AWS Cloud9 instance

1. Go to the AWS Cloud9 product page.
2. Choose **Create environment**.
3. Enter a name and description for the AWS Cloud9 environment.
4. Choose **Next step**.
5. Configure your Environment settings:
   a. Environment type—Create a new EC2 instance for environment (direct access)
   b. Instance type—m5.large
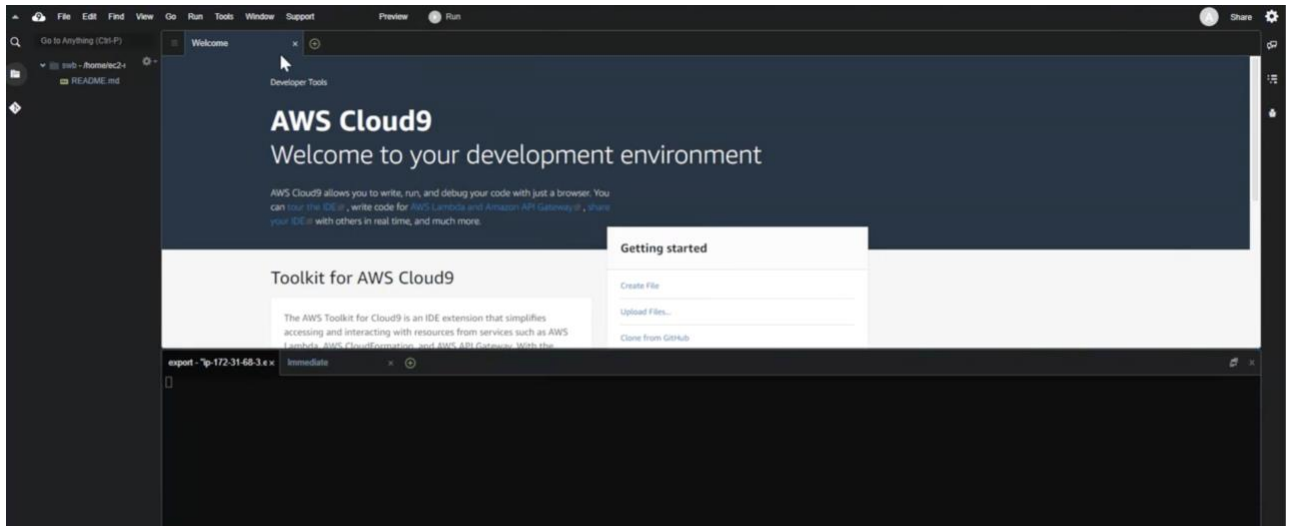   c. Platform—Amazon Linux 2

   ℹ **Note**
   Keep all other settings in the default selections.

6. Choose **Next step**.
7. Review your environment name and settings and click **Create environment**.
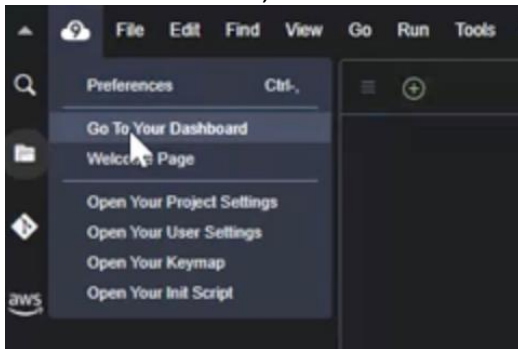
Your environment should load a welcome screen and command line interface.

Modify the volume

1. In the **Cloud9** menu, choose **Go To Your Dashboard**.

2. Use the **Services** menu to go to **EC2**.

3. From the left navigation, choose **Volumes**.

4. Select the volume associated with your AWS Cloud9 instance.

5. From the **Actions** menu, choose **Modify Volume**.

6. Increase the size to at least 40 GB.

7. Choose **Modify**, and then choose **Modify** again to confirm.

For more information about modifying the volume, see Resize an Amazon EBS volume used by an environment.

Configure the AWS Cloud9 environment

Return to the AWS Cloud9 command line environment to complete the following steps.

> **ℹ Note**
> Keep all other settings in the default selections.

---

1. Verify the file size by entering the command:
   ```
   df -hT
   ```

2. To increase the partition size, enter:
   ```
   sudo growpart /dev/nvme0n1 1
   ```

3. Increase the file system inside the partition:
   ```
   sudo xfs_growfs -d /
   ```

4. Verify that the file size increased by entering:
   ```
   df -hT
   ```

5. Check the node version by entering:
   ```
   node --version
   ```

⚠️ **Important**
Node.js v14.x is required. If you must revert to v14.x, see the steps in Troubleshooting.

6. Install the node package manager:
   ```
   npm install -g pnpm
   ```

7. A serverless packer is necessary to build AMIs. Follow the README instructions to install. Open a new terminal window after you complete the installation.

Clone the GitHub directory

1. From the terminal, clone the GitHub directory:
   ```
   git clone https://github.com/awslabs/service-workbench-on-aws.git
   ```

   The Git directory is now available on your instance.

2. Ensure you are working from the service-workbench-on-aws folder for the remainder of the process:
   ```
   cd service-workbench-on-aws/
   ```

Prepare the environment file

1. From the toggletree, navigate to select **example.yml**:
   main > **config** > **settings** > **example.yml**.

2. Copy the example.yml file and rename it.
   Example: dev.yml

3. In the copied file, uncomment the following:

   a. **awsRegion:** us-east-1

   b. **solutionName:** sw

   c. **envType:** dev

   d. **createServiceCatalogPortfolio:** true

4. Save the copied file.

ℹ️ **Note**
For more information on editing the environment file, see Environment file.

Run the install script

1. Ensure you are working from the service-workbench-on-aws folder:
    ```
    cd service-workbench-on-aws/
    ```

2. Deploy the install script:
    ```
    scripts/environment-deploy.sh <copied file name>
    ```
    Example: `scripts/environment-deploy.sh dev`

The install may take up to an hour. Once successfully installed you will receive a success message and login details.

For more information about logging in to Service Workbench, see Login and setup user account.

Installation Using EC2 Instance

Create an EC2 instance

If you do not already have an available EC2 instance, follow these steps to create one:

1. From the EC2 console, choose **Launch instances**.

2. Choose **Select** for Amazon Linux 2 AMI (HVM) – Kernel 5.10, SSD Volume Type.

3. Select an instance of **t2.medium** or larger.

4. Choose **Next: Configure Instance Details**.

5. Select an IAM role with administrative permissions.
   If you do not have an available IAM, choose **Create new IAM role**. Once you have created a role, return to Configure Instance Details and choose refresh next to IAM role. The role should now be available.

> **ⓘ Note**
> For more information on creating IAM admin roles, see Creating your first IAM admin user and user group.

6. Choose **Next: Add Storage**.

7. Change the size to at least 40 GB.

8. Choose **Review and Launch**.

9. Review the details of your instance and choose **Launch**.

10. Select an existing key pair or create a new key pair for your instance.

11. Choose **Launch Instances**.

Connect to an EC2 instance

1. From the EC2 console, select the instance that will host Service Workbench. Choose **Connect**.

2. Select your connection method, and follow the onscreen directions.

Install Node and Go

Install Node.js

> **⚠ Important**
> Node.js v14.x is required. If you need to revert to v14.x, see the steps in Troubleshooting.

```
curl -o- https://raw.githubusercontent.com/nvm-
sh/nvm/v0.35.3/install.sh | bash

source ~/.bashrc

nvm install 14
```

```
npm install -g serverless pnpm hygen
```

Install Go

```
cd ~
wget -c https://golang.org/dl/go1.15.2.linux-amd64.tar.gz
sudo tar -C /usr/local -xvzf go1.15.2.linux-amd64.tar.gz
echo "PATH=$PATH:/usr/local/go/bin" >> ~/.bashrc
source ~/.bashrc
go version
```

Clone the GitHub directory

1. From terminal, install git:
   ```
   sudo yum install -y git
   ```

2. Clone the git repository:
   ```
   git clone https://github.com/awslabs/service-workbench-on-
   aws.git
   ```

Prepare the environment file

1. Open the Service Workbench folder:
   ```
   cd service-workbench-on-aws/
   ```

2. Navigate to the main configuration directory:
   ```
   cd main/config/settings
   ```

3. Make a copy of the example.yml file and rename it. For this example, we will rename the file to dev.yml.
   ```
   cp example.yml dev.yml
   ```

4. Open the newly created configuration file to edit:
   ```
   vim dev
   ```

5. Uncomment and set the following values:

   a. awsRegion
      Ensure you use the same AWS Region where you use the AWS Management Console.

   b. solutionName

6. To quit editing, press ESC and then enter:
   ```
   :wq
   ```

> ℹ **Note**
> For more information on editing the environment file, see Environment file.

Run the install script

Return to the root directory before launching. If you changed the copied configuration file name to a name other than dev, use that in the place of dev.
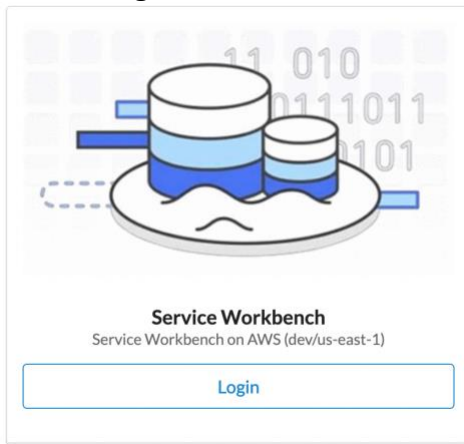
1. Run the installation script:
   `./scripts/environment-deploy.sh dev`

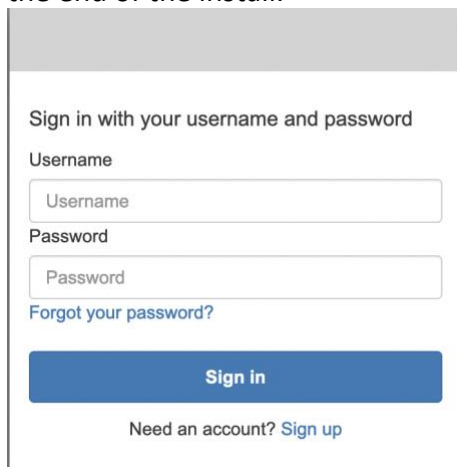2. Note the URL and password provided. You can display the URL and password again with:
   `scripts/get-info.sh dev`

Login and setup user account

1. Copy and paste the provided URL in a browser.

2. Choose **Login**.

3. Your user name will be root@example.com and the password will be the one provided at the end of the install.

4. Change your password when prompted.

Environment file

In addition to the required flags to be uncommented when preparing the environment file for deployment, there are additional custom flags that may be uncommented as well.

enableAMISharing

Enables the AMI sharing feature so that AMIs and Appstream images can be maintained in a central devops account.

For more information about AMIs, see Amazon Machine Images (AMI).

disableAdminBYOBSelfAssignment

Admins can only assign a user account when registering a study.

restrictAdminWorkspaceConnection

Restricts admin access to workspaces not owned by the admin.
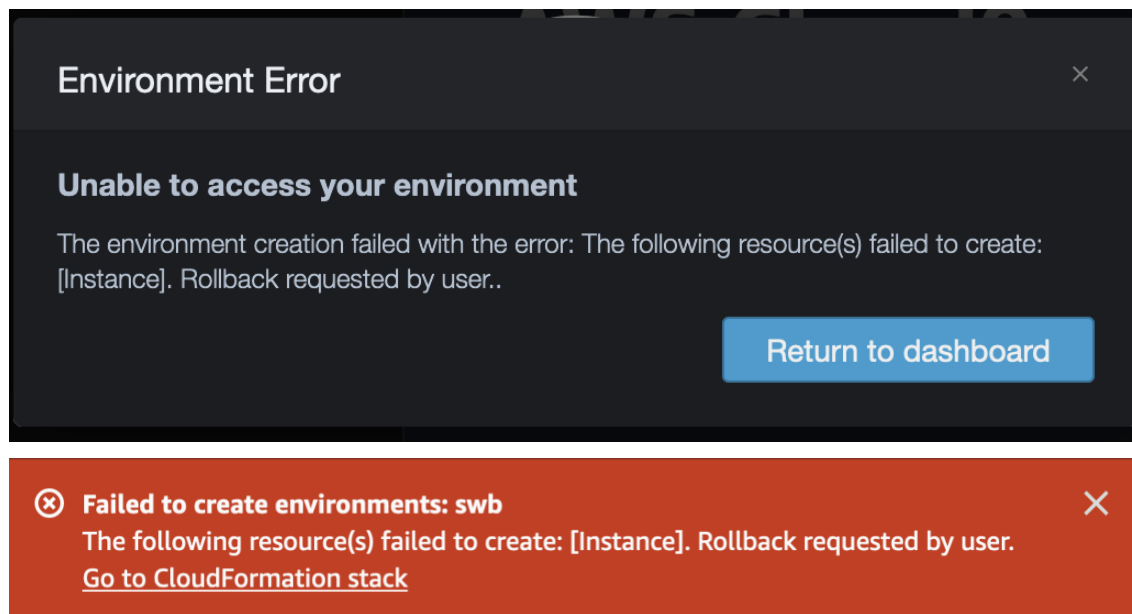
disableStudyUploadbyResearcher

A user will not be able to upload files or create a study unless they are an admin.

devopsProfile

Creates AWS IAM role in the DevOps account and adds a trust policy in the role to grant AssumeRole permissions to the main account.

Troubleshooting

Received Environment Error during install of AWS Cloud9

Go to Dashboard to view the error reason. Try again with designated AWS Region selected for subnet.
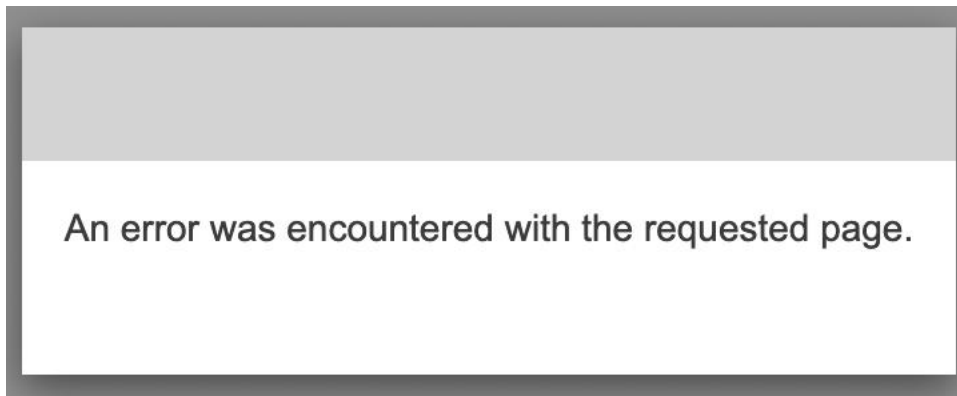
Install node version 14

```
npm install -g n
mkdir ~/.local
N_PREFIX=$HOME/.local n 14
sudo rm -rf /usr/local/node
echo 'export PATH="$HOME/.local/bin:$PATH"' >> ~/.bashrc
```

Close the current terminal and open a new terminal. Confirm the node version after.

Error logging into Service Workbench after install

Re-run the deploy script.

IAM role not available when creating EC2 instance

Choose the refresh button if the IAM role with admin permissions is not available. If it still does not appear, start over with the creation of the EC2 instance.

Best practice is to set up the IAM role in advance of the EC2 instance.