AWS SAM Installation

Along with other software required.

Contents

[Installing Docker 1](#_Toc38632764)

[Installing AWS CLI 1](#_Toc38632765)

[Installing SAM Local 1](#_Toc38632766)

[Installing IntelliJ IDEA and AWS Toolkit 1](#_Toc38632767)

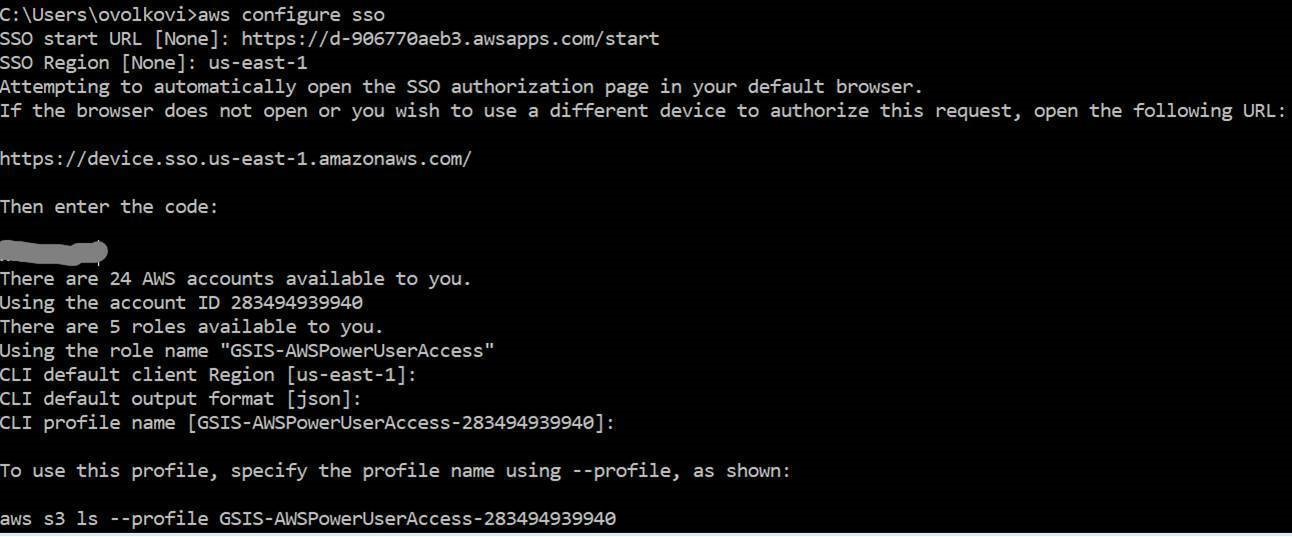
[Set Up A New Serverless Application 2](#_Toc38632768)

[Connecting SAM Local to an MySQL Database 3](#_Toc38632769)

# Installing Docker

1. Go to Docker’s official website and download Docker Desktop
2. During the installation DON’T select the option for having Windows Containers (keep Linux Containers)
3. After the installation is done, ask the IT person to help you add yourself in the docker-users group.
4. Go on the Docker website and create a Docker account. Log in to Docker with that account.
5. To verify the installation, follow the steps on <https://docs.docker.com/docker-for-windows/#test-your-installation>

# Installing AWS CLI

1. Download and Install the AWS CLI from this link <https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-windows.html#cliv2-windows-install>
2. ~~Configure the AWS CLI with your AWS account following the steps on this link [https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html#cli-quick](https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-configure.html" \l "cli-quick-configuration)-configuration~~
3. Please find below the link to the instructions explaining how to configure the AWS CLI to authenticate the user with AWS SSO to get short-term credentials to run AWS CLI commands.   
     
   <https://docs.aws.amazon.com/cli/latest/userguide/cli-configure-sso.html>. You should use **us as the SSO start URL and us-east-1 as the client region.**  
     
   Please find below the example:  
     
   

# Installing SAM Local

1. Install the SAM CLI from the following link <https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/serverless-sam-cli-install-windows.html> (install it at the default location provided by the installer)
2. To cover all the bases, also install the npm package npm install -g aws-sam-local

# Installing IntelliJ IDEA and AWS Toolkit

(IntelliJ IDEA is an IDE for Java development which AWS officially supports well, making SAM development easy)

1. Go to IntelliJ IDEA website and get the community version installed
2. During installation, select AWS Toolkit as additional packages to be installed.
3. Follow this link for setting up AWS Credentials: <https://docs.aws.amazon.com/toolkit-for-jetbrains/latest/userguide/setup-credentials.html>. After setting it up, set region as N. Virginia
4. Maven comes bundled with IDEA, but still if you want to set it up manually, go to File > Settings > Build, Execution, Deployment > Build Tools > Maven and set the Maven Home Repository
5. Go to File > Project Structure and set the appropriate JDK

# Set Up A New Serverless Application

1. Open IntelliJ IDEA. Go to File > New > Project...
2. On the side menu, select AWS > AWS Serverless Project
3. Complete the wizard (select runtime as java11 and select JDK 11 (as configured in the previous section)).
4. After the new project is setup, on the bottom right, select the appropriate region and profile.
5. The SAM Application has been created based on AWS SAM HelloWorld template.
6. On the top right, select “(Local) HelloWorldFunction” > Edit Configurations.
7. In the input section, select File. Navigate to (Project Root)\events\event.json so that the Lambda function has an input file. Click on Apply and OK
8. Go to pom.xml, and add these dependencies/plugins
   1. Gson Dependency (in the dependencies section)

<dependency>

<groupId>com.google.code.gson</groupId>

<artifactId>gson</artifactId>

<version>2.8.6</version>

</dependency>

* 1. Maven Surefire Plugin (in the plugins section)

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.19.1</version>

<configuration>

<testFailureIgnore>true</testFailureIgnore>

</configuration>

</plugin>

* 1. MySQL Connector

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.19</version>

</dependency>

Right Click on pom.xml > Maven > ReImport

1. Make sure Docker is running in the background. Hit run and the application should build and run.

# Connecting SAM Local to an MySQL Database

1. Create a database called “test” in MySQL. Create a table in it and put some values inside it.
2. Go to “[Local] HelloWorldFunction” on top right > Edit Configurations
3. Go to SAM CLI section and in Docker Network put the value as “host”.
4. (This step requires admin privileges, if this is skipped the program won’t run without the –use-container flag in the “sam build” command.)

Go to “C:\Program Files\Amazon\AWSSAMCLI\runtime\lib\site-packages\aws\_lambda\_builders\workflows\java\_maven\maven.py”. Comment out line 31 (there have been a lot of MavenBuildFailures if this is not done and the –use-container flag is not used).

1. In the lambda function, set the url as “jdbc:mysql://host.docker.internal:3306/test” (note the usage of host.docker.internal instead of localhost. This is because the SAM Application runs in a docker container, and so it can’t reach localhost).
2. Import “java.sql.\*”
3. Try Connection con = DriverManager.getConnection(url, user, password). It should work.
4. To verify we can try retrieving data from the table created. Make a string query.
5. Do Statement stmt = con.createStatement();
6. Get the results as ResultSet rs = stmt.executeQuery(query).

# CORS Issue

1. We encountered a CORS issue, we solved it by this.s
2. By following the steps mentioned in <https://enable-cors.org/server_awsapigateway.html> (the key is DEPLOYING the API after the changes have been made).
3. Putting 3 headers:
   1. headers.put("Access-Control-Allow-Origin", "\*");
   2. headers.put("Access-Control-Allow-Methods", "\*");
   3. headers.put("Access-Control-Allow-Headers","\*");

in the response body.