**All of the resources created below should be in the same region**

**Setting up CodeCommit:**

Follow this video to create a CodeCommit repo to host the source code.

<https://www.youtube.com/watch?v=W-SM4m19JMA>

There will be two branches. **Staging** will be connected to the staging environment and **Master** will be connected to the production environment.

**Setting up S3:**

Create two buckets for Production and Staging artifact storage and note down the bucket names. Sample names can be “or.serverless.stg” for **Staging** and “or.serverless.prod” for **Production**

**Setting up Cloud9:**

Open up cloud9 and upload the zipped artifact (scheduled-lambda.zip) & extract it into to cloud9 environment folder.

The zipped artifact contains a AWS Cloudformation stack which deploys a lambda function, a dynamodb table and a event bridge scheduler which runs the lambda every 5 minutes to insert a dummy record at dynamodb table.

Go back to 1.Getting familiar with AWS Serverless.docx if do not understand

It also has a buildspec.yaml for AWS Codebuild to build the dependencies and run test.

**Complete following steps to create the initial stacks for the stage and production env:**

Run all the commands from same terminal or run the **source set-env.sh** if you change the terminal

1. Open **set-env.sh** file and then set **STAGE=”Staging”, AWS\_REGION**  to the region you are in, **s**et up **STAGE\_BUCKET** name to “or.serverless.stg” and run **source set-env.sh.** Now run **deploy-stage.sh** to setup initial Staging environment
2. Open **set-env.sh** file and then set **STAGE=”Production” s**et up **STAGE\_BUCKET** name to “or.serverless.prod” and run **source set-env.sh.** Now run **deploy-stage.sh** to setup initial Production environment
3. When the deployment is complete you can check that a DynamoDB table named “simple-table-stg”, “simple-table-prod”, two event rule at Cloudwatch console **Events**>**Rules**  and two lambda at aws lambda console, connected to stage and production stack populating both database every 5 minutes with records. Check the AWS Lamada console for those two(stg and prod) functions and check the **Monitoring** tab for logs.
4. Now Push this source code to AWS CodeCommit’s **stage** and **master** branch

**To run test locally :**

* **Unit test:**
  + Do **cd sample-lambda/ && npm install** to install the dependencies
  + run unit test using **npm run test** at the “sample-lambda” directory
* **Lambda docker based local test**:
  + Do **sam local invoke ScheduledFunction** at the scheduled function directory. Go back to 1.Getting familiar with AWS Serverless.docx if do not understand