**AWS Lambda Function Node.js Boilerplate**

[AWS Lambda function](https://docs.aws.amazon.com/lambda/latest/dg/welcome.html) allows you to run a piece of code in a serverless manner. All you need to do is to upload your function to the cloud. It sounds easy. The problem is that the function seldom runs on its own.

Most of the time, it is only useful when you have other AWS services interacting with it. You need to create an event source, which can be events sent from services such as API Gateway, S3, CloudWatch etc. You may also need to have targets which are AWS services like RDS, Redshift, S3, DynamoDB, Load Balancer and so on. To access various services, the function needs to run in the correct subnet and have the right IAM roles.

When I started deploying lambda functions, I was in fact spending more time in configuring AWS resources than writing the function itself. Deployment was another issue. To have a good CD/CI pipeline, I needed to couple the lambda function deployment to the creation of all the related AWS resources.

The first ever lambda function I deployed was done with AWS CLI command. Yes, you can deploy lambda with a bunch of command lines (this is documented in the previous post [here](https://www.mydatahack.com/event-driven-data-ingestion-in-aws-s3-to-s3/)). Then, I discovered the framework called [serverless](https://serverless.com/) that makes packaging of lambda function easy. With serverless, you can configure your lambda function and create associated resources in ymal files. I have introductory lambda function posts that uses serverless with Python [here](https://www.mydatahack.com/advanced-aws-lambda-python-function-deployment-with-serverless/) and Node.js [here](https://www.mydatahack.com/event-driven-s3-data-ingestion-with-node-js-lambda-function-and-deploy-it-with-serverless/).

Configuring serverless for proper CD/CI deployment requires a bit of learning curve. You need to deploy the same lambda function and associated resources throughout the environment by changing the environment parameter.

For production grade code, you also want to include unit & integration tests as well as test coverage report.

After a few attempts, I now created a boilerplate that allows you to create production level Node.js lambda functions and deploy it with any CD/CI tool such as Jenkins.

Check this out on our git repo: [**nodejs-lambda-serverless-bilerplate**](https://github.com/mydatahack/nodejs-lambda-serverless-boilerplate)

For creating basic lambda function driven API, all you need to do is to change the place holders in the yaml files. The environment variable is passed through the stage argument in the sls deploy command.

I also curated the unit testing and code coverage tool ([mocha](https://mochajs.org/) and [Istanbul](https://istanbul.js.org/)) with test examples, so that you can get it started straight away.

Simply pull the repo and try running tests to see if it works!

git clone <https://github.com/mydatahack/nodejs-lambda-serverless-boilerplate.git>

npm install

npm test

npm run-script integration

See what you think. Let us know what you think of this boilerplate. Hopefully, it helps you to create awesome lambda functions!