Deployment guide for online order + queue system

1. Client (ReactJs)
2. Online order + queue system (Python3)
   1. Parameters
   2. Server startup
3. AWS deployment (Serverless framework)
   1. Prerequisite
   2. Folder structure
   3. Parameters
   4. Deployment
4. Client (ReactJs)

You can use http-server that simply the time for the deployment testing. To install, run “brew install http-sever” from the brew command (OSX)

The default port is 8080 in http-server in your localhost.

For the ReactJs build, please execute npm run build. The complied files will be in the folder /Client/build.

Copy the build files into http-server folder (Default is ~/.public) and reload the page for testing.

1. Online order + queue system (Python3)
2. Parameters

Inside /Tools, there is a local\_server.py for you to start local server to test, it will load the config in /Deploy/resourcs/ encrypt-dev.yml and env-dev.yml

|  |  |
| --- | --- |
| RDS\_XXXXXXX | Database configuration |
| SESSION\_EXPIRATION\_MINUTE | How long will the session being expired after finish queuing (i.e Session valid for user entering the order system to place order) |
| CONCURRENT\_MAXIMUM\_USERS | Maximum users that can enter the order system simutaineously |
| VPC\_XXXXXXX | AWS subnet id and security group id |

1. Server startup

To start the local server, simply apply python3 ./local\_server.py for testing

1. AWS deployment (Serverless framework)
   1. Prerequisite

There are few items you need to setup if you choose AWS deployment

* AWS account
* AWS IAM setup for policy

You can apply the policy in /Setup/03\_AWS\_IAM\_POLICY.json , this policy enables the deployment account having the sufficient right to access AWS S3, CloudFormation, CloudWatch Log, ApiGateway and Lambda

* AWS IAM programmable access key

Obtain the IAM role key from the AWS website, and place into your aws\_creditals file (inside ~/.aws)

Graphical user interface, text, application

Description automatically generated  
  
Text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

* AWS RDS

Remember to mark down the subnet id and security group id in the RDS portal, you will need these information inside env-dev.yml or env-prd.yml

Graphical user interface, application

Description automatically generated

* AWS ApiGateway after setup

Make sure you need to select “Deploy API” for the first time after Serverless deployment success to your AWS account

Graphical user interface, application

Description automatically generated

1. Folder structure

There is a Deployment script for the Serverless framework, which is located in Deploy folder

|  |  |  |
| --- | --- | --- |
| File / Folder | Description | Usage |
| Deploy.sh | Deployment script for packaging, project build and deploy to AWS server using servless/serverless.yml config | First time deploy :  Sh ./Deploy.sh deploy  Other deploy moment :   * Sh ./Deploy.sh deploy * Sh ./Deploy.sh deploy -g <group> * Sh ./Deploy.sh deploy function –function <function name> |
| serverless | Configuration of serverless deployment (serverless.yml) | * Serverless.yml (Servlerless configuration) * Package.json (Package json for serverless npm build) * Package-lock.json (Package versioning lock configurations) |
| Resources | Configuration of environment | * Env-dev.yml : Dev environment configuration * Encrypt-dev.yml : Encryption key for Dev environment configuration * Env-prd.yml : Production environment configuration * Encrypt-prd.yml : Encryption key for Production environment configuration |

1. Parameters

|  |  |
| --- | --- |
| Serverless.yml | Environment paramaeter inside provider.stage  Prd = it will load env-prd.yml and encrypt-prd.yml. If changing to dev, it will load the dev set respectively.  Text  Description automatically generated with medium confidence |
| Resources/encrypt-XXXX.yml | Encryption key  You can apply any string for encryption key or leave it blank |
| Resources/env-XXXX.yml | Config for the serverless deployment  If you enter the encryption key, make sure you will need to generate the according information inside Tools/encrypt\_string.py. You can test the encrypted message using Tools/decrypto\_string.py |

1. Deployment

|  |  |
| --- | --- |
| First time deployment | Sh ./Deploy.sh deploy |
| Deploy all items | Sh ./Deploy.sh deploy |
| Deploy group of functions | Sh ./Deploy.sh deploy -g <group>  Group can be all/other function names, you can check the deploy.sh for more information |
| Calling serverless function | Sh ./Deploy.sh XXXXX XXXXX XXXXXX  If will treat the same as serverless XXXXX XXXXX XXXXX. |