NAMASTE NODE.JS SEASON 3

Episode-5
Hand Written Notes

-By Shanmuga Priya

www.linkedin.com/in/shanmuga-priya-e-tech2

Episode - 5

Sending Emails using Amazon SES (simple Email service)

Step1: Set a new IAM user.

IAM (Identify and Access Management)

↳ it gives Permission to different services inside AWS.

→ To create a new user search "IAM" in search bar present in AWS console.

→ click on "user" then "create user" btn. add a username as you wish and click on next and set permission click on "attach policies directly" it will display a list of policies search "amazon SES" in it and select "AmazonSESFullAccess" and click next it will take you to the summary page click on "create new user".

→ Now user is successfully created.

Step 2: Setup SES account.

Once the IAM user is created search for "SES" in searchbar to setup SES.

step1: Account creation:

click on "view get setup page" in the account dashboard and it will take you to "get setup page" click on "create identity"

→ we can create identity either by using domain name (or) email address Since we created a domain name already click on "domain" option. and enter a domain name. and verify the domain name using "Easy DKIM" type. select the key length of your choice. and click on "create Identity".

→ our Identity is created but its verification is in pending state to complete its verification, configure DKIM in DNS record of cloudflare.

Step 2: configuring DKIM in cloudflare

→ copy the CNAME records provided by DKIM identify to cloudflare's DNS record.

→ In cloudflare, go to DNS Records click on add record choose the type as "CNAME" and copy the CNAME Records from DKIM identity and turn off the proxy. (3)

→ once the records are needed SES takes some time to verify the records. once the identity is verified we can ~~the~~ see the identity status as "verified."

Step 3: Request Production action

→ Move back to setup page and click on "Request production access" click on "Transactional" as mail type and provide our website url. and submit it.

→ It will take some time to grand permission.

Step 4: Using it in the code for sending Email

→ For that we need to get the credentials of the IAM user that we have created in step 1.

→ go to the user and click on "security credentials" click on "create Access key" and select the usecase as others and give a tag name for it as your wish (or) just skip it and click on "create access key"

→ Now, the secret key is generated copy it and place it in the .env file of our Project and also a access key.

step5: Copy the code from AWS SDK V3.

→ Google the AWS SDK V3 documentation and move onto Amazon SES search for sendEmail.

Step a: Create sesclient:

→ In utils folder of our project, create a new file called "sesclient.js" and copy the sesclient code from github repo.

→ Inorder to use this code we need to install Package

npm i @aws-sdk/client-ses

and edit the code to include access key.

eg: //sesclient.js

```
Const {SESClient} =require("@aws-sdk/client-ses")
Const REGION= "us-east-1" → it should be equal to the region where
                               we sending from.
Const sesclient = new SESclient({
      region: REGION,
      credentials: {
            accessKeyId:process.env. AWS_Access_KEY,
            secretAccesskey: Process.env. AWS_SECRET_KEY,
      },
})
module. exports = {sesclient}
```

(*)

stepb: write a code for sending emails.

→ copy the code of sendmail from github repo of aws SDK V3.
and edit the recipient & sender email address.

→ the email we are using here should be verified as user in IAM.
if not add a new user in IAM with this email.

step6: Using the sendemail fn in project:

whenever a connection req is ~~send~~ made it should send a email to the other user.

eg: //api for sending connection req.

```
~~step~~ const sendEmail = require("../utils/sendEmail")

    requestRouter.post("/request/send/:status/:touserId", userAuth,
     asyn(req, res)=>{
        try{
          - - - -
           - - - - Logic for sending a connection req ...
          - - - -
          const emailres = await sendEmail.run()
          console.log(emailres)

          - - - - sending res to user - -
          - - - - - - - -
        }catch(){
          - - -
        }
    })
```

whenever person1 makes connectn req to person 2, email is send to person 2 stating that person1 sent you a connection req.

——— × ———