1. Click on Services, go to Lambda service
2. Click on Create function
3. Select Author from scratch
4. Configure below settings:
    - Function name: FAQ
    - Runtime: NodeJS 10.x
    - Expand the dropdown with Choose or create an execution role
    - Execution role: Use an existing role
    - Existing role: lambda-basic-execution
    - Click on create function

    A page will be displayed with your function configuration.

5. Scroll down to function code and delete all of the existing code in code editor.
6. Function Code:

```
var json = {
  "service": "lambda",
  "reference": "https://aws.amazon.com/lambda/faqs/",
  "questions": [{
    "q": "What is AWS Lambda?",
    "a": "AWS Lambda lets you run code without provisioning or managing servers. You pay only for the compute time you consume - there is no charge when your code is not running. With Lambda, you can run code for virtually any type of application or backend service - all with zero administration. Just upload your code and Lambda takes care of everything required to run and scale your code with high availability. You can set up your code to automatically trigger from other AWS services or call it directly from any web or mobile app."
  },{
    "q":"What events can trigger an AWS Lambda function?",
    "a":"You can use AWS Lambda to respond to table updates in Amazon DynamoDB, modifications to objects in Amazon S3 buckets, logs arriving in Amazon CloudWatch logs, incoming emails to Amazon Simple Email Service, notifications sent from Amazon SNS, messages arriving in an Amazon Kinesis stream, client data synchronization events in Amazon Cognito, and custom events from mobile applications, web applications, or other web services. You can also invoke a Lambda function on a defined schedule using the AWS Lambda console."
  },{
    "q":"When should I use AWS Lambda versus Amazon EC2?",
    "a":"Amazon Web Services offers a set of compute services to meet a range of needs. Amazon EC2 offers flexibility, with a wide range of instance types and the option to customize the operating system, network and security settings, and the entire software stack, allowing you to easily move existing applications to the cloud. With Amazon EC2 you are responsible for provisioning capacity, monitoring fleet health and performance, and designing for fault tolerance and scalability. AWS Elastic Beanstalk offers an easy-to-use service for deploying and scaling web applications in which you retain ownership and full control over the underlying EC2 instances. Amazon Elastic Container Service is a scalable management service that supports Docker containers and allows you to easily run distributed applications on a managed cluster of Amazon EC2 instances. AWS Lambda makes
```

it easy to execute code in response to events, such as changes to Amazon S3 buckets, updates to an Amazon DynamoDB table, or custom events generated by your applications or devices. With Lambda you do not have to provision your own instances; Lambda performs all the operational and administrative activities on your behalf, including capacity provisioning, monitoring fleet health, applying security patches to the underlying compute resources, deploying your code, running a web service front end, and monitoring and logging your code. AWS Lambda provides easy scaling and high availability to your code without additional effort on your part."

},{

"q":"What kind of code can run on AWS Lambda?",

"a":"AWS Lambda offers an easy way to accomplish many activities in the cloud. For example, you can use AWS Lambda to build mobile back-ends that retrieve and transform data from Amazon DynamoDB, handlers that compress or transform objects as they are uploaded to Amazon S3, auditing and reporting of API calls made to any Amazon Web Service, and server-less processing of streaming data using Amazon Kinesis."

},{

"q":"What languages does AWS Lambda support?",

"a":"AWS Lambda supports code written in Node.js (JavaScript), Python, and Java (Java 8 compatible). Your code can include existing libraries, even native ones. Lambda functions can easily launch processes using languages supported by Amazon Linux, including Bash, Go, and Ruby. Please read our documentation on using Node.js, Python and Java."

},{

"q":"Can I access the infrastructure that AWS Lambda runs on?",

"a":"No. AWS Lambda operates the compute infrastructure on your behalf, allowing it to perform health checks, apply security patches, and do other routine maintenance."

},{

"q":"How does AWS Lambda isolate my code?",

"a":"Each AWS Lambda function runs in its own isolated environment, with its own resources and file system view. AWS Lambda uses the same techniques as Amazon EC2 to provide security and separation at the infrastructure and execution levels."

},{

"q":"How does AWS Lambda secure my code?",

"a":"AWS Lambda stores code in Amazon S3 and encrypts it at rest. AWS Lambda performs additional integrity checks while your code is in use."

},{

"q":"What is an AWS Lambda function?",

"a":"The code you run on AWS Lambda is uploaded as a Lambda function. Each function has associated configuration information, such as its name, description, entry point, and resource requirements. The code must be written in a stateless style i.e. it should assume there is no affinity to the underlying compute infrastructure. Local file system access, child processes, and similar artifacts may not extend beyond the lifetime of the request, and any persistent state should be stored in Amazon S3, Amazon DynamoDB, or another Internet-available storage service. Lambda functions can include libraries, even native ones."

},{

"q":"Will AWS Lambda reuse function instances?",

"a":"To improve performance, AWS Lambda may choose to retain an instance of your function and reuse it to serve a subsequent request, rather than creating a new copy. Your code should not assume that this will always happen."

},{
"q":"What if I need scratch space on disk for my AWS Lambda function?",
"a":"Each Lambda function receives 500MB of non-persistent disk space in its own /tmp directory."
},{
"q":"Why must AWS Lambda functions be stateless?",
"a":"Keeping functions stateless enables AWS Lambda to rapidly launch as many copies of the function as needed to scale to the rate of incoming events. While AWS Lambda's programming model is stateless, your code can access stateful data by calling other web services, such as Amazon S3 or Amazon DynamoDB."
},{
"q":"Can I use threads and processes in my AWS Lambda function code?",
"a":"Yes. AWS Lambda allows you to use normal language and operating system features, such as creating additional threads and processes. Resources allocated to the Lambda function, including memory, execution time, disk, and network use, must be shared among all the threads/processes it uses. You can launch processes using any language supported by Amazon Linux."
},{
"q":"What restrictions apply to AWS Lambda function code?",
"a":"Lambda attempts to impose few restrictions on normal language and operating system activities, but there are a few activities that are disabled: Inbound network connections are managed by AWS Lambda, only TCP/IP sockets are supported, and ptrace (debugging) system calls are restricted. TCP port 25 traffic is also restricted as an anti-spam measure."
},{
"q":"How do I create an AWS Lambda function using the Lambda console?",
"a":"You can author the code for your function using the inline editor in the AWS Lambda console. You can also package the code (and any dependent libraries) as a ZIP and upload it using the AWS Lambda console from your local environment or specify an Amazon S3 location where the ZIP file is located. Uploads must be no larger than 50MB (compressed). You can use the AWS Eclipse plugin to author and deploy Lambda functions in Java and Node.js. If you are using Node.js, you can author the code for your function using the inline editor in the AWS Lambda console. Go to the console to get started."
},{
"q":"How do I create an AWS Lambda function using the Lambda CLI?",
"a":"You can package the code (and any dependent libraries) as a ZIP and upload it using the AWS CLI from your local environment, or specify an Amazon S3 location where the ZIP file is located. Uploads must be no larger than 50MB (compressed). Visit the Lambda Getting Started guide to get started."
},{
"q":"Which versions of Python are supported?",
"a":"Lambda provides a Python 2.7-compatible runtime to execute your Lambda functions. Lambda will include the latest AWS SDK for Python (boto3) by default."
},{
"q":"How do I compile my AWS Lambda function Java code?",
"a":"You can use standard tools like Maven or Gradle to compile your Lambda function. Your build process should mimic the same build process you would use to compile any Java code that depends on the AWS SDK. Run your Java compiler tool on your source files and

```
include the AWS SDK 1.9 or later with transitive dependencies on your classpath. For more
details, see our documentation."
 },{
  "q":"What is the JVM environment Lambda uses for execution of my function?",
  "a":"Lambda provides the Amazon Linux build of openjdk 1.8."
 }
 ]
}

exports.handler = function(event, context) {
   var rand = Math.floor(Math.random() * json.questions.length);
   console.log("Quote selected: ", rand);

   var response = {
      body: JSON.stringify(json.questions[rand])
   };
   console.log(response);
   context.succeed(response);
};
```

The code defines a list of Frequently Asked Questions(FAQs).

Returns a random FAQ when asked.

7. Scroll down to the basic settings section

8. For Description enter Provide a random FAQ

9. Scroll up to the Designer Section

10. Click Add Trigger

- Select a trigger : API Gateway
- API: Create a new API
- Choose a template: REST API
- Security: Open
- Expand Additional Settings
- API name: FAQ-API
- Deployment Stage: myDeployment

11. Click Add

12. Click Save

13. Copy the API endpoint to your clipboard and then

- In a new browser tab, paste the API Endpoint
- Press Enter to go to the URL

14. Close the FAQ browser tab and return to Lambda on the management console.

15. At the top of screen, Click Test

- Event Name: BasicTest
- Delete all content and keep {} on the screen

16. In the Execution result: Suceeded windows Expand Details tab

17. Click Monitoring Tab

18. Click View Logs in Cloudwatch

19. Click on one of the log streams

You will be presented with same event data that was displayed in Lambda Console. Examine the contents of each line to view the log information.