

SkipQ

SCRUM

Document

Catalog

1. Orientation.....	2
1. 1. Cloud Computing.....	2
1. 1. 1. What is cloud computing.....	2
1. 1. 2. Benefits.....	2
1. 2. Amazon Web Services(AWS).....	2
1. 2. 1. What is AWS.....	2
1. 2. 2. Benefits.....	2
1. 3. DevOps.....	3
1. 3. 1. What is DevOps.....	3
1. 3. 2. Benefits.....	3
2. Sprint 1.....	3
2. 1. Technologies Used.....	3
2. 1. 1. Cloud9.....	3
2. 1. 2. Lambda.....	3
2. 1. 3. Cloud Watch.....	4
2. 1. 4. Dynamo DB.....	4
2. 2. Hello Lambda(Task 1).....	4
2. 2. 1. Implementation.....	4
2. 2. 2. Results.....	4
2. 3. Web Health Lambda(Task 2).....	5
2. 3. 1. Implementation.....	5
2. 3. 2. Results.....	5
2. 4. Web Matrix Lambda(Task 3).....	5
2. 4. 1. Implementation.....	5
2. 4. 2. Results.....	5
2. 5. Web Alarm Lambda(Task 4).....	6
2. 5. 1. Implementation.....	6
2. 5. 2. Results.....	6
2. 6. Dynamo DB & Web Lambda(Task 5).....	7
2. 6. 1. Implementation.....	7
2. 6. 2. Results.....	7
2. 7. Errors & Solution.....	8
3. Sprint 2.....	10
3. 1. Technologies Used.....	10
3. 1. 1. Pipelines.....	10
3. 1. 2. CloudFormation.....	11
3. 2. Pipeline Source & build (Task 1).....	11
3. 2. 1. Implementation.....	11
3. 2. 2. Results.....	11
3. 3. Beta Stage(Task 2).....	12
3. 3. 1. Implementation.....	12
3. 3. 2. Results.....	12
3. 4. Unit and integration tests(Task 3).....	13
3. 4. 1. Implementation.....	13
3. 4. 2. Results.....	13
3. 5. Production stage(Task 4).....	13
3. 5. 1. Implementation.....	13
3. 5. 2. Results.....	13
3. 6. Roll Back(Task 5).....	14
3. 6. 1. Implementation.....	14
3. 6. 2. Results.....	14
3. 7. Errors & Solution.....	14

1. Orientation

1. 1. Cloud Computing

1. 1. 1. What is cloud computing

The delivery of various services over the Internet is known as cloud computing. These resources include data storage, servers, databases, networking, and software, among other tools and applications. As long as an electronic gadget has internet access, it has access to data and the software programmers needed to execute it.

1. 1. 2. Benefits

1. 1. 2. 1. Cost saving

If you are worried about the price tag that would come with making the switch to cloud computing, you aren't alone 20% of organisations are concerned about the initial cost of implementing a cloud-based server. But those who are attempting to weigh the advantages and disadvantages of using the cloud need to consider more factors than just initial price they need to consider ROI.

1. 1. 2. 2. Security

Many organization have security concerns when it comes to adopting a cloud-computing solution. After all, when files, programs, and other data aren't kept securely onsite, how can you know that they are being protected? If you can remotely access your data, then what's stopping a cyber criminal from doing the same thing? Well, quite a bit, actually.

1. 1. 2. 3. Flexibility

Your business has only a finite amount of focus to divide between all of its responsibilities. If your current IT solutions are forcing you to commit too much of your attention to computer and data-storage issues, then you aren't going to be able to concentrate on reaching business goals and satisfying customers. On the other hand, by relying on an outside organization to take care of all IT hosting and infrastructure, you'll have more time to devote toward the aspects of your business that directly affect your bottom line.

1. 2. Amazon Web Services(AWS)

1. 2. 1. What is AWS

AWS (Amazon Web Services) is a comprehensive, evolving cloud computing platform provided by Amazon that includes a mixture of infrastructure as a service (IaaS), platform as a service (PaaS) and packaged software as a service (SaaS) offerings.

1. 2. 2. Benefits

1. 2. 2. 1. Easy to Use

AWS is designed to allow application providers, ISVs, and vendors to quickly and securely host your applications – whether an existing application or a new SaaS-based application. You can use the AWS Management Console or well-documented web services APIs to access AWS's application hosting platform.

1. 2. 2. 2. Reliable

With AWS, you take advantage of a scalable, reliable, and secure global computing infrastructure, the virtual backbone of Amazon.com's multi-billion dollar online business that has been honed for over a decade.

1. 2. 2. 3. Scale able

Using AWS tools, Auto Scaling, and Elastic Load Balancing, your application can scale up or down based on demand. Backed by Amazon's massive infrastructure, you have access to compute and storage resources when you need them.

1. 3. DevOps

1. 3. 1. What is DevOps

DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes.

1. 3. 2. Benefits

1. 3. 2. 1. Ensure Fast Deployment

Faster and more frequent delivery of updates and features will not only satisfy the customers but will also help your company take a firm stand in a competitive market.

1. 3. 2. 2. Stabilize Work Environment

Do you know that the tension involved in the release of new features and fixes or updates can topple the stability of your workspace and decreases the overall productivity? Improve your work environment with a steady and well-balanced approach of operation with DevOps practice.

1. 3. 2. 3. Improvement In Product Quality

Collaboration between development and operation teams and frequent capturing of user feedback leads to a significant improvement in the quality of the product.

2. Sprint 1

2. 1. Technologies Used

2. 1. 1. Cloud9

AWS Cloud9 is a cloud-based integrated development environment (IDE) that lets you write, run, and debug your code with just a browser. It includes a code editor, debugger, and terminal. Cloud9 comes prepackaged with essential tools for popular programming languages, including JavaScript, Python, PHP, and more, so you don't need to install files or configure your development machine to start new projects. Since your Cloud9 IDE is cloud-based, you can work on your projects from your office, home, or anywhere using an internet-connected machine. Cloud9 also provides a seamless experience for developing serverless applications enabling you to easily define resources, debug, and switch between local and remote execution of serverless applications. With Cloud9, you can quickly share your development environment with your team, enabling you to pair program and track each other's inputs in real time.

2. 1. 2. Lambda

Lambda is a compute service that lets you run code without provisioning or managing servers. Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging. With Lambda, you can run code for virtually any type of application or backend service. All you need to do is supply your code in one of the languages that Lambda supports.

2. 1. 3. Cloud Watch

Amazon CloudWatch is a monitoring and observability service built for DevOps engineers, developers, site reliability engineers (SREs), IT managers, and product owners. CloudWatch provides you with data and actionable insights to monitor your applications, respond to system-wide performance changes, and optimize resource utilization. CloudWatch collects monitoring and operational data in the form of logs, metrics, and events. You get a unified view of operational health and gain complete visibility of your AWS resources, applications, and services running on AWS and on-premises. You can use CloudWatch to detect anomalous behavior in your environments, set alarms, visualize logs and metrics side by side, take automated actions, troubleshoot issues, and discover insights to keep your applications running smoothly.

2. 1. 4. Dynamo DB

Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. DynamoDB lets you offload the administrative burdens of operating and scaling a distributed database so that you don't have to worry about hardware provisioning, setup and configuration, replication, software patching, or cluster scaling. DynamoDB also offers encryption at rest, which eliminates the operational burden and complexity involved in protecting sensitive data. For more information, see DynamoDB Encryption at Rest.

2. 2. Hello Lambda(Task 1)

2. 2. 1. Implementation

For our first task on AWS we implemented a Hello world lambda function, as beginners do in any new programming field. Created the directory for project imported the important libraries to our stack file and created a Lambda file in resources folder to start the process. First of all Defined a handler function for our Lambda file in stack. And in that file defined all the parameters required for a lambda function invoke in our stack file. In lambda file defined function for event and context and then given two strings to it for printing in console. User name concatenated with the hello word to print my first cloud computing message.

2. 2. 2. Results

Results of first hello lambda function are given in figure 1.

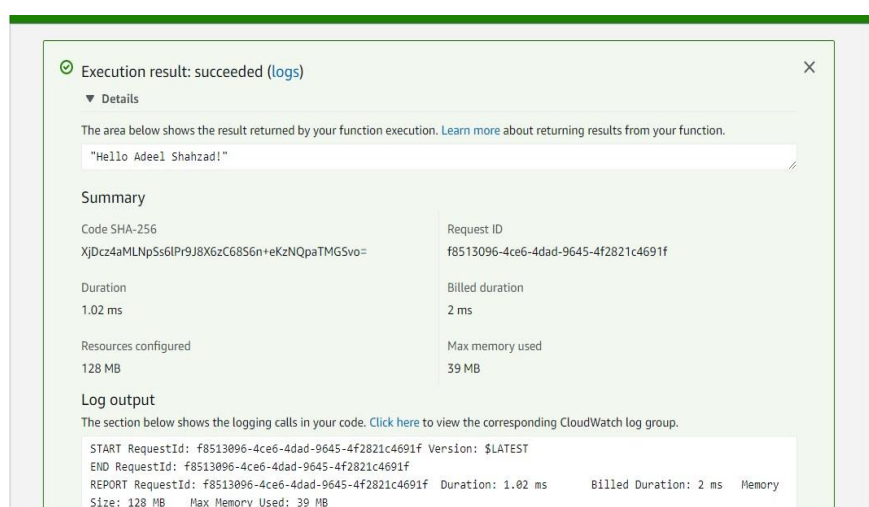


Figure 1 : Hello lambda results

2. 3. Web Health Lambda(Task 2)

2. 3. 1. Implementation

In this task we tried to implement a web health lambda function. This function returns the latency and availability of a given URL in Numeric values. As the same way started the lambda function and stack file for this task. Our stack files remains same because we don't need any extra libraries for this task. We made changes in lambda file and created two functions. First one is for calculating the availability of given URL in terms of ones and zeros. Used built in function for that which returns boolean output, so using if conditions converted them in ones and zeros. If a Web is not down it returns one else zero. And then created a function for latency of that web. It returns float values between zero and one. For latency requested a response from the given URL and saved the time before and after the response. Taken their difference and converted it to seconds as latency value.

2. 3. 2. Results

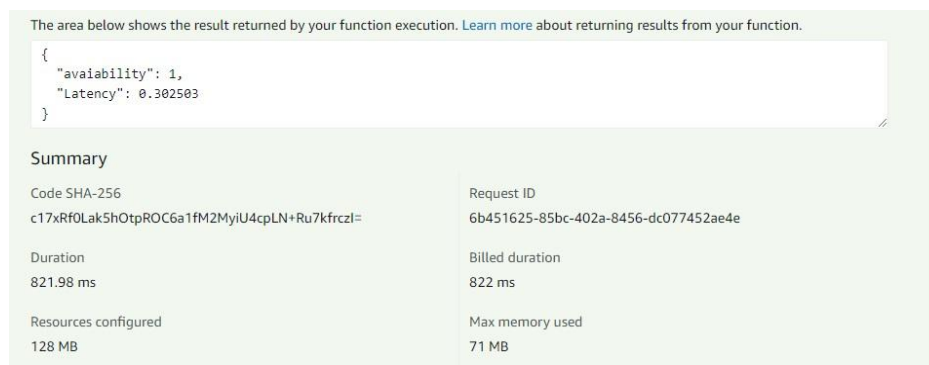


Figure 2: web health parameter

2. 4. Web Matrix Lambda(Task 3)

2. 4. 1. Implementation

In this task created a periodic lambda function that invokes after a certain duration to show the availability and latency of URL in a matrix graph to observe the trends related to that two parameters. For this task again used the functions from previous file that returns the values of latency and availability in numeric form. Created a matrix and given the values to that matrix. To be plotted in a graph and shown in cloud watch. In stack file created schedule for the lambda function to invoke after certain amount of time intervals. Imported cloud watch and related libraries in the lambda file. Initialized lambda rule function in the stack file to get access to cloud watch services.

2. 4. 2. Results

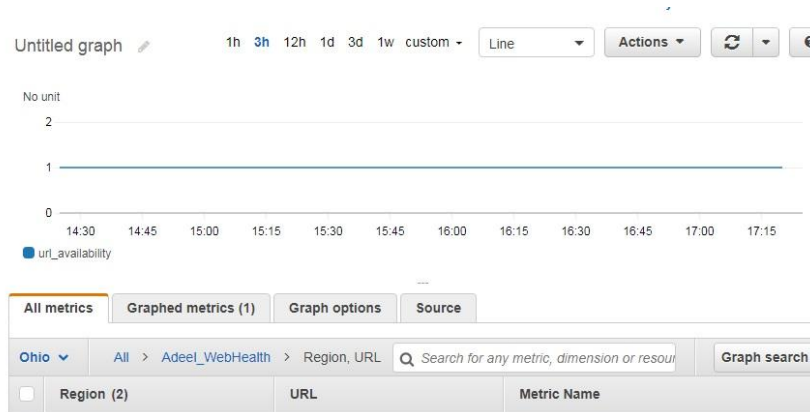


Figure 3:Availability Matrix

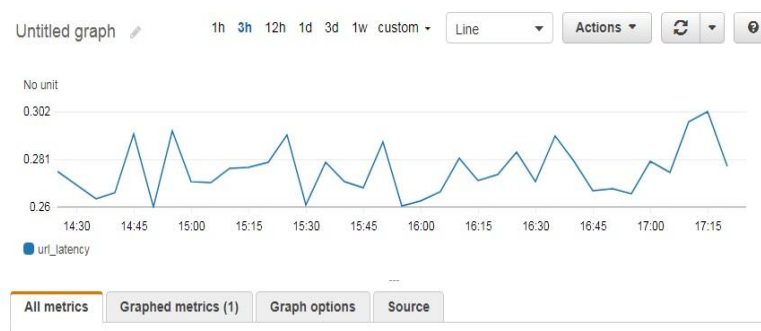


Figure 4:Latency Matrix

2. 5. Web Alarm Lambda(Task 4)

2. 5. 1. Implementation

This function not only shows the values of latency and availability of URL but creates an alarm If the values go beyond a certain threshold. For this function we used the functions from previous task. To implement it created another matrix in stack file and given it the same parameters as the matrix in lambda function. So that both of them get merged. Generated a threshold for each parameter and defined a alarm function from AWS libraries to create an alarm, Given this function all the parameters required to generate an alarm including the matrix and threshold. After that generated a action for each alarm by sending an email to the related person . For this imported the actions and subscriptions libraries. Subscribed to the email of given user and created a action for that email. So it sends data of the alarm to all subscribed emails.

2. 5. 2. Results

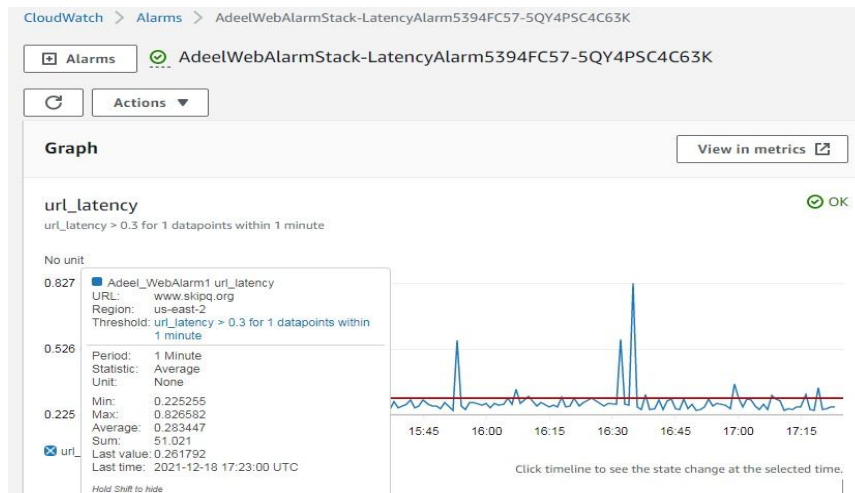


Figure 5:Latency Alarm Matrix

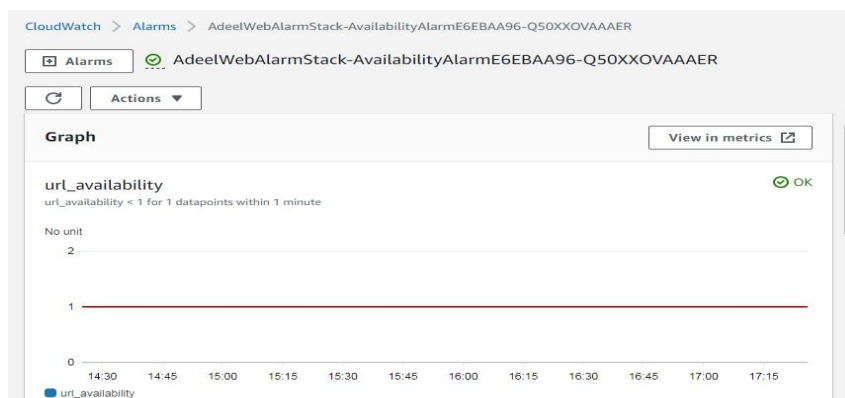


Figure 6:Availability Alarm Matrix

2. 6. Dynamo DB & Web Lambda(Task 5)

2. 6. 1. Implementation

During this task initialized a new bucket of multiple urls for alarm generation and that created a table for dynamoDB database to update the alarm values in it.First of all created the table for dynamoDB and then given lambda function all the access to read and write in that function. Created a separate lambda file for adding sns alarm values to it created events for message of sns alarm and provided the message values to our table to be stored in dynamoDB database. Created a bucket to read more than one urls

2. 6. 2. Results

► AdeelAlarmdynamo View table details

Expand to query or scan items.

Items returned (300) Actions Create item

< 1 ... > ⚙️ 🗖️

<input type="checkbox"/>	Timestamp ▾	Reason ▾
<input type="checkbox"/>	2021-12-2...	Threshold Crossed: 1 out of the last 1 datapoints [0.318658 (22/12/21 03:43:00)] was grea...
<input type="checkbox"/>	2021-12-2...	Threshold Crossed: 1 out of the last 1 datapoints [0.347319 (22/12/21 17:21:00)] was grea...

Figure 7:Dynamo table

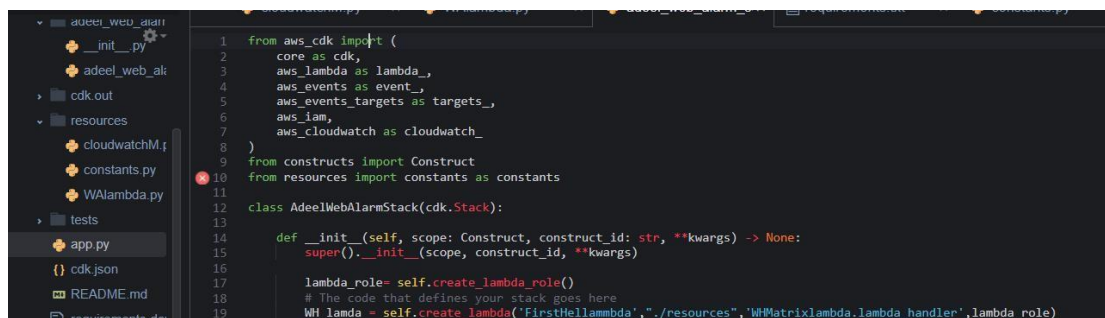
2. 7. Errors & Solution

```
(.venv) adeelshahzadskipq:~/environment/MyDemoRepo (master) $ cdk synth
Traceback (most recent call last):
  File "app.py", line 12, in <module>
    from my_demo_repo.my_demo_repo_stack import MyDemoRepoStack
  File "/home/ec2-user/environment/MyDemoRepo/my_demo_repo/my_demo_repo_stack.py", line 1, in <module>
    from aws_cdk import (
ImportError: cannot import name 'aws_es2' from 'aws_cdk' (unknown location)
Subprocess exited with error 1
```

Figure 8:CDK library missing

Solution:

This was the first error I encountered. My Ide was not able to find CDK module from the Libraries installed so I first used PIP commands to install the requirements to get started with the coding.

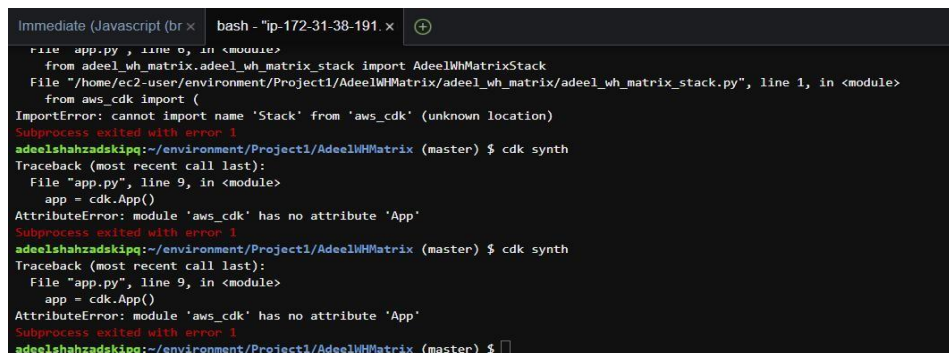


```
1 from aws_cdk import (
2     core as cdk,
3     aws_lambda as lambda_,
4     aws_events as event_,
5     aws_events_targets as targets_,
6     aws_lambda as lambda_,
7     aws_cloudwatch as cloudwatch_
8 )
9 from constructs import Construct
10 from resources import constants as constants
11
12 class AdeelWebAlarmStack(cdk.Stack):
13
14     def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
15         super().__init__(scope, construct_id, **kwargs)
16
17         lambda_role = self.create_lambda_role()
18         # The code that defines your stack goes here
19         WH_lambda = self.create_lambda('FirstHelloLambda', './resources', 'WHMatrixLambda.lambda_handler', lambda_role)
```

Figure 9:Not picking up the constants files

Solution:

My module wasn't able to find any constants file from the resources folder, but it was right there. This error occurred due to multiple files with same names. So I changed the names and got rid of it.



```
adeelshahzadskipq:~/environment/Project1/AdeelWHMatrix (master) $ cdk synth
Traceback (most recent call last):
  File "app.py", line 9, in <module>
    app = cdk.App()
AttributeError: module 'aws_cdk' has no attribute 'App'
Subprocess exited with error 1
adeelshahzadskipq:~/environment/Project1/AdeelWHMatrix (master) $ cdk synth
Traceback (most recent call last):
  File "app.py", line 9, in <module>
    app = cdk.App()
AttributeError: module 'aws_cdk' has no attribute 'App'
Subprocess exited with error 1
adeelshahzadskipq:~/environment/Project1/AdeelWHMatrix (master) $
```

Figure 10:App.py file was not working fine

Solution:

Same as first error but occurred due to file name was given in a wrong way inside the stack file. Changed the way it was given and got rid of it

```
(.venv) adeelshahzadskipq:~/environment/MyDemoRepo (master) $ cdk deploy
MyDemoRepoStack: deploying...

✓ MyDemoRepoStack (no changes)

Stack ARN:
arn:aws:cloudformation:us-east-2:315997497220:stack/MyDemoRepoStack/34f30740-5d72-11ec-ba6e-02bc742e1cea
(.venv) adeelshahzadskipq:~/environment/MyDemoRepo (master) $
```

Figure 11:No changes gets detected

Solution:

Different functions started to get deployed without any change. And all the changes made to them were not detected so I figured that in my main machine another file with same name is present so changed the name and got it working fine.

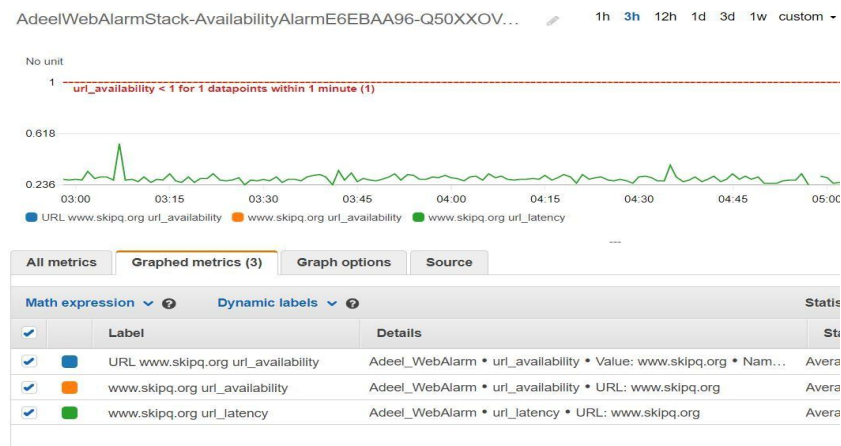


Figure 12:Alarm matrix not merging

Solution:

Both matrices failed to merge with each other all the latency and availability values from lambda file shown on one matrix and alarm and threshold from stack file shown in an other matrix. So to solve this error looked deeply into the code and found out that parameters for both matrices from lambda file and stack file are different given them same parameters and and run the code smoothly.

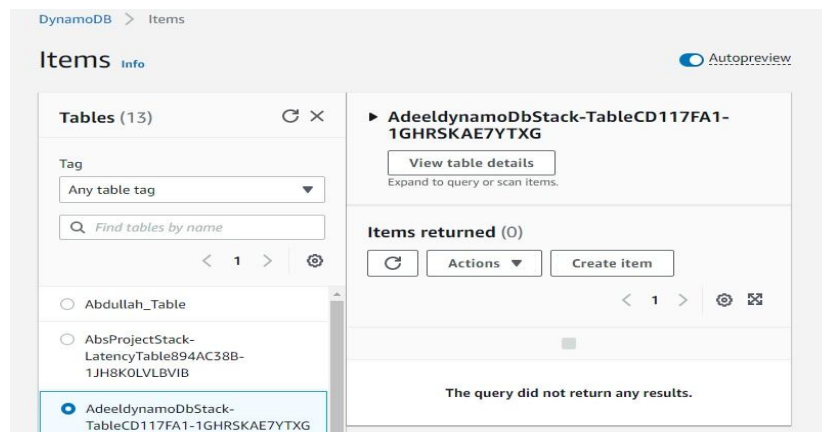
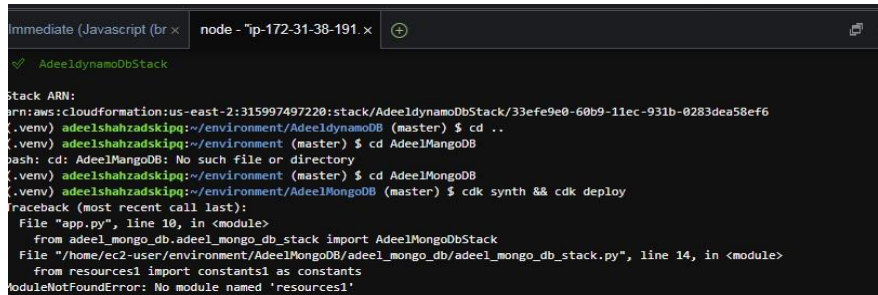


Figure 13:Table is not showing the values

Solution:

Table is created successfully but failed to get values. Get to now the index of message that was going to be printed in the table. Given it the right way and solved it.



```
node - "ip-172-31-38-191.x"
AdeelMongoDBStack
Stack ARN:
arn:aws:cloudformation:us-east-2:315997497220:stack/AdeelMongoDBStack/33efe9e0-60b9-11ec-931b-0283dea58ef6
(.venv) adeelshahzadskipq:~/environment/AdeelMongoDB (master) $ cd ..
(.venv) adeelshahzadskipq:~/environment (master) $ cd AdeelMongoDB
bash: cd: AdeelMongoDB: No such file or directory
(.venv) adeelshahzadskipq:~/environment (master) $ cd AdeelMongoDB
(.venv) adeelshahzadskipq:~/environment/AdeelMongoDB (master) $ cdk synth && cdk deploy
Traceback (most recent call last):
  File "app.py", line 10, in <module>
    from adeel_mongo_db.adeel_mongo_db_stack import AdeelMongoDBStack
  File "/home/ec2-user/environment/AdeelMongoDB/adeel_mongo_db/adeel_mongo_db_stack.py", line 14, in <module>
    from resources1 import constants1 as constants
ModuleNotFoundError: No module named 'resources1'
```

Figure 14:folder not found in directory

Solution:

Failed to find the resources file So changed the names to solve the issue. App file was not given any CDK library. Defined a proper library and done.

3. Sprint 2

3.1. Technologies Used

3.1.1. Pipelines

This walkthrough builds a pipeline for a sample WordPress site in a stack. The pipeline is separated into three stages. Each stage must contain at least one action, which is a task the pipeline performs on your artifacts (your input). A stage organizes actions in a pipeline. CodePipeline must complete all actions in a stage before the stage processes new artifacts, for example, if you submitted new input to rerun the pipeline.

- By the end of this walkthrough, you'll have a pipeline that performs the following workflow:
- The first stage of the pipeline retrieves a source artifact (an AWS CloudFormation template and its configuration files) from a repository.
- You'll prepare an artifact that includes a sample WordPress template and upload it to an S3 bucket.
- In the second stage, the pipeline creates a test stack and then waits for your approval.
- After you review the test stack, you can choose to continue with the original pipeline or create and submit another artifact to make changes. If you approve, this stage deletes the test stack, and then the pipeline continues to the next stage.
- In the third stage, the pipeline creates a change set against a production stack, and then waits for your approval.
- In your initial run, you won't have a production stack. The change set shows you all of the resources that AWS CloudFormation will create. If you approve, this stage executes the change set and builds your production stack.

3. 1. 2. CloudFormation

Developers can deploy and update compute, database, and many other resources in a simple, declarative style that abstracts away the complexity of specific resource APIs. AWS CloudFormation is designed to allow resource lifecycles to be managed repeatably, predictable, and safely, while allowing for automatic rollbacks, automated state management, and management of resources across accounts and regions. Recent enhancements and options allow for multiple ways to create resources, including using AWS CDK for coding in higher-level languages, importing existing resources, detecting configuration drift, and a new Registry that makes it easier to create custom types that inherit many core CloudFormation benefits.

3. 2. Pipeline Source & build (Task 1)

3. 2. 1. Implementation

First of all started with the basic implementation of pipeline. In this task tried to implement the first three stages of pipelines. First of all wrote the code for source stage in this stage all the important libraries get installed in the pipeline environment and behaviors are defined. Then comes the build stage in which code is picked up from the github repository and synth the code for any errors to build a exe file for the code so that code can be available at any time in forms of executable file. If there are problems or errors in the build, the process stops and issues are reported back to the developers for remediation. Typical problems include functional errors, such as a divide-by-zero math error, or missing components -- for example, a required library or module is not present in the build manifest.

We must first construct a pipeline source. GitHub will be a third-party source that we will integrate into our workflow. Build the source after you've added it. The source code is presented below. Add a repository, a branch, and the name of the secret where you've saved your personal access token to the repo string. Code Pipeline examines the source for changes on a regular basis using GitHub's "POLL" feature.

3. 2. 2. Results



Figure 15:Source state



Figure 16:Build State

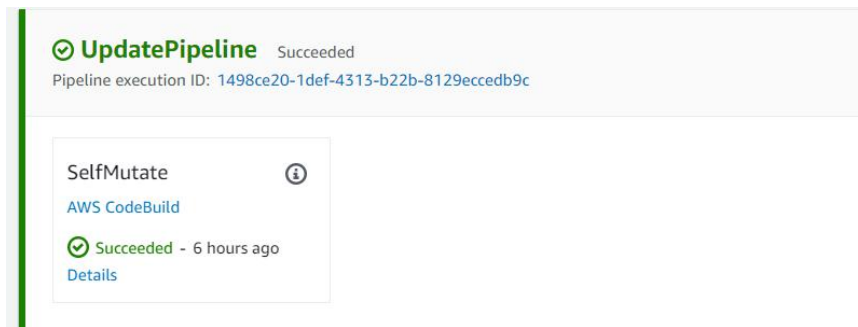


Figure 17:Update pipeline step

3. 3. Beta Stage(Task 2)

3. 3. 1. Implementation

We must now develop the Beta Stage. In addition, a unit test is built. The test is set to pre when adding the Beta stage. It signifies that if the test succeeds, the code will be deployed. Commit the code once again and upload it to GitHub. On Code Pipeline, you'll find these results.

3. 3. 2. Results

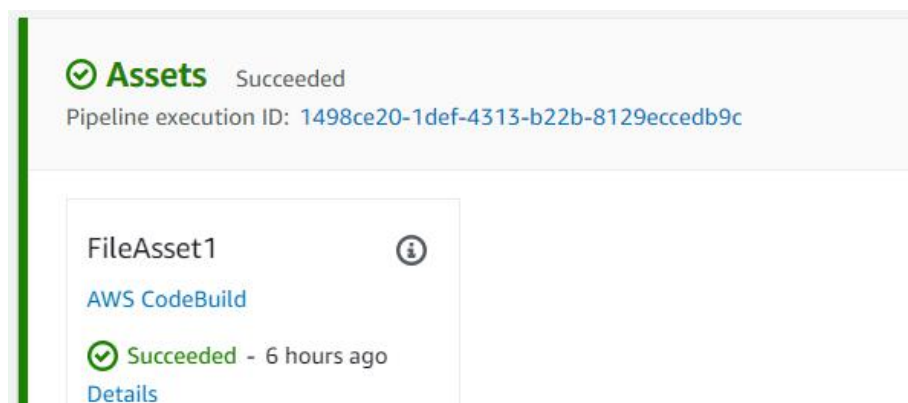


Figure 18:Assets update step



Figure 19:Beta stage

3. 4. Unit and integration tests(Task 3)

3. 4. 1. Implementation

We have deployed the code in two stages: beta and production. For both the Beta and Production stages, we collect Availability and Latency Metrics. The availability graphs for Beta and Prod for one URL are provided below. Metrics and alarms are built in the same way for both the Beta and Prod phases. Separate Dynamo DB tables are also generated. As a result, we update the dynamo Lambda code to reflect this. Alarm generates in the Pro or Beta level, according to Alarm. It saves the details of the alert in the relevant database. To reach these results we added different test in our stages so that a desired output acceptable by the client can be obtained on the screen.

3. 4. 2. Results



Figure 20:Unit test

3. 5. Production stage(Task 4)

3. 5. 1. Implementation

The final step in the Pipeline is to add production. We established a production stage and added it to the pipeline as a pre with manual permissions. It implies it will ask the user for approval before deploying the code to the server. The code for adding a production stage may be found here. Commit and send the code to the GitHub repository once more. These are the outcomes you'll receive if you use Code Pipeline.

3. 5. 2. Results

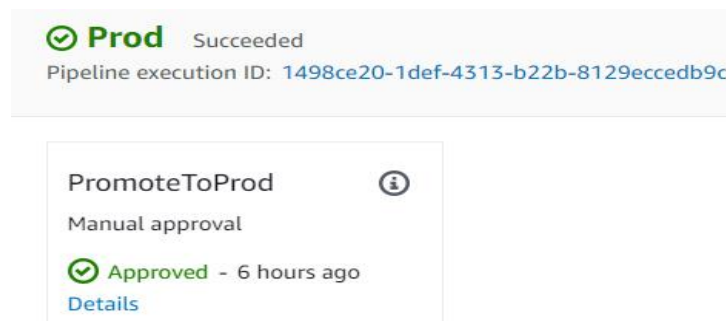


Figure 21:Production Step

3. 6. Roll Back(Task 5)

3. 6. 1. Implementation

Added a roll back method to the pipeline so that whenever an error appears it should roll back to previous state and avoid the new state. This thing is done for the sake of successful pipeline setup. In this task we added a simple roll back for time error. If our lambda function takes more time than the added threshold to update than we roll it back to previous version and send the traffic to new lambda function according to our desired limit.

3. 6. 2. Results



Figure 22:Alias

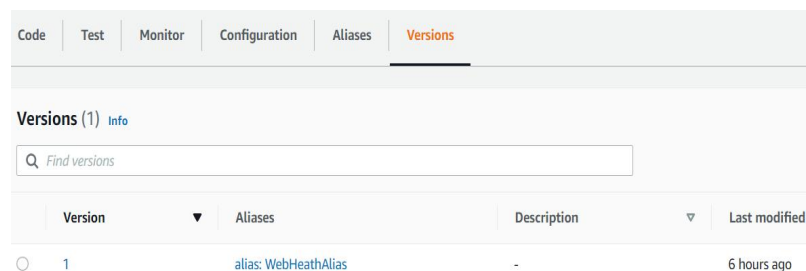


Figure 23:Versions

3. 7. Errors & Solution

```
error: failed to push some refs to 'https://github.com/muhammadskipq2021/ProximaCentauri.git'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Figure 24:Token stopped working

Solution:

Git hub started to reject my supplied token key . I checked and found out that it was expired, I renewed and got the process working.

```
raise JSIIError(resp.error) from JavaScriptError(resp.stack)
jsii.errors.JSIIError: Object of type @aws-cdk/core.Stack is not convertible to @aws-cdk/core.Stage
```

Figure 25:Not controvertible to stage

Solution:

Given wrong name to my stage file. So changed the name form stack to stage and that error got resolved.

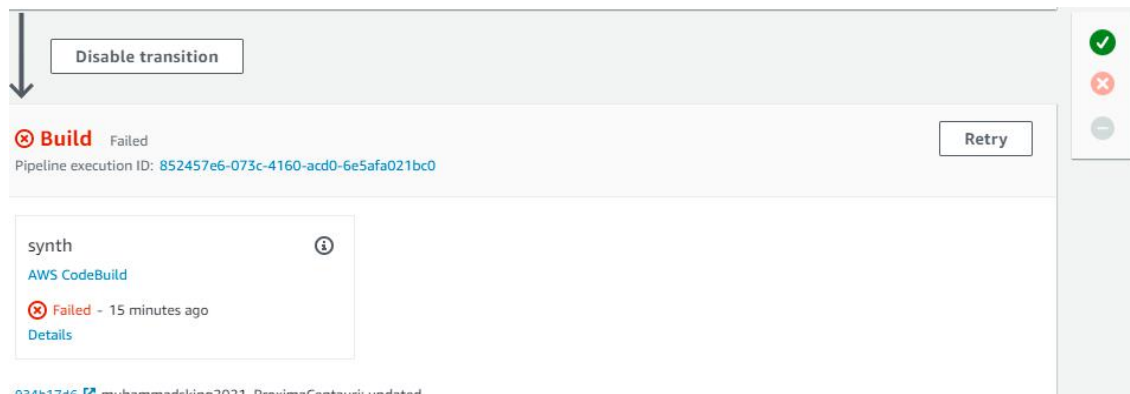


Figure 26:Build stage failed

Solution:

My build state started to get failed. I looked into the event logs and found out that it was an error related to wrong names specification in the code. So changed the names and got this error resolved.

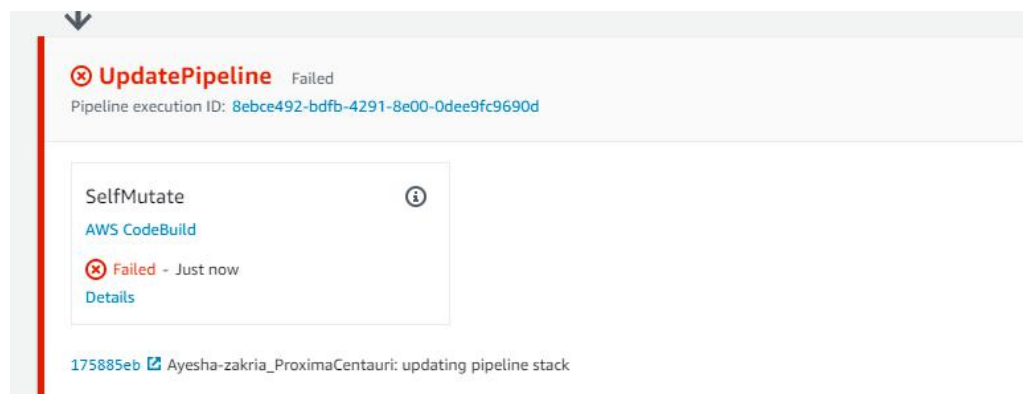


Figure 27:Update step failed

Solution:

Update pipeline started to give the same error. Resolved it same way.


```

    inst = super().__call__(*args, **kwargs)
File "/codebuild/output/src388266275/src/Adeel/Sprint2/AdeeldynamoDB/adeeldynamo_db/adeeldynamo_db_stack.py", line
__init__
    Url_Monitor = bo().bucket_as_list()
File "/codebuild/output/src388266275/src/Adeel/Sprint2/AdeeldynamoDB/resources1/bucket.py", line 7, in __init__
    self.Object = boto3.client('s3').get_object(Bucket='adeelskipq',Key='urls.json')
File "/root/.pyenv/versions/3.9.5/lib/python3.9/site-packages/botocore/client.py", line 388, in _api_call
    return self._make_api_call(operation_name, kwargs)
File "/root/.pyenv/versions/3.9.5/lib/python3.9/site-packages/botocore/client.py", line 708, in _make_api_call
    raise error_class(parsed_response, operation_name)
botocore.exceptions.ClientError: An error occurred (AccessDenied) when calling the GetObject operation: Access Denied
Subprocess exited with error 1
[Container] 2021/12/23 05:52:24 Command did not exit successfully cdk synth with status 1

```

Figure 28: Acces denied to Some functions

Solution:

It was the error related to access of different objects in pipeline environment. Added some policies in it to get access and run the code smoothly.

```

}
--app points to a cloud assembly, so we bypass synth
No stacks match the name(s) AdeelPipelineStack2
Error: No stacks match the name(s) AdeelPipelineStack2
    at CdkToolkit.validateStacksSelected (/usr/local/lib/node_modules/aws-cdk/lib/cdk-toolkit.ts:545:13)
    at CdkToolkit.selectStacksForDeploy (/usr/local/lib/node_modules/aws-cdk/lib/cdk-toolkit.ts:492:10)
    at CdkToolkit.deploy (/usr/local/lib/node_modules/aws-cdk/lib/cdk-toolkit.ts:120:20)
    at initCommandLine (/usr/local/lib/node_modules/aws-cdk/bin/cdk.ts:267:9)

[Container] 2021/12/23 05:52:24 Command did not exit successfully cdk -a . deploy AdeelPipelineStack2 --require-
approval=never --verbose exit status 1
[Container] 2021/12/23 05:52:24 Phase complete: BUILD State: FAILED
[Container] 2021/12/23 05:52:24 Phase context status code: COMMAND_EXECUTION_ERROR Message: Error while executing
cdk -a . deploy AdeelPipelineStack2 --require-approval=never --verbose. Reason: exit status 1

```

Figure 29:No Module name found

Solution:

Pipeline failed to detect my stack file. It was the error related to git hub push. I have pushed the latest changes to github and got it working.

```

AdeeldynamoDBStack(app, AdeeldynamoDBStack ,
File "/home/ec2-user/environment/AdeeldynamoDB/.venv/lib64/python3.7/site-packages/jsii/runtime.py", line 86, in __c
inst = super().__call__(*args, **kwargs)
File "/home/ec2-user/environment/AdeeldynamoDB/adeeldynamo_db/adeeldynamo_db_stack.py", line 46, in __init__
    Url_Monitor = bo('Adeelskipq').bucket_as_list('urls.json')
File "/home/ec2-user/environment/AdeeldynamoDB/resources1/bucket.py", line 10, in bucket_as_list
    data = Object['body']
TypeError: 's3.Object' object is not subscriptable
Subprocess exited with error 1
(.venv) adeelshahzadskipq:~/environment/AdeeldynamoDB (master) $

```

Figure 30:Not able to subscribe to S3

Solution:

S3 bucket in which I have stored my links started to give errors. It was due to permission setting. So changed the permission setting and code started working.

```

Traceback (most recent call last):
  File "app.py", line 31, in <module>
    PipelineStack(app,'AdeelPipelineStack',env = core.environment(account='315997497220',r
AttributeError: module 'aws_cdk.core' has no attribute 'environment'
Subprocess exited with error 1
adeelshahzadskipq:~/environment/ProximaCentauri/Adeel/Sprint2/AdeeldynamoDB (main) $

```

Figure 31:environment error

Solution:

Environment started to give errors because different stages had different env to run. Deployed them in same env and solved the error.