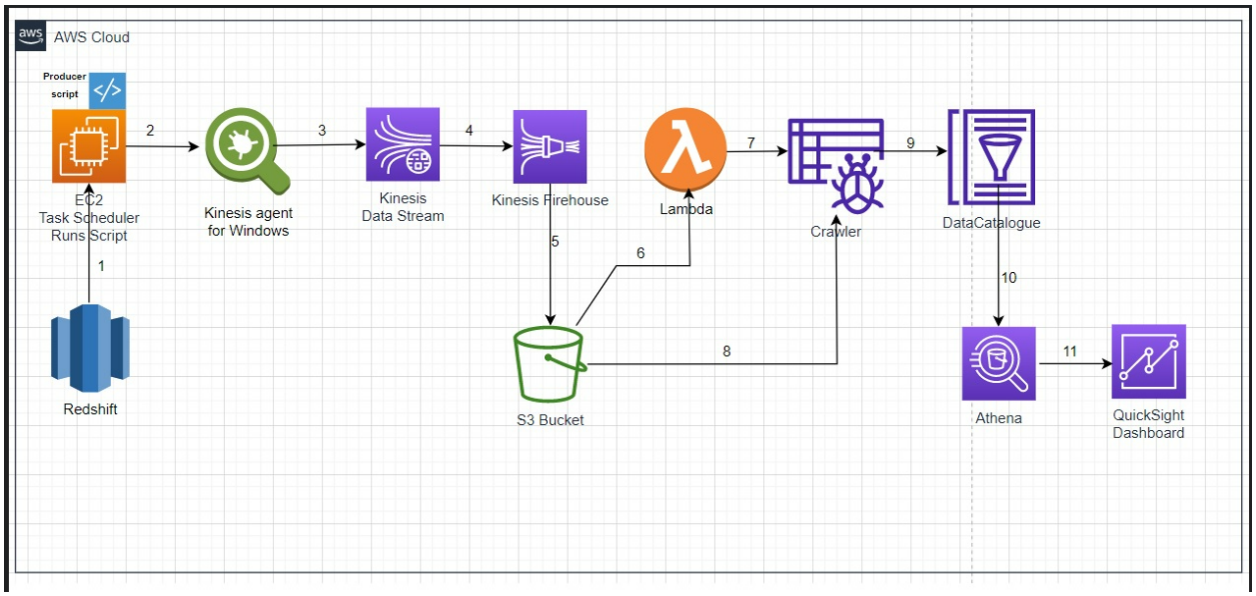
**LIVE DATA STREAM WITH KINESIS FROM REDSHIFT AND DELIVERY TO QUICKSIGHT DASHBOARD**

**Warning:** This project will pick up where Project 3 ended - Redshift table. To successfully complete this project you must have finished Project 3. This project is using EC2, Security Group, some IAM Roles, Redshift and S3 buckets that were created in Project 3.
Addition to 'enriched' glue job from Project 3 will be performed to produce a trigger file for the next pipeline to start once pipeline from Project 3 completes.

**Purpose**: Design an aws cloud based data pipeline that will read a table from redshift database, simulate streaming data delivery into S3 bucket, from where crawlers will read the data and create metadata table called - catalog, and finally quicksight dashboard will pick up streaming data from data catalog though direct athena queries when an update is performed.

Take a look at the following Diagram to get a high level glimpse into the expected work awaiting us:




**Prerequisites**: AWS Account, Installed SQL Workbench, Python, Basic Network Configuration, json, SQL, RDC, Successful pipeline from Project 3, downloaded files from folder 'Project 4' in GitHub link provided: https://github.com/Myself1214/Upwork.git

**Plan of Work (Pseudo Work)**:

1. Implement additional code to ‘Enriched’ glue job from Project 3 to produce trigger file in S3 for this pipeline to start
2. Assign your EC2 a ‘key:value’ pair tag for Systems Manager to find it when running remote commands on it
3. Program a data producer and load it on EC2 through a shared path that is mounted from local to EC2.
4. Create a role for producer to write data on Kinesis Data Stream and attach it to EC2
5. Create a folder in your S3 raw bucket and name it ‘from-firehouse’
6. Create Kinesis Data Stream
7. Create Kinesis Firehose
8. Install and configure Kinesis Agent for Windows on your EC2 to stream data from EC2 server into kinesis data stream and then to firehose delivery stream which will deliver data to S3 bucket in ‘from-firehouse’ folder. Run the agent
9. Create crawler to crawl streaming data from raw S3 bucket and create table metadata as a catalog
10. Create Lambda function to trigger crawler every time when new data is streamed and loaded in S3 raw bucket
11. Manually run the pipeline to produce data for next step - Dashboard
12. Create Quicksight dashboard to be updated through direct athena queries
13. Create an event based EventBridge rule to trigger execution of ‘producer.py’ file that in turn will trigger the entire data pipeline.

**Actual Steps:**

1. Add new code to ‘enriched’ glue job from Project 3 for trigger file
   - From github link provided, under ‘Project 4’ download ‘producing_trgger_file.py’ and add it to existing script of ‘enriched’ glue job from previous project at the end of code
   - Log in to the S3 console, inside the ‘Enriched’ bucket from Project 3 create folder ‘for_trigger_file’ by selecting the ‘Create folder’ button on the top right section, then giving it the mentioned name.
2. Switch to EC2 console, select your EC2 created during project 3, under ‘Actions’ select ‘Instance settings’ and choose ‘Manage tags’. Click on ‘Add new tag’, enter a name for ‘key’ and ‘value’ fields and click on save.
3. Get producer and load it onto EC2 mounted shared path from local

- From the github link provided, under ‘Project 4’ download file ‘producer.py’, edit it and add your redshift cluster connection strings where indicated and table name. Since you already came here after finishing Project 3, you already have a Redshift cluster with a loaded table - ‘911 calls’ file in it. (If you have not gone through Project 3, you will not be able to continue). Once your edited producer, save it and copy it into path in your local machine that is mounted on your EC2 Windows instance that you created in Project 3 and it will be accessible on your remote server

4. Create a role for producer to write data on Kinesis Data Stream and attach it to EC2
   - From github link provided, under ‘Project 4’ download file ‘ec2_kinesis_role_policy.json’
   - Log in to the IAM console on aws, select ‘Roles’ on the left pane. Click on ‘Create role’. Under common use cases select ‘EC2’ and on the next screen select ‘Create policy’. Now switch to the ‘JSON’ tab and paste the code from the file downloaded from github and click next twice. On review page give your role name ‘kinesis_role’ and click on ‘Create policy’
   - Switch to EC2 console, select your windows instance created in Project 3, and from right top select ‘Actions’. From the list select ‘Security’ and then ‘Modify IAM role’. In new screen from drop-down list select ‘kinesis_role’ created earlier and click on ‘Update role’
5. Create a folder in your S3 raw bucket and name it ‘from-firehouse’
   - Switch to the S3 console, select the ‘Raw’ bucket created in Project 3, then click on ‘Create folder’. Give it a name ‘from-firehouse/’ and create.
6. Create Kinesis Data Stream
   - Switch to Kinesis console, on the left pane select ‘Data Streams’, then ‘Create data stream’. Give it a name of “MyFirstDataStream’, scroll down and click on ‘create data stream’
7. Create Kinesis Firehose
   - On the left pane select ‘Delivery Streams’, then click on ‘create delivery stream’. For the ‘Source’ select ‘Amazon Kinesis Data Stream’ and for ‘Destination’ select ‘Amazon S3’
   - For ‘Source settings’ select the data stream created in step 5, then give delivery steam a name of your choice.
   - For ‘Destination Settings’ select ‘Raw’ S3 bucket and for ‘S3 bucket prefix’ put name of folder ‘from-firehouse/’ created in step 4 and click on ‘Create delivery steam’.
8. Install and configure Kinesis Agent for Windows on your EC2

- To install kinesis agent for ms windows on your EC2, go to: https://s3-us-west-2.amazonaws.com/kinesis-agent-windows/downloads/index.html, and follow instructions for 'Install using PowerShell'
- Now we need to create configuration file for Kinesis Windows Agent that will help it find necessary file to read from - 'Source', and target destination where it should stream data to - 'Sink'
- Once installation is done, open the notepad and create a Kinesis Agent for Windows configuration file called 'appsettings.txt'. Inside this file copy and paste content of file 'appsettings.txt' that you'll download from the github link provided under Project 4. Replace following values in your 'appsettings.txt' file with your own:
    - 'id' - pick any id name
    - 'Directory' - path to source file from which kinesis will read data - this is same path where 'proucer.py' file will write 'json_file.txt' file in
    - 'SourceRef' - id name that you assigned to 'Sources'
    - 'SinkRef' - id name that you assigned for 'Sink'
    - 'StreamName' - name of your Data Stream created in step 5
    - 'AccessKey - your account access key
    - 'SecretKey' your account secret key
    - 'Region' - you account region
- Once you are done with all changes to the 'appsettings.txt' file, save it in the following directory: *C:\Program Files\Amazon\AWSKinesisTap* . If such a file with the same name already exists in the above directory, replace it with yours.
- Now we need to start Kinesis agent by executing following command in PowerShell: *Start-Service -Name AWSKinesisTap*
(To stop Kinesis agent, use this command: *Stop-Service -Name AWSKinesisTap*)
Now the agent will be continuously running in the background and waiting for the 'json_file.txt' file in the directory that we specified in the 'appsettings.txt' file. The file 'json_file.txt' will be produced by our 'producer.py' file created in step 2.

9. So far we have pipeline set up to read data from Redshift and produce file 'json_file.txt' content of which then gets streamed to Kinesis data stream by Kinesis Agent and then from Data stream picked by firehose and loaded to S3 bucket into 'from-firehose/' folder. Now we need to create a glue crawler that will crawl data in 'from-firehouse/' folder and create a data catalog that could be queried by AWS Athena service. Later we will use QuickSight dashboard that will update by running Athena queries against data in 'from-firehose/' folder through the data catalog created by crawler
- Switch to Glue console, on the left pane select 'Crawlers', select 'Create crawler'. Give it a name and click 'Next', on the following page select 'add a data source'.

For data source select ‘S3’ and below put a checkmark on ‘In this account’, then “Browse" to location of ‘from-firehose/’ folder in your raw S3 bucket, click ‘add data source’ and click ‘Next’.
- On the security settings page, choose ‘Create New IAM role’, give it a name extension and click ‘Next’.
- On this page select ‘add database’ and on the newly popped window give it a name and click ‘Create database’ and click ‘Next’ again. Review your entries and click on ‘create crawler’
- Now we need to give the IAM role of our crawler additional permissions to be able to crawl ‘from-firehose/’ folder in the raw bucket. To do that switch to IAM console and select ‘Roles’ on the left pane, then from the list find the IAM role you created for crawler and click on ‘add permissions’ then ‘attach policies’. Then select ‘create policy’, switch to ‘JSON’ tab, copy and paste content of file ‘crawler_role_policy.json’ then click ‘Next’ twice, give the policy a name, and click on ‘Create policy’. Go back to the crawler role page and add your newly created policy to it.

10. Create Lambda Function as a trigger for crawler
- Switch to Lambda console, click on ‘create function’, give it a name, for runtime select ‘Python 3.9’, let it create basic lambda role, then click on ‘create function’.
- Once function is created, under the ‘Code’ tab delete all existing code, then copy and paste content of file ‘lambda_code.py’ downloaded from the github link provided, remember to change ‘your_crowler_name’ to the name of your crawler created in step 8. This code will first check crawler status, and only if it is in ‘Ready’ state will start the crawler
- Since this lambda will server as crawler trigger that is based on event - objects created in S3 raw bucket (from-firehose/ folder), it will need permissions to interact with crawler and S3 bucket, thus we need to attach to lambda this policy: ‘AWSGLueConsoleFullAccess’ which will provide both permissions. To do that switch to the ‘Configuration’ tab and select the ‘Permissions’ section. Here click on the role name under ‘Execution role’ and it will open the default role created for this lambda. Here, click on ‘add permissions’, then ‘attach policies’, in search field type ‘glue’ and from filtered list select ‘AWSGLueConsoleFullAccess’ and select ‘attach policies’ in the bottom
- To trigger this lambda based on S3 object created events switch back to the lambda console, select ‘add trigger’ then choose ‘S3’ from the list. Under ‘Bucket’ select Raw S3 bucket where Kinesis Delivery stream loads data to, for event type
- ‘All object create events’, and for prefix add ‘from-firehose/’ and click ‘Add’ in the bottom

- Now under the ‘Code’ tab click on ‘Test’, give it a name under ‘Event name’ and click ‘Save’. Now our lambda is ready to go

11. Now we will manually run the pipeline so that it produces data in our ‘from-firehose/’ folder in the raw S3 bucket by reading ‘911_calls’ data from Redshift for our dashboard to use in the next step. To do this, we need to log on to the EC2 instance and manually execute the ‘producer.py’ script. You can do this by double clicking on the producer file or running this file from your terminal window by switching to the directory where the file resides and typing ‘py producer.py’.
12. Create Quicksight dashboard
    - To create a Quicksight dashboard, you first need to sign up for Amazon QuickSight Subscription. Follow instructions on this page to sign up: https://docs.aws.amazon.com/quicksight/latest/user/signing-up.html. For the sake of tutorial I choose ‘Standard’ subscription
    - Once you have subscribed log on to the main page of QuickSight, select “Datasets", click on ‘New dataset’, select ‘Athena’ and give it a name under ‘data source name’ then click ‘Create data source’. On the next page under ‘Database’ list select database you created in step 8 when creating a crawler and check mark its table in the list below and click ‘select’. On the next page put a checkmark on ‘Direct query your data’ and click on ‘Visualize’. Now dataset is created and we are on Analysis phase now
    - Here we’ll create a sample visualization for our dashboard. Under ‘visual types’ select ‘Filled map’ visual, then on top select ‘Field wells’ tab. Now under ‘Fields list’ scroll down and select ‘zop_code’ field and drag it under ‘location’ area of ‘Field wells’ section. Now you can see the map of all zip codes in Detroit. You may create other visuals as you wish but for our tutorial this sample is enough.
    - On the top right corner select ‘Share’ icon and then ‘Publish dashboard’ then give it a name and click on publish dashboard. Now if you switch to the dashboard page you will see your created dashboard.
    *Note: For free tier automatic dashboard updates are not available, but once pipeline is done you may update your dashboard by just reloading the browser page*
13. Create an event based EventBridge rule to trigger execution of ‘producer.py’ file
    - Switch to EventBridge console, on the left pane select “Rules’ and click on ‘Create rule’. Give it a name then scroll down and click ‘Next’. For ‘Event Source’ select ‘Other’, and for ‘Creation method’ select ‘Custom pattern JSON editor’ and in the field ‘Event pattern’ copy and paste content of file ‘event_bridge_rule_policy.json’ and don’t forget to change bucket name to your S3 enriched bucket name. This json string creates an event based rule based on

any file created in our enriched bucket under the ‘for_trigger_file’ folder. When done, click ‘Next’. On this page, for ‘Target’ select ‘AWS Service’, for ‘select a target’ choose ‘System Manager Run Command’, for ‘Document’ select ‘AWS-RunPowrShellScript’, for ‘Target key’ enter ‘tag:<your instance tag key’ created in step 2 and for ‘Target value/s’ enter instance’s tag value. Then, under ‘Configure Automation parameters’ select ‘Constant’ and in the field ‘Commands’ copy and paste your python executable’s full path, space, and full path of ‘producer.py’ file. (To get python executabe’s full path run the following command in your PowerShell terminal: ‘(gcm py).source’). For ‘Executions role’ leave default selection, scroll down and click on ‘Next’, then ‘Next’ again, review your entries and click on ‘Create rule’.

- Once your rule is created, it automatically creates the default execution role, which you can check in the rule's settings. Open this rule in IAM console and attach additional following two policies:
    - AmazonEC2RoleforSSM
    - AmazonSSMManagedInstanceCore
- Now, the EventBridge rule is created with proper role, we need to enable EventBridge Notifications on S3 enriched bucket’s properties to allow it to send ‘Objects created’ notifications to our EventBridge rule. To do this, switch to S3 Console, select enriched bucket, under ‘Properties’ tab navigate to ‘AmazonEventBridge’ sections and enable it though ‘Edit’ button.
- Now since our EventBridge rule will, once triggered, run remote command (executing ‘producer.py’ script) on an EC2 instance through AWS Systems Manager, we need to ensure that our instance has an SSM agent (needed for Systems Manager to communicate with Instance) installed and running. Nowadays, most AWS Instances come with SSM Agent preinstalled. You can check it by running the following command: *Get-Service AmazonSSMAgent.* In case if it is not installed or not running, follow instruction on following page to install and/or run it: https://docs.aws.amazon.com/systems-manager/latest/userguide/ami-preinstalled-agent.html

Now, we are done with building our pipeline. You can test it by re-running your pipeline from ‘Project 3’, which, when finished, will produce a trigger file for this pipeline to start. Steps that will happen:

- EventBridge rule will pick up the trigger file in enriched bucket and with help of AWS Systems Manager will execute ‘producer.py’ file remotely in EC2 instance.
- Once ‘producer.py’ file runs it will read data from Redshift table ‘911_Calls’ and produce ‘json_file.json’ in specified path in EC2.

- Kinesis agent will pick up updates in ‘json_file.json’ file and will keep streaming it to Kinesis Data Stream.
- Kinesis Data Stream then will receive and pass on data to Kinesis Firehose Delivery stream
- Kinesis Firehose Delivery Stream will then deliver data to ‘from/firehose’ folder in S3 row bucket
- Every time Firehose delivers data to S3 raw bucket Lambda function will run and start Crawler to read data in ‘/from-firehose’ folder and create Glue Data Catalogue
- Every time when we update our Quicksight Dashboard it will send new Athena query against data stored in ‘/from-firehose’ folder through Glue Data Catalogue created/updated by Glue Crawlers every time new data is streamed

If you enjoyed this process and love gaining new skills and knowledge please check out my other projects under ‘Projects’ tab.