Project Name: INJOY – An Event Management System

User story / Tasks Name: INJOY – EVENT MANAGEMENT SYSTEM

Summary: An event management system has been created where admins can sign in and create events, update and manage them. Users can come and register for these events after they sign-in/sign-up to the system. AWS services such as Cognito, API Gateway, Secrets manager, RDS, DynamoDB, S3, Lambda, Step functions, EC2, SQS, SNS, X-RAY have been integrated to develop this system. Python programming language is incorporated throughout this project along with a basic web page using HTML and CSS, to display event images in S3. MySQL Workbench is integrated to create the RDS, and Postman is used to deploy the APIs.

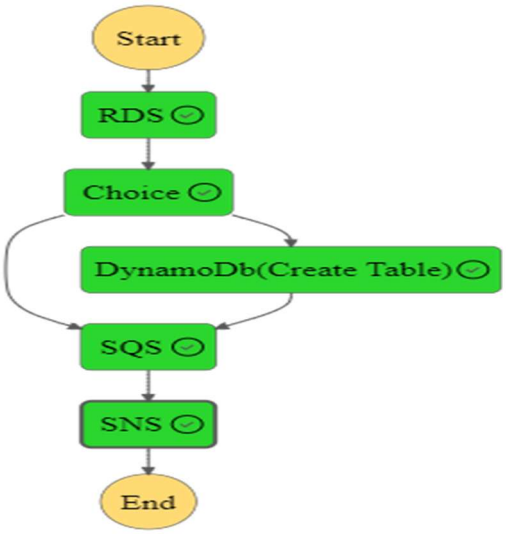
Work Performed

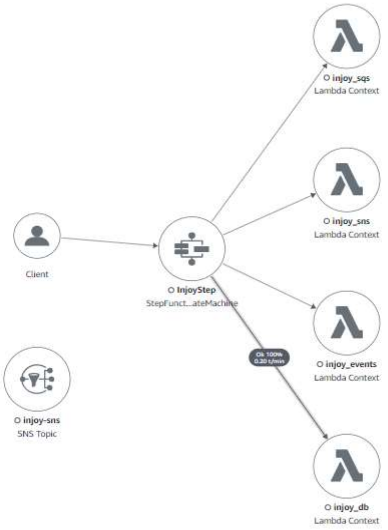
1. Create Event Step Function

- 'InjoyStep' step function has been created integrating 4 lambda functions
- The 4 lambdas created are:
    i. **Create RDS event –** An Event is added into the RDS database (database-2) with the necessary event details.
    ii. **Create Event Tables in DynamoDB** – A table is created in DynamoDB after verification with the EventID as its name to store registering user details every time an event is created, by lambda.
    iii. **Create Event SQS** – A lambda function to add the event status to SQS queue is integrated with 'injoy-sqs'
    iv. **Create Event Status SNS** – Event status message is retrieved from queue and a mail is sent to subscribed users integrated with 'injoy-sns' topic.
- An API Gateway 'StepAPI' has been created and integrated with the step function and is deployed in POST method using Postman.
- An X-RAY has been created to show the step function.

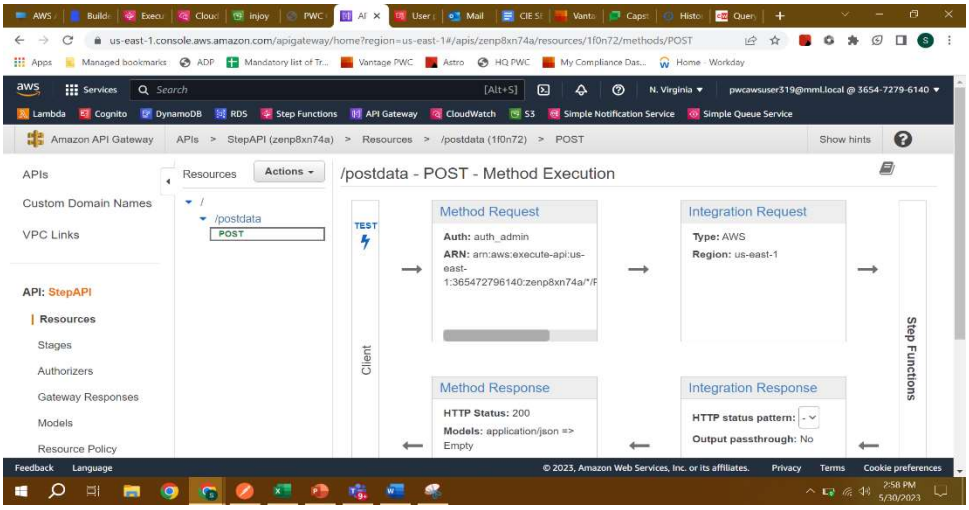
Step Function execution

Step Function X-Ray

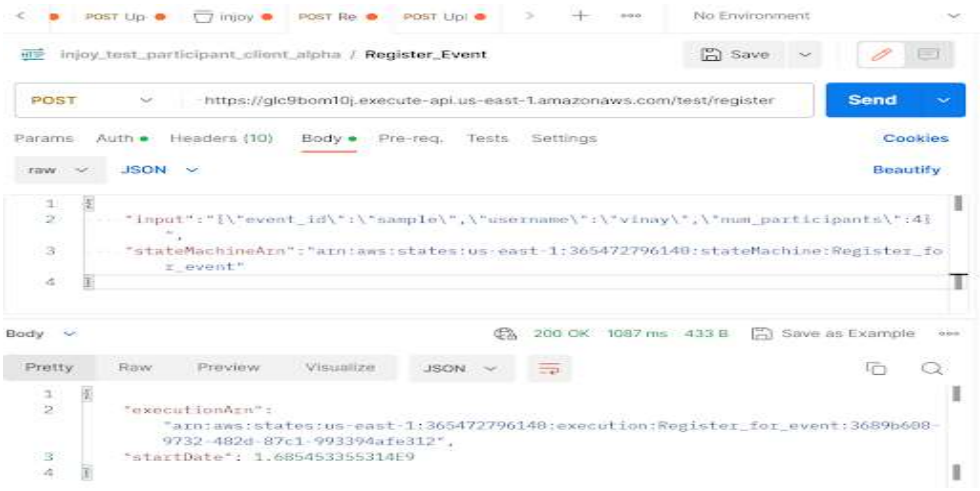




StepAPI

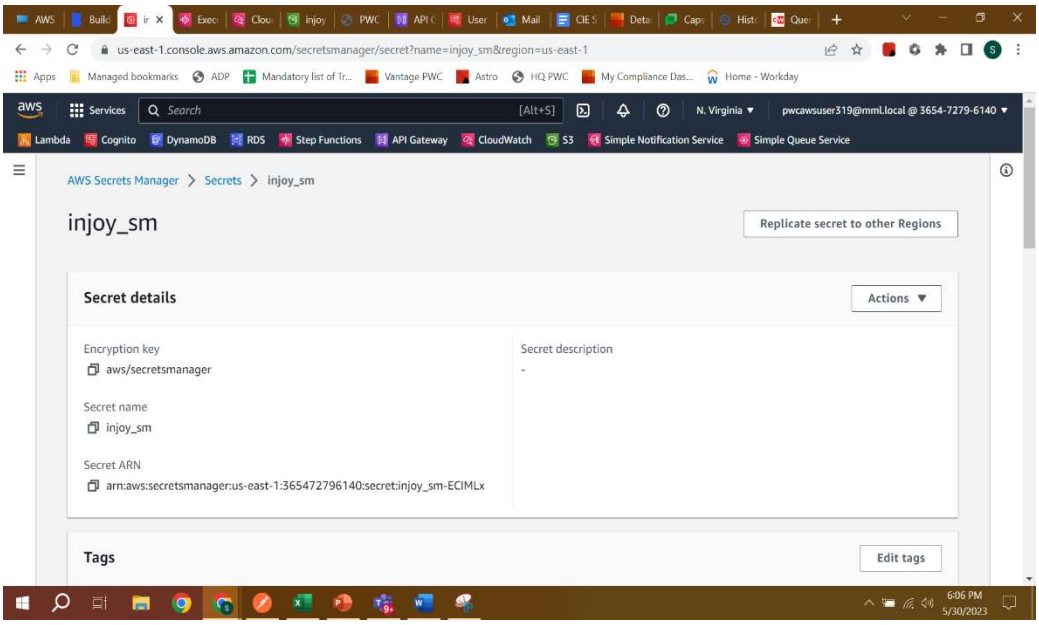


StepAPI deployed in postman



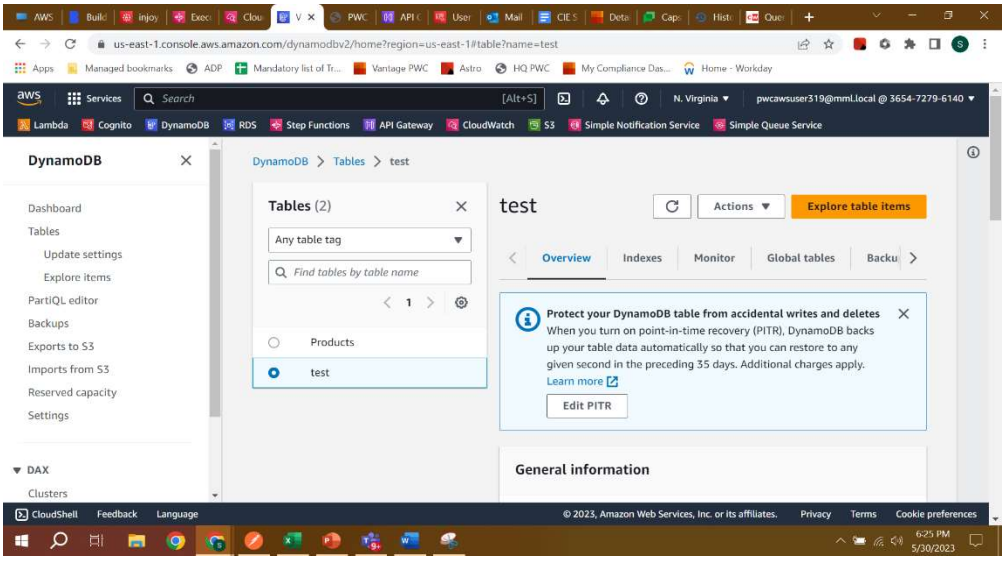
2. Secrets Manager

- Created a secret in secrets manager ‘injoy-sm’ to store the RDS database credentials and securely access the data.
- Implemented code in lambdas which connect RDS to fetch credentials via secrets manager.



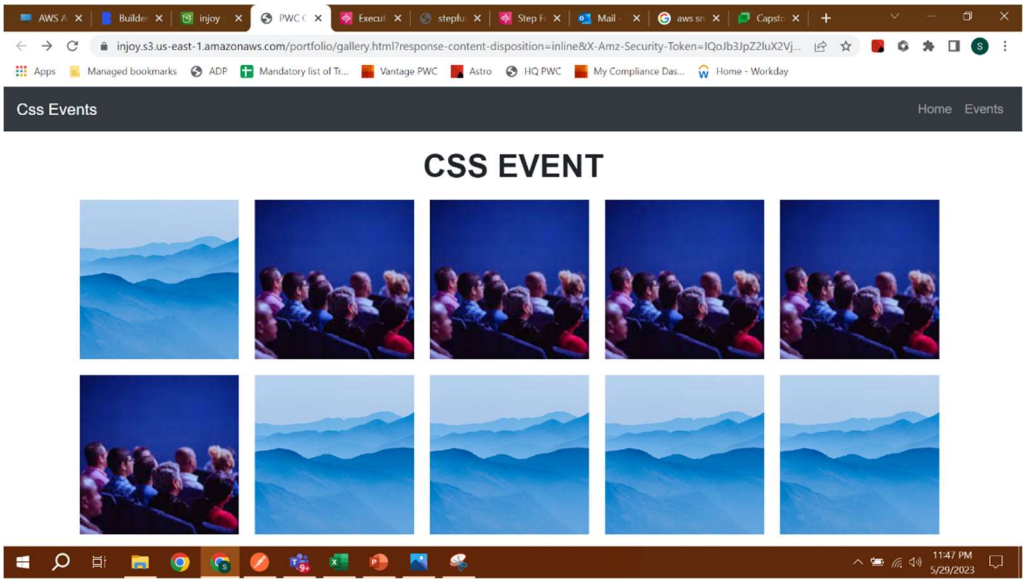
3. DynamoDB Event Tables created

- Created lambda function to retrieve eventId and create a table with eventId as its name every time an event is created.
- These tables created for each event, by the lambda function, is to store user details who register to an event.
  Table ‘test’ created with lambda (eventId-‘test’)



4. S3 Web page

- Worked on creating a portfolio page in ‘injoy’ s3 bucket to display all the event images inside the images folder in that bucket



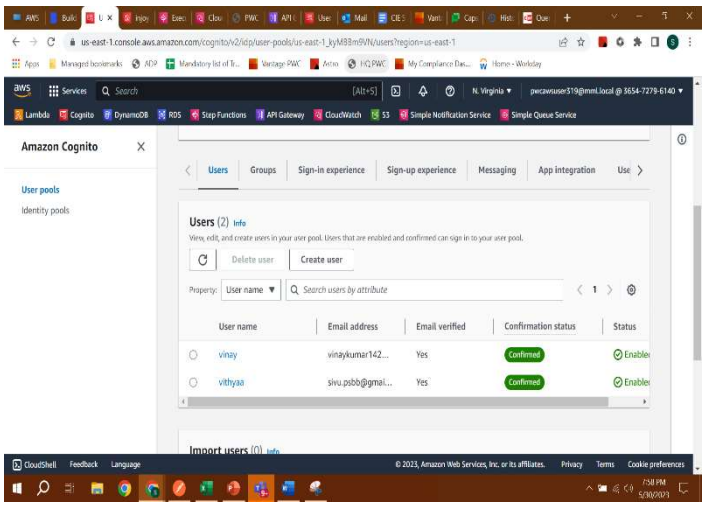
5. SQS and SNS Event Status message

- Worked on lambda functions to push event message into an SQS Queue and retrieve the message from queue and send a mail via SNS.

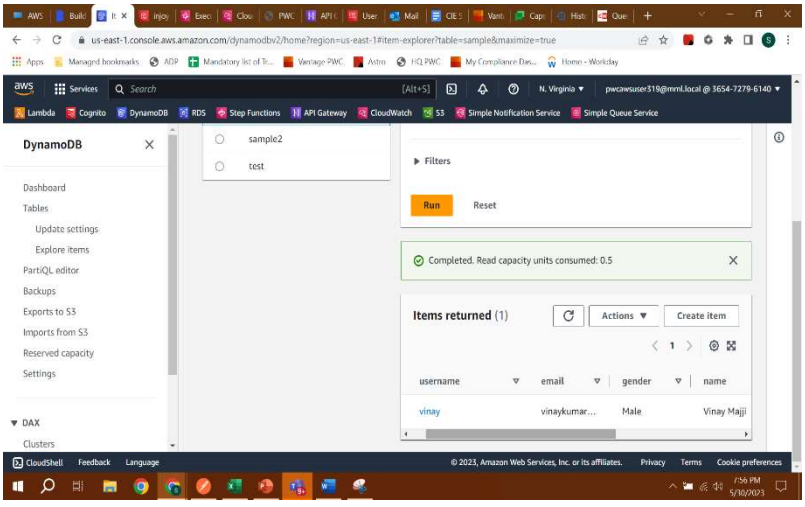
6. Cognito Integration

- User details from Cognito are retrieved in lambda function along with other details fetched from the registering user and the user is added into the DynamoDB event table which is created with lambda.

Signed in Users in ‘injoy_userpool’



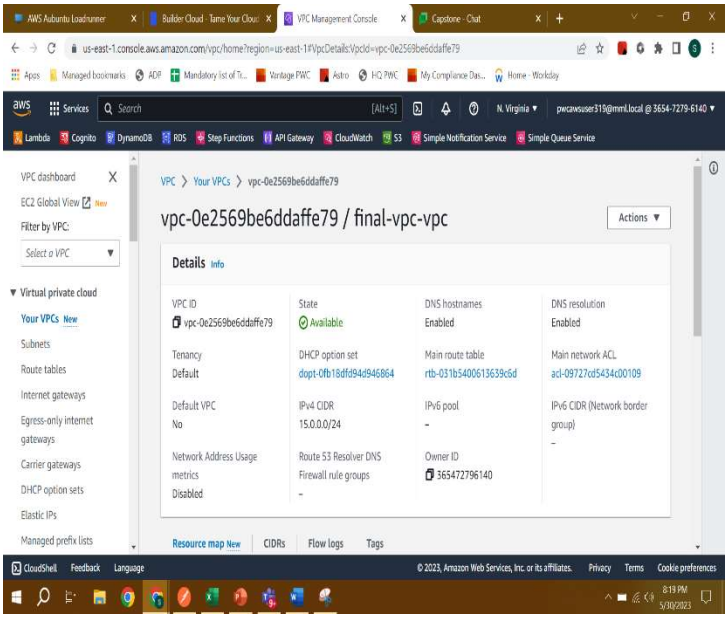
Details of user (vinay) fetched from Cognito into ‘sample2’ event table using lambda



7. Private RDS

- A VPC ‘final-vpc’ has been created within which are multiple public and private subnets.
- This VPC is integrated with an EC2 instance.
- Subnet groups are created for the RDS database and integrated with the VPC.
- Endpoint is created to access the private subnet via public subnets in the VPC
- This is done to enable access to the data within the ‘database-2’ private database within the VPC.

VPC ‘final-vpc’



private rds ‘database-2’