AWS Assignment 3 - Detailed

Below is an assignment that integrates Step Functions, CloudWatch, SES, EventBridge Scheduler, EventBridge, SNS, and SQS into a cohesive serverless order processing pipeline. This scenario will give hands-on experience with orchestrating workflows, event-driven architectures, and monitoring/logging across multiple AWS services.

Assignment: Build a Serverless Order Processing Pipeline

Objective

Develop an end-to-end serverless application that simulates order processing. When a new order is received, it will be placed into an SQS queue, processed via a Step Functions workflow (with multiple Lambda tasks), and monitored with CloudWatch. The solution should also include:

- Sending order confirmation emails with SES.
- Triggering events using EventBridge (both rules and scheduled tasks).
- Notifying stakeholders using SNS.

Assignment Tasks

1. SQS – Order Queue

- **Create an SQS Queue:**
  - Name: `OrderQueue`
  - Configure attributes such as Visibility Timeout and (optional) Redrive Policy for failed messages.
- **Test the Queue:**
  - Write a simple script or use the AWS CLI to send test order messages in JSON format (e.g., `{ "orderId": "12345", "customerEmail": "customer@example.com", "items": [...], "total": 99.99 }`).

---

2. Step Functions – Workflow Orchestration

- **Design a State Machine (`OrderProcessingStateMachine`)** that orchestrates the order processing workflow. The workflow should include:
    - **Order Validation:** A Lambda function to check the order data structure.
    - **Inventory Check:** A Lambda function that simulates inventory verification.
    - **Payment Processing:** A Lambda function that simulates payment processing.
    - **Notification:** A task that sends an order confirmation via SES.
- **Error Handling:** Include retry logic and error states. (For bonus points, route failed orders to a dead-letter queue.)
- **Triggering the Workflow:** Configure a Lambda function (or use an EventBridge rule) to poll the SQS queue and start a new Step Functions execution with the order details.

---

3. Lambda Functions

Develop and deploy Lambda functions for each workflow step:

- **Order Validation Function:**
    - Validate required fields (e.g., `orderId`, `customerEmail`, etc.).
    - Log details to CloudWatch.
- **Inventory Check Function:**
    - Simulate inventory verification (return success or trigger an error if inventory is low).
    - Log outcomes to CloudWatch.
- **Payment Processing Function:**
    - Simulate payment processing (you can assume a successful transaction for this assignment).
    - Log the payment status.
- **Notification Function:**
    - Prepare to trigger SES (see next task) or directly integrate SES API call to send an email.
    - Log email sending status.
- **Note:** Ensure each function's IAM role has permissions for its respective operations (e.g., SQS access, Step Functions execution, SES sending).

---

4. SES – Email Confirmation

- **Set Up SES:**
    - Verify a sender email address.
    - If needed, verify the recipient email address (depending on your SES sandbox settings).

- **Integrate with Lambda/Step Functions:**
    - In the Notification task of your workflow, use SES to send an order confirmation email containing details like order ID and order summary.
- **Test the Email Functionality:**
    - Verify that emails are received when an order is processed successfully.

---

5. EventBridge – Event Routing

- **Create an Event Bus:**
    - Name: `OrderEventsBus`
- **Set Up EventBridge Rules:**
    - **Order Placement Rule:** Configure a rule that catches new order events (from your application or a test script) and forwards them to:
        - Either the SQS `OrderQueue` (to decouple order ingestion)
        - Or directly to a Lambda function that triggers the Step Functions workflow.
- **Event Format:** Ensure that events follow a consistent JSON schema (e.g., with detail-type, source, and detail fields).

---

6. EventBridge Scheduler – Scheduled Tasks

- **Create a Scheduler Rule:**
    - Schedule a task (e.g., every hour or every few minutes for testing) that triggers a Lambda function.
- **Scheduled Lambda Function:**
    - This function can perform health checks on your order processing pipeline (e.g., reviewing CloudWatch logs/metrics) or send a summary report of orders processed in the last period.
- **Test the Scheduled Task:**
    - Confirm that the Lambda function executes according to the schedule.

---

7. SNS – Stakeholder Notifications

- **Create an SNS Topic:**
    - Name: `OrderNotifications`
- **Subscribe an Endpoint:**
    - Add at least one email subscription (verify the email if necessary).
- **Integration with Workflow:**

    - At the end of the Step Functions workflow or within one of the Lambda functions, publish a message to the SNS topic to notify stakeholders (e.g., “Order 12345 processed successfully”).
- **Test the SNS Notification:**
    - Confirm that subscribers receive the notifications.

---

8. CloudWatch – Monitoring & Logging

- **Enable Logging:**
    - Ensure all Lambda functions and the Step Functions state machine send logs to CloudWatch.
- **Create Dashboards/Alarms:**
    - Set up CloudWatch dashboards to monitor key metrics such as:
        - Number of processed orders.
        - Failed execution counts in Step Functions.
        - SES sending errors.
        - SQS queue metrics (e.g., ApproximateNumberOfMessages).
    - Create alarms for critical failures (e.g., if order processing failures exceed a threshold).
- **Documentation:**
    - Provide screenshots or a summary of your CloudWatch dashboards and alarms setup.

Submission Requirements

1. **Documentation:**
    - A detailed report describing your architecture, workflow, and integration of each AWS service.
    - Diagrams illustrating service interactions (SQS → Step Functions → Lambda → SES, SNS, EventBridge, etc.).
2. **Screenshots:**
    - SQS configuration and sample messages.
    - Step Functions state machine definition and execution details.
    - Lambda function code snippets and CloudWatch logs.
    - SES configuration and a sample confirmation email.
    - EventBridge rules and Scheduler configuration.
    - SNS topic configuration and subscription confirmation.
    - CloudWatch dashboards and alarms.
3. **Source Code:**
    - Provide links to your repository containing Lambda code, configuration files, and any deployment templates (CloudFormation/SAM/Terraform if used).

4. **Deployment Instructions:**
    - A step-by-step guide on how to deploy and test your solution.

Evaluation Criteria

- **Service Integration:** Correct use and configuration of Step Functions, CloudWatch, SES, EventBridge Scheduler, EventBridge, SNS, and SQS.
- **End-to-End Functionality:** Successful simulation of the order processing pipeline from order reception to stakeholder notifications.
- **Monitoring & Error Handling:** Effective use of CloudWatch for logging and monitoring, with proper error handling in the workflow.
- **Documentation & Diagrams:** Clear and comprehensive documentation and architecture diagrams.
- **Bonus Enhancements:** Additional features or automation that demonstrate advanced understanding.

This assignment is designed to provide practical experience with AWS's serverless and event-driven services while building a real-world order processing pipeline. Feel free to reach out with any questions or for further guidance. Happy building!