Here’s how you can **integrate AWS SNS with Email Subscription in a Spring Boot application** using the AWS SDK for Java.

**✅ Steps to Implement AWS SNS Email Subscription in Spring Boot**

1. **Set up AWS Credentials** (IAM user with SNS permissions).
2. **Create an SNS Topic** in AWS.
3. **Subscribe an Email Address** to the SNS Topic.
4. **Send Notifications** from Spring Boot.

**1️⃣ Add Dependencies in pom.xml**

Make sure your **pom.xml** includes the AWS SDK dependency:

<dependency>

<groupId>software.amazon.awssdk</groupId>

<artifactId>sns</artifactId>

<version>2.20.0</version>

</dependency>

**2️⃣ Configure AWS SNS in application.properties**

Replace with your **AWS credentials** and **region**:

aws.region=us-east-1

aws.accessKeyId=YOUR\_ACCESS\_KEY

aws.secretAccessKey=YOUR\_SECRET\_KEY

sns.topic.arn=arn:aws:sns:us-east-1:123456789012:MySNSTopic

**3️⃣ Create a Spring Boot Service to Handle SNS**

**SNS Configuration**

package com.example.snsdemo.config;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import software.amazon.awssdk.auth.credentials.AwsBasicCredentials;

import software.amazon.awssdk.auth.credentials.StaticCredentialsProvider;

import software.amazon.awssdk.regions.Region;

import software.amazon.awssdk.services.sns.SnsClient;

@Configuration

public class AwsSnsConfig {

@Bean

public SnsClient snsClient() {

return SnsClient.builder()

.region(Region.US\_EAST\_1)

.credentialsProvider(StaticCredentialsProvider.create(

AwsBasicCredentials.create("YOUR\_ACCESS\_KEY", "YOUR\_SECRET\_KEY")))

.build();

}

}

**Subscribe an Email Address to the SNS Topic**

package com.example.snsdemo.service;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Service;

import software.amazon.awssdk.services.sns.SnsClient;

import software.amazon.awssdk.services.sns.model.SubscribeRequest;

import software.amazon.awssdk.services.sns.model.SubscribeResponse;

@Service

public class SnsSubscriptionService {

private final SnsClient snsClient;

@Value("${sns.topic.arn}")

private String topicArn;

public SnsSubscriptionService(SnsClient snsClient) {

this.snsClient = snsClient;

}

public String subscribeEmail(String email) {

SubscribeRequest request = SubscribeRequest.builder()

.protocol("email")

.endpoint(email)

.returnSubscriptionArn(true)

.topicArn(topicArn)

.build();

SubscribeResponse response = snsClient.subscribe(request);

return response.subscriptionArn();

}

}

**Send a Notification via SNS**

package com.example.snsdemo.service;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.stereotype.Service;

import software.amazon.awssdk.services.sns.SnsClient;

import software.amazon.awssdk.services.sns.model.PublishRequest;

import software.amazon.awssdk.services.sns.model.PublishResponse;

@Service

public class SnsNotificationService {

private final SnsClient snsClient;

@Value("${sns.topic.arn}")

private String topicArn;

public SnsNotificationService(SnsClient snsClient) {

this.snsClient = snsClient;

}

public String sendNotification(String message, String subject) {

PublishRequest request = PublishRequest.builder()

.message(message)

.subject(subject)

.topicArn(topicArn)

.build();

PublishResponse response = snsClient.publish(request);

return response.messageId();

}

}

**4️⃣ Create a Controller to Handle Requests**

**REST API to Subscribe and Send Emails**

package com.example.snsdemo.controller;

import com.example.snsdemo.service.SnsNotificationService;

import com.example.snsdemo.service.SnsSubscriptionService;

import org.springframework.web.bind.annotation.\*;

@RestController

@RequestMapping("/sns")

public class SnsController {

private final SnsSubscriptionService snsSubscriptionService;

private final SnsNotificationService snsNotificationService;

public SnsController(SnsSubscriptionService snsSubscriptionService, SnsNotificationService snsNotificationService) {

this.snsSubscriptionService = snsSubscriptionService;

this.snsNotificationService = snsNotificationService;

}

@PostMapping("/subscribe")

public String subscribeEmail(@RequestParam String email) {

return snsSubscriptionService.subscribeEmail(email);

}

@PostMapping("/notify")

public String sendNotification(@RequestParam String message, @RequestParam String subject) {

return snsNotificationService.sendNotification(message, subject);

}

}

**5️⃣ Testing the API**

**Subscribe an Email**

Run the following **cURL command** or use **Postman**:

curl -X POST "http://localhost:8080/sns/subscribe?email=example@email.com"

📩 **Check your email** and **confirm the subscription** from AWS SNS.

**Send a Notification**

curl -X POST "http://localhost:8080/sns/notify?message=Hello!&subject=Test Notification"

📧 **The email should be delivered to all subscribed users.**

**✅ Summary**

1. **Created an SNS topic** and **subscribed an email**.
2. **Sent notifications via SNS** from Spring Boot.
3. **Confirmed subscription and tested email notifications.**

🚀 **Now, you have a working AWS SNS + Email Subscription in Spring Boot!** 🚀