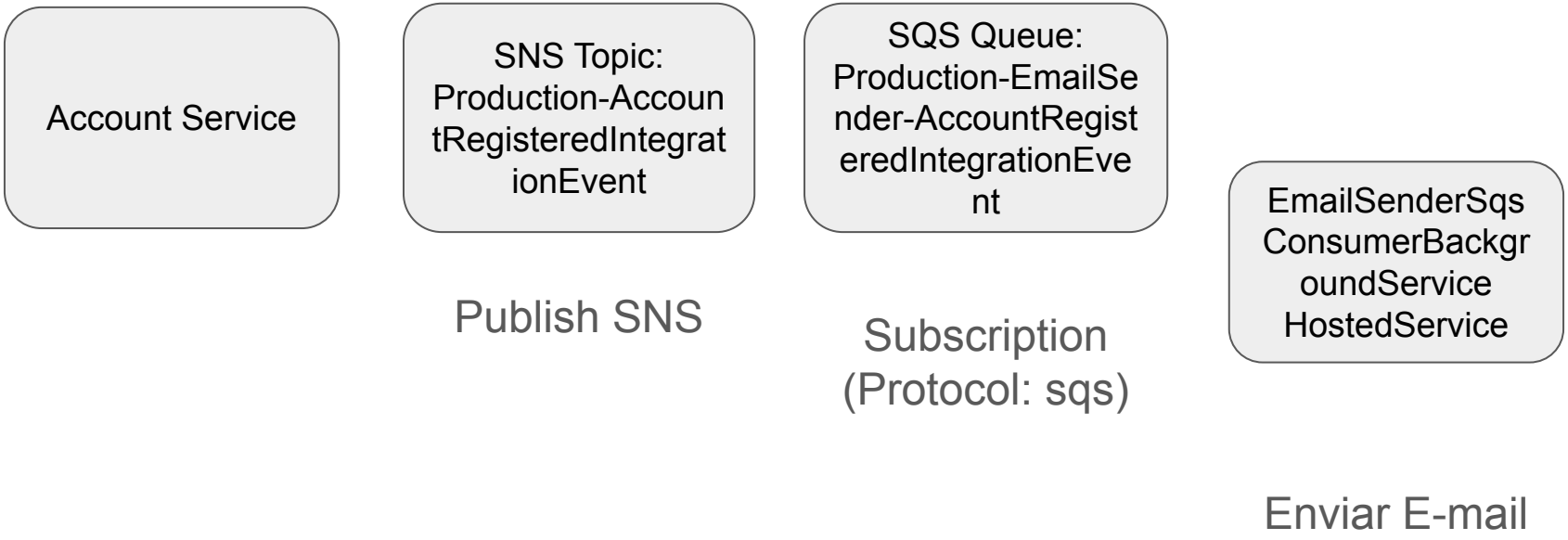
# Apresentação: Arquitetura de Envio de E-mails via Eventos (AWS SNS + SQS + HostedService)



## Contexto Geral

- Objetivo: desacoplar o envio de e-mails do fluxo principal da aplicação (ex: criação de conta).
- Arquitetura **event-driven (baseada em eventos)** com **AWS SNS** e **AWS SQS**.

# Conceitos Principais

## 🟣 AWS SNS (Simple Notification Service)

- Serviço de **publicação e assinatura (Pub/Sub)**.
- Responsável por **distribuir mensagens (eventos)** a vários consumidores simultaneamente.
- Permite **fan-out** — um evento pode chegar a várias filas (múltiplos microserviços)

## 🟢 AWS SQS (Simple Queue Service)

- Serviço de **fila de mensagens**.
- Garante **entrega confiável e assíncrona** das mensagens enviadas pelo SNS.
- Armazena as mensagens até que sejam processadas com sucesso.

## 🔗 Subscription (Assinatura SNS → SQS)

- Liga o **tópico SNS** à **fila SQS**.
- Tudo que é publicado no tópico SNS é automaticamente **entregue à fila**.
- O protocolo da assinatura é `sqs`.

📄 **Exemplo ambiente:**

| Componente | Valor |
| --- | --- |
| SNS Topic | `Production-AccountRegisteredIntegrationEvent` |
| SQS Queue | `Production-EmailSender-AccountRegisteredIntegrationEvent` |
| Subscription | `Protocol: sqs` conectando os dois |

# Fluxo do Processo de Envio de E-mails

1. **Usuário é criado** na aplicação.
2. O microserviço de “Contas” publica o evento `AccountRegisteredIntegrationEvent` no **SNS**.
3. O SNS entrega o evento para a fila **SQS** assinada.
4. Um **HostedService** (background service) consome a fila e processa as mensagens.
5. O **serviço de e-mail** envia o e-mail de boas-vindas.
6. A mensagem é **removida da fila** após o processamento com sucesso.

# Benefícios da Arquitetura

| Benefício | Descrição |
| --- | --- |
| 🧩 **Desacoplamento** | O serviço de envio de e-mails não depende do serviço de contas. |
| 🔁 **Escalabilidade** | Cada serviço pode processar mensagens no seu ritmo. |
| 📬 **Confiabilidade** | SQS garante que nenhuma mensagem se perca. |
| 🕒 **Assincronicidade** | O usuário não precisa esperar o e-mail ser enviado. |
| 📣 **Fan-out** | O mesmo evento pode acionar outros serviços (CRM, Analytics, etc.). |

## Boas Práticas

- Sempre **configurar DLQ (Dead Letter Queue)** para capturar mensagens com falhas.
- **Evitar lógica pesada no SNS publisher** — publique e retorne rápido.
- Monitorar métricas via **CloudWatch** (mensagens não processadas, falhas, tempo médio de consumo).

# Resumo Final

| Etapa | Serviço | Responsabilidade |
|---|---|---|
| 1 | SNS | Publicar evento `AccountRegisteredIntegrationEvent` |
| 2 | Subscription | Enviar evento para fila SQS |
| 3 | SQS | Armazenar evento até o processamento |
| 4 | HostedService | Consumir fila e processar (envio de e-mail) |
| 5 | CloudWatch / DLQ | Monitorar e tratar falhas |