## Challenge Name: Email Service for Salesforce with Amazon SES

Challenge Details

<http://www.cloudspokes.com/challenges/1555>

Solution Submitter Name

Romin Irani

Solution Submitter ID

romin

Solution Submitted Profile URL

<http://www.cloudspokes.com/profile.html?username=romin>

Solution File Attached

**EmailServiceSF\_Amazon\_SES\_RominIrani\_romin\_version2.zip**

**Note that I had submitted a partially working solution to meet the cutoff of June 13. Subsequent to that, I got the correct email address that works with Amazon SES and now I am able to not just send off messages successfully but they also have attachments being processed and attached. So this is my second and latest version of the submission. Please consider this version as my final solution.**

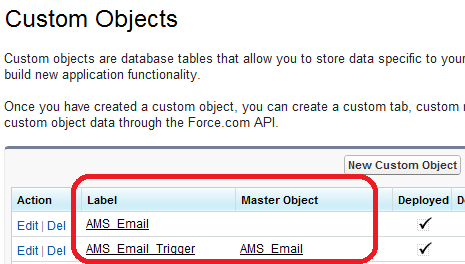
**Overview**

The ZIP file contains:

* This document
* **EmailService.zip**: A Java Project that can be installed to Heroku via the Heroku Toolbelt. This is a Maven based project that you can use as is to deploy the project to Heroku.

# Salesforce Objects

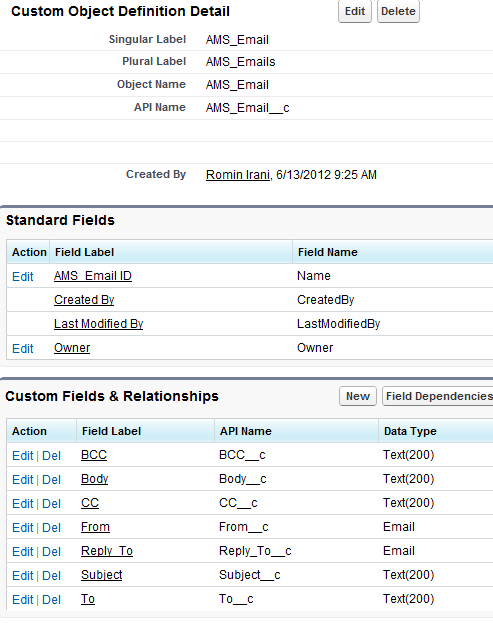
As per the requirements of the challenge, I have created two custom objects in Salesforce:



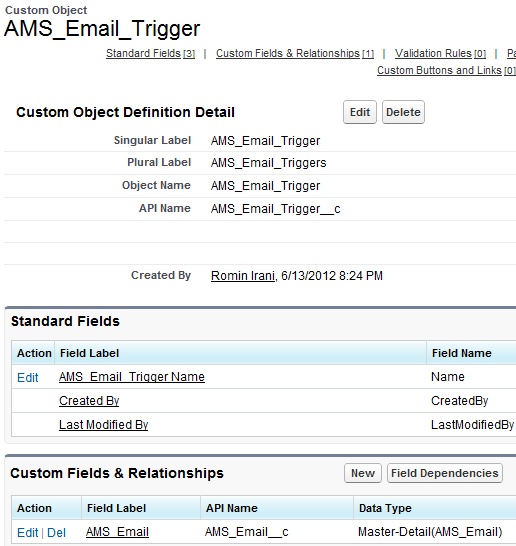
1. **AMS\_Email:** This object that has all the custom fields and it **also includes the Notes and Attachments**. The Attachments will be where all the attachments for the email record will be placed, so that it is associated here itself with the **AMS\_Email** Object.

I have currently made the following assumptions:

* From, To, ReplyTo, Subject and Body fields are made mandatory. The CC and BCC fields are optional
* The From and ReplyTo fields are of Email Type. The To, BCC and CC fields are currently of Text type and limited to a total of 200 characters. You can give a comma separated list of email addresses.



1. **AMS\_Email\_Trigger:** This is another Custom Object. And is primarily used to trigger the Email to be sent. Note that it has a Master Detail Relationship with **AMS\_Email**. I have taken this approach so that I can first use the normal Salesforce GUI to create the **AMS\_Email** Object and all its attachments. Then I will simply insert a record in the **AMS\_Email\_Trigger** which has a push topic associated with it.



A push topic is created as given below. The Apex code is given below and you will need to run it via the Developer Console in your DE Org.

**PushTopic pushTopic = new PushTopic();**

**pushTopic.ApiVersion = 23.0;**

**pushTopic.Name = 'AllAMSEmailsTriggerTest';**

**pushTopic.Description = 'All records for the AMS\_Email\_Trigger object';**

**pushtopic.Query = 'SELECT Id, Name, AMS\_Email\_\_c FROM AMS\_Email\_Trigger\_\_c';**

**insert pushTopic;**

**System.debug('Created new PushTopic: '+ pushTopic.Id);**

What the above means is that only when there is a new / updated record for **AMS\_Email\_Trigger** will it trigger the listener which can then access the details of the **AMS\_Email** Record.

# Heroku Project

The Heroku Project is provided in the zip file : **EmailService.zip.**

The ZIP file contains the following folders/files:

* **src:** The main file to look at over here is the **main.java.AMSEmailTriggerSubscriber**
  + This file does the Salesforce Authentication first
  + Then it starts up a listener to listen to the Push Topic that we created earlier i.e. **AMSAllEmailsTriggerTest**
  + On receipt of a message, it will check if it is an **updated** or a **created** type. We are interested in the **created** type, so that we do not incorrectly process any **updated** events.
  + It will then extract out all the fields like To, Bcc, From, Body, Subject and the Attachments Body URLs if present.
  + It will then invoke the **send\_email()** method that will use the Amazon SES to send out the email.
  + The send\_email() method uses Java Mail to compose a multi-part message. It also processes attachments if present in the message. It uses the correct Content Type as provided by the Attachment Sales Force record.
* **pom.xml :** This is the maven build file. All the Dependencies (JAR files) are mentioned here.
* **Procfile:** Heroku file to launching the worker client

# Deploying and running the Heroku Project and Application

Assuming that Heroku Toolbelt is installed, you can deploy the project as per the steps given below:

1. Unzip the EmailService.zip file
2. Traverse to the root folder “EmailService” and put all the files in Git. Stay at the root folder and type on command prompt
   1. **git init**
   2. **git add .**
   3. **git commit –m “new release”**
3. Create the app on Heroku by typing : **heroku create –-stack cedar**
4. Deploy your code by typing : **git push heroku master**
5. Run the Java Client by typing : **heroku run sh target/bin/workerClient**

Now all you need to do is go to Salesforce and create some **AMS\_Email** Custom Objects. Make sure that you try out some with attachments and without attachments also.

Then to trigger the whole process, simply add an **AMS\_Email\_Trigger** object on the particular AMS\_Email Object. This will result in a notification to the workerClient and you will see it send out the email.

Please let me know if you have any questions. Just email me at : [romin.k.irani@gmail.com](mailto:romin.k.irani@gmail.com)