Message Recall: Deployment Guide

Last Updated: Wed May 25, 2016

CONTENTS

Overview

Message Recall is a Google-AppEngine-hosted application designed to run in a **single Google Apps domain**. The following instructions describe a detailed process for deploying the application:

1. Create a Google Apps Domain Super-Admin ‘Role’ account
2. Create the App Engine hosting environment
3. Configure the Google APIs Console Project
4. Configure your Google Apps Domain for the App Engine application
5. Prepare the source code
6. Download the Google App Engine SDK
7. Upload the source code into the App Engine host
8. Test the running application

The end to end process should take about 30 minutes (not including study time to learn about App Engine or applications).

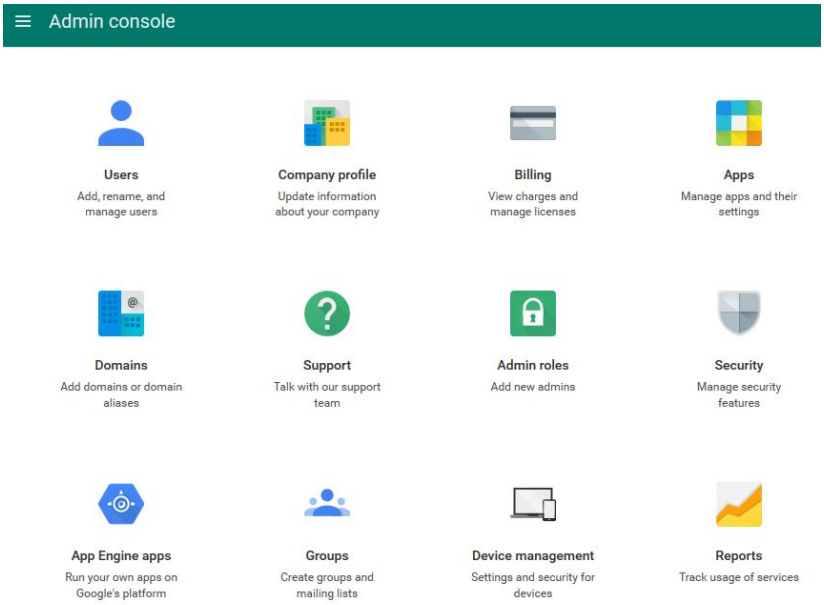
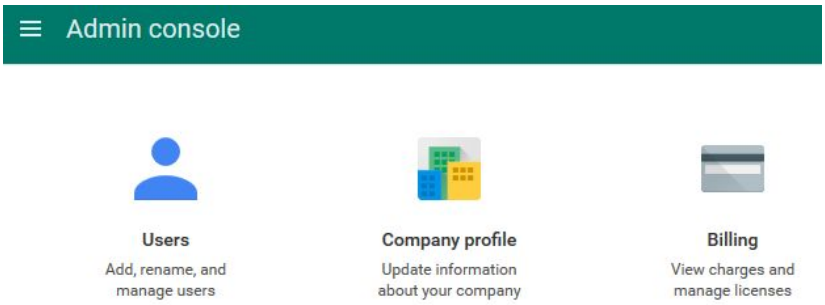
NOTE: We will show examples using a Google Apps Domain named gappslabs.com.

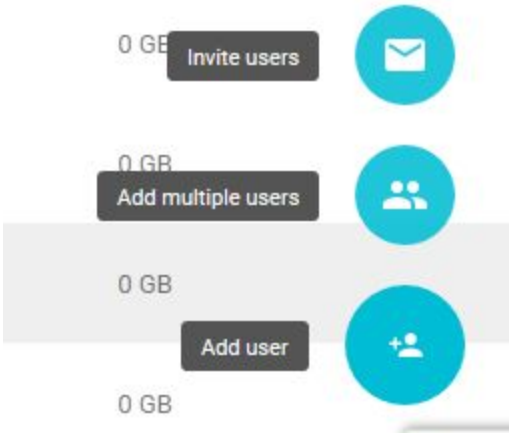
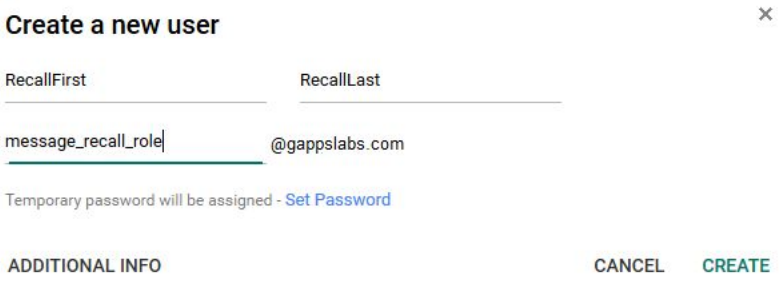
---

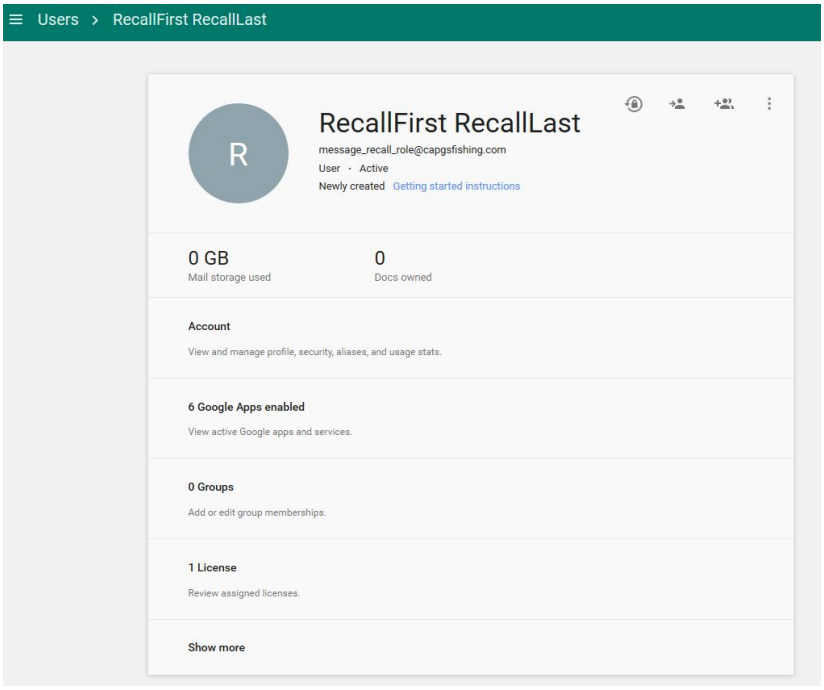
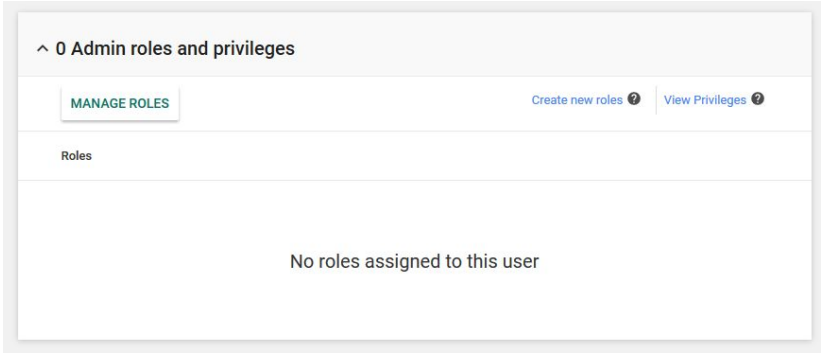
1. Create a Google Apps Domain Super-Admin ‘Role’ account

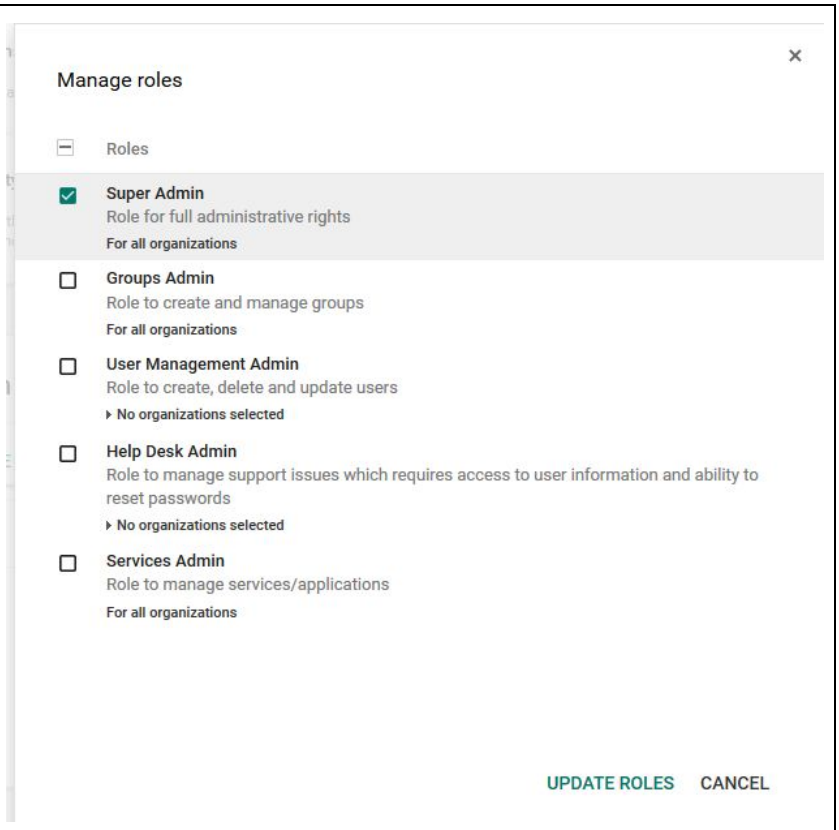
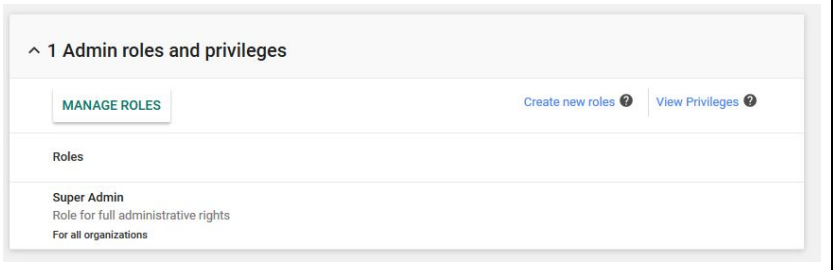
For additional reference, the following articles from the Google Support center discuss accounts and Admin roles:

- Adding users individually
- Pre-built administrator roles
- Assigning Administrator roles to users

In your Google Apps Domain, while logged in as a Domain Admin, navigate to the Admin console at: https://admin.google.com.	
Select ‘Users’ to create the Role account.	

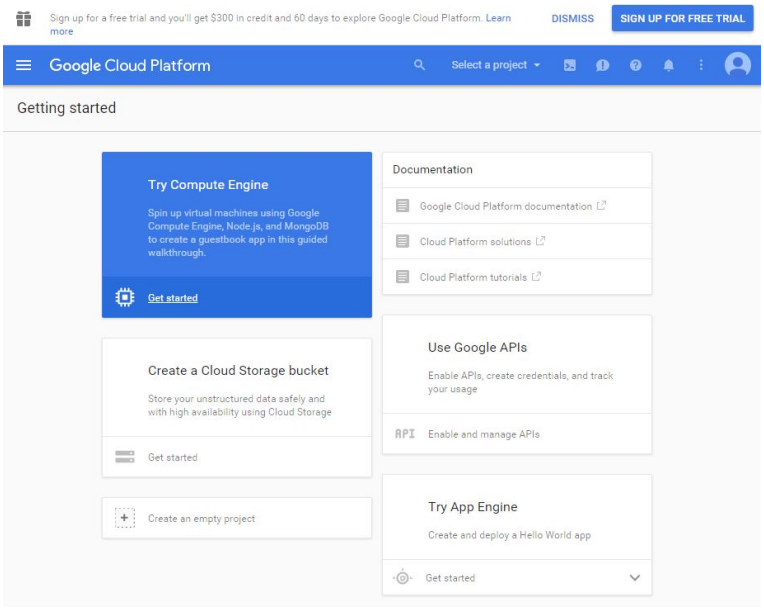
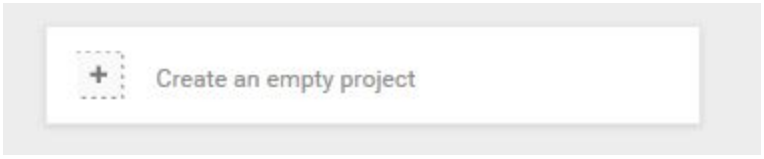
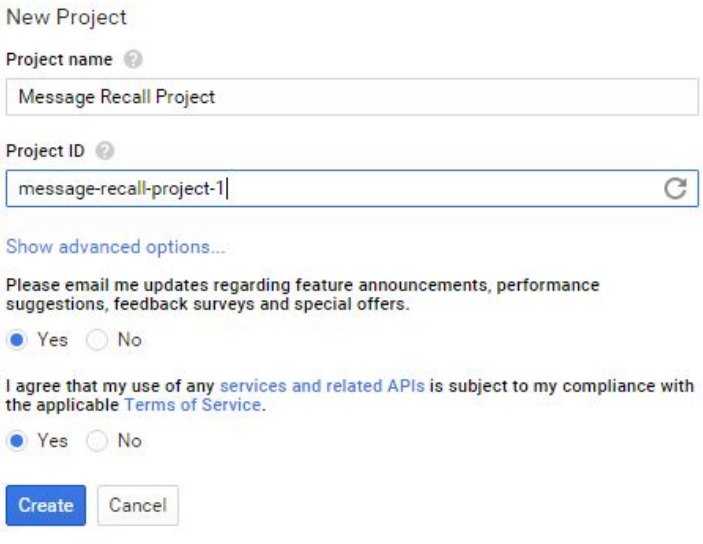
Add a new user by clicking ‘Add user’ that appears after clicking the + graphic. 	
Enter ‘First name’, ‘Last name’ and ‘Primary email address’ (we used message_recall_role for this example) then choose ‘Create’. On the next confirmation page, note the Password generated for the new account. You may choose to send email or print details. Choose Done.	

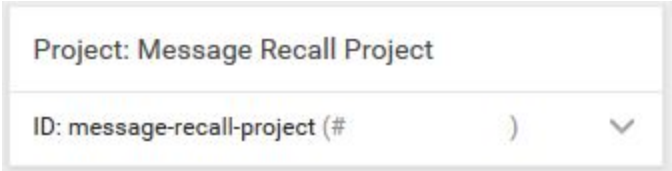
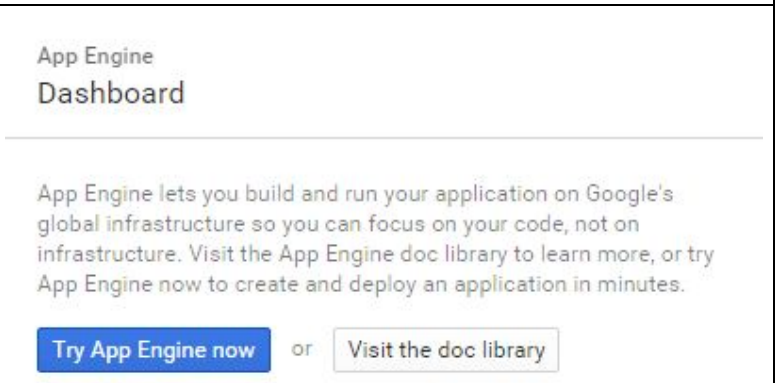
Select the new user, choose ‘Show more’ details.	
Choose: ‘Admin roles and its privileges’.	
Choose: ‘Manage roles’.	

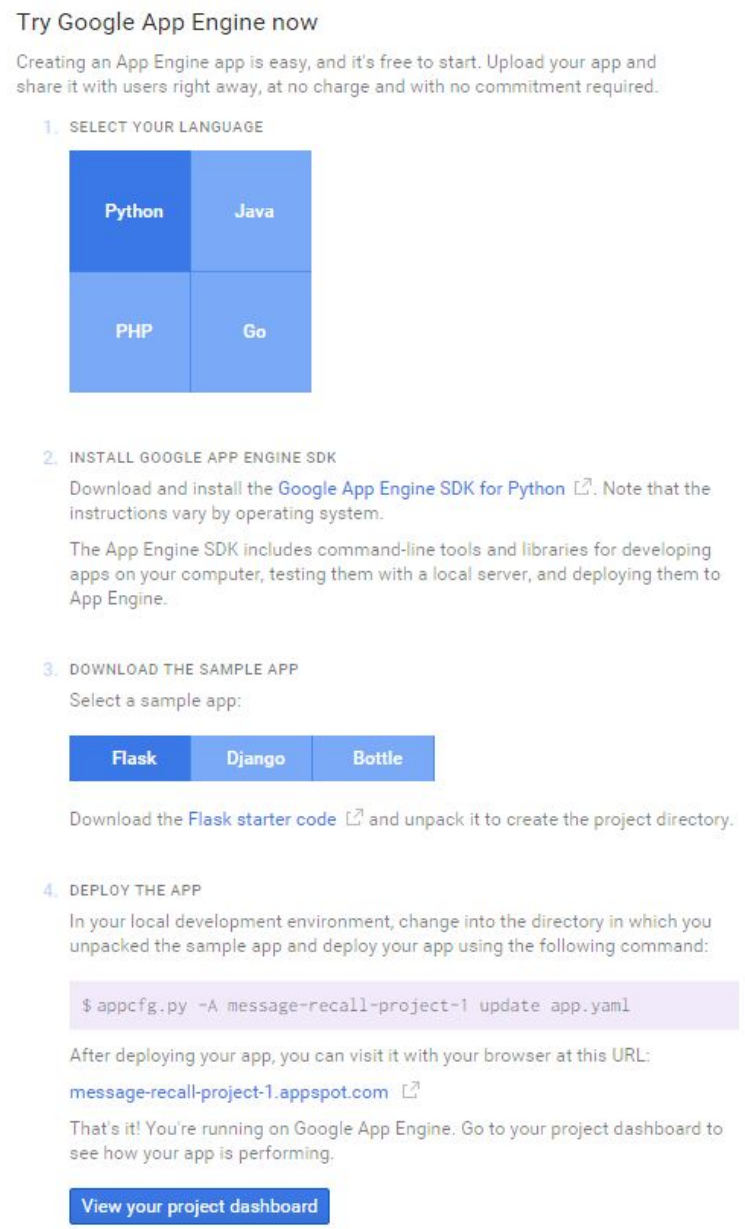
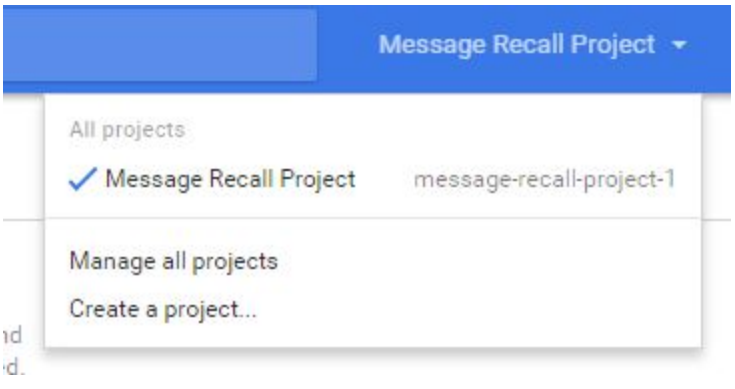
Select: ‘Super Admin’ and choose ‘Update Roles’.	
You should notice your user now has ‘1’ next to ‘Admin roles and its privileges’.	
Sign out. Sign back in as message_recall_role @yourdomain.com. [You will need to Accept the Google Apps (Online) Agreement].	

2. Create the App Engine hosting environment

2.1 Create the application in App Engine

While logged in as message_recall_role, navigate to the 'Google Cloud Platform' console at https://console.cloud.google.com .	
Choose 'Create an empty project'. (You may be requested to verify your account via text message or voice call.)	
Fill out the new project form. A Project name like 'Message Recall <yourdomainname>' is recommended. This maps to an application 'Project ID' of message-recall-<yourdomain>. You'll need this 'Project ID' later. Select 'I agree...' Choose 'Create'.	

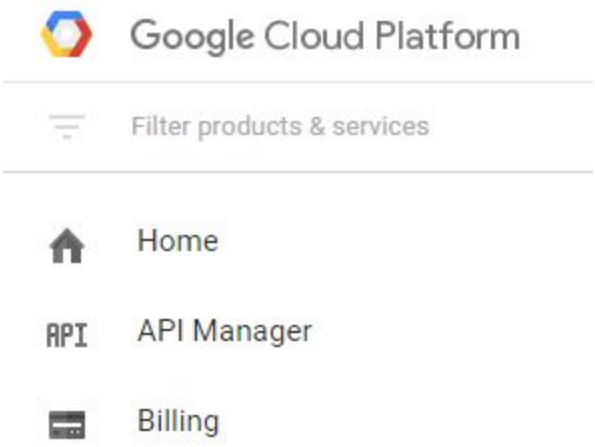
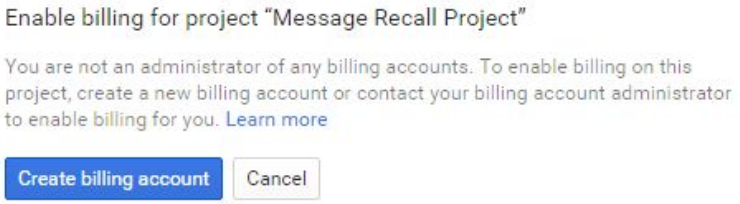
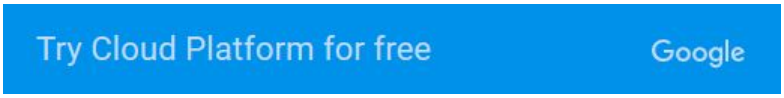
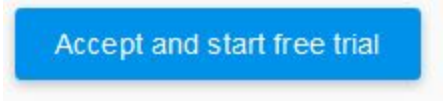
You should see the new Project.	
Now navigate to App Engine. Open the left-sidebar menu and choose: ‘COMPUTE’ > ‘App Engine’.	
Choose ‘Try App Engine now’	

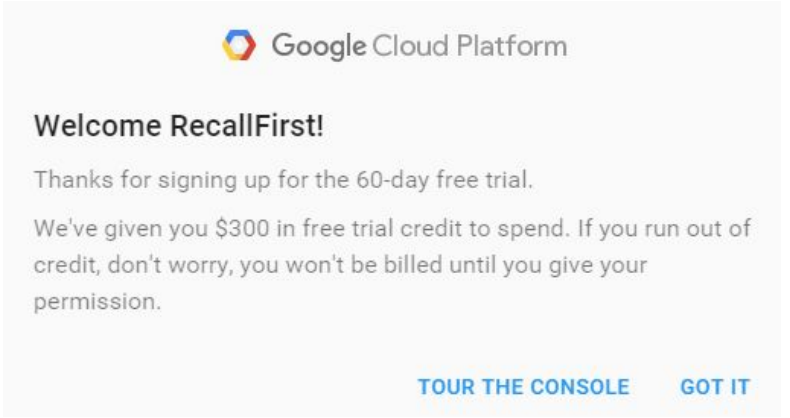
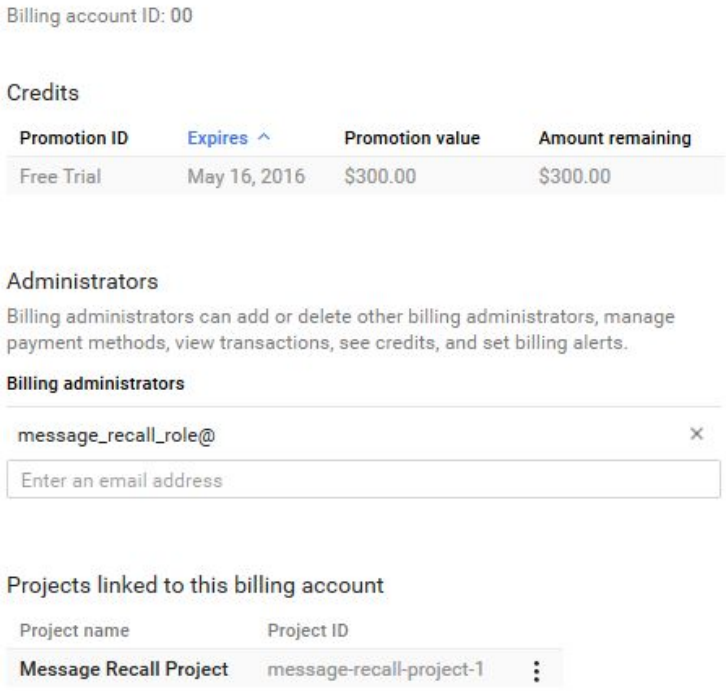
Choose ‘View your project dashboard’. Make sure ‘Python’ is selected.	
At this point the App Engine project is created. This can be observed by clicking the ‘Message Recall Project’ dropdown.	

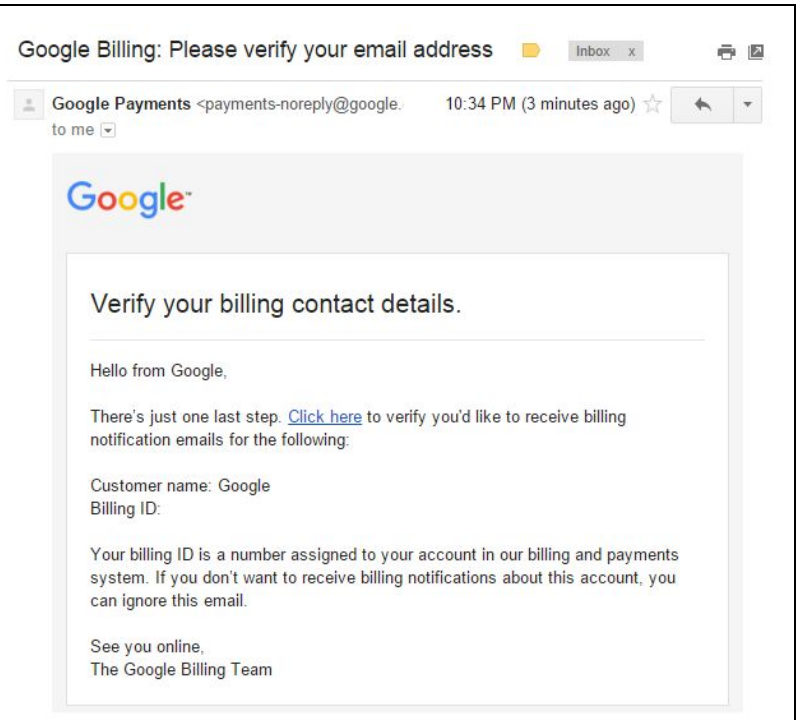
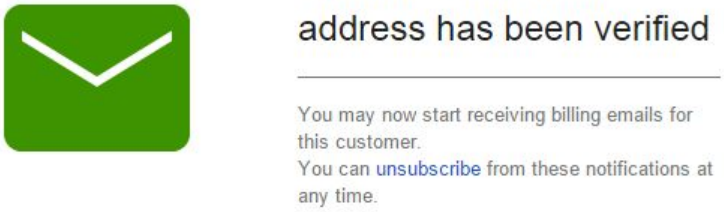
2.2 Enable billing for your application

This application uses the IMAP mail API which requires us to ‘Enable Billing’ in the application. Our use of this API is very small so any actual billing is very small.

However, Message Recall’s use of backends is not free. It is **highly recommended** that you track your Billing Status on the application dashboard frequently. Running out of daily billing quota will cause your application to show a **OverQuotaError**.

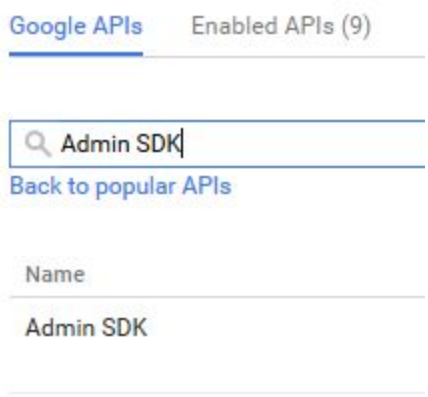
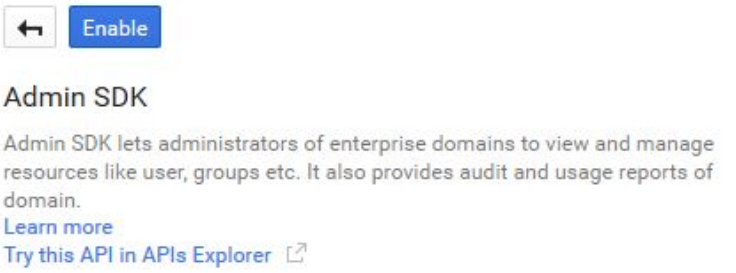
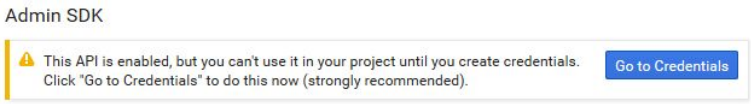
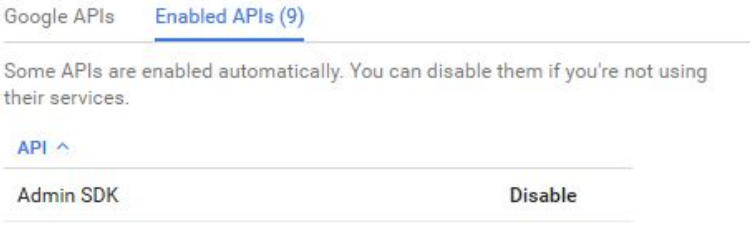
Open the sidebar menu and choose ‘Billing’.	
Choose ‘Create billing account’.	
Enter billing details (even though you won’t be charged during your free trial).	
Choose ‘Accept and Start Free Trial’.	

You can ‘Tour the Console’ or choose ‘Got it’.	
You should see billing details.	

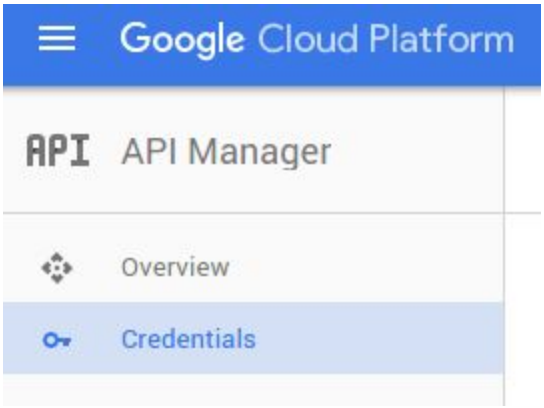
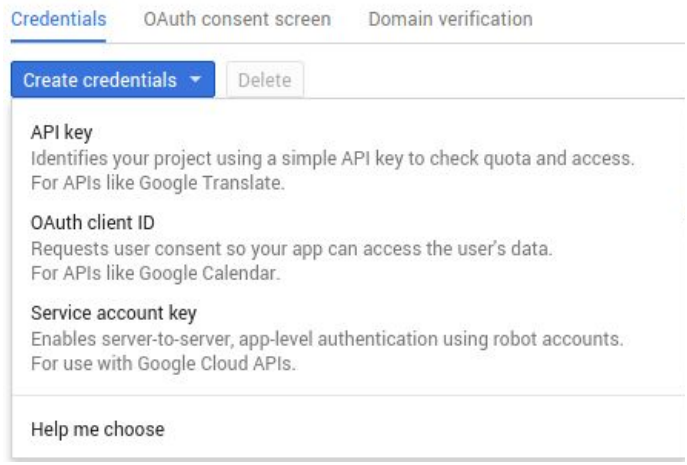
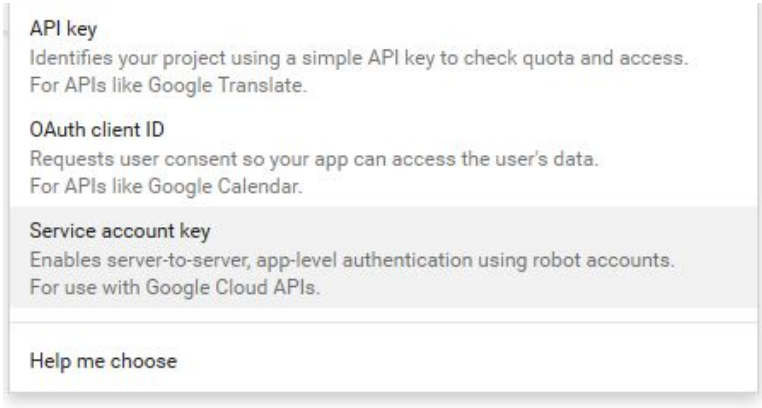
You should also receive an email to confirm your billing contact details. Make sure you choose ‘Click here’ in the email to confirm your billing details.	
You should see a confirmation that your billing details are confirmed.	

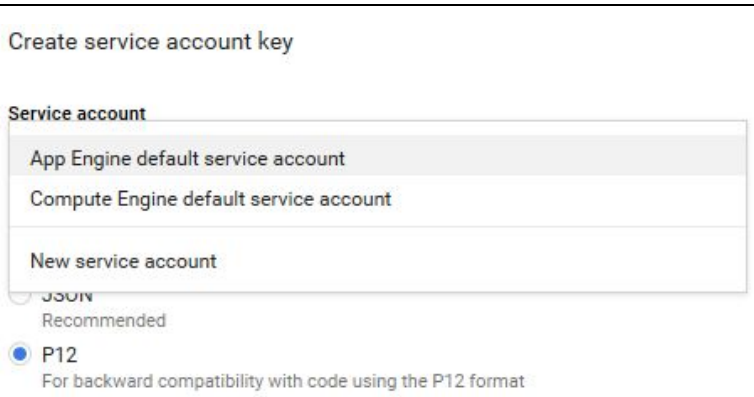
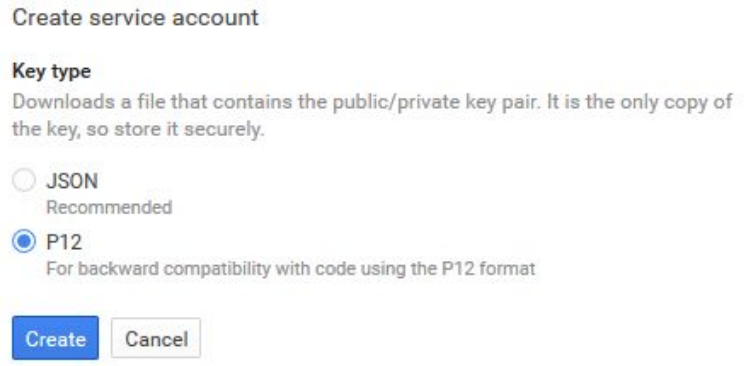
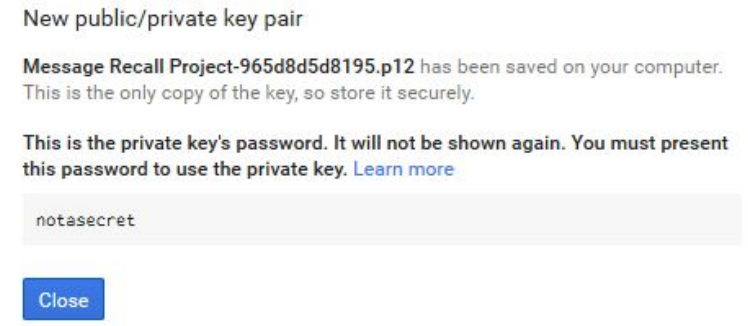
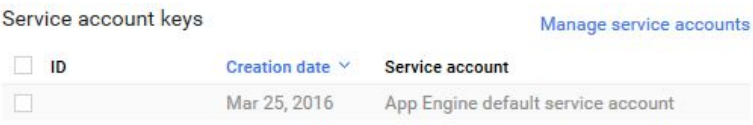
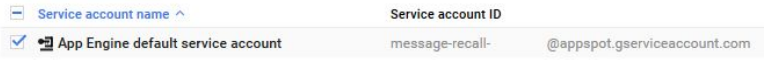
3. Configure the Google APIs Console Project

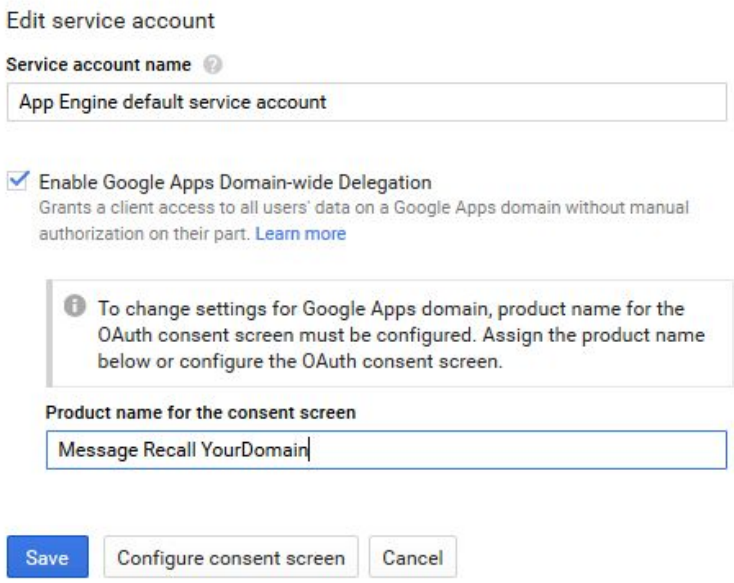
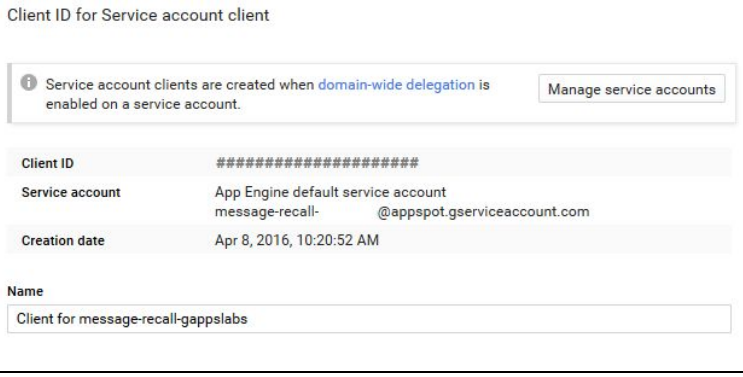
3.1 Enable the Admin SDK API

In the Cloud Console open the sidebar menu and choose ‘API Manager’.	
Type ‘Admin SDK’ in the Search box. Choose ‘Admin SDK’ in the results. Choose ‘Enable’ to enable this API.	
The API is enabled, but we must create credentials now for our server-to-server interactions with APIs. We will do that next.	
Navigate back to ‘Overview’ in the ‘API Manager’. The ‘Admin SDK’ should now be listed in the ‘Enabled APIs’ list.	

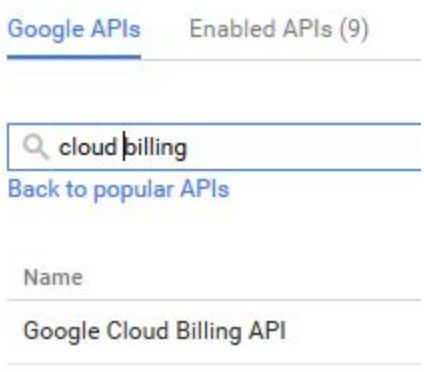
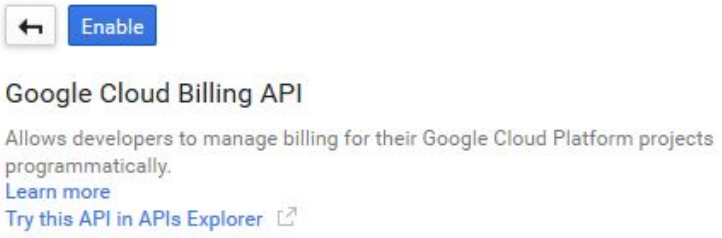
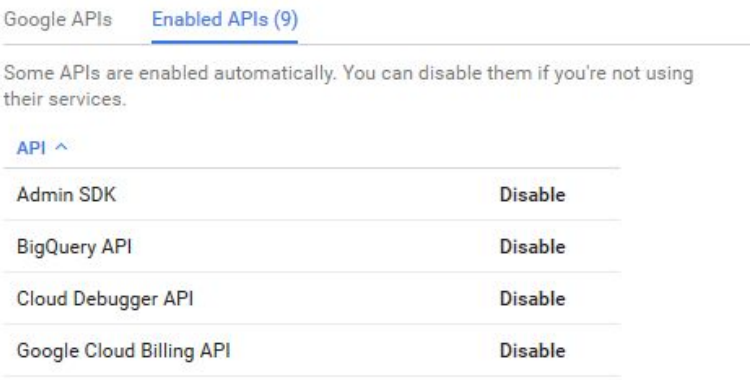
3.2 Create Credentials (a Project Service Account) File

Choose ‘Credentials’ under the ‘API Manager’ to create your new OAuth 2.0 API application client.	
Then, click the ‘Create credentials’ drop-down.	
Choose ‘Service account key’.	

Choose ‘App Engine default service account’.	
Choose ‘P12’ for backward compatibility. Click ‘Create’.	
A file is generated and downloaded to your local system with your private key material. The file has a name of the form ‘Message Recall Project-X.p12’. You may need to click a ‘Save’ button. Choose ‘Close’ to proceed.	
Your OAuth 2.0 client information is now displayed in the Cloud Console. Click the ‘Manage service accounts’ to see these details.	
Take special note of the ‘Service account ID’ (e.g. message-recall-xxxxx@appspot	

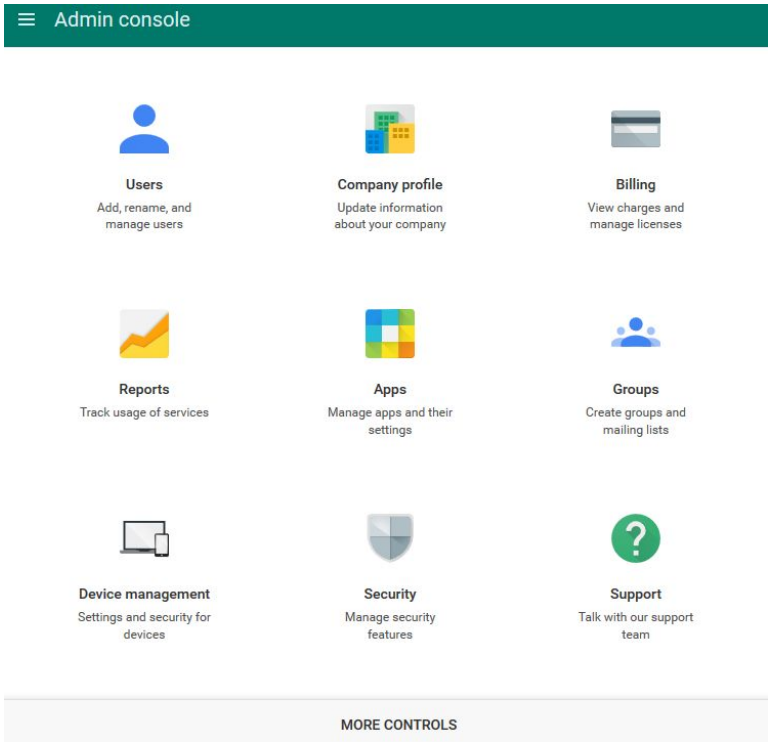
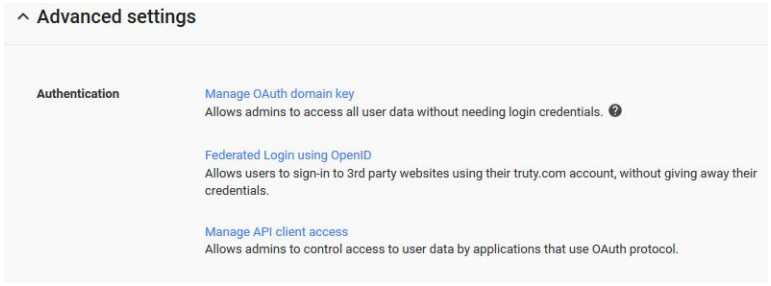
.gserviceaccount.com) it will be needed later.	
We need to allow this service account domain-wide access in the Google Apps domain. Under Options click the ⋮ and select ‘Edit’. Check ‘Enable Google Apps Domain-wide Delegation’. This creates a ‘service account client’ that will be used for API access. Click ‘Save’.	
Take special note of the generated (numeric) ‘Client ID’ for new ‘service account client’. It is used to register the service account in the Google Apps domain later.	

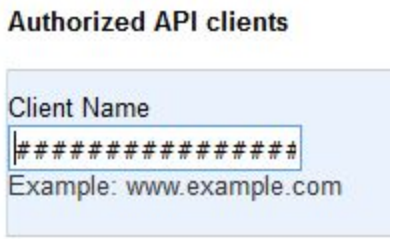
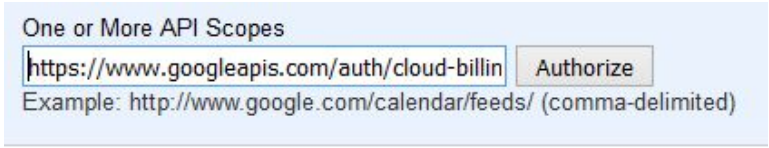
3.3 Enable the Google Cloud Billing API

Return to the Google APIs view within the ‘API Manager’ of the Cloud Console.	
Type ‘Cloud Billing’ in the Search box. Choose ‘Google Cloud Billing API’ in the results. Choose ‘Enable’ to enable this API.	
Navigate back to ‘Overview’ in the ‘API Manager’. The ‘Google Cloud Billing API’ should now be listed in the ‘Enabled APIs’ list.	

4. Configure your Google Apps Domain for the App Engine application

4.1 Add the App Engine service-account to the domain security model

Navigate back to the ‘Admin console’ for your domain at https://admin.google.com. Select ‘Security’.  	
In Security, select ‘Show more’. Select ‘Advanced settings’. Select ‘Manage API client access’.	

In the ‘Client Name’ field, enter the Service Account client ‘Client ID’ from the very end of step 3.2 earlier. It should be of the form: ###################### (only numbers).	
In the ‘One or More API Scopes’ box, enter the following scopes string: https://www.googleapis.com/auth/cloud-billing.readonly,https://mail.google.com/,https://www.googleapis.com/auth/admin.directory.user.readonly Choose ‘Authorize’	
You should notice an entry for your service account client now listed with enabled scopes.	

5. Prepare the source code

5.1 Get the source code

Browse to the github.com page for [googleapps-message-recall]().	google / googleapps-message-recall
Click the 'Download ZIP' button.	Download ZIP
Create an empty folder for the project source code. Copy the downloaded googleapps-message-recall-master.zip file to the empty folder. Unzip the project source code.	`$ unzip ./googleapps-message-recall-master.zip`

Notice the source files under a new folder: googleapps-message-recall-master/ message_recall.	`$ ls -l ./googleapps-message-recall-master/`
	`-rw-r--r--@ 1 adminuser xxxxx 1697 Mar 18 11:29 README.md`
	`drwxr-xr-x@ 28 adminuser xxxxx 952 Mar 18 11:29 message_recall`

5.2 Make local updates

In the extracted source, edit the app.yaml file.
 	
Change the 'application' from 'message_recall' to your app engine 'Application Identifier'.
 	
Save your new app.yaml file.
	From:
	`application: message-recall`
	
	To:
	`application: message-recall-gappslabs`

In the extracted source, edit the service_account.py file.
 	
Change the SERVICE_ACCOUNT_NAME to reflect the 'Service account ID' discovered in step 3.3.
	From:
	`SERVICE_ACCOUNT_NAME =`
	`('000000000000-xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'`
	`'@developer.gserviceaccount.com')`
	
	To:
	`<your Service account ID (Email) address>`

5.3 Create a service account certificate .pem file

Find the file downloaded 'XXXX-privatekey.p12' file and copy it to your application folder.	`googleapps-message-recall-master/message_recall/`
Use the openssl tool to convert the PKCS12 certificate file to PEM format. Note: openssl is available on Linux and Windows Note the output filename of 'messagerecall_privatekey.pem' is required.	`$ openssl pkcs12 -in xxxx-privatekey.p12 -out messagerecall_privatekey.pem -nodes -nocerts`
When prompted for a password, supply the one presented earlier: *notasecret*.	`Enter the password: notasecret`
You should see the following output response and a new file created: messagerecall_privatekey.pem.	`MAC verified OK`
You can now remove the privatekey.p12 file.	`$ rm xxxx-privatekey.p12`

Edit the new messagerecall_privatekey.pem file.
 	
Delete the Bag Attributes lines.
 	
Delete the Key Attributes line.
 	
The .pem file should begin with the 'BEGIN PRIVATE KEY' line and end with the 'END PRIVATE KEY' line.
	`Bag Attributes`
	`    friendlyName: privatekey`
	`    localKeyID: 54 69 6D 65 20 31 33 38 31 37 37 36 30 34 33 31 37 34`
	`Key Attributes: <No Attributes>`
	
	`-----BEGIN PRIVATE KEY-----`
	`MIICdwIB……`
	`-----END PRIVATE KEY-----`

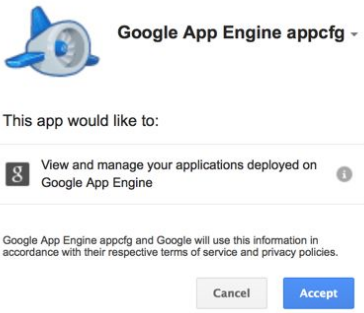
The edited file must be placed in the root source folder next to the app.yaml file.	`googleapps-message-recall-master/message_recall/`

6. Download the Google App Engine SDK

Find your platform, download and install the ‘Google App Engine SDK for Python’	https://developers.google.com/appengine/downloads#Google_App_Engine_SDK_for_Python
Your installation process may be a single unzip or an executable installation program.	

---

7. Upload the source code into the AppEngine host

Deploy the front-end code. From your application source root folder (the folder that contains app.yaml), run the appcfg.py command that was installed with the 'Google App Engine SDK for Python'. Authenticate as your message_recall_role@yourdomain.com user.	`$ appcfg.py --noauth_local_webserver update .`
When prompted, choose 'Accept' to allow the appcfg program to upload the source code to your app engine hosting environment.	

You should notice the following progress on your console confirming the initial application upload (deployment).	`Compilation starting.`
	`Compilation completed.`
	`Starting deployment.`
	`Checking if deployment succeeded.`
	`Deployment successful.`
	`Checking if updated app version is serving.`
	`Completed update of app:`
	`  message-recall-gappslabs, version: 1`
	`Uploading index definitions.`
	`Uploading task queue entries.`

Now deploy the back-end code.	`$ appcfg.py backends . update`

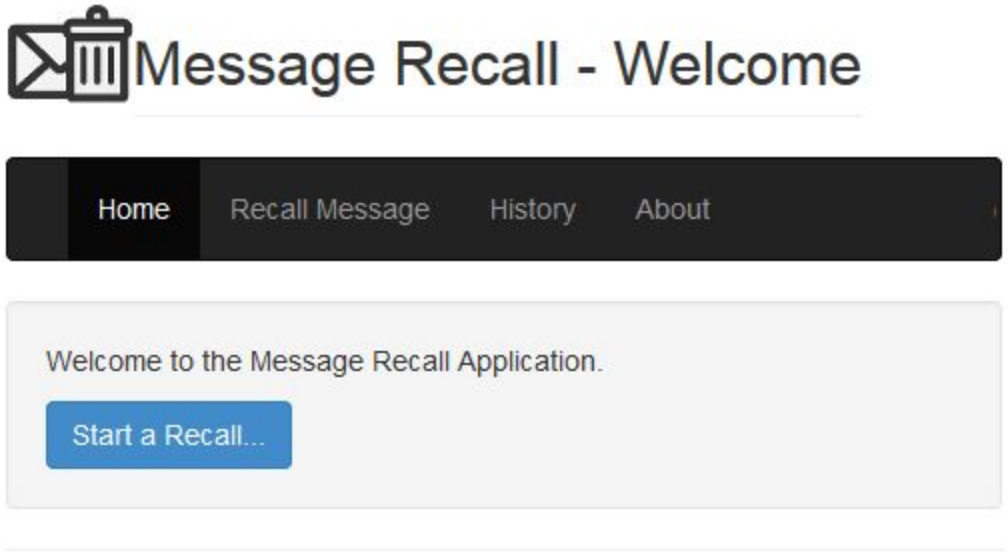
You should notice the following progress on your console confirming the deployment.	`Starting update of app: message-recall-gappslabs, backend: recall-backend`
	`Getting current resource limits.`
	`Scanning files on local disk.`
	`...`
	`Compilation starting.`
	`Compilation completed.`
	`Starting deployment.`
	`Checking if deployment succeeded.`
	`Deployment successful.`

	Checking if updated app version is serving. Completed update of app: message-recall-gappslabs, backend: recall-backend

---

8. Test the running application

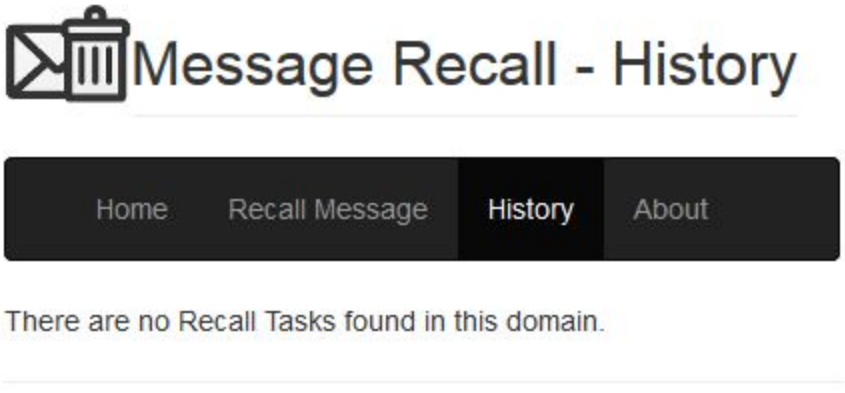
A. Navigate to the application and view the landing page:
https://message-recall-gappslabs.appspot.com



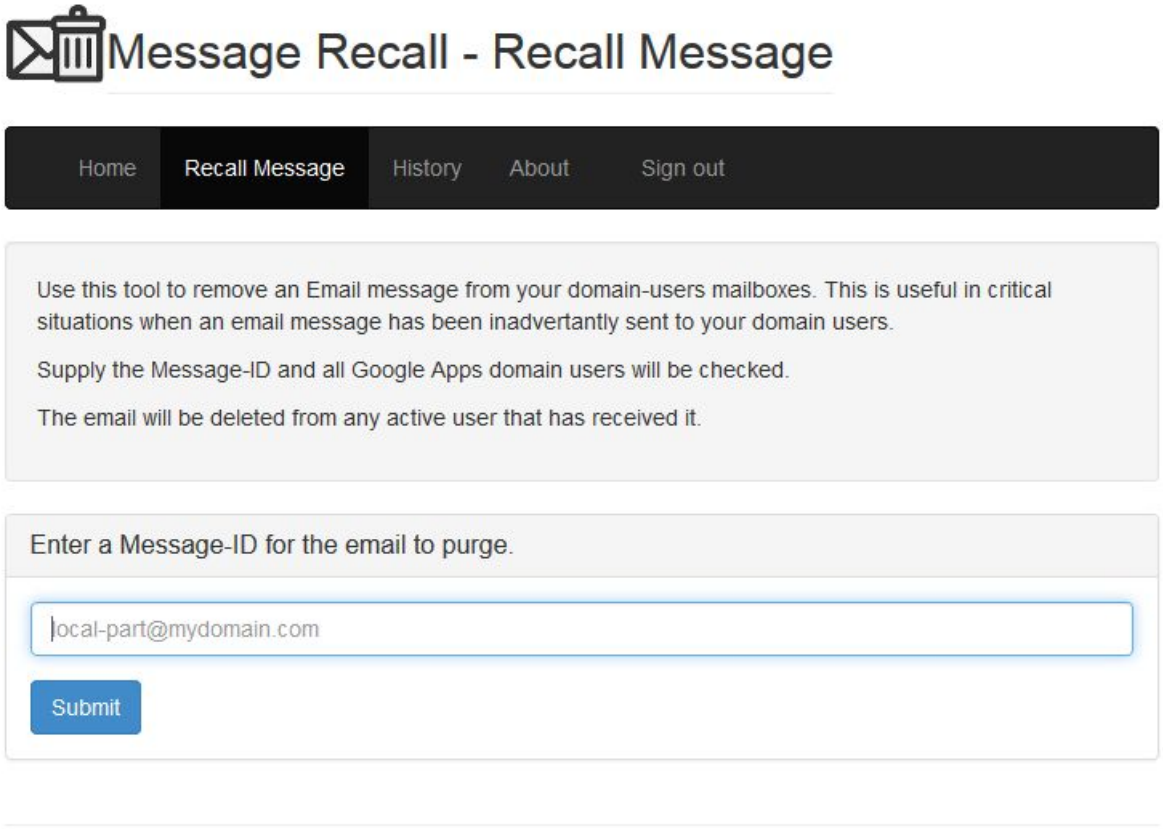
B. Navigate to the History page:

NOTE: The initial application data store indices need to be constructed. If you see the following message on the History page, initial index construction has not yet completed.

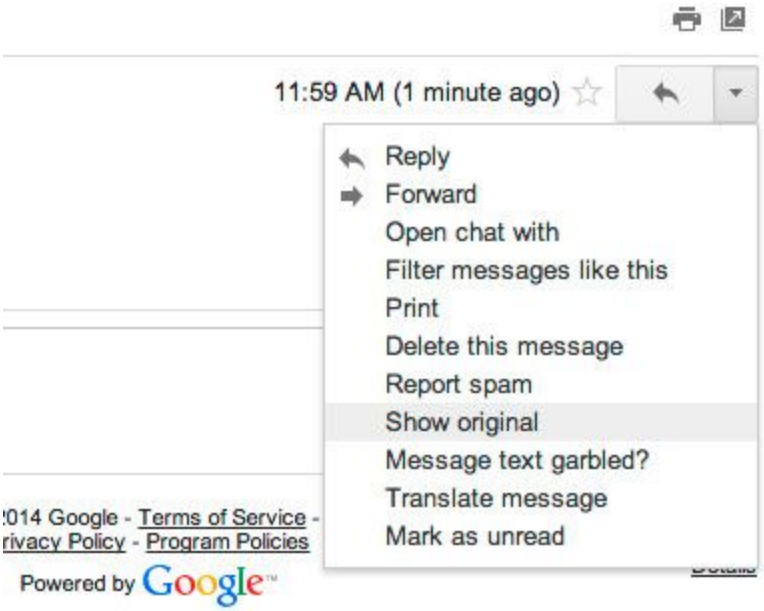
You can also check progress on initial index creation from the AppEngine administrator page under 'Datastore Indexes'.



C. Try recalling a message:



1. From Gmail, send a message to any domain user.
2. From your Gmail ‘Sent Mail’ folder, open the sent message.
3. You can see detailed information about your message by clicking the ‘Show original’.



4. Note the **Message-ID** of your sent message (it is the text inside of the <> brackets including '@mail.gmail.com').
5. Navigate to the 'Recall Messages' page of your application.
6. Enter the Message-ID of your sent message in the textbox.
7. Click 'Submit'.
8. You are dropped into the Message Recall 'Task' page. Refresh it frequently to see status of your recall task.
9. You may choose the 'View Report' page to see a different view of progress.
10. The Task state starts at '*Getting Users*' and progresses through *'Recalling'* to *'Done'*.

11. When the application completes, you should notice the following on the 'Task' page if the message was only sent to 1 user. This indicates the user was identified, is not suspended, and the message was able to be deleted.

Message Recall - Task

Home | Recall Message | History | About | Sign out

Tip: Refresh frequently to see updates.

State	
Owner	
Message-ID	
Start (UTC)	20151024 10:20:49
Stop (UTC)	20151024 10:21:06
Elapsed (m:s)	
Total Users	
Users Processed	
Users with Messages Recalled	1

View Users | View Report | Debug Task

Common Problems

The following problems arise if one of the setup steps is missed or incomplete.

Problem	**Correction**
“unauthorized_client” error.	Add the App Engine service account to your Google Apps domain security with scopes.
“You are not an authorized user of this application.”	This occurs if you have not enabled the ‘Admin SDK’ API properly.
“This AppEngine application requires billing status: “Billing Enabled”.	Walk through the Billing pages in the Cloud Console to ensure your project has a Billing Account ID that is enabled.

Appendices

Getting Help

If you need additional assistance setting up and/or configuring your application, please try the following resources:

Help with Google App Engine
The Google App Engine SDK for Python
Google Apps Documentation & Support: Sign in to your Admin console
Google APIs Console Help
The Google Apps Admin SDK
Service Account Information
The googleapps-message-recall Google Group