

# Password Generator: Deployment Guide

## CONTENTS

[Overview](#)

[Key Features](#)

[Create the App Engine Application](#)

[Configure the App Engine Application](#)

[Configure The Google Apps Domain](#)

[Authorize the App Engine application](#)

[Create Google Apps Admin role account](#)

[Add a new user](#)

[Edit the new user, show more details](#)

[Assign the new user the Super Admin role](#)

[Download and Configure Password Generator](#)

[Download the Password Generator Source via Git or as a zip file from the project site.](#)

[Configure the Password Generator source](#)

[Download Password Generator 3rd Party Libraries](#)

[Convert the service account Certificate - PKCS12 to PEM](#)

[Edit privatekey.pem to remove non-essential details:](#)

[Cleanup the json and .p12 file](#)

[Deploy the App Engine Application](#)

[Download Google App Engine SDK](#)

[Deploy the default state of the App Engine app](#)

[Configure the Application Admin Setting](#)

[Appendix:](#)

[3rd Party Libraries](#)

## Overview

Many large Google Apps customers want to allow non-SAML capable devices to login to Google Apps (iOS, IMAP, etc), however they do not want to sync their corp passwords to Google. These customer also cannot use ASPs (application specific password) because they do not have ways to restrict how many ASPs are used by a user or how often ASPs are created.

This Password Generator solution provides a self-service application customers can deploy to their end users to enable users to create a Google Apps password for the use with iOS, IMAP or other clients that require a password to be stored at Google.

For developers this project code also includes examples of cross-site scripting (XSS) and cross-site request forgery (XSRF) protections implemented in an App Engine project.

## Key Features

- End user self-service password tool for creating Google Apps password.
- Automatically generate and configure iOS devices
- Supports configuring multiple iOS devices for a single user
- Google Group based access control
- Detailed Reporting

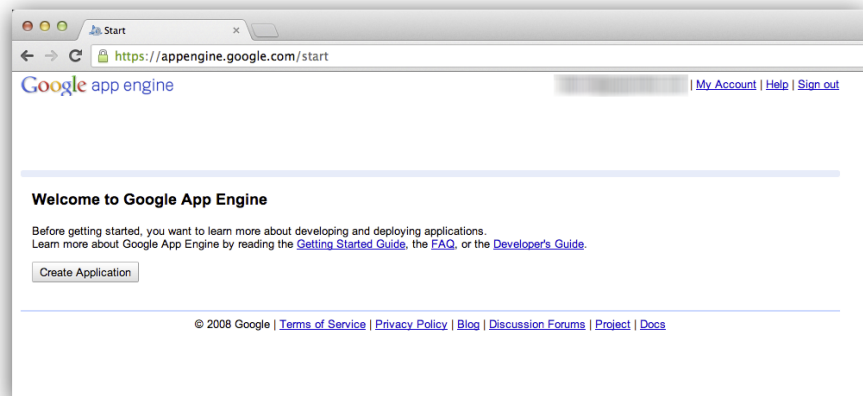
## Create the App Engine Application

Create a domain administrator role account in the Google Apps domain that will perform the admin actions of the App Engine application e.g. PasswordTool@altostrat.com.

It is recommend you create a role account not an end user Google Apps Administrator account.

Goto <https://AppEngine.google.com/>

Select Create Application



## Configure the App Engine Application

Complete the new application form

### Create an Application

You have 10 applications remaining.

Application Identifier:

pwgentool1 .appspot.com [Check Availability](#)

All Google account names and certain offensive or trademarked names may not be used as Application Identifiers. You can map this application to your own domain later. [Learn more](#)

Application Title:

Password Generator

Displayed when users access your application.

Authentication Options (Advanced): [Learn more](#)

Google App Engine provides an API for authenticating your users, including Google Accounts, Google Apps, and OpenID. If you choose to use this feature for some parts of your site, you'll need to specify now what type of users can sign in to your application.

☐ Open to all Google Accounts users (default)

If your application uses authentication, anyone with a valid Google Account may sign in.

☒ Restricted to the following Google Apps domain:

.com

e.g. foo.com

If your application uses authentication, only members of this Google Apps domain may sign in. If your organization uses Google Apps, use this option to create an application (e.g. an HR tracking tool) that is only accessible to accounts on your Google Apps domain. This option cannot be changed once it has been set.

☐ (Experimental) Open to all users with an OpenID Provider

If your application uses authentication, anyone who has an account with an OpenID Provider may sign in.

Terms of Service:

### Your Agreement with Google

This License Agreement for Google App Engine (the "Agreement") is made and entered into by and between Google and the business entity agreeing to these terms ("Customer"). "Google" means either (i) Google Ireland Limited, with offices at Gordon House, Barrow Street, Dublin 4, Ireland, if Customer's billing address is in any country within Europe, the Middle East, or Africa ("EMEA"), (ii) Google Asia Pacific Pte. Ltd., with offices at 8 Marina View Asia Square 1 #30-01 Singapore 018960, if Customer's billing address is in any country within the Asia Pacific region ("APAC"), or (iii) Google Inc., with offices at 1600 Amphitheatre Parkway, Mountain View, California 94043, if Customer's

☒ I accept these terms.

[Create Application](#)

[Cancel](#)

Click the dashboard link

### Application Registered Successfully

The application will use **pwgentool** as an identifier. This identifier belongs in your application's configuration as well. Note that this identifier cannot be changed. [Learn more](#)

The application uses the **High Replication** storage scheme. [Learn more](#)

If you use Google authentication for your application, **Password Generator** will be displayed on Sign In pages when users access your application.

Choose an option below:

- View the [dashboard](#) for Password Generator.
- Use [appcfg](#) to upload and deploy your application code.
- Add [administrators](#) to collaborate on this application.

Select Application Settings and click Save Settings to create an API console project and link

The screenshot shows the 'Basics' tab of the Google App Engine Application Settings page. On the left is a navigation menu with sections: 'Main' (Dashboard, Instances, Logs, Versions, Cron Jobs, Task Queues, Quota Details), 'Data' (Datastore Indexes, Datastore Viewer, Datastore Statistics, Blob Viewer, Prospective Search, Text Search, Datastore Admin, Memcache Viewer), and 'Administration' (Application Settings, Permissions, Blacklist, Admin Logs). The 'Application Settings' link is highlighted. The main content area has a light blue header 'Basics'. It contains several sections: 'Application Title' with a text input 'Password Generator 1'; 'Application Identifier' with the value 'pwgentool1'; 'Service Account Name' with the value 'pwgentool1@appspot.gserviceaccount.com'; 'Application Default Version URL' with the value 'http://pwgentool1.appspot.com'; 'Application Identifier Alias' with the value 'pwgentool1.appspot.com'; and 'Datastore Replication Options' set to 'High Replication'. On the right side, there are settings for 'Cookie Expiration' (Default (1 Day)), 'Authentication Type' (Google Apps domain), and 'Authentication Domain' (a text input with '.com'). A 'Save Settings' button is located at the bottom of the main content area. Below the 'Basics' tab is a 'Performance' tab.

Click on the API project console number link

The screenshot shows a dialog box titled 'Google APIs Console Project Number:'. It displays a blue hyperlink '496564023428'. Below the link is a short paragraph: 'The Google APIs Console allows you to configure other Google APIs, like Translate or Cloud Storage.' At the bottom of the dialog is a 'Save Settings' button.

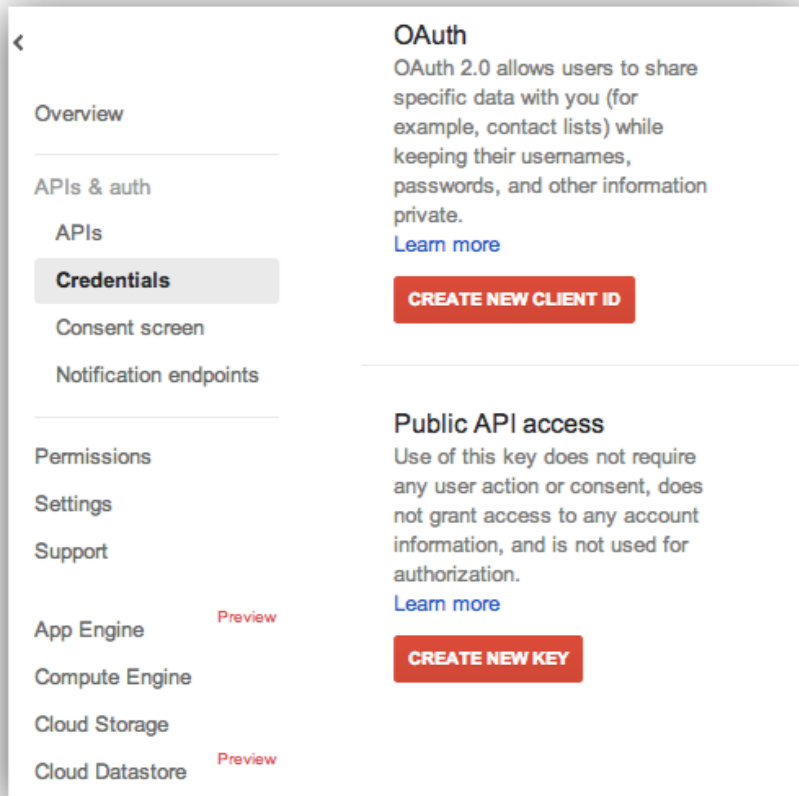
Click on APIs & auth |  
Consent screen and  
enter a Project Name

The screenshot shows the 'Consent screen' configuration page in the Google Cloud Console for project 'pwgentool1'. The left sidebar contains a navigation menu with options: Overview, APIs & auth, APIs, Registered apps, **Consent screen**, Notification endpoints, Permissions, Billing, Settings, and Support. Below this is a section for 'App Engine', 'Compute Engine', 'Cloud Storage', 'Cloud Datastore', 'Cloud SQL', 'BigQuery', and 'Cloud Development'. The main content area is titled 'Consent screen' and includes a note: 'The consent screen will be shown to users whenever you request access to their private data using your client ID. Note: This screen will be shown for all of your applications registered in this project.' The configuration fields include: 'EMAIL ADDRESS' (a dropdown menu), 'PRODUCT NAME' (a text input containing 'Password Generator 1'), 'HOMEPAGE URL' (a text input containing 'https:// or http://'), 'LOGO' (a placeholder image with a 'Max size: 120x60 px' note and an 'Update' button), 'PRIVACY POLICY URL' (a text input containing 'https:// or http://'), 'TERMS OF SERVICE URL' (a text input containing 'https:// or http://'), and 'GOOGLE+ PAGE' (a text input containing 'plus.google.com/' and a 'Page ID' field). At the bottom are 'Save' and 'Cancel' buttons. On the right, a preview of the consent screen is shown, featuring a 'Product Name' header, a 'Developer info' section with an email field, and a list of permissions with checkboxes for 'Know your name, basic info, and list of people you're connected to on Google+', 'Make your app activity and reviews available via Google, visible to you and...', 'Your circles', and 'Only you'. The preview also includes 'Cancel' and 'Accept' buttons.

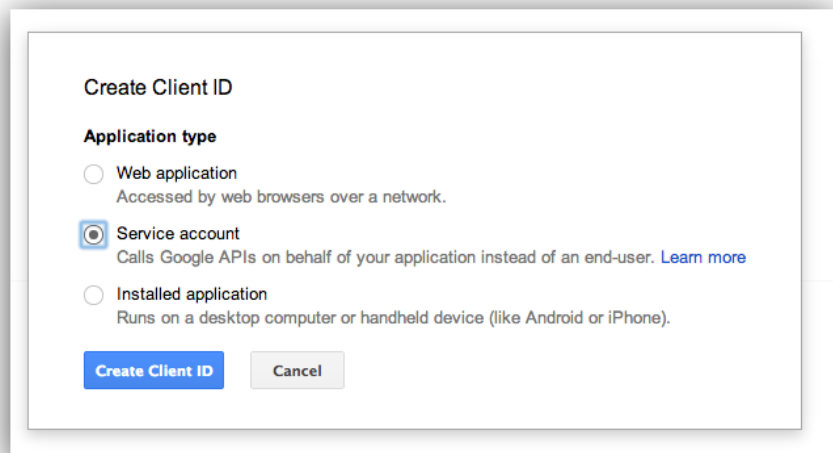
Click on the APIs option  
and enable the Admin  
SDK

	NAME	STATUS
Overview	<a href="#">Admin SDK</a>	ON
APIs & auth	<a href="#">Translate API</a>	ON
<b>APIs</b>	<a href="#">Acquisitions Storage API</a>	OFF
Credentials	<a href="#">Ad Exchange Buyer API</a>	OFF
Consent screen	<a href="#">Ad Exchange Seller API</a>	OFF
Notification endpoints	<a href="#">AdSense Host API</a>	OFF
Permissions	<a href="#">AdSense Management API</a>	OFF
Settings		
Support		

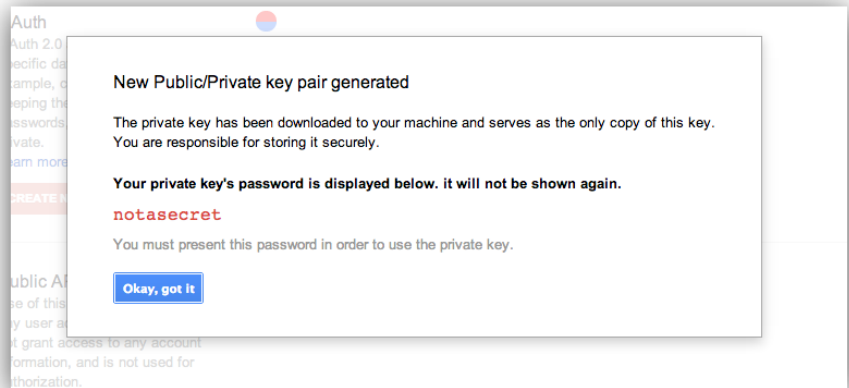
Click on Credentials and select Create New Client ID



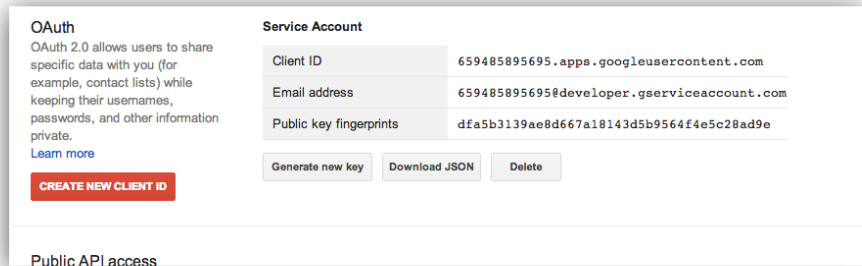
Select Service account



The .p12 certificate for the service account will download



Note the Client ID and EMail Address these two items will be needed in later configuration steps.

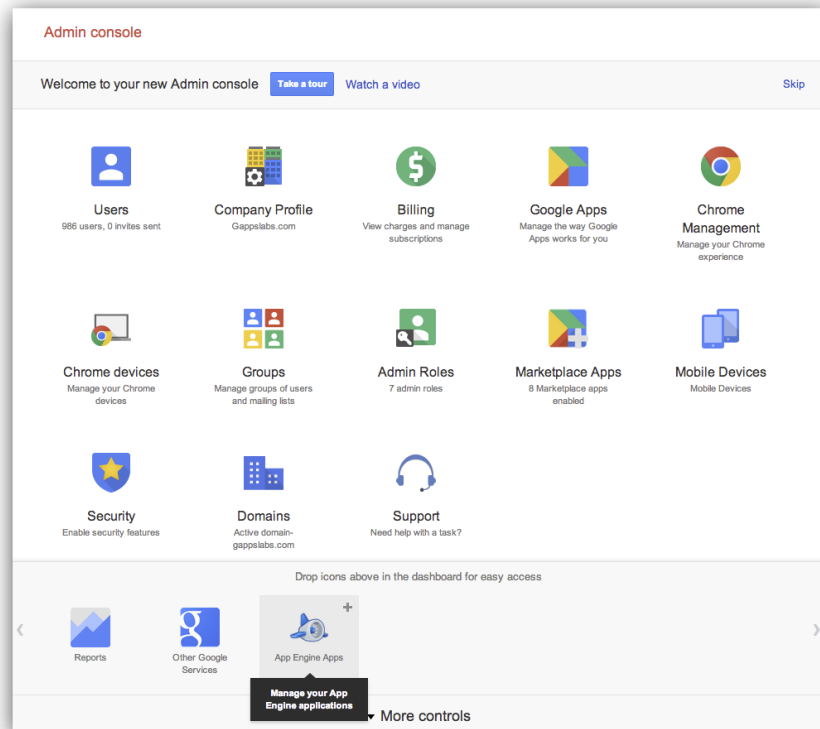


## Configure The Google Apps Domain

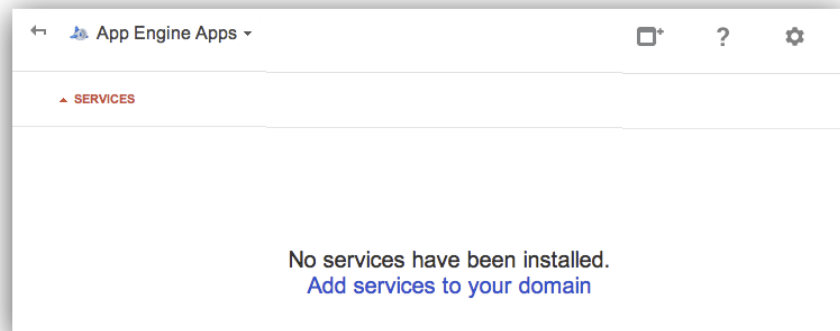
Add the App Engine app to the domain  
Login to

[Admin.Google.com](https://admin.google.com)

Click on More controls  
and select App Engine

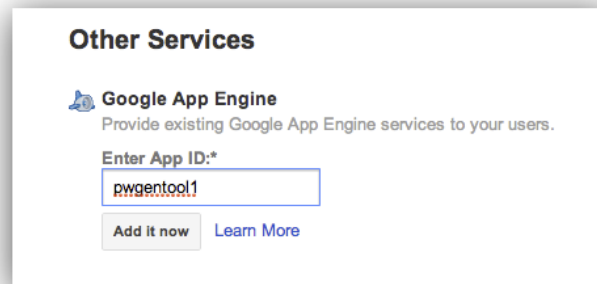


Click Add services to  
your domain






Enter the App Engine ID and click Add it now



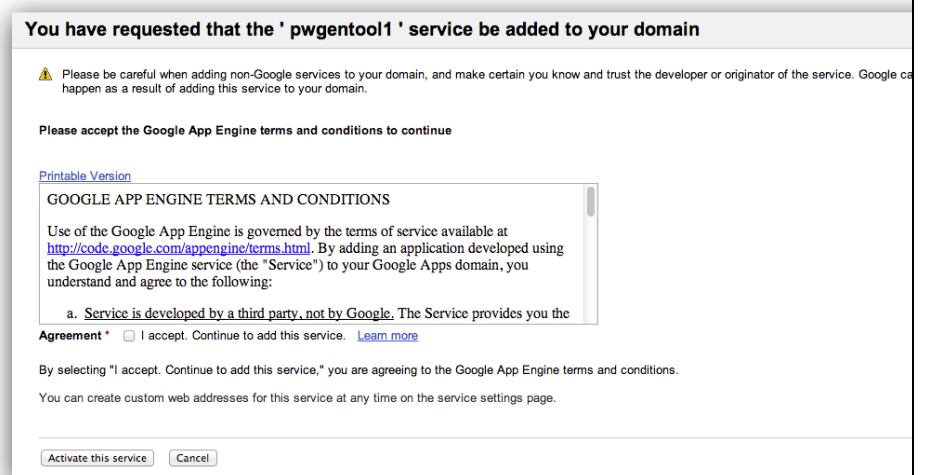
**Other Services**

 **Google App Engine**  
Provide existing Google App Engine services to your users.


Enter App ID:\*

[Learn More](#)

Click I accept and Active this service



**You have requested that the ' pwgentool1 ' service be added to your domain**

 Please be careful when adding non-Google services to your domain, and make certain you know and trust the developer or originator of the service. Google can't control what happens as a result of adding this service to your domain.

Please accept the Google App Engine terms and conditions to continue

[Printable Version](#)

GOOGLE APP ENGINE TERMS AND CONDITIONS

Use of the Google App Engine is governed by the terms of service available at <http://code.google.com/appengine/terms.html>. By adding an application developed using the Google App Engine service (the "Service") to your Google Apps domain, you understand and agree to the following:

a. Service is developed by a third party, not by Google. The Service provides you the

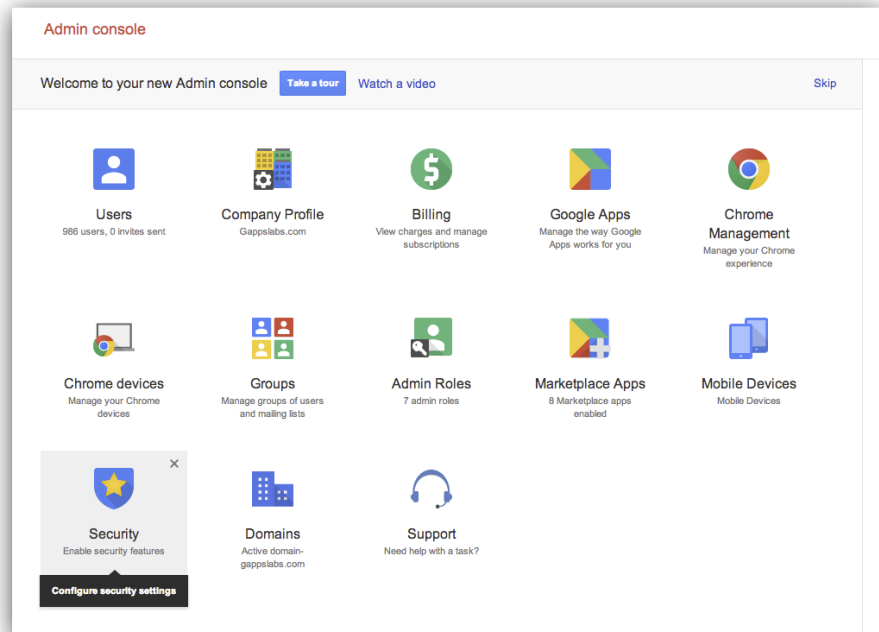
**Agreement \*** ☐ I accept. Continue to add this service. [Learn more](#)

By selecting "I accept. Continue to add this service," you are agreeing to the Google App Engine terms and conditions.

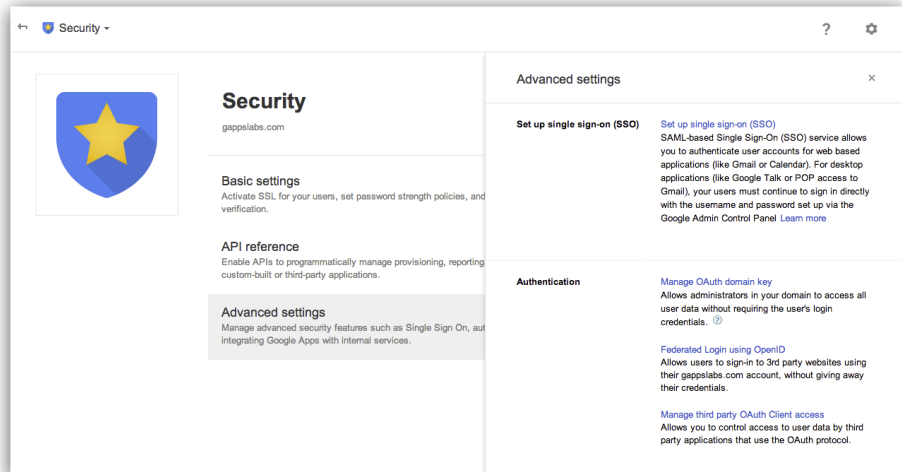
You can create custom web addresses for this service at any time on the service settings page.

## Authorize the App Engine application

In the Google App Admin console select Security



Select Advanced Settings and click on manage third party OAuth Client access

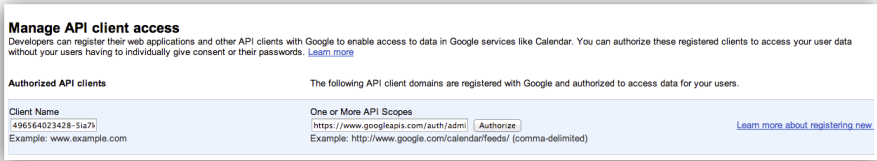
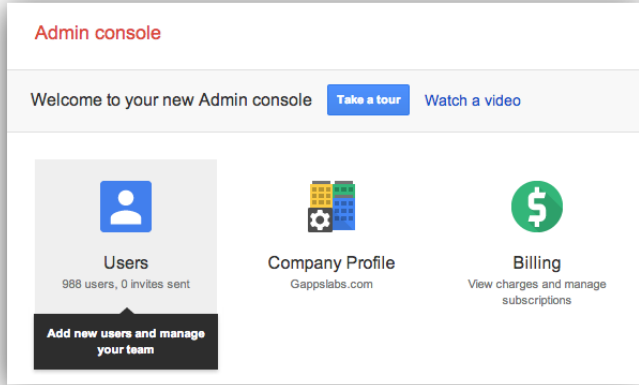

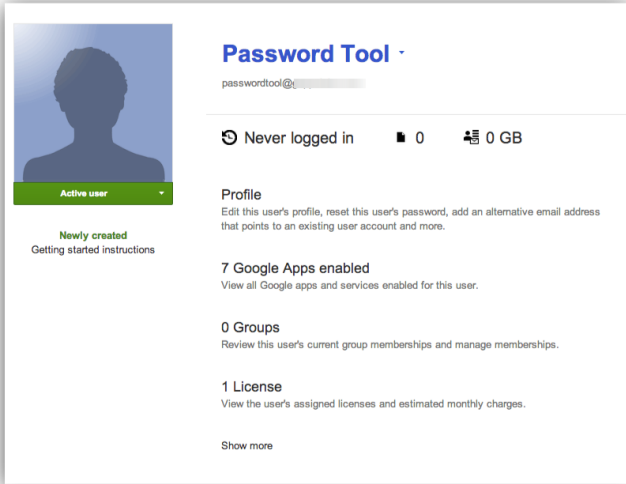


Enter the Client\_ID from the downloaded Certificate JSON file into the Client Name field.

Enter the comma separated API scopes listed below in the API

Client Name: (Client\_ID value from Certificate JSON file)  
496564023428-5ia7km9b4t1plbslkgla2b95ehjrdmnc.apps.googleusercontent.com

Scopes:  
https://www.googleapis.com/auth/admin.directory.group.member.readonly,https://www.googleapis.com/auth/admin.directory.user

<p>Scopes field.</p>	
<p><b>Create Google Apps Admin role account</b></p> <p>In the Google App Admin console select Users</p>	
<p>Add a new user</p>	
<p>Edit the new user, show more details</p>	

Select Admin roles	
Assign the new user the Super Admin role	

## Download and Configure Password Generator



<p><b>Download the Password Generator Source via Git or as a zip file from the project site.</b></p> <p>GitHub Project Site:  <a href="https://github.com/google/googleapps-password-generator">https://github.com/google/googleapps-password-generator</a></p>	<pre>\$ git clone https://github.com/google/googleapps-password-generator.git</pre>
<p><b>Configure the Password Generator source</b></p> <p>In the extracted source directory edit the app.yaml file</p> <p>In the extracted source</p>	<pre>\$ vim app.yaml</pre> <pre>application: {CHANGEME}</pre> <p>Example:</p> <pre>application: pwgentool</pre>

<p>directory edit the app.yaml file/ Change the application name entry to the App Engine identifier you created</p>	
<p><b>Download Password Generator 3rd Party Libraries</b></p> <p>Run the deploy_download.sh script downloads and copies all dependency items into the proper directory locations. For a list of all <a href="#">3rd Party Libraries</a> please refer to the appendix section.</p>	<p>From within the setup sub-directory:</p> <pre>\$ cd ~/pwgtool \$ \$ ./download_dependencies.sh</pre>
<p><b>Convert the service account Certificate - PKCS12 to PEM</b></p> <p>Run openssl to convert the PKCS12 certificate file to PEM format</p>	<pre>openssl pkcs12 -in 725ff72c089fe8a8149a5c3cfb8e21b1a0e605f5-privat ekey.p12 -out privatekey.pem -nodes -nocerts</pre> <p>Enter the password: notasecret</p>
<p>Edit privatekey.pem to remove non-essential details:</p> <p>Delete the Bag and Key Attributes lines</p>	<pre>\$ vim privatekey.pem</pre> <pre>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: &lt;No Attributes&gt; -----BEGIN PRIVATE KEY----- MIICdwIB.....</pre>

	<pre>-----END PRIVATE KEY-----</pre>
Cleanup the json and .p12 file	<pre>\$ rm client_secret_496564023428-2vl2prl8lupuercm4ls2 ipsu203i6nbf.apps.googleusercontent.com.json  \$ rm 25ff72c089fe8a8149a5c3cfb8e21b1a0e605f5-private key.p12</pre>

## Deploy the App Engine Application

<b>Download Google App Engine SDK</b>  <a href="https://developers.google.com/AppEngine/downloads">https://developers.google.com/AppEngine/downloads</a>	<pre>\$ curl http://googleApp Engine.googlecode.com/files/google_appengine_1 .9.1.zip &gt;google_appengine_1.9.1.zip  \$ unzip google_appengine_1.9.1.zip</pre>
<b>Deploy the default state of the App Engine app</b>	<pre>\$ ./appcfg.py --oauth2 update ~/pwgtool/ or \$ /appcfg.py --oauth2 --noauth_local_webserver update ~/pwgtool/</pre>

	 <p><b>Google App Engine appcfig</b></p> <p>This app would like to:</p> <div>  <p>View and manage your applications deployed on Google App Engine</p> </div> <p>Google App Engine appcfig and Google will use this information in accordance with their respective terms of service and privacy policies.</p> <div> <span>Cancel</span> <span>Accept</span> </div> <pre> Authentication successful. 01:53 PM Scanning files on local disk. 01:53 PM Cloning 4 application files. 01:53 PM Uploading 1 files and blobs. 01:53 PM Uploaded 1 files and blobs 01:53 PM Compilation starting. 01:53 PM Compilation completed. 01:53 PM Starting deployment. 01:53 PM Checking if deployment succeeded. 01:53 PM Deployment successful. 01:53 PM Checking if updated app version is serving. 01:53 PM Completed update of app: pwgentool, version: 1 01:53 PM Uploading index definitions. </pre>
--	--

## Configure the Application Admin Setting

<p>In a browser access the App Engine app admin settings page.</p>	<pre>https://&lt;appname&gt;.appspot.com/admin</pre> <p>In this example we used the name pwgentool so the URL for the App Engine application /Admin page is:  <a href="https://pwgentool.appspot.com/admin">https://pwgentool.appspot.com/admin</a></p> <p><i>To login to this page you must be a Google Apps admin in the Google Apps domain you registered the App Engine application.</i></p> <p><i>There is also an App Engine admin which can be different from the Google Apps domain admin account. The App Engine admin account is the owner of the Password Generator App Engine</i></p>
--	---

	<p><i>service. The App Engine admin can be configured in the App Engine Console under Administration &gt; Permissions</i></p>
Configure the Admin settings.	<ul style="list-style-type: none"> <li>• Configure the Service Account based on EMail address associated with the configured OAuth Key.  Example: <a href="mailto:659485895695@developer.gserviceaccount.com">659485895695@developer.gserviceaccount.com</a></li> <li>• Configure the Private Key filename  Example: <code>privatekey.pem</code></li> <li>• Configure the Domain Admin Account  Example: <a href="mailto:passwordtool@domain.com">passwordtool@domain.com</a></li> <li>• Configure the Google Group for access control to the tool:  Example: <code>passwordtool_users@domain.com</code>  NOTE: You need to create this Google Group in the Google Apps domain and add the users of the domain you want into the group so they will have access to the Password Generator tool.</li> <li>• Review all other settings and message text and adjust them to meet your needs.</li> <li>• Save the settings</li> </ul>



## Appendix:

### 3rd Party Libraries

- [Google API Client](#): [Download URL](#) : License [Apache License 2.0](#)
- [Google OAuth2 Client](#): [Download URL](#) : License [Apache License 2.0](#)
- [HttpLib2](#) : [Download URL](#) : License [MIT License](#)
- [Noty](#): [Download URL](#) : License [MIT License](#)
- [jQuery Core 1.10.2](#): [Download URL](#) : License [MIT License](#)
- [jQuery MAP 1.10.2](#): [Download URL](#) License [MIT License](#)
- [WTFForms](#): [Download URL](#) : [BSD License](#)
- [URITemplate](#): [Download URL](#) : [License: Apache v2.0](#)