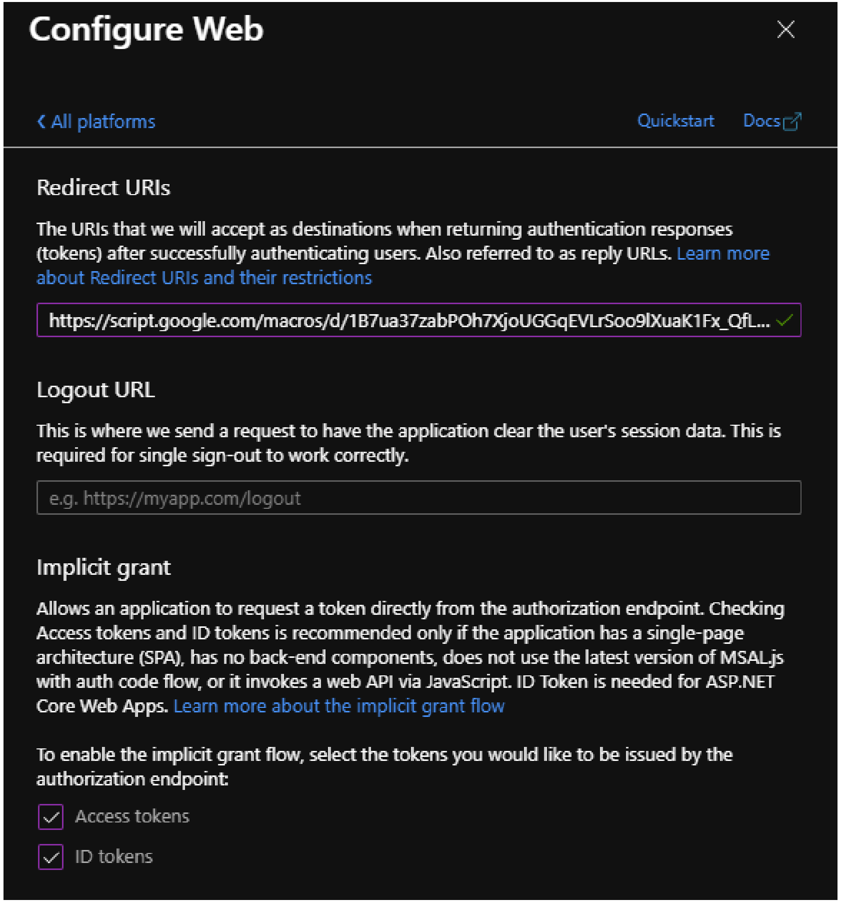
How to deploy the G suite add-on

You need to create an azure app registration for the add-on to be able to access your Microsoft account

Link to Microsoft docs: https://docs.microsoft.com/en-us/azure/active-directory/develop/quickstart-register-app

1. Log into portal.azure.com and go to App registrations https://portal.azure.com/#blade/Microsoft_AAD_RegisteredApps/ApplicationsListBlade
2. Select new Registration, give it a name and click register at the bottom of the page
3. Go to the Authentication tab and click add a platform, a flyout will open from the side of the page
4. Enter https://script.google.com/macros/d/{SCRIPT ID}/usercallback as the Redirect URI, check the Access tokens and ID tokens check boxes and click configure at the bottom of the page
    a. The id used for the redirect uri is the Script ID from project properties of the add on. Replace {SCRIPT ID} with the Script ID from your google project
    b. The script Id can be found by going to file => project properties from the google project



Configure Web

Redirect URIs

The URIs that we will accept as destinations when returning authentication responses (tokens) after successfully authenticating users. Also referred to as reply URLs. Learn more about Redirect URIs and their restrictions

https://script.google.com/macros/d/1B7ua37zabPOh7XjoUGGqEVLrSoo9lXuaK1Fx_QfL... ✓

Logout URL

This is where we send a request to have the application clear the user's session data. This is required for single sign-out to work correctly.

e.g. https://myapp.com/logout

Implicit grant

Allows an application to request a token directly from the authorization endpoint. Checking Access tokens and ID tokens is recommended only if the application has a single-page architecture (SPA), has no back-end components, does not use the latest version of MSAL.js with auth code flow, or it invokes a web API via JavaScript. ID Token is needed for ASP.NET Core Web Apps. Learn more about the implicit grant flow

To enable the implicit grant flow, select the tokens you would like to be issued by the authorization endpoint:

☑ Access tokens

☑ ID tokens

5. Go to the API permissions tab, click add permission, select Microsoft Graph, select Delegated permissions, search for and add the following permissions
    a. offline_access
    b. OnlineMeetings.ReadWrite

    c. openid
    d. profile
    e. User.Read
6. Go to the Certificates & secrets tab, click the New client secret button, give the secret a description, select how long until it expires, then click add
7. Copy the secret as we will need it for step 8
8. In the add on go to the Authentication file and in the get0AuthService function do the following
    a. .createService (‘give the service a name, Recommended to be the same as app registration’)
    b. .setClentId(‘set this to the application (client) ID from the appRegistration’)
    c. .setClientSecret(‘paste the secret from step 7 here’)
9. Save the add-on so the changes will not be lost

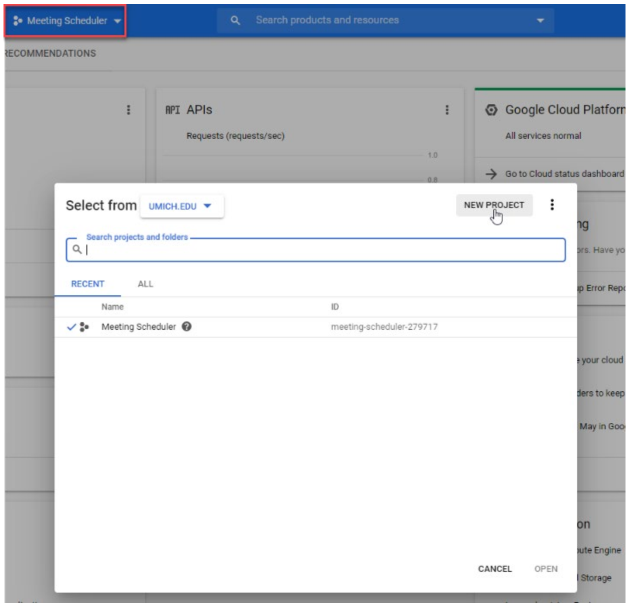
Next will be the steps on how to deploy the add-on to the g suite marketplace

Since this will be a private app you do not need to do the steps that get you ready to do an add-on review

Link to google docs: https://developers.google.com/gsuite/add-ons/how-tos/publishing-gsuite-addons

1. Create a google cloud platform account that you can create and host a project in
    a. Will be able to go to https://console.cloud.google.com/ when this is complete

2. Select the projects dropdown from the top of the page then click the new project button

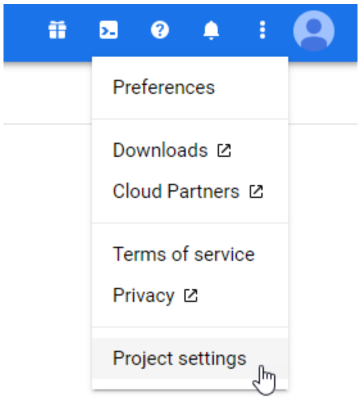


Name	ID
Meeting Scheduler	meeting-scheduler-279717

3. Give the project a name and click create. *Note: that the name cannot be changed later.

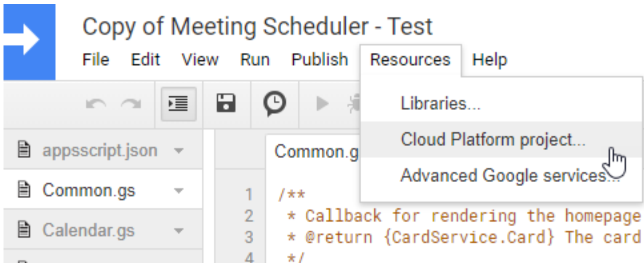
4. Once the project is created go to that project’s dashboard either from the notification or by selecting it from the project selector where you clicked create new project

5. In order to convert the G Suite project to the Google Cloud Platform project click the vertical ellipsis on the top right of the Google Cloud Platform header and select Project settings



6. Copy the Project number and open the G Suite project you wish to use from https://script.google.com/home

7. Select Resources => Cloud Platform project…, paste the Project number in the Change Project text box and click the set project button



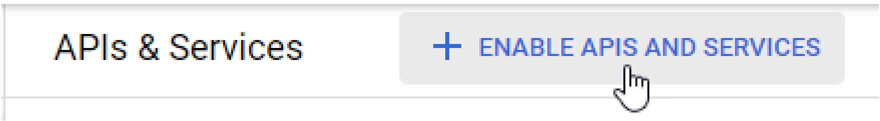
```
/**
 * Callback for rendering the homepage
 * @return {CardService.Card} The card
 */
```

8. Go back to https://console.cloud.google.com/ Open the hamburger menu on the left side and open “APIs & Services => OAuth consent screen”

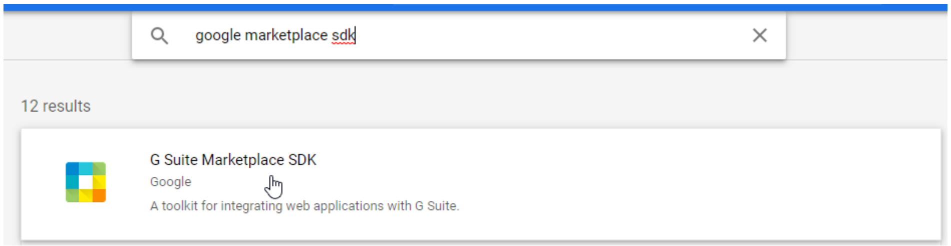
9. Select the user type of Internal and select create. *Note: this cannot be changed and if you select External you must pass an app review by google.

10. Fill out the OAuth consent screen and click save. *Note: minimum required info is Application name

11. Go to “APIs & Services => Dashboard” and click the Enable APIs and Services button at the top of the page



12. Search for and enable the G Suite Marketplace SDK

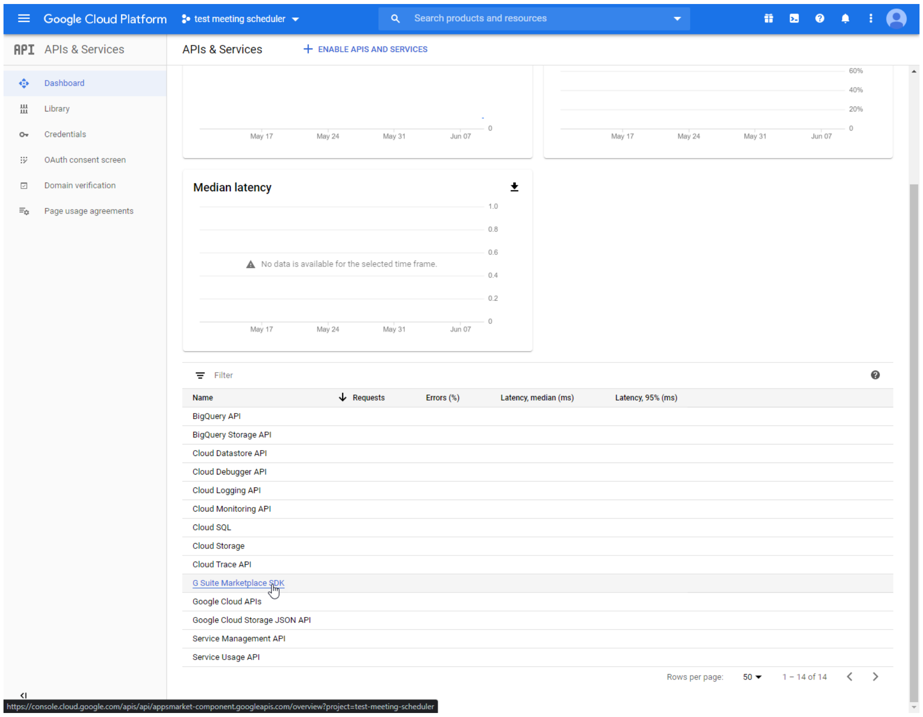


google marketplace sdk

12 results

G Suite Marketplace SDK
Google
A toolkit for integrating web applications with G Suite.

13. When you enable the SDK it will bring you to the G Suite Marketplace SDK overview screen, if you need to get back to the overview screen you can go to the APIs & Services => Dashboard and select the marketplace SDK from the list below



No data is available for the selected time frame.

Name	Requests	Errors (%)	Latency, median (ms)	Latency, 95% (ms)
BigQuery API				
BigQuery Storage API				
Cloud Datastore API				
Cloud Debugger API				
Cloud Logging API				
Cloud Monitoring API				
Cloud SQL				
Cloud Storage				
Cloud Trace API				
G Suite Marketplace SDK				
Google Cloud APIs				
Google Cloud Storage JSON API				
Service Management API				
Service Usage API				

Rows per page: 50 1 – 14 of 14

https://console.cloud.google.com/apis/api/appsmarket-component.googleapis.com/overview?project=test-meeting-scheduler

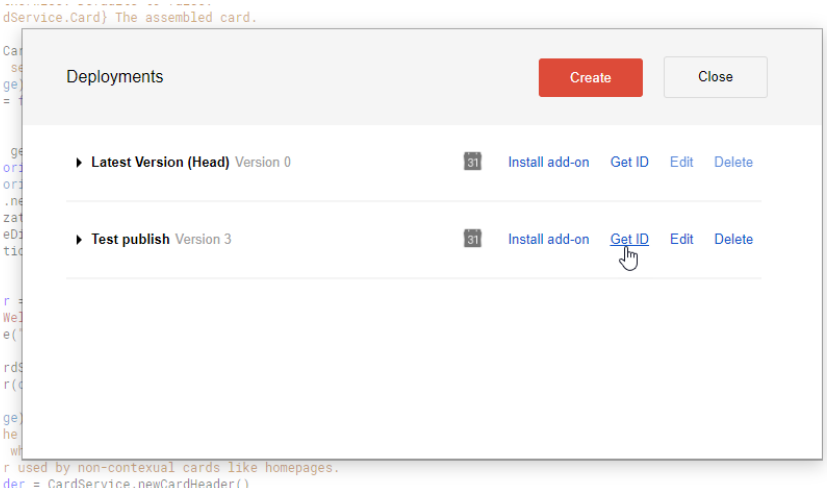
14. From the G Suite Marketplace SDK overview screen go to the Configuration tab and fill out all the required fields
    a. Under extensions the only box you need to check is the G Suite Add-ons Extension
    b. Steps 15 -17 will explain how to get the deployment ID to paste here

    c. It is recommended that you set the visibility to My Domain so that only people in your domain can see the app

15. Go back to the project in https://script.google.com/home and select publish from the top bar and go to deploy form manifest

16. Click the create button at the top of the modal that opened, give the deployment a name and click save

17. Now that you have a deployment created, select Get ID on the deployment you just created and copy and paste the deployment ID in the field from step 14 *Note don't use Latest Version as it may not always be stable for deployment


dService.Card} The assembled card.
Deployments
Create
Close
Latest Version (Head) Version 0
Install add-on
Get ID
Edit
Delete
Test publish Version 3
Install add-on
Get ID
Edit
Delete
r used by non-contexual cards like homepages.
der = CardService.newCardHeader()

18. Once you have successfully saved the Configuration settings go to the publish tab for the G Suite Marketplace SDK

19. Fill out the required fields and select publish at the bottom of the page.
    a. You will now be able to find the app in the g suite marketplace either by searching for it or by going to a direct link if you checked the Unlist Application checkbox
    b. You may need an account admin to install the add-on depending on organization rules