**How to Install Nodemailer**

Install nodemailer using the following command:

npm install nodemailer

Nodemailer’s API is pretty simple and requires us to do the following:

1. Create a **Transporter**object
2. Create a **MailOptions**Object
3. Use the **Transporter.sendMail** method

To create a transporter object, we do the following:

let transporter = nodemailer.createTransport({

service: 'gmail',

auth: {

type: 'OAuth2',

user: process.env.MAIL\_USERNAME,

pass: process.env.MAIL\_PASSWORD,

clientId: process.env.OAUTH\_CLIENTID,

clientSecret: process.env.OAUTH\_CLIENT\_SECRET,

refreshToken: process.env.OAUTH\_REFRESH\_TOKEN

}

});

✋ Pay attention, as apart from the user and the pass keys, which are your own credentials for your gmail account, the other three keys need to be retrieved after setting up OAuth.

As we stated in the beginning of this article, we will be using Gmail for our mail sending needs. As you may have guessed, Gmail has a high level of security when it comes to mail sent by/to a user’s account.

There are a number of ways we can overcome this obstacle (some better than others), and we will choose the one that requires us to set up a project in the **Google Cloud Platform**. We need to do that in order to have credentials for the OAuth security enabled by Gmail.

If you want to read more about the complexities of using Gmail with nodemailer, go [here](https://nodemailer.com/usage/using-gmail/).

The next steps will require some configurations instead of coding, so brace yourselves.

**Google Cloud Platform Configurations**

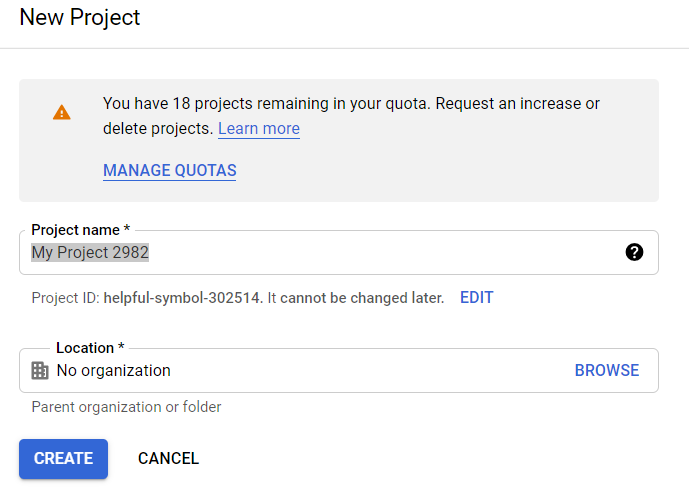
If you don’t have a [Google Cloud Platform](https://console.cloud.google.com/home) account, be sure to set one up as a prerequisite. Once you have that set up, create a new project by clicking on the dropdown menu in the upper left corner.

1_a4fnFLNMoTtLJuqsKilVnA

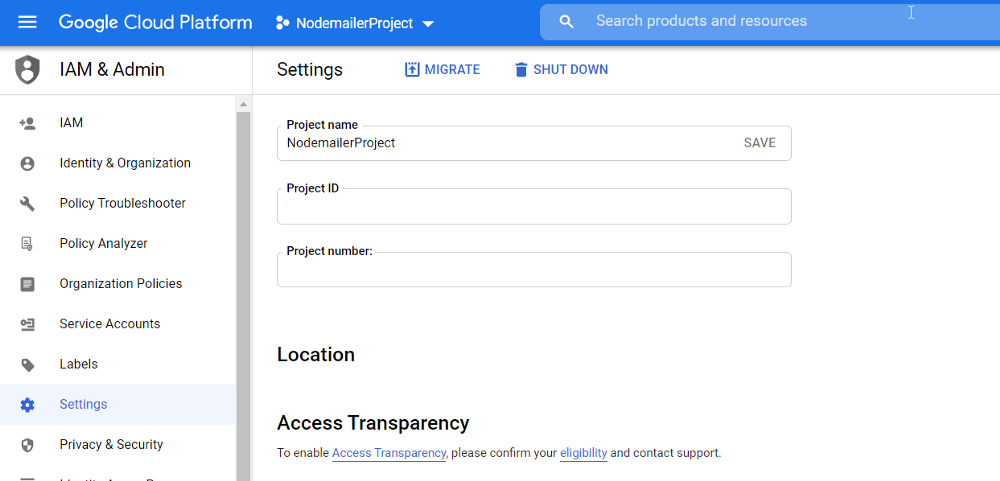
Select the New Project option:

1_HNwUG3wPdbrwc3JB5D7_tg

In the next window, we will have to give our project a name. Pick whatever you like, but we will continue with our **NodemailerProject** name. For the location property, you can leave it as No organization.



It may take a few seconds to for the project to be set up, but after that you will be able to see this screen:

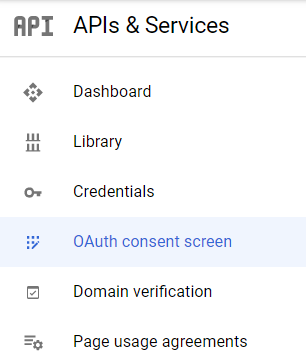


Open up the navigation menu by clicking the three dashed lines in the top left corner and select **APIs and Services:**

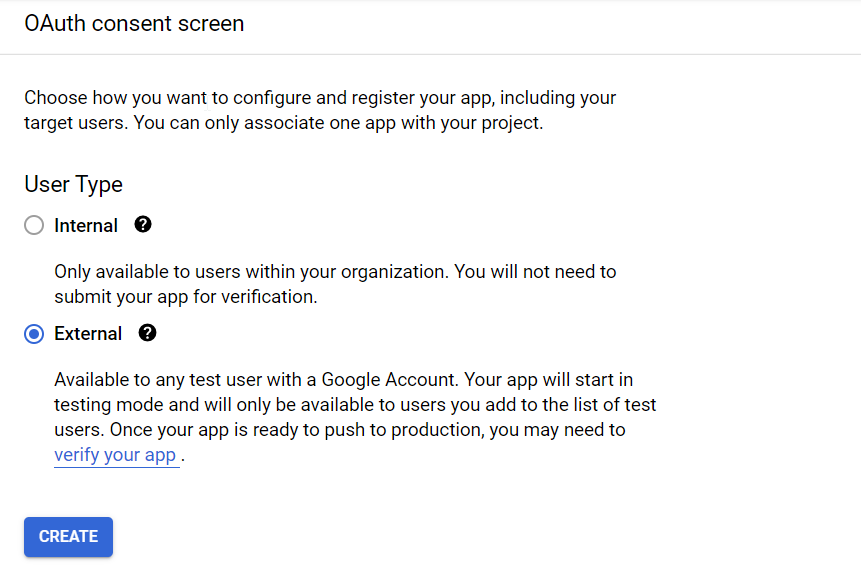
1_qPaPpPadHQLdKCQbhjND7Q

In order to be able to use Nodemailer and Gmail we will have to use OAuth2. If you aren’t familiar with OAuth, it is a protocol for authentication. I won’t get into the specifics here as it is not necessary, but if you want to understand more about it, go [here](https://oauth.net/2/).

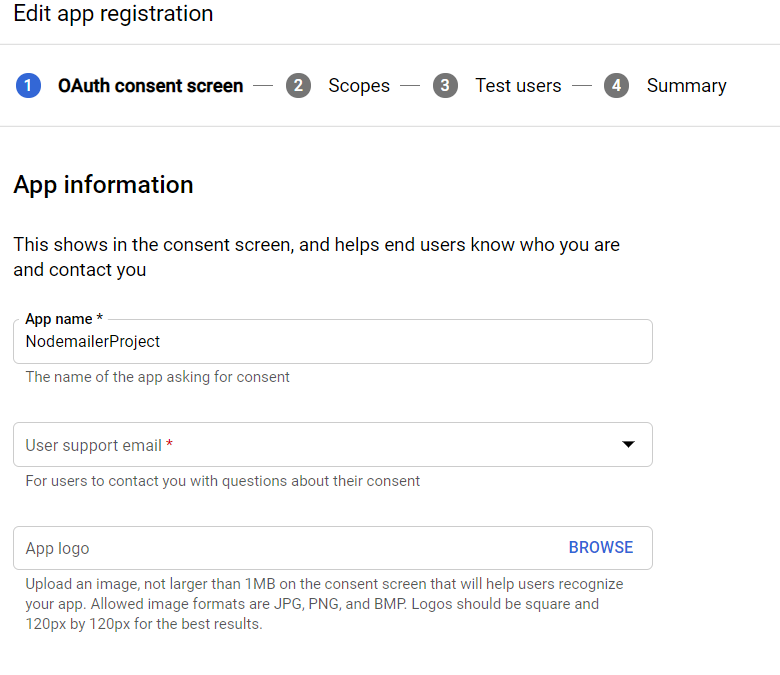
First we will have to configure our OAuth Consent Screen:



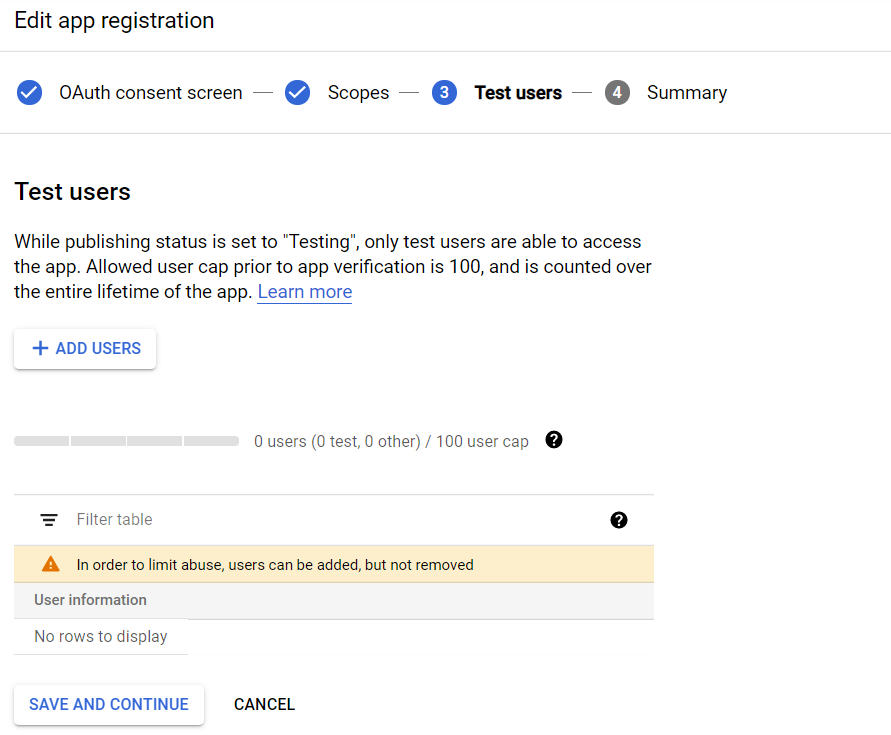
If you are not a G-Suite member, the only option available will be External for User Type.



After clicking create, the next screen requires us to fill out the application’s information (our server):



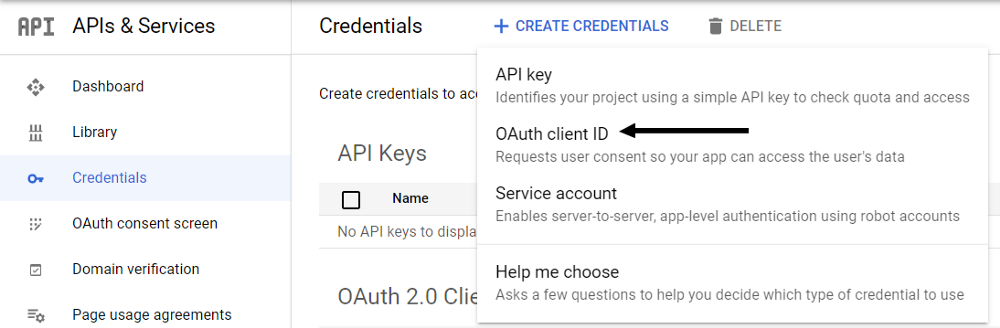
Fill in your email in the User support email field and also in the Developer contact information field. Clicking Save and Continue will bring us to the Scopes phase of this configuration. Skip this phase, as it is not relevant for us, and head into the Test Users phase.



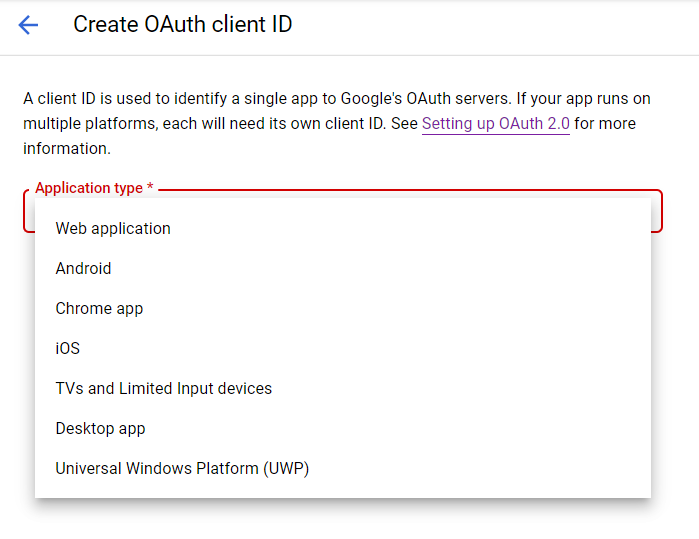
Here, add yourself as a user and click Save and continue.

**How to Configure Your OAuth Settings**

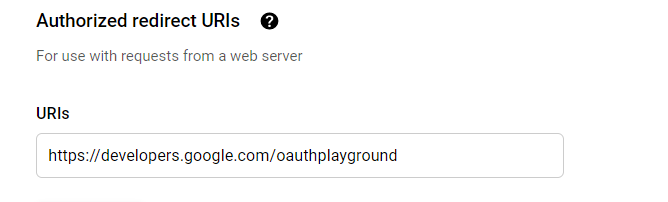
In this phase will we create OAuth credentials to be used with Nodemailer. Head over to the Credentials tab above OAuth Consent Screen. Click on the plus (➕) sign that has the text **Create Credentials**and choose OAuth Client ID.



In the Application type dropdown menu, choose **Web Application**:



In the **Authorized Redirect URIs** section, make sure to add OAuth2 Playground ([https://developers.google.com/oauthplayground](https://developers.google.com/oauthplayground/)) as we will use it to get one of the keys that was mentioned in the beginning of this article.

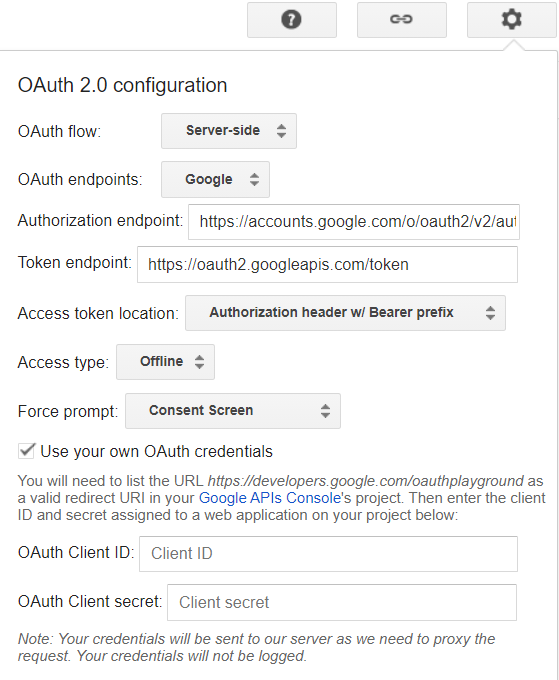


After clicking create, you will be presented with your client id and client secret. **Keep these to yourself and never expose them in any way, shape, or form**.

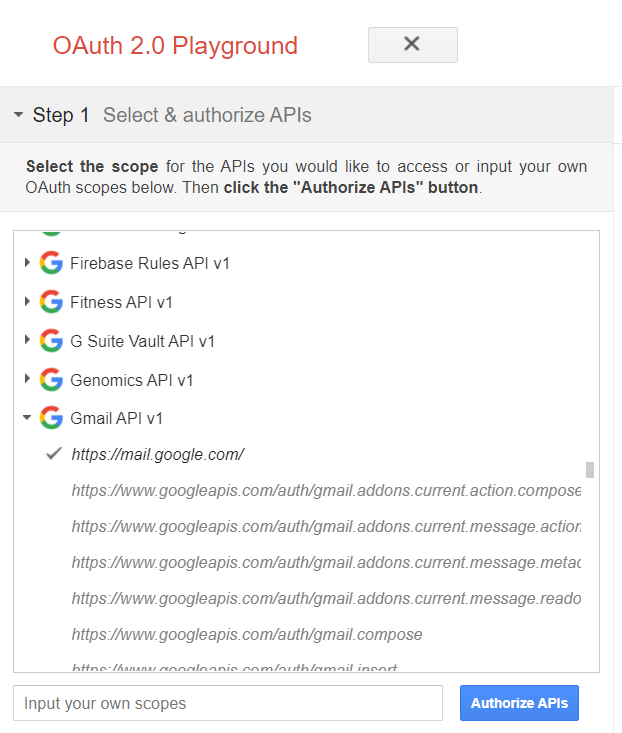
**Get Your OAuth Refresh Token**

To get the refresh token, which we will use within the transporter object in Nodemailer, we need to head over to the OAuth2 Playground. We approved this URI for this specific purpose in an earlier stage.

1. Click on the gear icon to the right (which is OAuth2 Configuration) and check the checkbox to use your own OAuth2 Credentials:



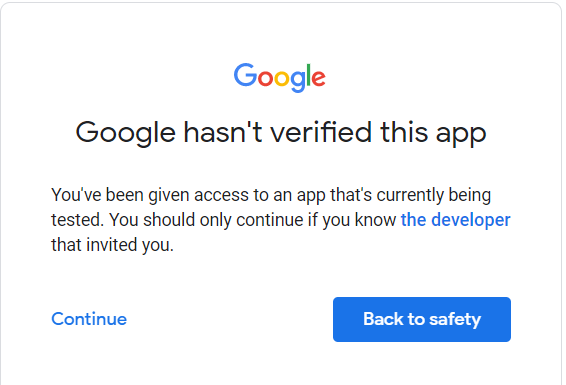
2. Look over to the left side of the website and you will see a list of services. Scroll down until you see Gmail API v1.



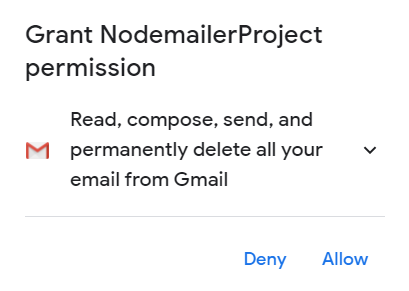
3. Click **Authorize APIs**

You will be presented with a screen to login to any of your Gmail accounts. Choose the one you listed as a Test user.

4. The next screen will let you know that Google still hasn’t verified this application, but this is ok since we haven’t submitted it for verification. Click continue.



5. In the next screen, you will be asked to grant permission to your project to interact with your gmail account. Do so.



6. Once that is done, you will be redirected back to the OAuth Playground and you can see that there is an authorization code in the menu to the left. Click on the blue button labelled **Exchange authorization code for tokens**.

The fields for the refresh token and the access token will now be filled.