


COSC 4P02 Web Summarizer - Setup Guide

Web Summarizer

[Login](#) [Pro](#)

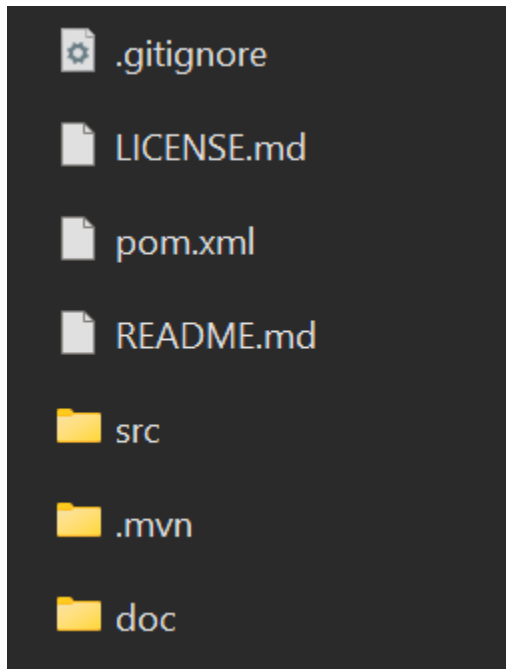


What would you like to summarize?

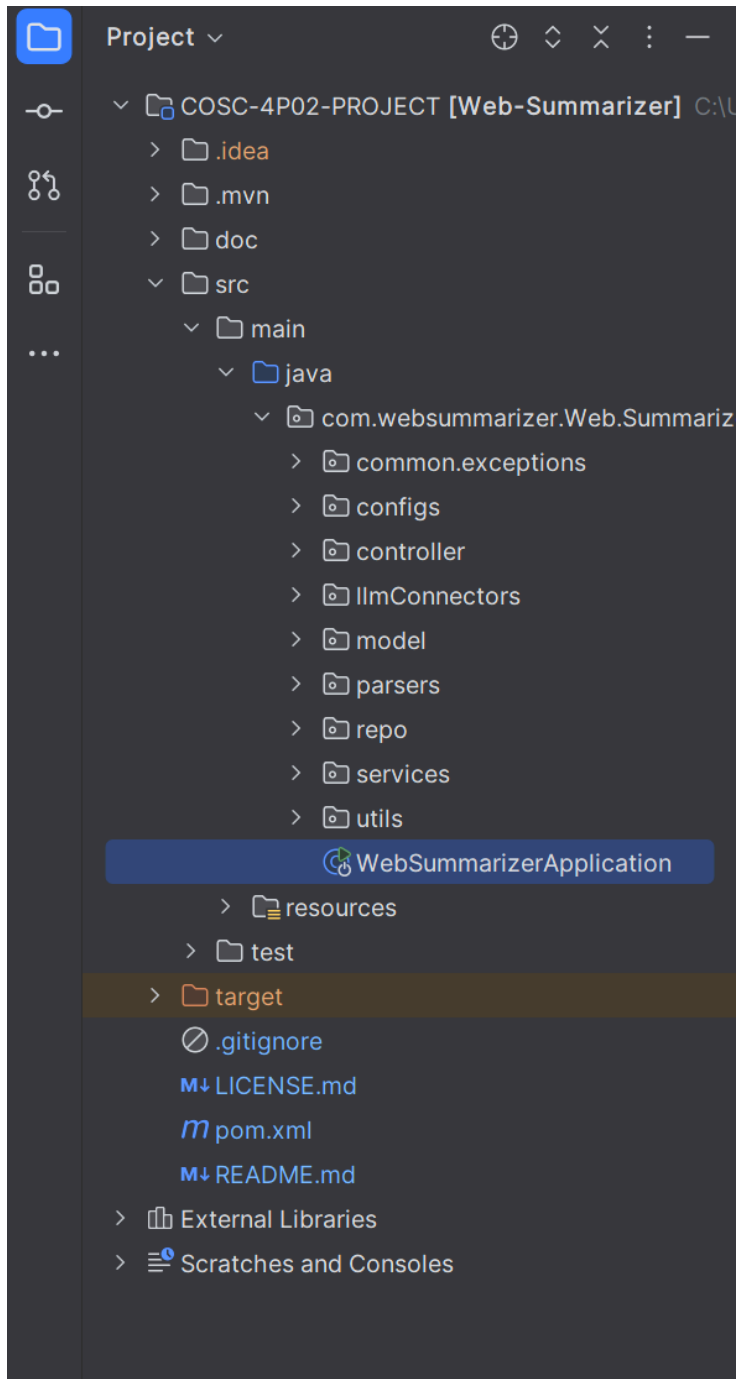
Summarize

Running the program

1. Download the source code (ZIP Folder) and extract it. The directory should look like this:



2. Open the extracted copy of the program in your preferred choice of IDE (Ex: IntelliJ). The Java file needed to run the overall program is "WebSummarizerApplication" which is found here:



3. Run WebSummarizerApplication.
(IMPORTANT: See *Setting Up Database* if this is your first time running the program)

Setting Up Database

This program uses PostgreSQL as its database to store user accounts and chat/summarization histories. To install it, click on this link and choose the version meant for your operating system:

<https://www.enterprisedb.com/downloads/postgres-postgresql-downloads>

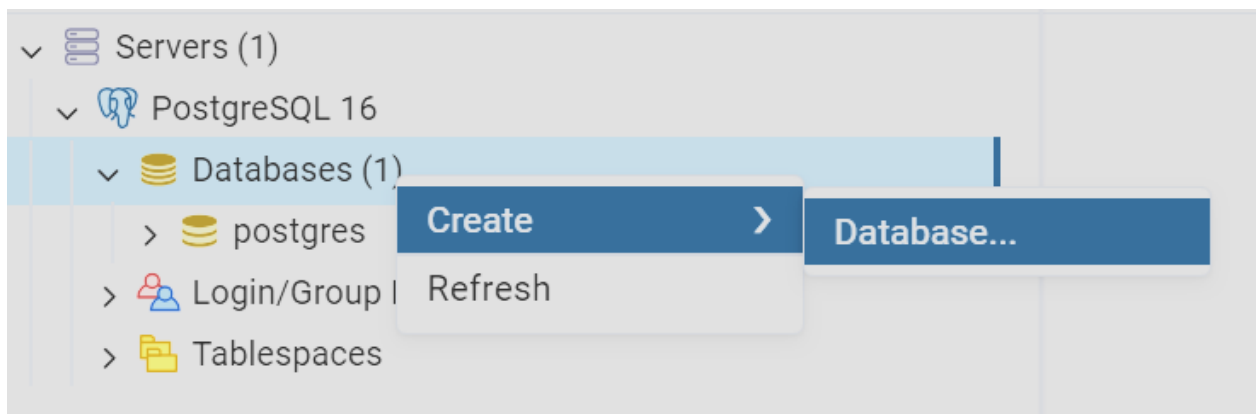
This installs PostgreSQL on your system along with pgAdmin4.

Run the installer you downloaded while also keeping a note of the following information:

- PostgreSQL Username and Password
- Port Number (Default is 5432. This program assumes you are using the default port number)

Once you have everything setup, follow these steps to connect the program to the database:

1. Open pgAdmin4 and enter your credentials if prompted
2. Right-click Databases and click Create:



3. Name the new database "web_summarizer_uat" and click save:

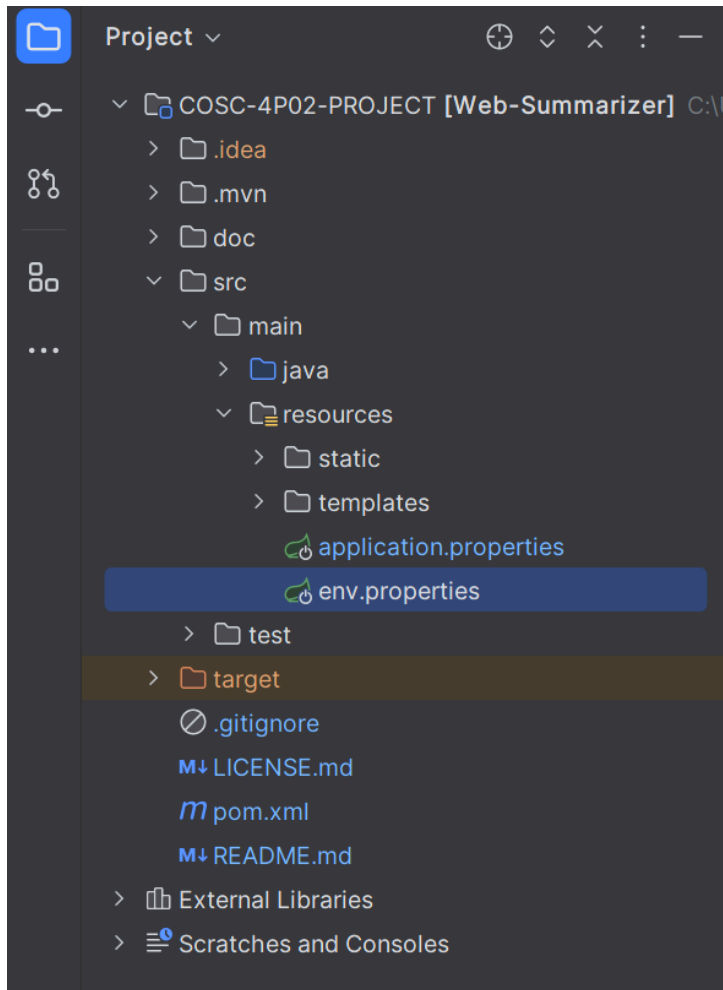
The screenshot shows the 'Create - Database' dialog box in pgAdmin4. The 'General' tab is active, displaying the following fields:

- Database:** web_summarizer_uat
- OID:** (empty)
- Owner:** postgres
- Comment:** (empty)

At the bottom of the dialog, there are three buttons: 'Close', 'Reset', and 'Save'.

Feel free to exit pgAdmin4 at this point.

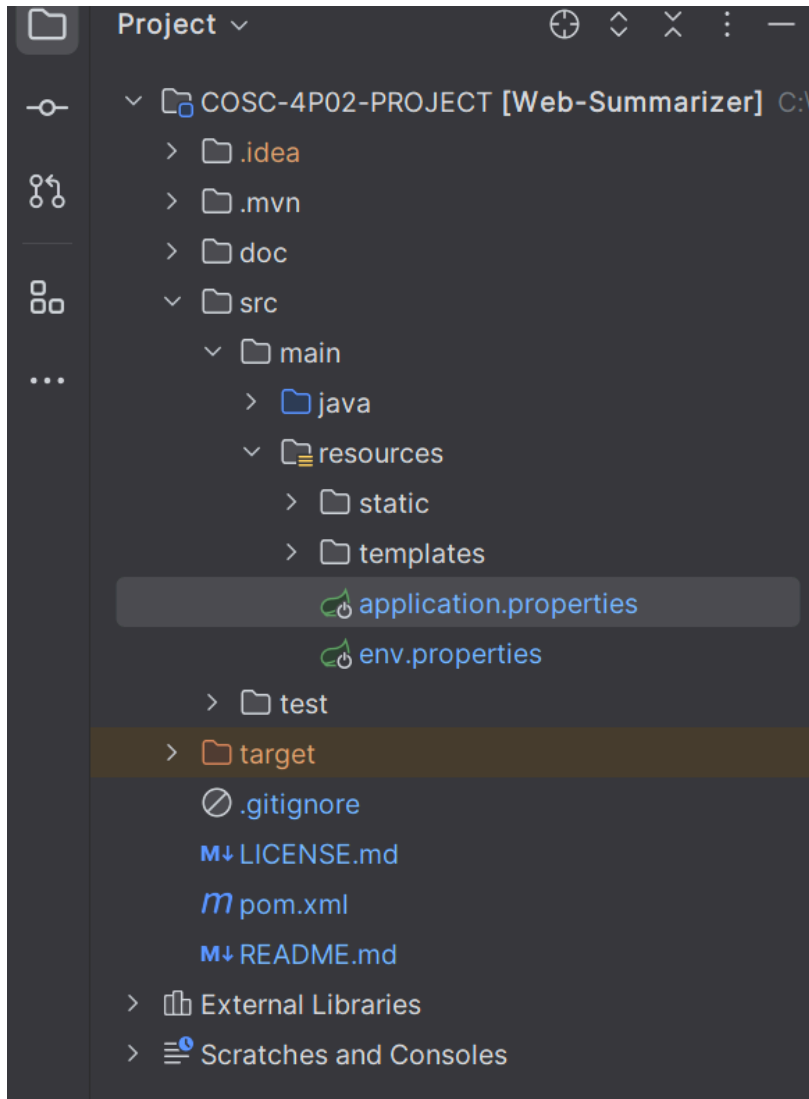
4. Open the program folder in your preferred choice of IDE and open ***env.properties***:



5. Edit the values "DB_USERNAME_UAT" and "DB_PASSWORD_UAT" with your PostgreSQL credentials:

```
1  #API CREDENTIALS:
2  #BART
3  API_AUTH_TOKEN = ##
4  #OPEN AI
5  API_AUTH_TOKEN_OPEN_AI = ##
6
7  #DB CREDENTIALS
8  DB_USERNAME_UAT = postgres
9  DB_PASSWORD_UAT = 123
10
11 #GITHUB CREDENTIALS
12 GITHUB_CLIENT_ID = ##
13 GITHUB_CLIENT_SECRET = ##
14
15 #GOOGLE CREDENTIALS
16 GOOGLE_CLIENT_ID = ##
17 GOOGLE_CLIENT_SECRET = ##
18
19 #FACEBOOK CREDENTIAL
20 FACEBOOK_CLIENT_ID = ##
21 FACEBOOK_CLIENT_SECRET = ##
22
23 #SMTP CREDENTIALS GMAIL
24 SMTP_USERNAME = ##
25 SMTP_PASSWORD = ##
26
27 BITLY_AUTH_TOKEN = ##
```

6. (OPTIONAL: Only do this if you are using a different port number or database)
In the ***application.properties***, you will find the database url.



Look at the section labeled “DATABASE CONNECTION INFO : UAT”:

```
20 #DATABASE CONNECTION INFO : UAT
21 spring.datasource.driver-class-name=org.postgresql.Driver
22 spring.datasource.url=jdbc:postgresql://localhost:5432/web_summarizer_uat
23 spring.datasource.username=${DB_USERNAME_UAT}
24 spring.datasource.password=${DB_PASSWORD_UAT}
25
```

By default, the datasource url is

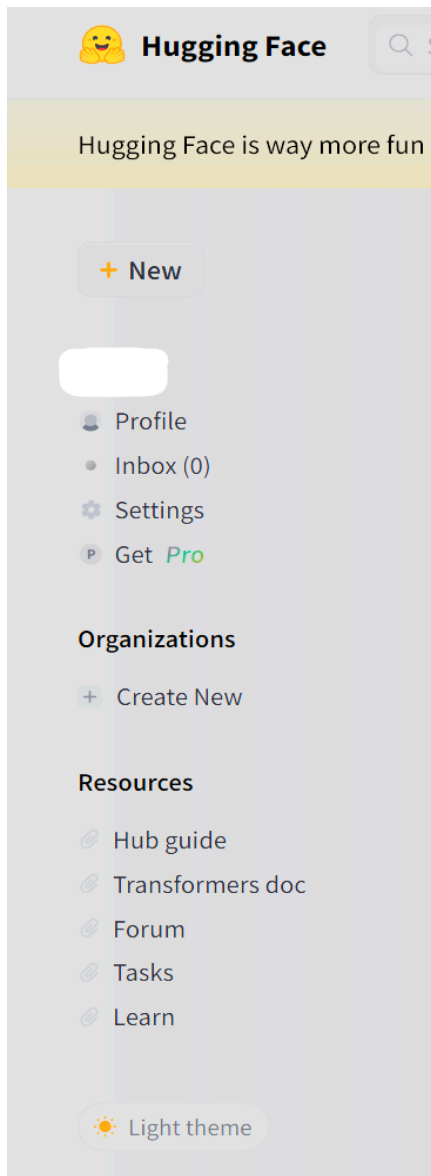
jdbc:postgresql://localhost:5432/web_summarizer_uat

If you are using a different port number, replace 5432 with the new number. If you want to use a different database, change “web_summarizer_uat”.

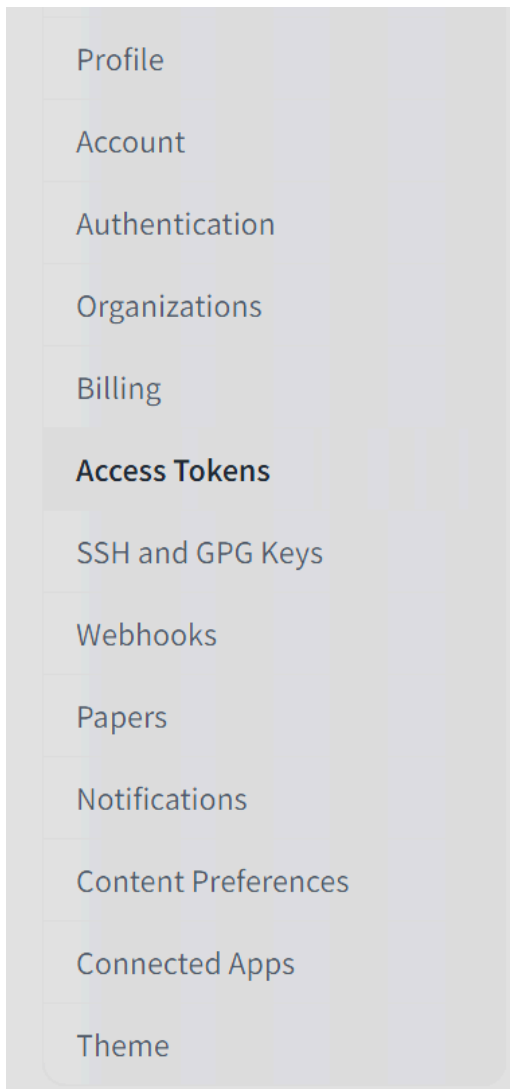
Get HuggingFace API Token

By default, the program uses two providers for the text summarization: HuggingFace and OpenAi. This section goes over how to retrieve and implement your HuggingFace API token. This is needed because the main/default LLM used for the text summarization is HuggingFace's [facebook/bart-large-cnn](https://huggingface.co/facebook/bart-large-cnn) model.

1. Go on the HuggingFace website and create your account: <https://huggingface.co/join>
2. On the left hand side or from click your profile on the top right, click Settings:



3. On the left-hand side, click “Access Tokens” and create a new token (“Type” can just be Read):



4. Copy the token



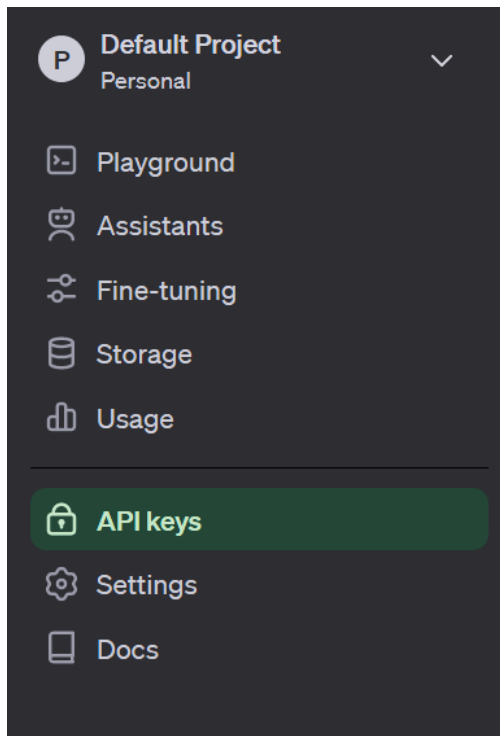
5. In your preferred IDE, open ***env.properties*** and paste the token for API_AUTH_TOKEN

```
1 #API CREDENTIALS:
2 #BART
3 API_AUTH_TOKEN = insert_token_here|
4 #OPEN AI
5 API_AUTH_TOKEN_OPEN_AI = ##
6
```

Get OpenAI API Token

By default, the program uses two providers for the text summarization: HuggingFace and OpenAi. This section goes over how to retrieve and implement your OpenAI API token. This is needed because this program offers an alternative LLM for text summarization as a pro feature.

1. Go to the OpenAI developer platform and create an account:
<https://platform.openai.com/>
2. Once you are logged in, select API keys from the left hand side of the screen:



3. Create a new token (Default settings are sufficient). The token will show up in another window pop-up with the option to copy it:

Save your key

Please save this secret key somewhere safe and accessible. For security reasons, **you won't be able to view it again** through your OpenAI account. If you lose this secret key, you'll need to generate a new one.

Copy

Permissions

Read and write API resources

Done

4. In your preferred IDE, open ***env.properties*** and paste the token for `API_AUTH_TOKEN_OPEN_AI`

```
1 #API CREDENTIALS:
2 #BART
3 API_AUTH_TOKEN = ##
4 #OPEN AI
5 API_AUTH_TOKEN_OPEN_AI = insert_token_here
```

Get SMTP Credentials (Google Account)

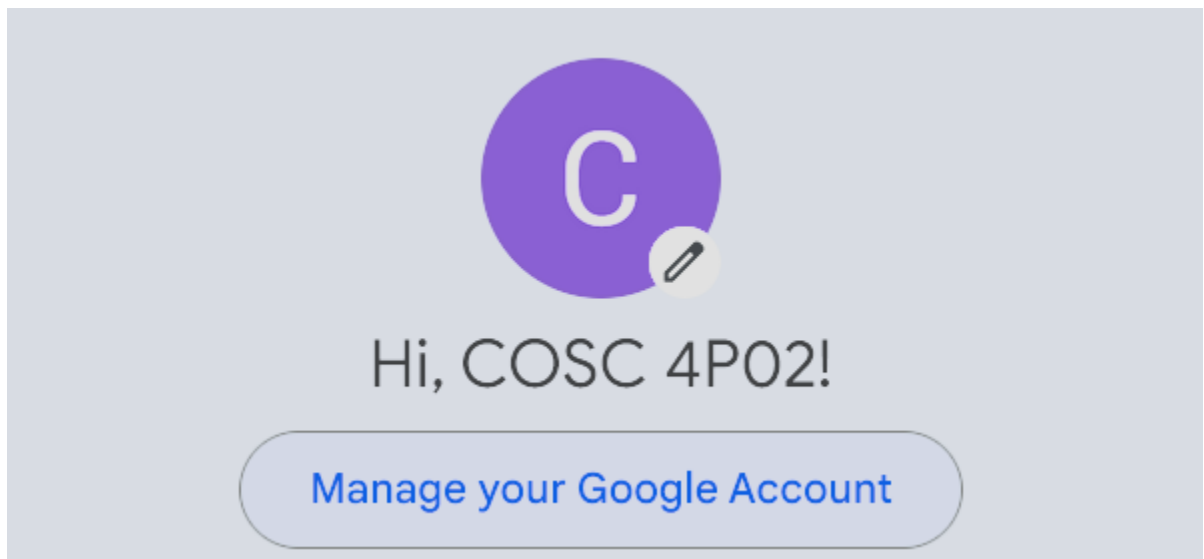
This program utilizes SMTP to allow users to reset their account password if forgotten. While there are many SMTP services you could use, this program was mainly tested using the Gmail SMTP server.

Your Google Account needs to have two-factor authentication enabled. Follow this link if you don't know how to set it up:

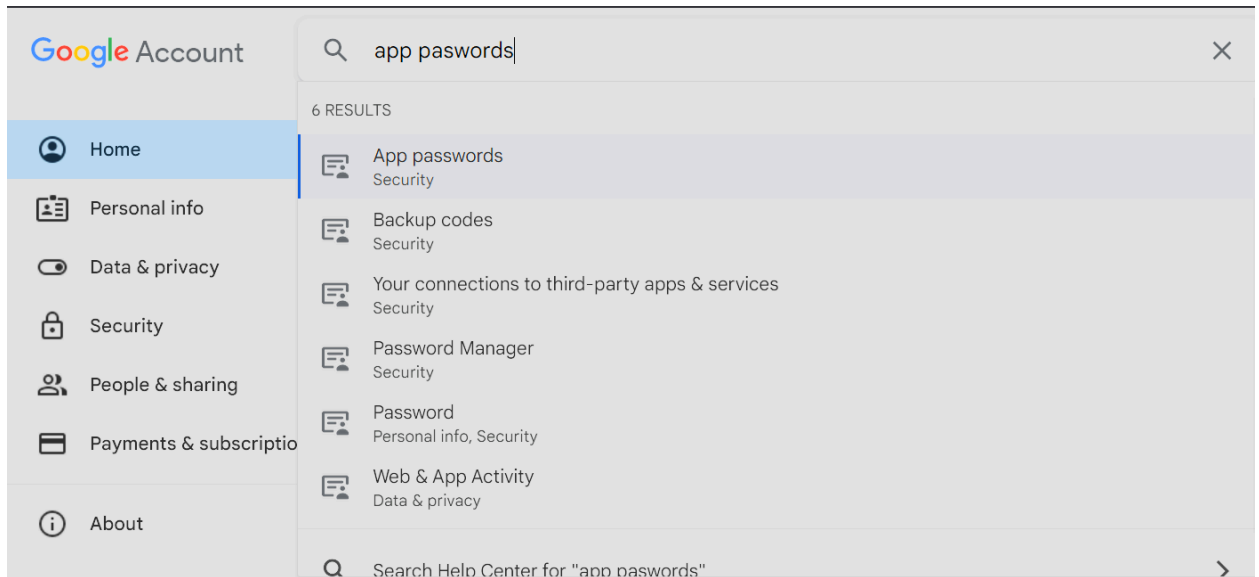
<https://support.google.com/accounts/answer/185839?hl=en&co=GENIE.Platform%3DDesktop>

Here is how to get the credentials:

1. Log into the Google Account you want to use. Once logged in, click your profile and select "Manage your Google Account"

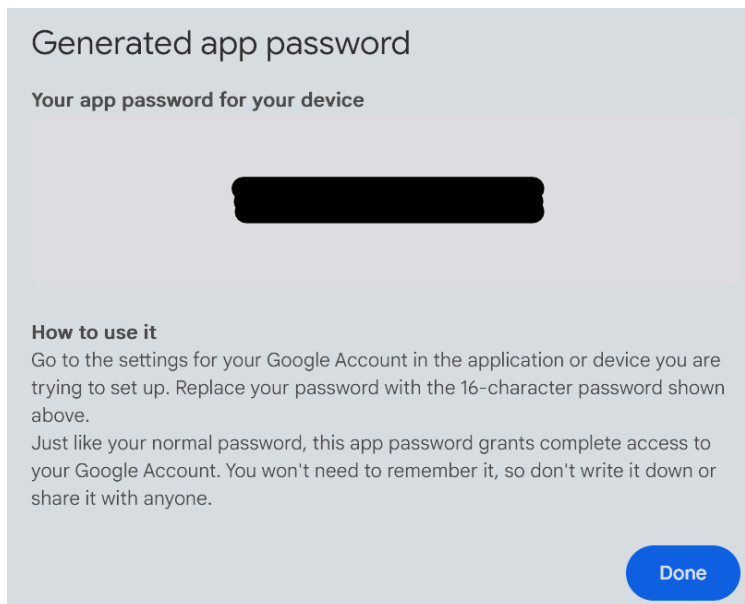


2. In the settings search bar, look up "App passwords" and select it:



If this option does not show up, it is most likely because you have not enabled two-factor authentication.

3. To create a new application to use your SMTP credentials, enter a name. Once you click create, there will be a new pop-up with the app password:



4. In your preferred IDE, open **env.properties**. Find the section labeled “SMTP CREDENTIALS GMAIL” and input your email address and the 16-character password generated (NOT your normal Google Account password):

```
23 #SMTP CREDENTIALS GMAIL
24 SMTP_USERNAME = address@email.com
25 SMTP_PASSWORD = abcd efgh ijkl mnop
```


Get OAuth2 Credentials

As an alternative to account creation, users can opt to create an account and log in using a social login option such as Google and Github. The process to setting it up is a bit more in-depth than the other instructions in this guide, but thankfully there is well-written documentation for these.

The quick summary is that you need to create an OAuth2 application with each provider. A successful application should contain a “client id” and “client secret” which you would input into their respective variables in order to make it work:

```
11      #GITHUB CREDENTIALS
12      GITHUB_CLIENT_ID = ##
13      GITHUB_CLIENT_SECRET = ##
14
15      #GOOGLE CREDENTIALS
16      GOOGLE_CLIENT_ID = ##
17      GOOGLE_CLIENT_SECRET = ##
18
19      #FACEBOOK CREDENTIAL
20      FACEBOOK_CLIENT_ID = ##
21      FACEBOOK_CLIENT_SECRET = ##
```

Below are the instructions for your reference:

Google: <https://support.google.com/cloud/answer/6158849?hl=en>

Github:

<https://docs.github.com/en/apps/creating-github-apps/registering-a-github-app/registering-a-github-app>

(There is an option for Facebook but the current release of the program does not support it. Just for reference, it will also be included here)

Facebook:

<https://developers.facebook.com/docs/development/create-an-app/facebook-login-use-case>