Deploy a Flask App to Render.com

If freezing your app won't work, follow these instructions

Render.com is a site where you can register and deploy web apps in the cloud. With a low-cost shared-hosting account, you have access to a web server, but you don't have access to all aspects of the server. Specifically, you don't have *root access,* and that means you can't install and configure the various libraries and scripts a Python app (for example) needs to run.

Render isn't the only option for cloud services, but it offers a reasonable free tier, which many others do not, and Render makes it fairly easy to get started.

Create a new account at Render.com

Log into your **GitHub** account *first,* and then set up a new **Render** account using your GitHub login. Thereafter, you'll always sign in by clicking a GitHub button at the Render website.

Configure your Python app

Go to the Render Quickstarts page. Inside the card labeled **Python,** click the word **Flask.**

The page **Deploy a Flask App** gives you instructions, but you're going to deviate from those. Why? Because *you already have a Flask app that runs*. Instead of deploying the example app, you're going to deploy your own.

First, you have to make a few changes to your app. So you'll ignore Render for a bit.

Install Gunicorn

Gunicorn is a Python web server. You know whenever you fire up localhost for a Flask app, you see a warning message that says the Flask server is not intended to run out in the wild. Well, Gunicorn is *safe* to run in the wild.

1. Change (`cd`) into your Flask directory.
2. Activate your Flask virtual environment in the usual way.
3. Then do the install:

```
pip install gunicorn
```

Create the requirements.txt file

This file will tell the server in the cloud all the libraries you need to run your app. They will all need to be installed in your app's cloud container, and this simple file makes that happen. Since

you're already in your Flask directory *and* your virtual environment is already activated, this command will create the file:

```
pip freeze > requirements.txt
```

(This has *no relation* to Frozen-Flask, by the way.)

Now you have a new file named *requirements.txt* in your Flask folder. But wait! Your app project is probably inside a folder *inside* the Flask folder. If so, you must copy that *requirements.txt* file and paste it inside the project folder. The project folder and its contents are what you will deploy — ***not*** your entire Flask directory with all its contents and projects.

Add a .gitignore file

This new file also must go *inside* the project folder. A *.gitignore* file (DO NOT omit the dot!) tells git which files to ignore when updating a git repo. This always includes your *env/* folder, *__pycache__/,* and other files that **have no business being on the server.**

To create an empty *.gitignore* file in Atom, right-click on your project folder, choose New File, and type: `.gitignore`

The filename must be exactly that. DO NOT omit the dot!

Now copy the complete contents of a good Python *.gitignore* file — like this one — and paste them into your empty file. Save and close. Make sure the file is *inside* your app project folder.

Create a new GitHub repo (or push to an existing one)

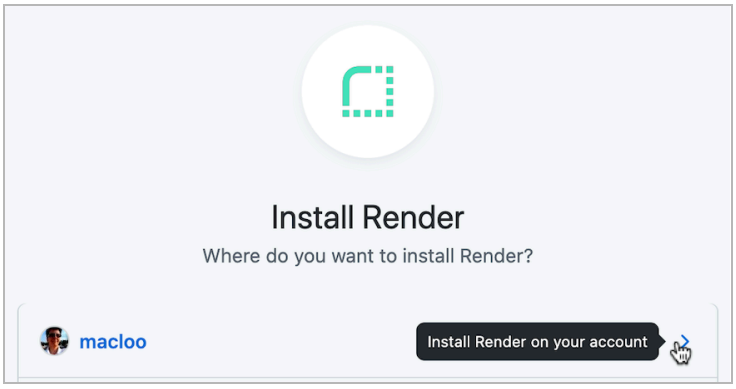
If your app project has a repo online at GitHub already, commit and push so that your new files are in place.

If not, make a new repo using your app project folder, commit and publish — and be sure the code is *public*. Here's an example of a correctly published Flash repo. Note that you can see the *requirements.txt* file and the *.gitignore* file at the top level of the repo.

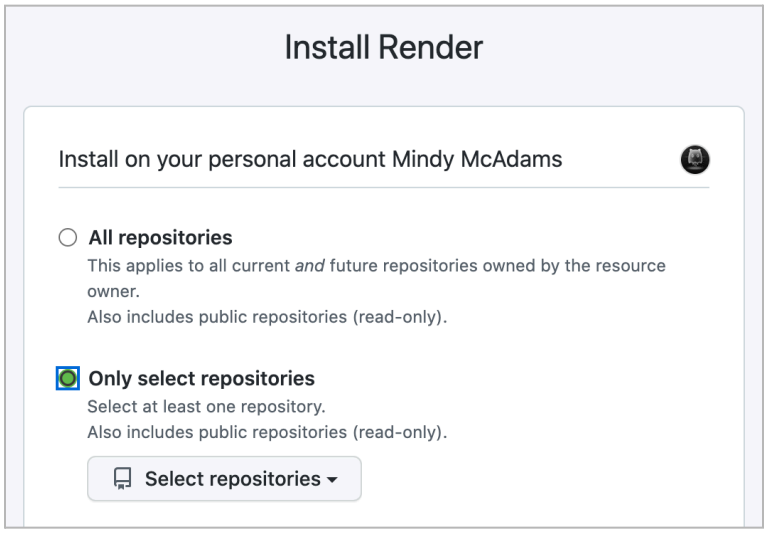
Use Render to deploy your app

Now go back to the **Deploy a Flask App** page at Render.com. You will skip Step 1 because you're using your own app.

Follow steps 2 and 3. When it says "Create a new Web Service on Render," it means go to your Render dashboard, click **New Web Service,** and (first) configure your GitHub account. There's a link for this.



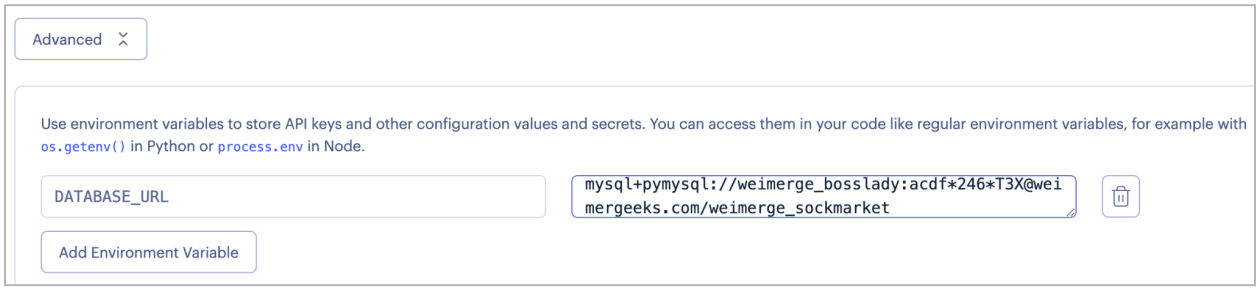
1. After you click the link to “Configure account,” you’ll be sent to this page. Click the > as shown above.



2. Make sure to click the box shown: “Only select repositories.” Then, select *the one repo* you want to deploy right now. Use the menu to ensure you’re selecting the repo you want.

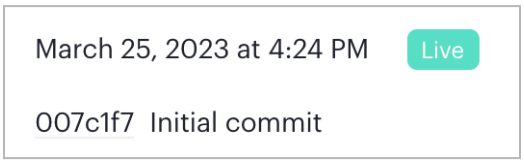
3. Click the **Install** button at the bottom of that page.
4. Enter your GitHub password when asked.
5. Click the **Connect** button.
6. You should be fine with the defaults with **one important exception**: your main app file is likely ***not*** named *app.py*. Mine, for example, is named *presidents.py*. This requires you to change the line that says `gunicorn app:app` to (for example) `gunicorn presidents:app` **DO NOT ADD any spaces around the colon!**
   For the “name” at the top, name it something sensible. I named mine “Presidents Search.” This is *not a filename,* so you can use spaces. This name will appear in your main Render Dashboard. Note that this name becomes part of your URL: https://presidents-search.onrender.com/
7. IF YOU NEED TO CREATE AN ENVIRONMENT VARIABLE, scroll to where you see the button labeled **Advanced**. Click it, then click **Add Environment Variable**. You will paste

in the exact name of your variable and the exact value for it. See Set environment variable on MacOS for instructions. PLEASE test the environment variable *locally* before you try to deploy!



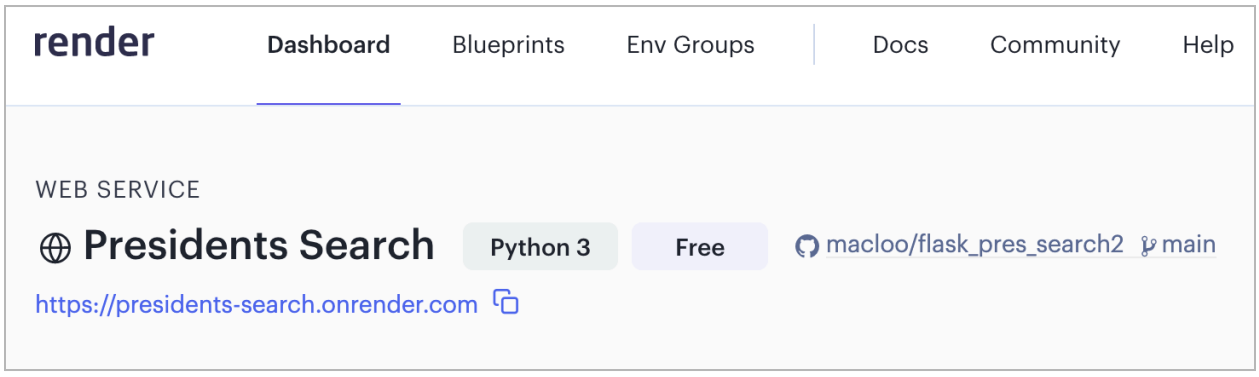
The illustration above is not from the Presidents app.

8. Click the **Create Web Service** button.
9. Then you'll go to a dashboard page for your app. You'll need to wait a bit for it to be fully deployed.



When you see a message that says "Live," it means your app has successfully deployed! If you get "Failed" instead, read the final line inside the terminal window on the Render page. It will usually tell you exactly what went wrong.

You can find your live URL at upper left:



This is a live URL that anyone can see. Your app will be updated automatically and restarted any time you push to that GitHub repo. There might be a time lag before you will see changes, so I would recommend that you make sure your app is finished before you deploy.

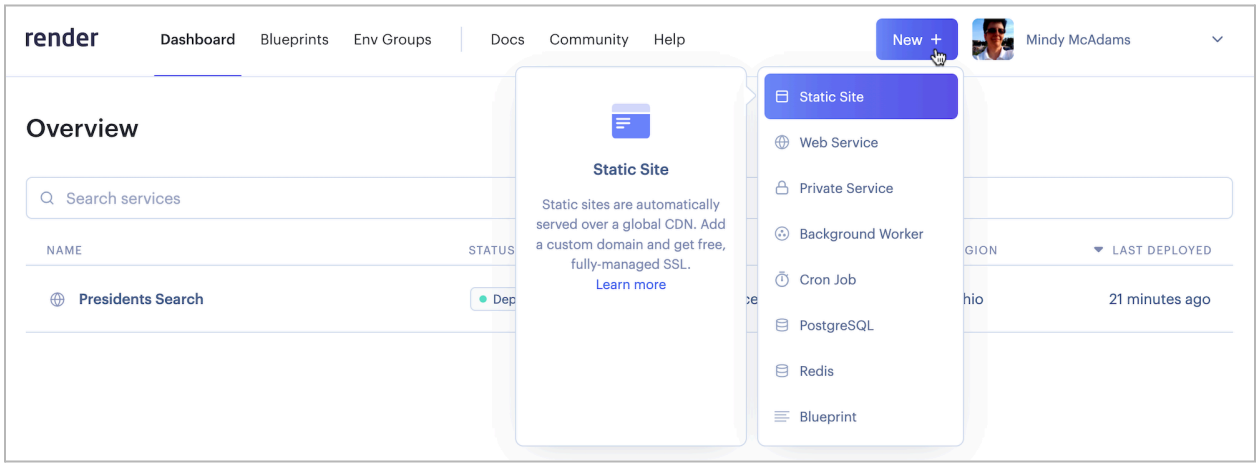
If your Flask routes lead to other pages in the app, you can share links to those as well: https://presidents-search.onrender.com/search

The default page will be determined by the route with **this decorator**: `@app.route('/')`

Your *next* Render deployment

You can access all your Render apps from your own dashboard: https://dashboard.render.com/

To create a new Web Service (or anything else), use the **New** button at top right on your dashboard.



Bear in mind that if you create and deploy a lot of things, at some point Render is going to require you to pay!