Josh's Self Replicating App

Intro

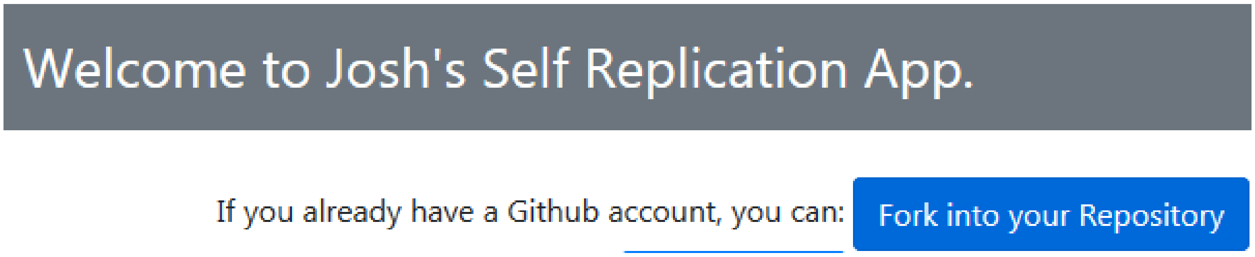
The Self Replicating App is a simple app that offers a web interface for users to copy its own code into a user's GitHub repository via the GitHub API. Once copied, the user can follow the steps in this document to get their own instance of the Self Replicating app running at a new URL by deploying the code into a new Heroku app.

View the original version here:

https://joshselfreplication.herokuapp.com/

Part 1: Forking into your Repository

The first part of the app is a web interface that allows a user to copy the Self Replicating App by clicking a button.



Welcome to Josh's Self Replication App.

If you already have a Github account, you can:

The user will be prompted to allow access into their GitHub account. If successful, the app uses the GitHub API to fork a copy of the Self Replication app into their repositories.

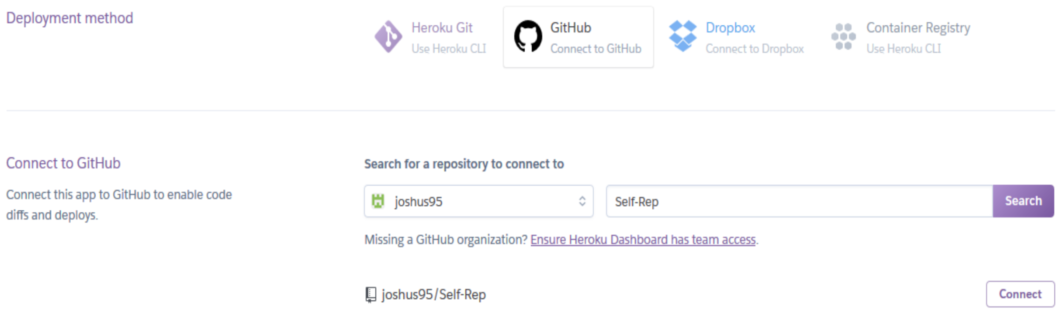
The an easy and free way to spin up a second instance of the Self-Rep app is by creating a new Heroku app for it.

Step 2: Hosting the app at a new URL

I have decided that, since my app is built using Django, hosting it as a Heroku app would be the easiest way for a person without much technical knowledge to set up an instance of the app. Also, it is free. Detailed below are the steps to do so:

1. Go to: https://signup.heroku.com/login
2. Fill in the information fields, register an email, and sign up for a Heroku account

3. Once logged into your new account, go to: https://dashboard.heroku.com/new-app
4. Select a name for your app and click "Create App". Whatever name you pick here will be used as part of the URL for the app.
5. On the next page, under "Deployment Method", click "GitHub". Then click, "Connect to GitHub". A popup, will appear asking for authorization. Click "Authorize Heroku"
6. In the "Connect to Github" section, type in the "Self-Rep" to search for the self replicating repository and click "connect"



Deployment method

Connect to GitHub

Connect this app to GitHub to enable code diffs and deploys.

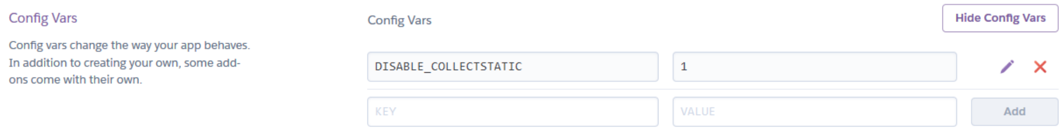
Search for a repository to connect to

joshus95 | Self-Rep

Missing a GitHub organization? Ensure Heroku Dashboard has team access.

joshus95/Self-Rep

7. To make the process as simple and streamlined as possible, we will not configure our Heroku app to serve static files. By default, the Heroku app will attempt to run a collect static command upon deployment, which will cause an error. Disable this from the by clicking "settings" while in the Heroku dashboard, and then the "Reveal Config Vars" button.



Config Vars

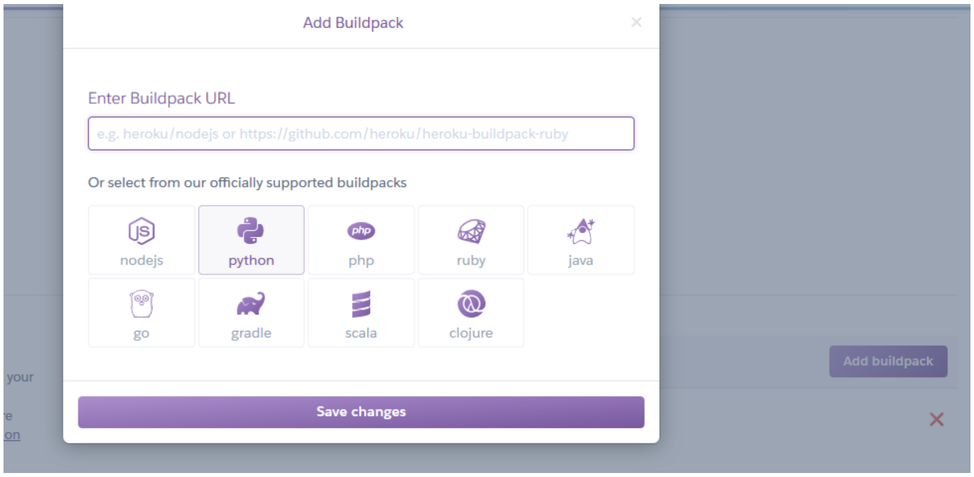
Config vars change the way your app behaves. In addition to creating your own, some add-ons come with their own.

Config Vars

DISABLE_COLLECTSTATIC	1
KEY	VALUE

Add " DISABLE_COLLECTSTATIC" with a value of "1" as a new Config Var.

8. While you are in the "settings" section, add the Python Buildpack. Do this by scrolling to the " Buildpacks" section and clicking "Add Buildpack". Select "python" and save changes.



Add Buildpack

Enter Buildpack URL

e.g. heroku/nodejs or https://github.com/heroku/heroku-buildpack-ruby

Or select from our officially supported buildpacks

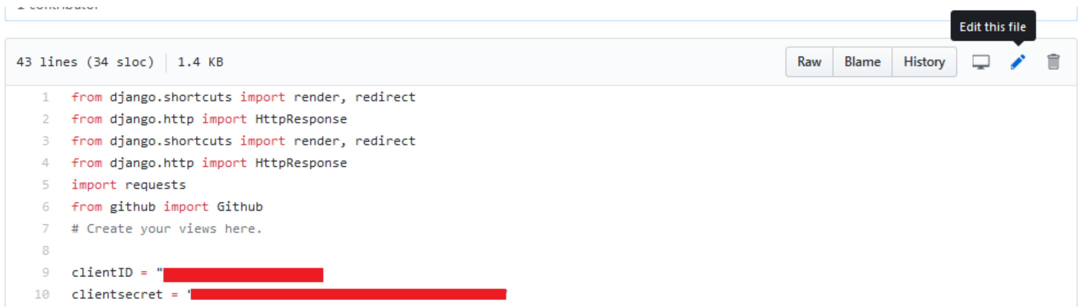
9. Click "Deploy" from the dashboard menu. Scroll down to the Manual Deploy section and click "Deploy Branch"

Congratulations! You should now have a new instance of the Self Replicating App running! You can view you new app by clicking the "Open App" button.

Step 3: Using your own GitHub Client ID

At this point, you may notice that the replicated version of the app is not completely standalone. When a user uses the "fork" button on your replicated app, the request will get routed through the original instance of the Self Replication App to handle the request. Step 3 is an optional step that you can go through to make your replicated version of the Self Replicating App completely standalone.

1. Go to: https://github.com/settings/applications/new
2. Fill in the fields. The Application name is not important, but pay attention to the Homepage and callback URL fields. The homepage is the URL that your new Heroku app is a hosted at. It should have the format "https://{YOUR APP NAME HERE}.herokuapp.com". The Callback URL is your homepage URL, but with a "/callback" at the end of it. It should take the format: "https://{YOUR APP NAME HERE}.herokuapp.com/callback".
3. Once registered, you will be given a "Client ID" and a "Client Secret". Takes note of these.
4. From github.com, navigate to your repositories and locate the "Self-Rep" repository. Then find the "launcher" folder in the repository. Alternatively, you can go to the URL: https://github.com/{Insert your GitHub username here}/Self-Rep/tree/master/launcher
5. Start editing the views.py file by clicking "views.py" and the edit button.

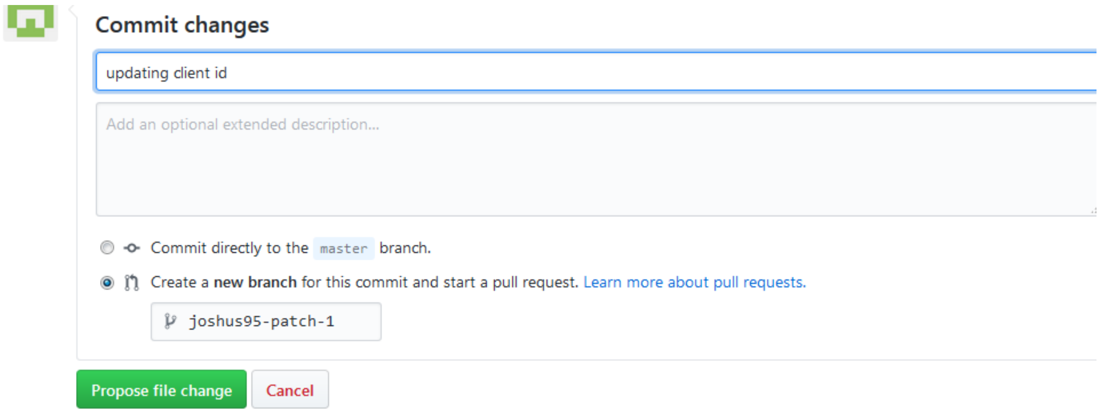


```
from django.shortcuts import render, redirect
from django.http import HttpResponse
from django.shortcuts import render, redirect
from django.http import HttpResponse
import requests
from github import Github
# Create your views here.

clientID = "[redacted]
clientsecret = '[redacted]
```

6. Change the clientID and clientsecret fields into the "Client ID" and "Client Secret" that you were given at step 3. Go to the "commit changes" section at the bottom of the page. Give your change a name and select the "Create a new branch" option. "Click the Propose file change"

button.



Commit changes

updating client id

Add an optional extended description...

○ Commit directly to the `master` branch.

● Create a **new branch** for this commit and start a pull request. Learn more about pull requests.

joshus95-patch-1

7. On the next page, click "create pull request" and then "merge pull request". Finally, click "confirm merge". This updates the master branch of your GitHub copy of the Self Replicator to use your own GitHub API key.
8. Now we need to re-deploy the Heroku app so it reflects the change you just made. Go back to your Heroku Dashboard. Click "Deploy" from the dashboard menu. Scroll down to the Manual Deploy section and click "Deploy Branch".

Congratulations! You now have a standalone instance of the self replication app running. Even if the main instance ever goes down, yours will still be functional.