Deploy a Flask App on Heroku

Name: Arda Cem Özmen

Batch Code: LISP01

Submission Date: 29-03-2021

Submitted To: Data Glacier

Heroku lets you deploy a Flask app online for **free**.

Heroku is a container-based cloud Platform as a Service (PaaS). Developers use Heroku to deploy, manage, and scale modern apps. Our platform is **elegant**, **flexible**, and **easy to use**, offering developers the simplest path to getting their apps to market.

Overview

1. Prerequisites
2. Install Heroku
3. Getting Started With Heroku
4. Using Existing The Flask App (or You can use create new one)
5. Make Required Files to Deploy to Heroku
6. Deploy Your App to Heroku
7. Kill The App

Prerequisites

Before you can start, you need to do three things:

- Install Git
- Install Python

- Sign-up Heroku account

Install Heroku

If you are using Mac, you can download HomeBrew from your Terminal.

```
$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

Then run this command to install Heroku.

```
$ brew tap heroku/brew && brew install heroku
```

If you want to install Heroku on Windows, go to this link: https://devcenter.heroku.com/articles/heroku-cli#other-installation-methods

Getting Started With Heroku

Start with an empty project in your preferred text editor. In this tutorial, I am using VSCode.

Create a New Working Directory

```
$ mkdir <your-app-name>
$ cd <your-app-name>
```

Create Virtual Environment

It is recommended to create a virtual environment so that there is no conflict with Heroku during the deployment.

If you have installed Python with Anaconda, you can create your empty environment this way.

```
$ conda create --name ml-model-heroku python=3.6
```

Activate the environment.

```
$ conda activate ml-model-heroku
```

or you can use venv. I recommend this way.

```
$ virtualenv venv
$ . venv/bin/activate
```

Install Requests, Flask and Gunicorn

To run the app, we need to install the Flask Python Framework with which we will build our app and Gunicorn to be our Python WSGI HTTP Server.

```
$ pip install requests[security] flask gunicorn
```

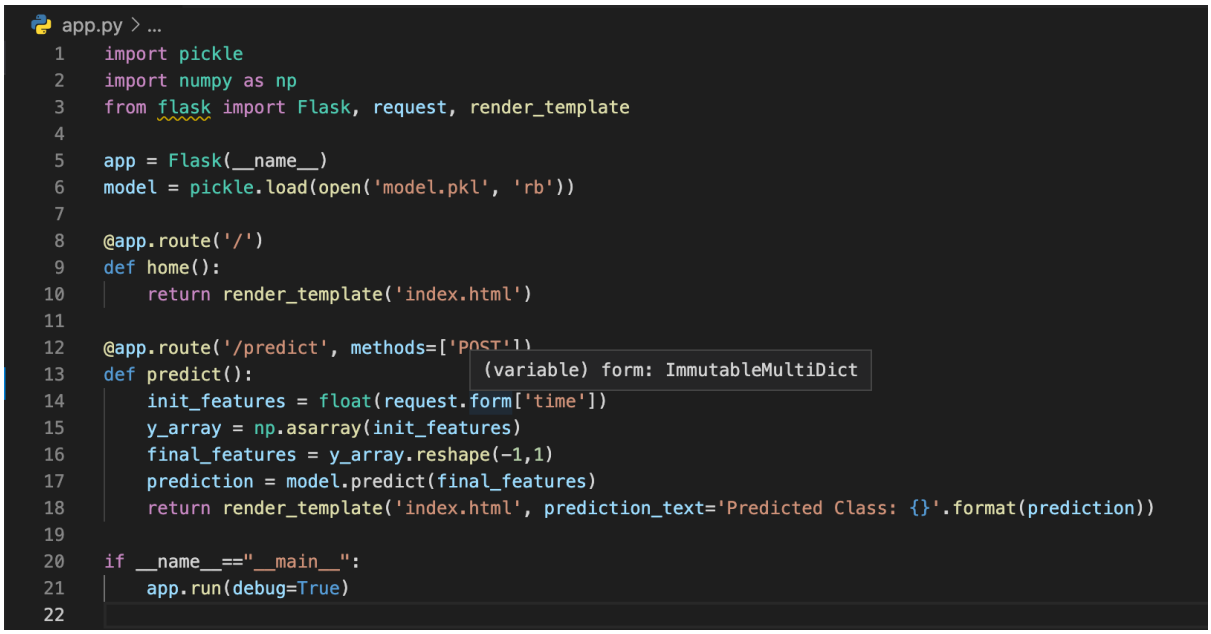
Using Existing The Flask App

Now, let's use existing Flask app. Also, you can use create a new one in this way.

Create App Folder

```
$ mkdir app
$ cd app
```

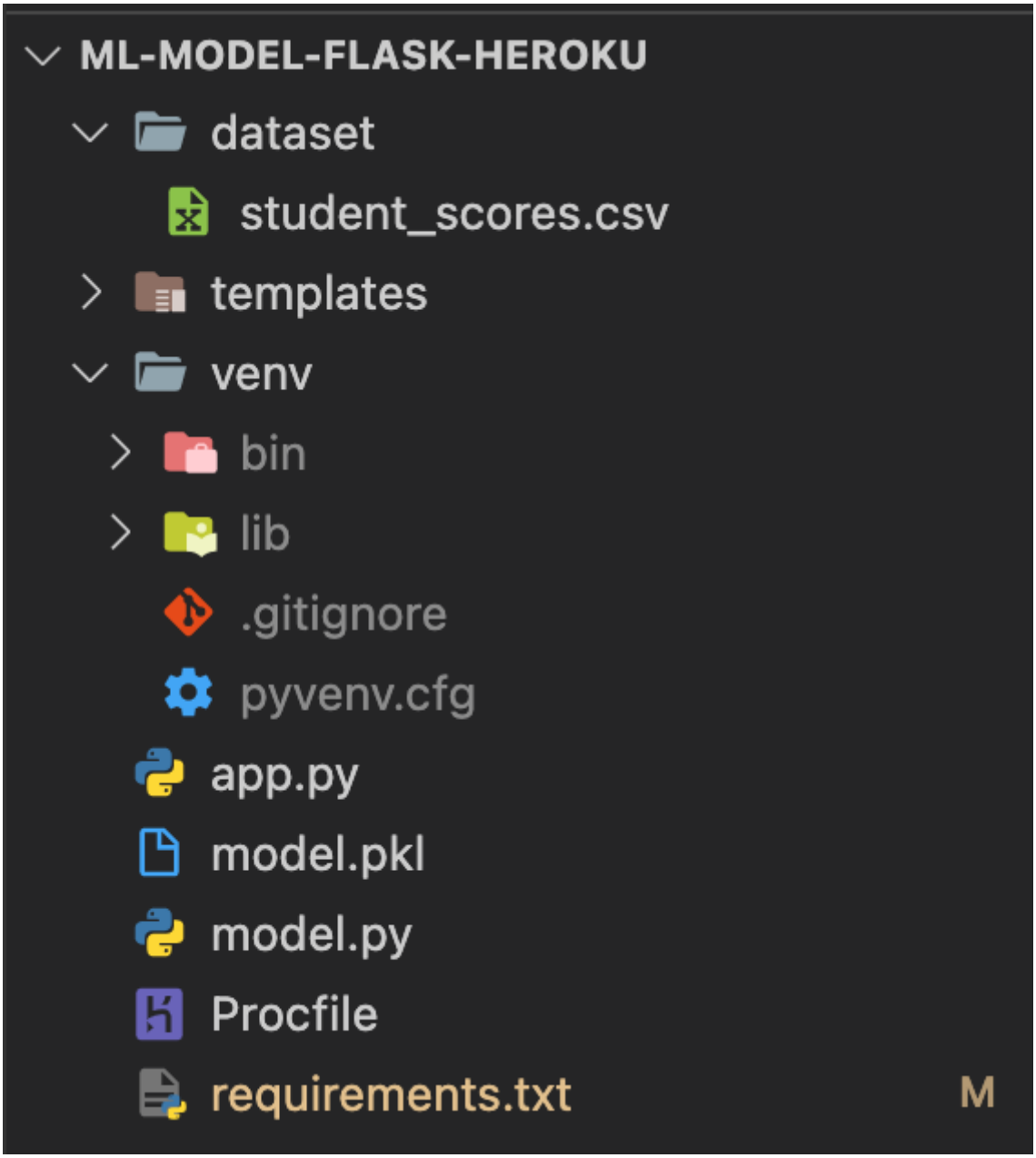
Write Your Flask app



Run Flask App

Move to the parent directory.

```
$ cd ../
```



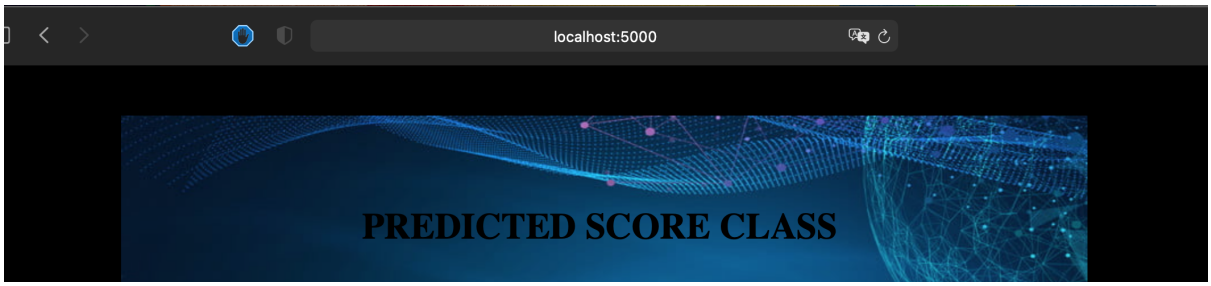
Check the files are in the correct location!

To verify if the app works, try running the app first.

```
$ python app.py
```

In your browser, go to the IP given and see if the app works.

```
(base) Ardas-MacBook-Pro:ml-model-flask-heroku ardaozmen$ . venv/bin/activate
(venv) (base) Ardas-MacBook-Pro:ml-model-flask-heroku ardaozmen$ python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 187-240-900
```



Make Required Files to Deploy to Heroku

Create Requirements.txt

Now, let's freeze the environment and add it to the requirements.txt so that the required packages are installed when deploying the app.

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

Create Procfile

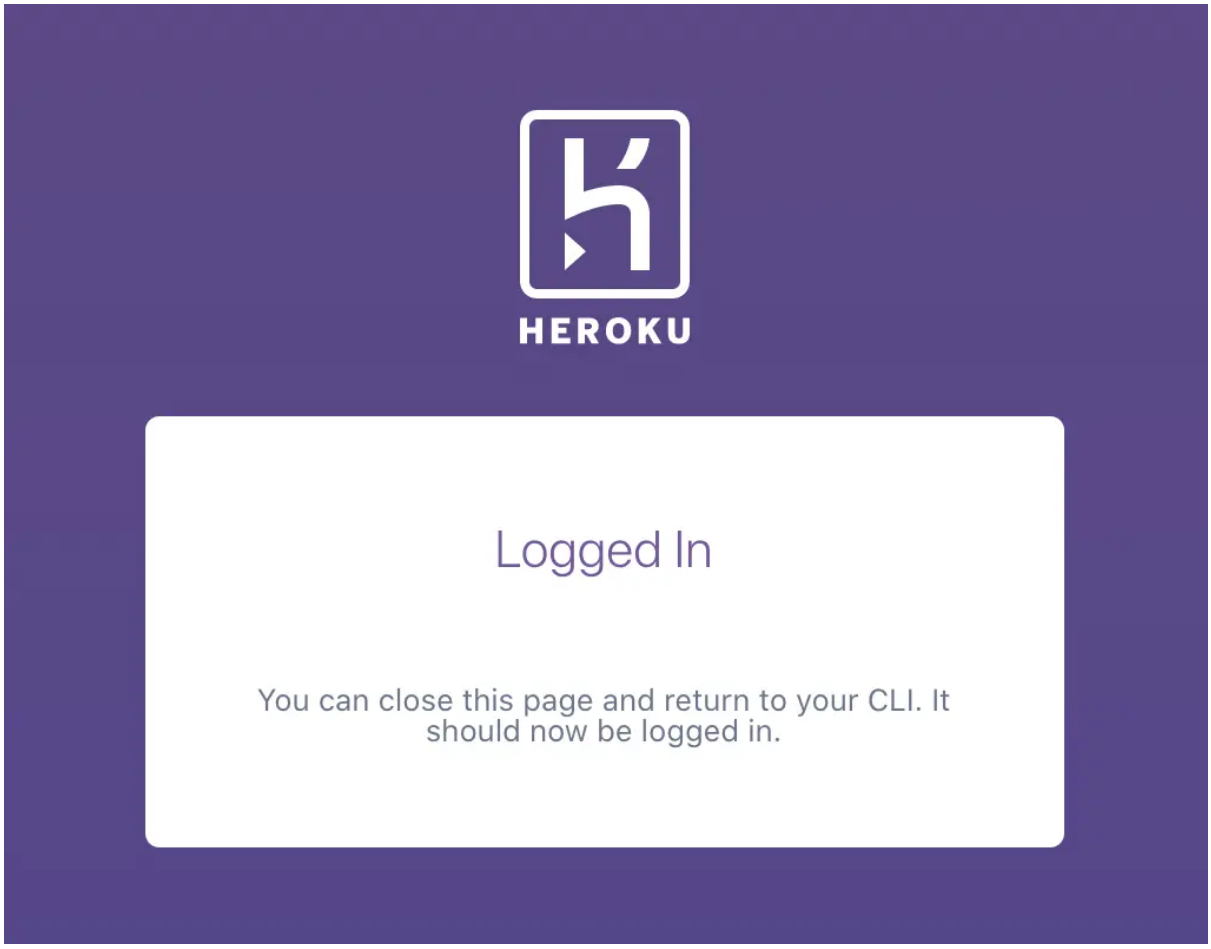
We need to create a Procfile that contains the commands that Heroku will need to run.

- To create the Procfile to run the app.py app:

```
$ echo web: gunicorn run:app >> Procfile
```

Deploy Your App to Heroku

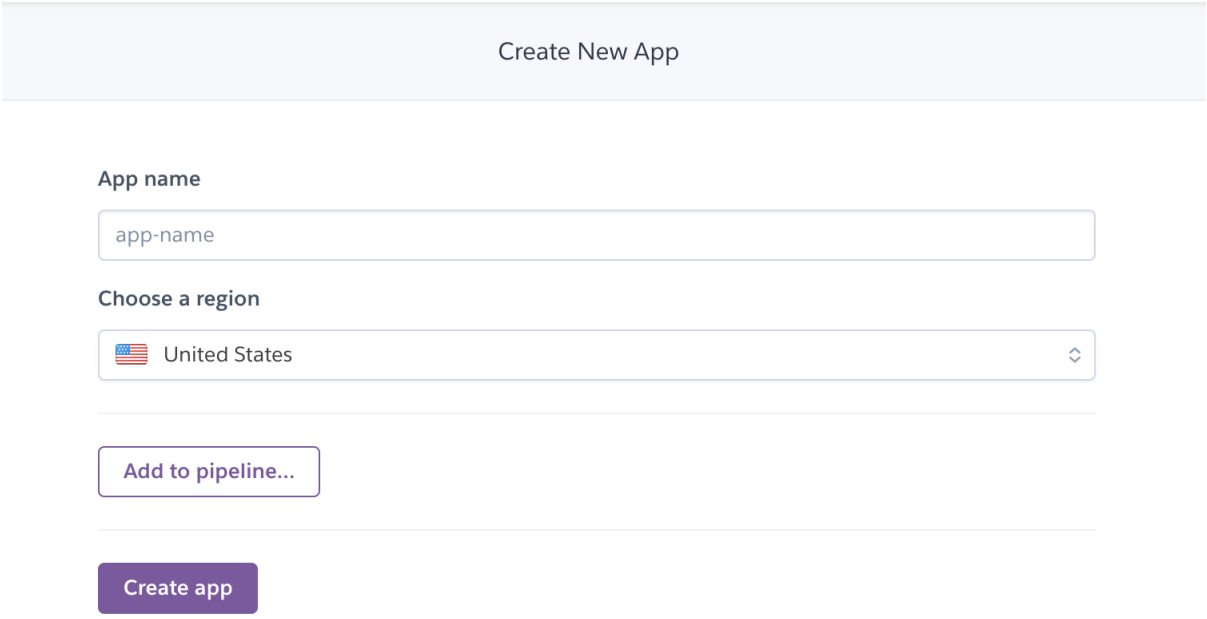
Login Heroku



Create Heroku App

```
$ heroku create <app-name>
```

or Go to Heroku Dashboard



Add Heroku App to Remote

```
$ git init
$ heroku git:remote -a <heroku-app-name>
$ git remote -v
```

Deploy the Flask App to Heroku

```
$ git add .
$ git commit -m "Init deploy"
```

```
$ git push heroku master
```

All done, the Flask app should be live.

Kill The App

Deactivate app

```
$ heroku ps:scale web=0
```

Reactivate App

```
heroku ps:scale web=1
```

Delete App

```
$ heroku apps:destroy <heroku-app-name>
```

Conclusion

LOOK AT THE PROJECT ON HEROKU