DataGlacier: Week #5

Cloud and API deployment

Name: Ethan Dy
Batch Code: LISUM39
Date: December 5, 2024
Submitted to: Github Repository to DataGlacier

Website URL: https://lisum39-mlapp.onrender.com/
(may take some time to initially load ~2 minutes)

Step 1 (Prepare Your Flask App):

- Ensure your Flask app is fully functional locally.
- Has all the required files (app.py, model.py, etc)
- Needs the file **requirements.txt** for production

datasets	12/4/2024 9:54 AM	File folder	
templates	12/4/2024 9:54 AM	File folder	
app.py	12/4/2024 10:30 AM	Python File	3 KB
model.py	12/4/2024 9:54 AM	Python File	2 KB
requirements.txt	12/4/2024 10:31 AM	Text Document	1 KB

- To create a requirements.txt, use the "**pipreqs**" library to automatically generate all used imports.
    - In CLI, use the command "**pip install pipreqs**"
    - Generate the requirements.txt by doing "**pipreqs .**"

- requirements.txt is used by Render (Our deployment service) to know which Python packages to install in the deployment environment. Without it, your app won't function because necessary libraries like Flask or scikit-learn wouldn't be installed.

- This is how your requirements.txt should look:

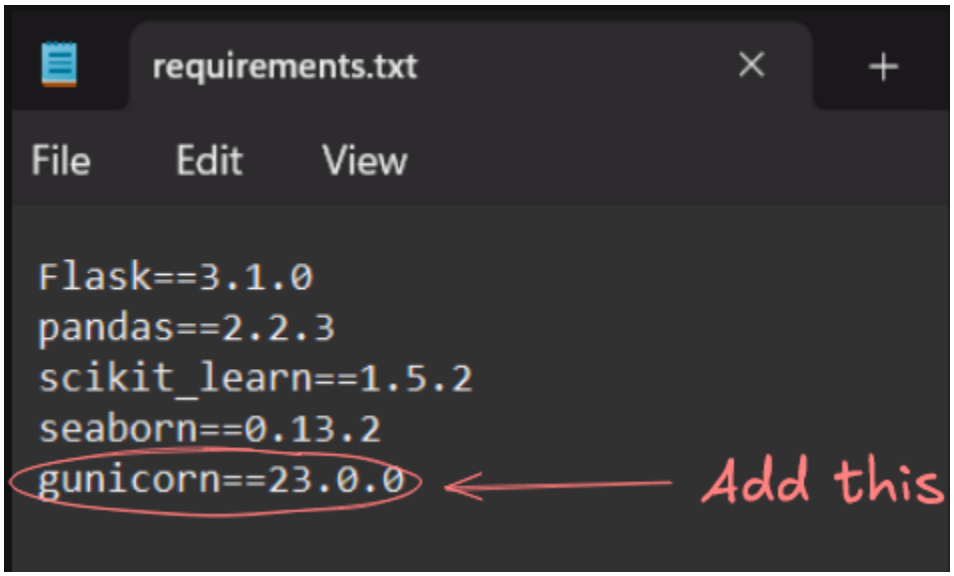
requirements.txt

File Edit View

```
Flask==3.1.0
pandas==2.2.3
scikit_learn==1.5.2
seaborn==0.13.2
gunicorn==23.0.0
```

- Next add the “**gunicorn**” package by also using the CLI command, “**pip install gunicorn**”
    - Inside the requirements.txt add the gunicorn package to install it alongside other dependencies in the form: “**gunicorn==<version>**”

- Gunicorn is a production-grade WSGI server (Web Server Gateway Interface) that is much more efficient and robust than Flask's built-in development server. Using gunicorn ensures that your app can handle multiple users and operate reliably in production.



Step 2 (Pushing Code to GitHub):

- In your folder directory, initialize a git repository with “**git init”**
- Add all your files onto the git stage with “**git add .”**
- Add a commit message that tells what you’re doing like: “**git commit -m "Initial commit for Flask app"**”

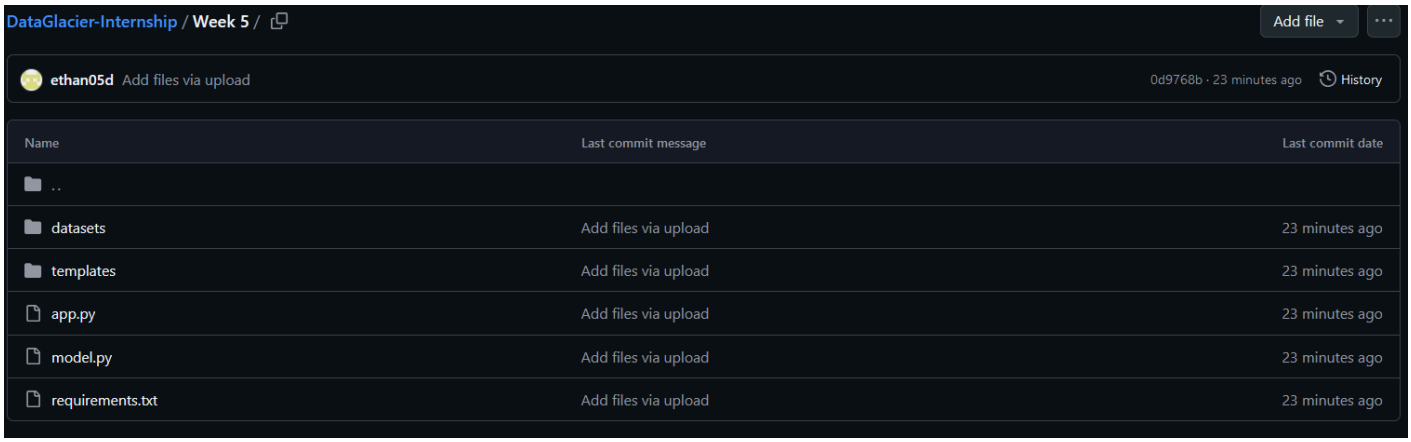
- Now push your code to a git repository, by first setting the origin your pushing towards:

    - **“git remote add origin <url>”**

- Then change your branch to a certain branch “main” in our case
    - “**git branch -M main**”

- Then push onto the origin on the branch you’re on
    - “**git push -u origin main**”

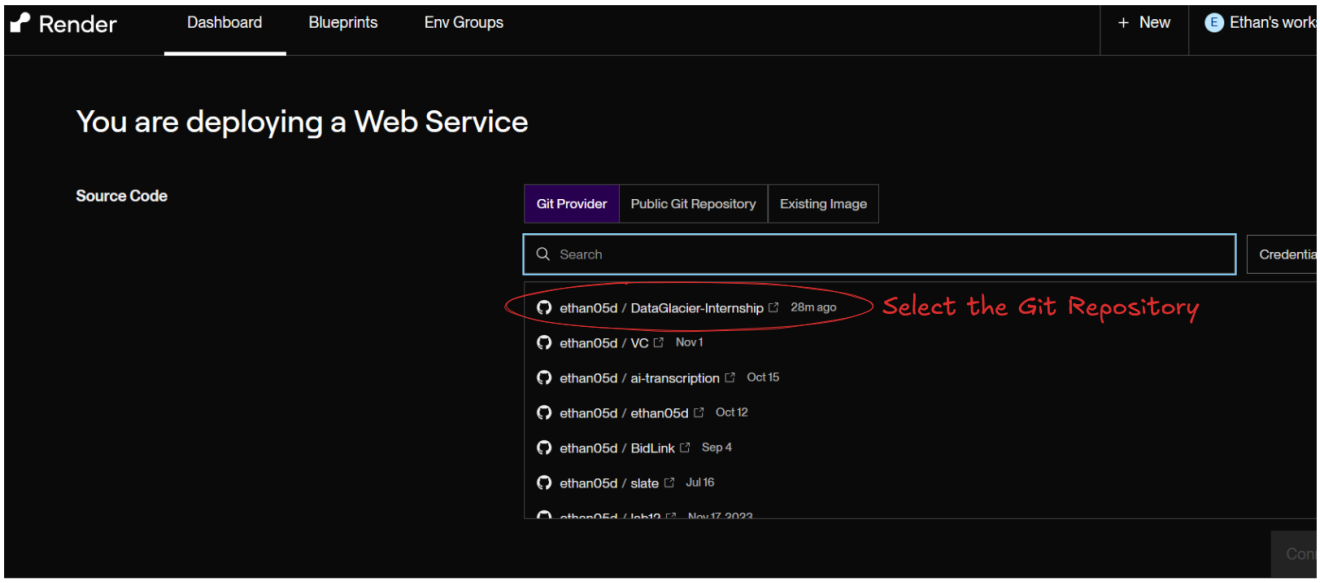
- Pushing your code to GitHub allows Render to access your repository and deploy the app. It also provides a version control system to track changes and collaborate with others.
- You should see something like this on your github repository:



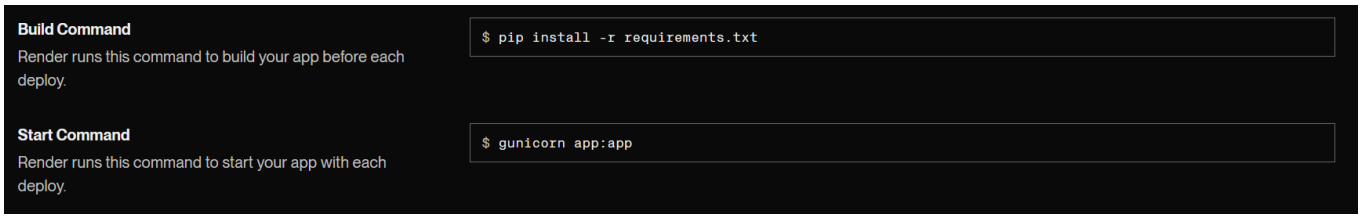
Step 3 (Deploying to Render):

- First create an account at Render using https://render.com/
- Then create a new “**Web Service**” and connect your github account to Render

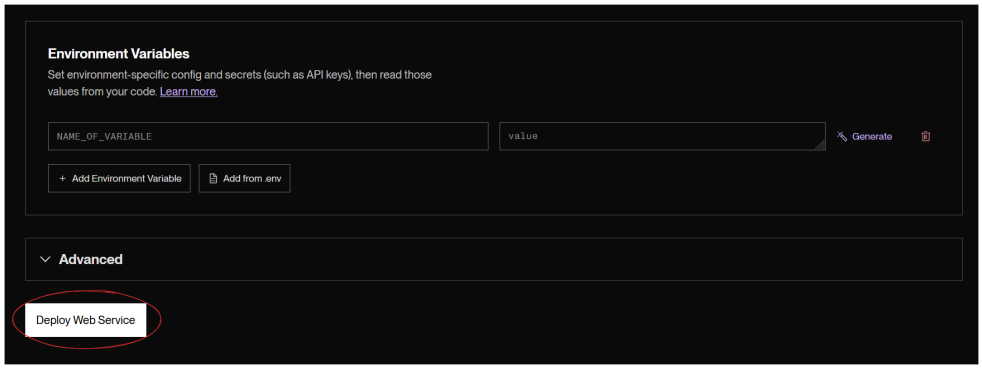
- To configure Render deployment, connect the github repository you pushed all your code onto:



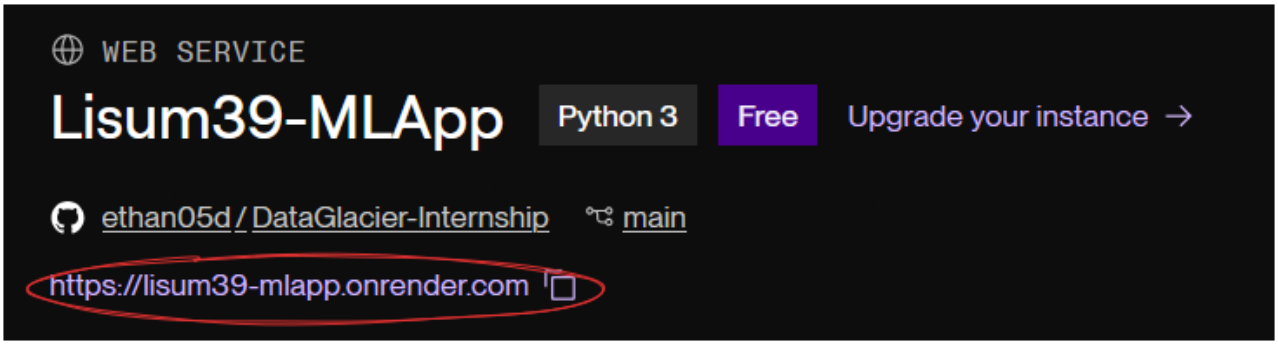
- Then set the “Build Command” to install your requirements.txt
  - “**pip install -r requirements.txt**”
- And the “Start Command” to use gunicorn to run the app
  - “**gunicorn app:app**”



Lastly, hit “Deploy Web Service” to spin up the application:



- Lastly, checking the deployed app with your domain:
- On dashboard you can check the url on the top left:



- I hosted my URL on https://lisum39-mlapp.onrender.com/
- Finally, you can see your deployed web app: