

Project description

This project aims to provide you with additional practice on common software engineering tasks. These tasks will augment and complement your data skills, and make you a more attractive job candidate to potential employers.

You will be asked to develop and deploy a web application to a cloud service so that it is accessible to the public.

In this project, we will provide you with the dataset on car sales advertisements, which you've already worked with in the past.

The project is split into several steps that replicate the process described in our blog post [here](#), but we will be using the [Render platform](#) instead of Heroku.

Demo of the web application you'll build in this project.

Instructions

Step 1. Getting started

- Create an account on [github.com](#)
- Create a new git repository with a `README.md` file and a `.gitignore` file (choose a Python template)
- Make a new Python environment. Install the following packages: `pandas`, `streamlit`, `plotly.express`, and `altair`. Feel free to install more packages if you want to implement additional features in your web app
- Create an account on [render.com](#) and link it to your GitHub account
- Use git to clone your new project's git repository to your local machine. From now on, you'll be working on the project on your local machine, and then committing and pushing the changes back to the GitHub repository
- [Install VS Code](#) and load the project into VS Code (by opening the project directory with VS Code)
 - If you have not used VS Code before, check out the extra lessons on Working in VS Code and Source Control in VS Code
- In VS Code, set the Python interpreter to the one used by the virtual environment. Make sure you have the necessary packages installed

Step 2. Download the data file

- [Download the car advertisement dataset](#) (`vehicles_us.csv`) or find your own dataset in a CSV format.
- Place the dataset in the root directory of the project.

Note: Check out [this video](#) for project ideas and inspiration.

Step 3. Exploratory Data Analysis

Create an `EDA.ipynb` Jupyter notebook in VS Code.
Save the notebook in the `notebooks` directory of your project.
Perform some basic exploratory analysis of the dataset in the notebook.
Create a couple of histograms and scatterplots using `plotly-express` library.

Note:

if you are using the car advertisement dataset, it won't be sufficient to simply recreate the plots described in the blog post to complete the project. You'll have to get creative and come up with your own plots and histograms.
it's often very convenient to experiment with data visualizations in Jupyter, and then copy-paste code into a web application file later

Step 4. Develop the web application dashboard

Note that this is not the same development flow that we used in the lesson on rendering.

- Create an `app.py` file in the root of the project's directory. The following steps (2-4) will require you to write code into this `app.py` file.
- In the `app.py` file, import `streamlit`, `pandas`, and `plotly.express`.
- Read the dataset's CSV file into a Pandas DataFrame.
- From the Jupyter notebook, create and/or copy:
 - at least one `st.header` with text
 - at least one Plotly Express histogram using `st.write` or `st.plotly_chart`
 - at least one Plotly Express scatter plot using `st.write` or `st.plotly_chart`
 - at least one checkbox using `st.checkbox` that changes the behavior of any of the above components

Don't forget to update the `README` file when you are done. It should contain a basic description of the project, explaining that this is a tool to simulate random events, and the methods and libraries used to implement it. It should also contain instructions on how other people could launch your project on their local machine if they wanted to.

IMPORTANT: Your project will only build on the online service if all project files are present in your GitHub repository. Therefore, you must commit and push each new file change to your repository as soon as you've completed it.

Notes:

As you add new Streamlit components to develop your application, you can run the `streamlit run app.py` command from the terminal to see what the result will look like. Do this on your local machine (preferably from a system terminal) to test that everything works before committing and pushing your changes to GitHub. Upon reaching certain application development milestones (e.g., adding a working component and having the application run without errors), it's good practice to commit and push your work to a remote repository on GitHub. Don't forget to write a meaningful commit message!

In this case, Render will fail to build your project unless all project files are pushed.

Step 5. Deploy the application to Render

- To make Streamlit compatible with Render, we need to create two new files: `requirements.txt` and `.streamlit/config.toml`.

First, we need to create the `requirements.txt` file. Create this new file in your project folder's root level. Then, add your project's required packages. It should look something like this (although you can include other packages):

```
pandas==2.0.3
scipy==1.11.1
streamlit==1.25.0
altair==5.0.1
```

```
plotly==5.15.0
```

Second, we need to add the configuration file to your git repository. Create the `.streamlit` directory, then add the `config.toml` file there (this can all be done with the right-click menu in the left-hand tab of VS Code).

Add the following content to the `.streamlit/config.toml` file:

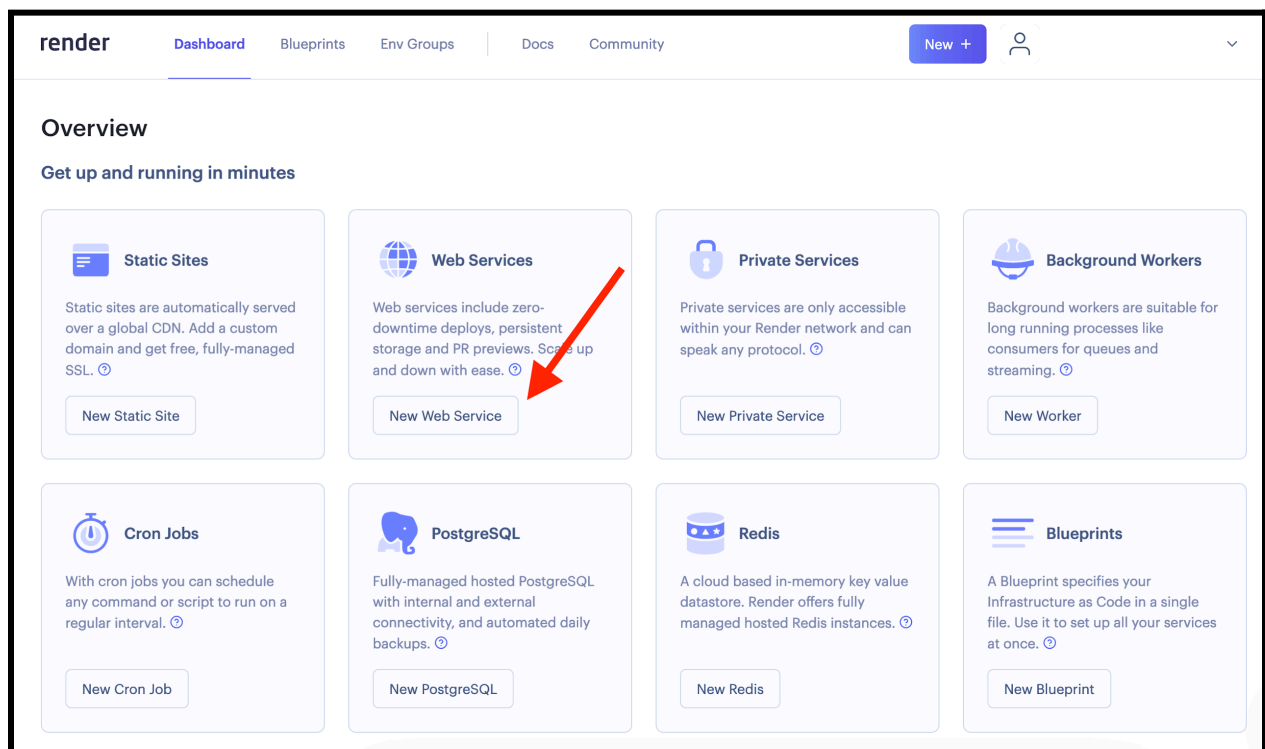
```
[server]
headless = true
port = 10000

[browser]
serverAddress = "0.0.0.0"

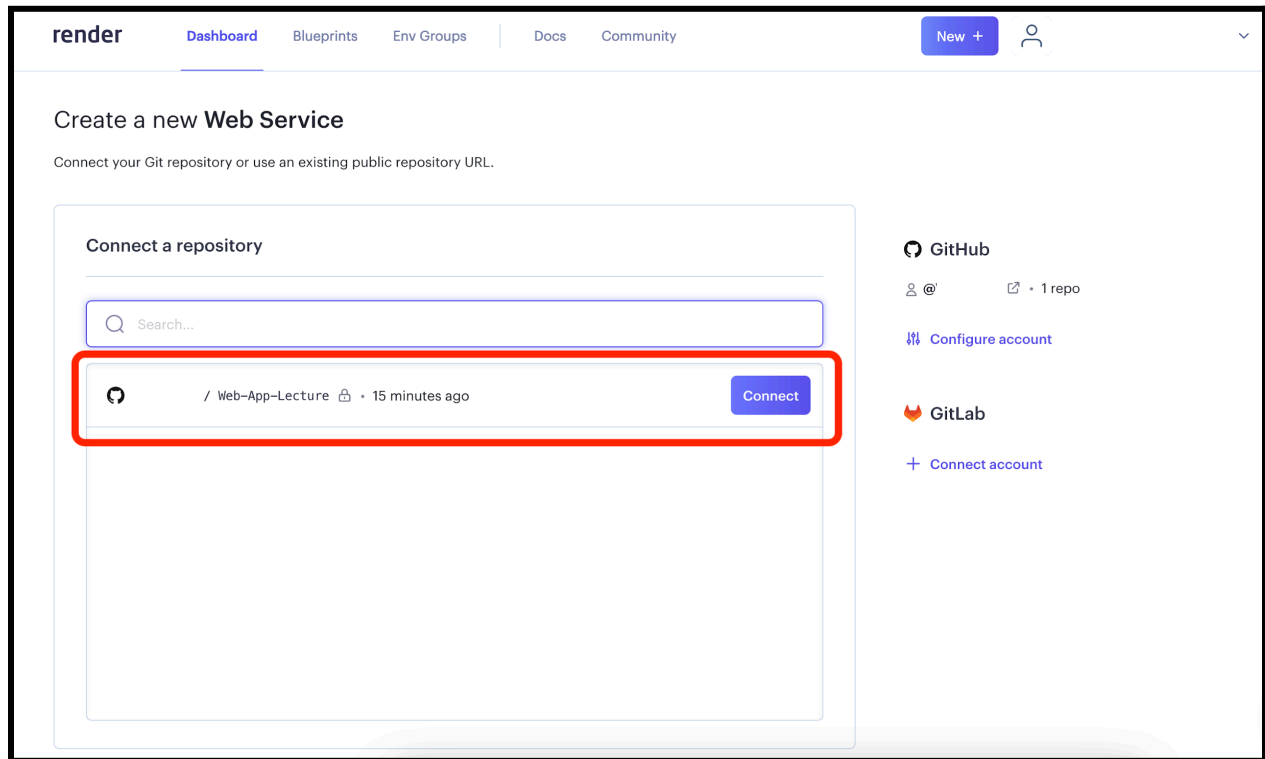
serverPort = 10000
```

This configuration file will tell Render where to look in order to listen to your Streamlit app when hosting it on its servers.

Open your account on render.com and create a new web service:



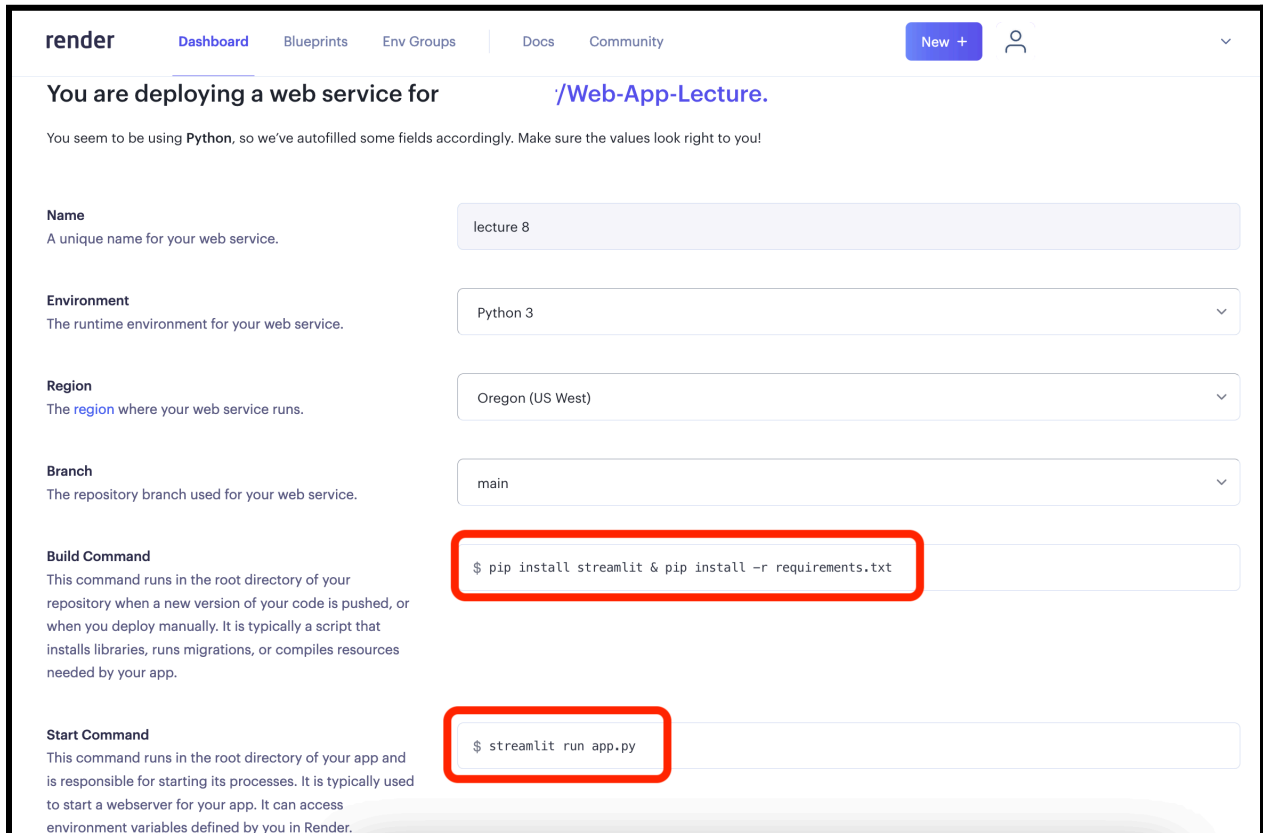
Create a new web service linked to your GitHub repository:



Configure the new Render web service. To your Build Command, add

```
pip install streamlit & pip install -r requirements.txt
```

To your Start Command, add: `streamlit run app.py`. It should look like this:



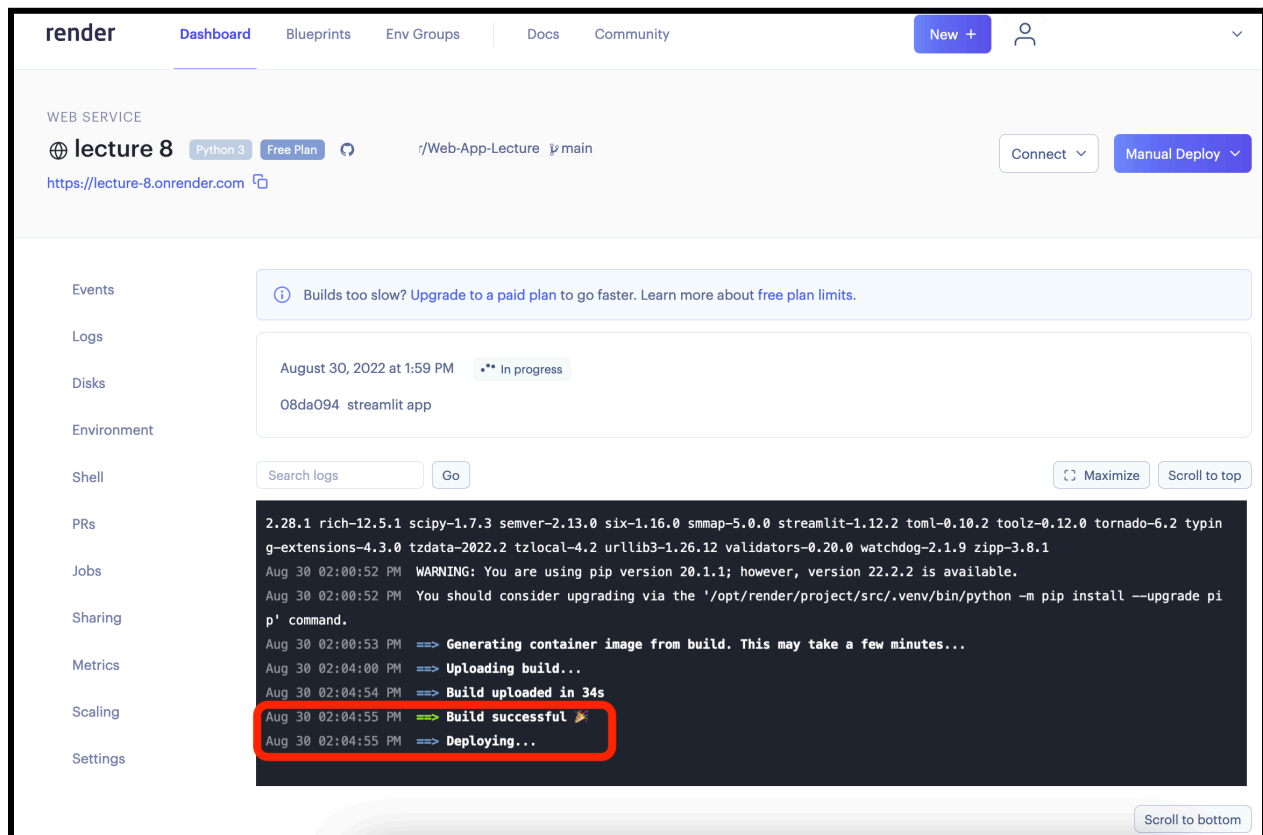
The screenshot shows the Render dashboard for configuring a new web service. The page title is "You are deploying a web service for /Web-App-Lecture." Below the title, a message states: "You seem to be using Python, so we've autofilled some fields accordingly. Make sure the values look right to you!"

The configuration form includes the following fields:

- Name:** A unique name for your web service. The value is "lecture 8".
- Environment:** The runtime environment for your web service. The value is "Python 3".
- Region:** The region where your web service runs. The value is "Oregon (US West)".
- Branch:** The repository branch used for your web service. The value is "main".
- Build Command:** This command runs in the root directory of your repository when a new version of your code is pushed, or when you deploy manually. It is typically a script that installs libraries, runs migrations, or compiles resources needed by your app. The value is `$ pip install streamlit & pip install -r requirements.txt`.
- Start Command:** This command runs in the root directory of your app and is responsible for starting its processes. It is typically used to start a webserver for your app. It can access environment variables defined by you in Render. The value is `$ streamlit run app.py`.

Both the "Build Command" and "Start Command" input fields are highlighted with red rectangles in the image.

Deploy to Render, wait for the build to succeed:



Verify that your application is accessible at the following URL:

https://<APP_NAME>.onrender.com/

Note: it can take several minutes after a successful deployment for the app to be available online on a free tier. Also note that apps go “asleep” after being inactive for a few minutes. If so, just load and refresh your app a few times for it to get awoken.

How to submit my project

You’ll need to submit a link to your GitHub repository. Please also add the URL of your app on Render to your project’s [README.md](#).

How will my project be evaluated?

We've put together some project assessment criteria. Read over them carefully before you make a submission.

Here's what project reviewers look for when assessing your project:

- Does the project repository contain at least the following files?

```
# The minimal expected project structure
$ tree
```

```
.
├── README.md
├── app.py
├── <name_of_your_dataset>.csv
├── notebooks
│   └── EDA.ipynb
└── .streamlit

    └── config.toml
```

- Is the web app accessible via a browser?
- Does the web app contain the following?
 - at least one header with text
 - at least 1 histogram
 - at least 1 scatter plot