BETSOL CAMPUS WORKSHOP HANDS-ON GUIDE

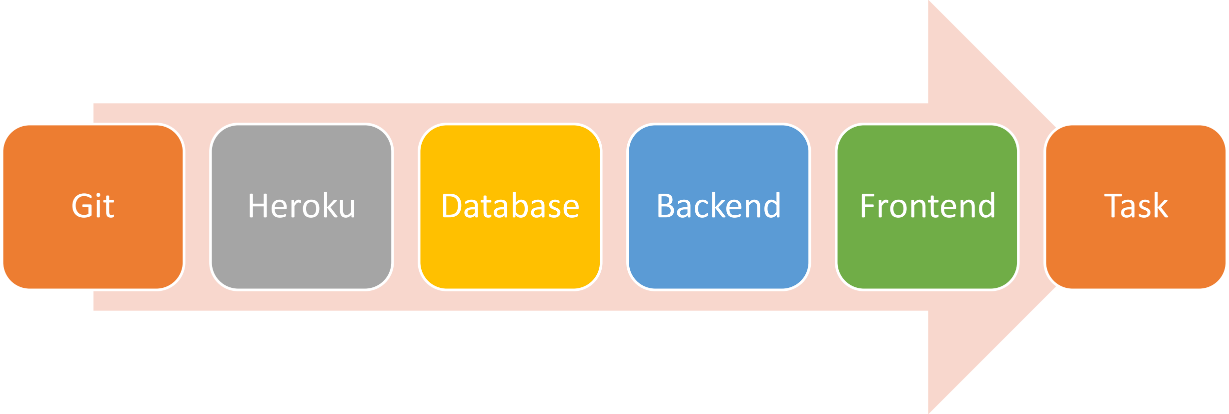
WORKSHOP HANDS-ON

Version 1.0

A Data Management and Intelligent Automation Company

CONTENTS

CAMPUS WORKSHOP

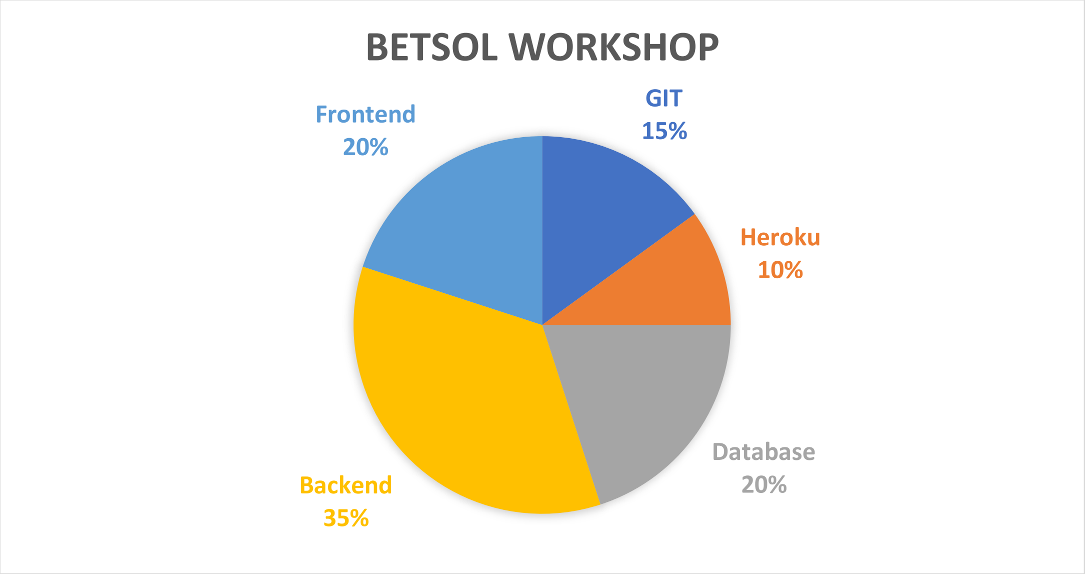


WHAT WE WILL BE LEARNING IN WORKSHOP?

We'll build a simple **To-do app** and deploy the database and backend on the cloud. We'll be using Python & Flask framework to build the backend, React JS to build the frontend, PostgreSQL for the database and finally Heroku to deploy our finished database and backend on the cloud.

PREREQUISITES FOR THE WORKSHOP

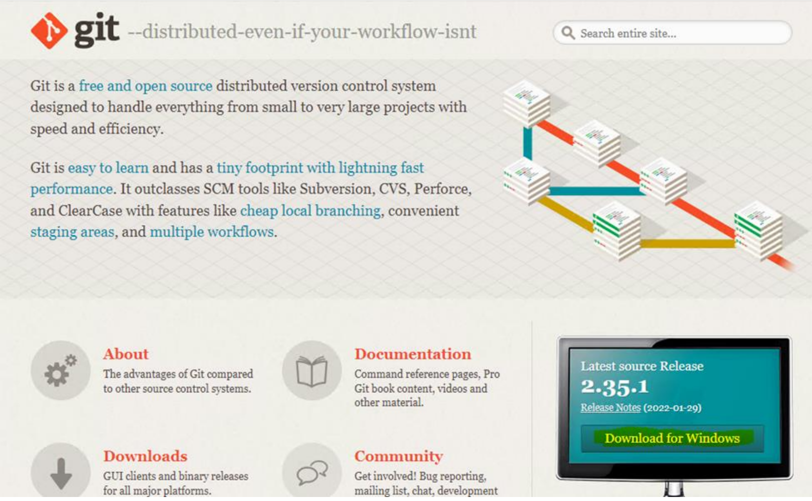
Topic	**Prerequisites**
GitHub	• Create an account at: https://github.com/ • Download and install git: https://git-scm.com/downloads
Cloud	• Heroku Account (Free account) Heroku | Sign up • Heroku CLI https://devcenter.heroku.com/articles/heroku-cli
Database	• https://www.pgadmin.org/download/
Python	• Download and install python from Download Python | Python.org preferably >=3.6 • Pycharm community version • Postman
React	• Download & install NodeJS and NPM from: https://nodejs.org/en/download/ • Install Visual Studio Code from: https://code.visualstudio.com/download • Versions: Node - 14.16.1 and npm - 6.14.12



INSTALLATION AND SETUP OF GIT

Step 1: Go to: Git (git-scm.com)

Step 2: Click on the Download option on the bottom left



Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

About
The advantages of Git compared to other source control systems.

Documentation
Command reference pages, Pro Git book content, videos and other material.

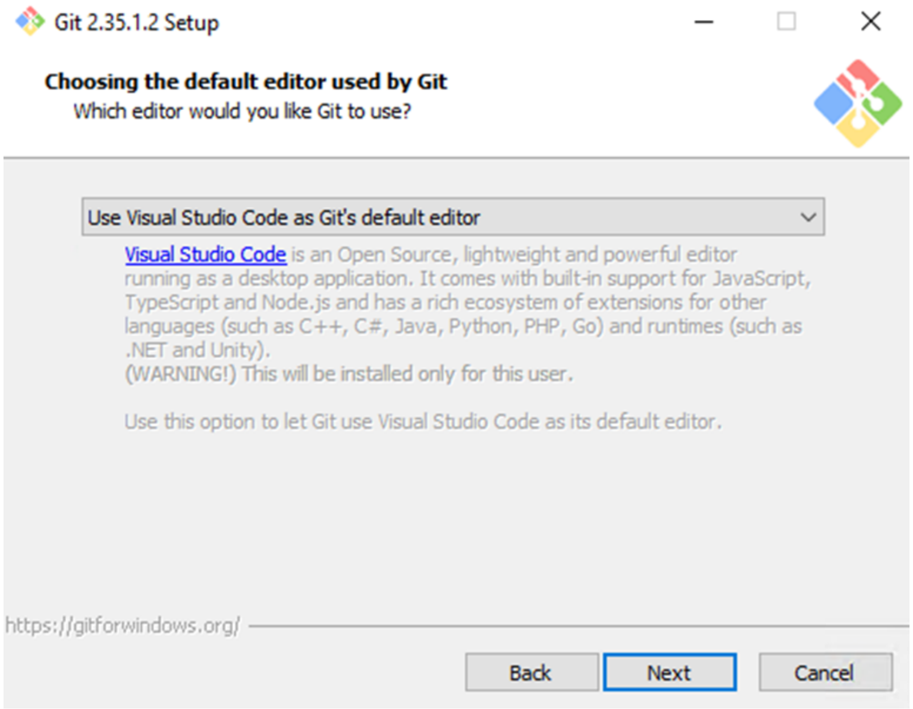
Downloads
GUI clients and binary releases for all major platforms.

Community
Get involved! Bug reporting, mailing list, chat, development

Latest source Release
2.35.1
Release Notes (2022-01-29)
Download for Windows

Step 3: Start installation

- While choosing your editor, select your preferred IDE and click next.
- Feel Free to install any other options than the selected default options



Git 2.35.1.2 Setup

Choosing the default editor used by Git

Which editor would you like Git to use?

Use Visual Studio Code as Git's default editor

Visual Studio Code is an Open Source, lightweight and powerful editor running as a desktop application. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity).
(WARNING!) This will be installed only for this user.

Use this option to let Git use Visual Studio Code as its default editor.

Step 4: On Installation, select Launch Git Bash option and click on Finish.

INITIALIZE A GIT REPOSITORY

Step 1: Create a folder:

```
mkdir hello-world
```

Step 2: Change directory into the folder created and initialize the Directory

```
cd hello-world
git init
```

This will initialize a local repository. The .git folder is a special folder tracked by the git utility to track changes made and state of the repo.

THE FIRST COMMIT

Step 3: Add a file called README.md and add content use the command

```
notepad README.md
```

Step 4: Type some Content in the file

Step 5: Check status of git repo: `git status`

Here we can see that there is a new file that is currently untracked by git. You may make more changes to this new file.

Step 6: Get ready with your changes by staging it:

```
git add .
```

The dot represents the current folder and will add all the untracked changes in that folder to the staging area.

Step 7: Check the git status again: you can see that there are list of files which are ready to be committed.

Step 8: Commit

```
git commit -m "my first commit"
```

Committing the changes will add the changes to the branch of the local repository.

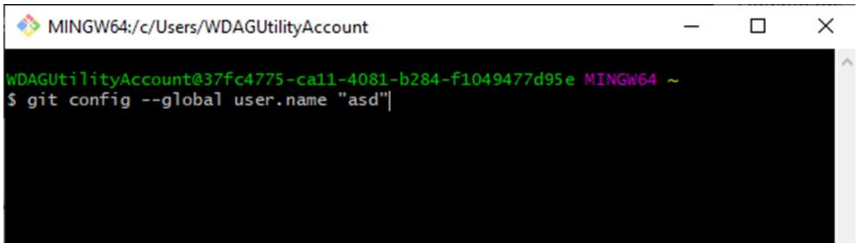
CONNECTING TO GITHUB AND CLONING THE APPLICATION DIRECTORY TO WORK WITH

Step 0: Create a GitHub account

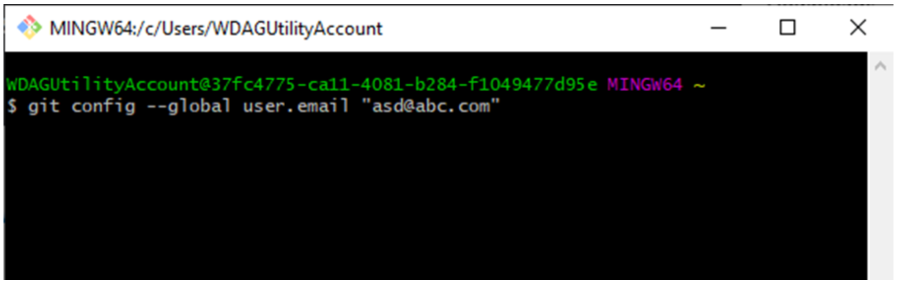
To be able to use GitHub, you will have to create an account first. You can do that on the website: http://github.com/

Step 1: Add Your Global Username and email of Git hub using following command:

```
git config --global user.name “<firstName lastName>”

git config --global user.email "<name@example.com>"
```

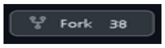


```
WDAGUtilityAccount@37fc4775-ca11-4081-b284-f1049477d95e MINGW64 ~
$ git config --global user.name "asd"
```



```
WDAGUtilityAccount@37fc4775-ca11-4081-b284-f1049477d95e MINGW64 ~
$ git config --global user.email "asd@abc.com"
```

Step 2: Fork the repository which you want to work on.

Click on the Fork  button fork to copy the repository to your account. This will be essential for deployment to Heroku.

Step 3: Goto your account and you can find the copy of your repository.

Step 4: Click on the code and copy the link present in it.

Step 5: open the Git Bash

Step 6: Cloning a Git Repo:

Locate to the directory you want to clone the repo:

```
cd D:/
mkdir To-doApp
cd To-doApp
```

Step 7: Copy the link of your repository which you want to work with:

```
git clone Pastethelinkhere
```

Step 8: use to following command to avoid ssh errors

```
git config http.sslVerify "false"
```

Step 9: Type the ls command to verify that you have the repository

```
ls
```

Step 10-: Enter the folder by typing

```
cd Todo-Backend/
```

Step 11: Type the ls command to verify the files inside the Todo-backend folder

```
ls
```

Step 12: Edit a README.md file and save it.

```
notepad README.md
```

Note : ***If you are using git with your Github account for the first time*** then you can click on this document , which details the steps to configure your git settings so that you can start pulling and pushing data to your remote repository

ADD AND COMMIT FILE

When we first initialized our project, the file was not being tracked by Git. To do that, we use this command git add .

The period or dot that comes after add means all the files that exist in the repository. If you want to add a specific file, maybe one named about.txt, you use git add about.txt

Step 1: **Add files to the Staging Area for commit:**

```
git add .
      # Adds all the files in the local repository and stages
      them for commit
      or
git add README.md
      # To add a specific file
```

Step 2: **Commit files in Git:** The next state for a file after the staged state is the committed state.

```
git commit -m "First commit"
 #The message in the " " is given so that the other users can read
 the message and see what changes you made
```

Step 3: **The git push command** pushes the changes in your local repository up to the remote repository you specified as the origin.

```
git push -u origin master #pushes changes to origin
```

NOTE: Each time you make changes that you want to be reflected on GitHub, the following are the most common flow of commands:

```
git add .
git status #Lists all new or modified files to be committed
git commit -m "Second commit"
git push -u origin master
```

Using Branches

- To check for all the branches in you project type

```
git branch
```

- To create a new branch, use the following command

```
git branch branchname
```

- To Switch between the branches, use the following command

```
git checkout branchnameyouwanttoswichto
```

SETUP HEROKU & POSTGRES ON HEROKU

INSTALL PGADMIN

- Download the latest version of pgAdmin:

  https://www.pgadmin.org/download/pgadmin-4-windows/

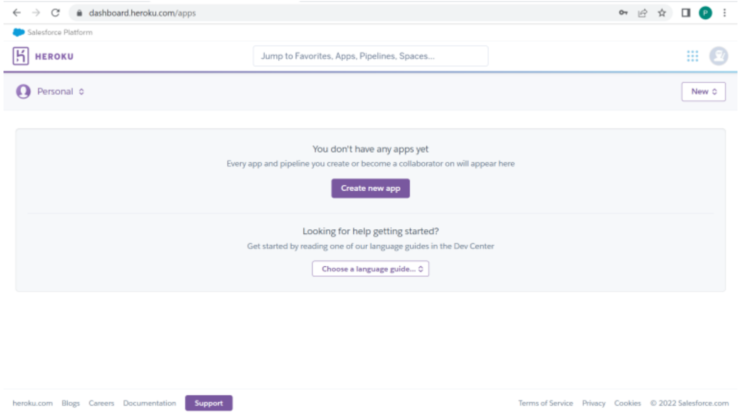
SETUP HEROKU

Step 0: Create an Account: https://signup.heroku.com/

Note: You must leave the Company Name blank.

Step 1: Login to Heroku and go to the Application dashboard

Step 2: Create a new app with the name get-my-db (any other name which is available)


dashboard.heroku.com/apps
Salesforce Platform
HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Personal
New
Create new app
Choose a language guide...
heroku.com Blogs Careers Documentation Support
Terms of Service Privacy Cookies © 2022 Salesforce.com

You don't have any apps yet

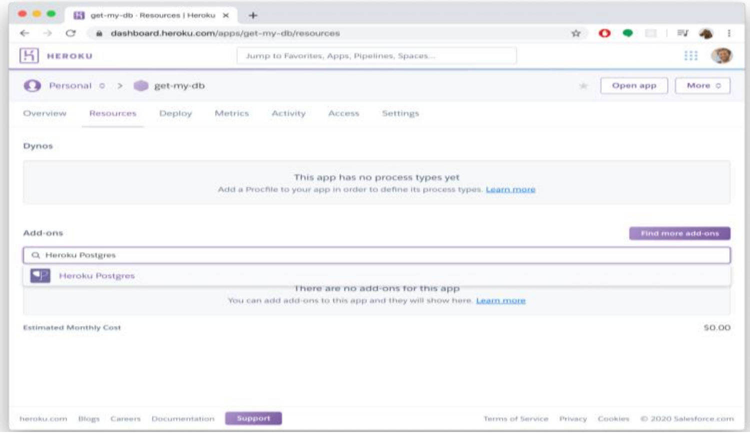
Every app and pipeline you create or become a collaborator on will appear here

Looking for help getting started?

Get started by reading one of our language guides in the Dev Center

Step 3: Navigate to the **Resources tab** of Application dashboard

Step 4: Add Heroku Postgres Add-on


get-my-db · Resources | Heroku
dashboard.heroku.com/apps/get-my-db/resources
HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Personal
get-my-db
Open app
More
Overview
Resources
Deploy
Metrics
Activity
Access
Settings
Dynos
Add-ons
Find more add-ons
heroku.com Blogs Careers Documentation Support
Terms of Service Privacy Cookies © 2020 Salesforce.com

This app has no process types yet

Add a Procfile to your app in order to define its process types. Learn more

Heroku Postgres

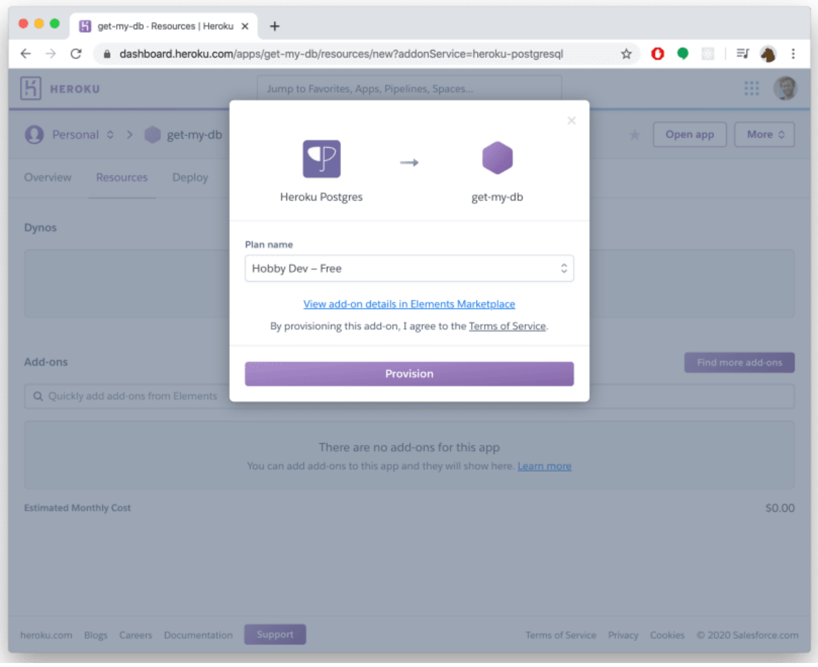
Heroku Postgres

There are no add-ons for this app

You can add add-ons to this app and they will show here. Learn more

Estimated Monthly Cost $0.00

Step 5: Select Hobby Dev - Free in the Plan Name Dropdown


get-my-db - Resources | Heroku
dashboard.heroku.com/apps/get-my-db/resources/new?addonService=heroku-postgresql
HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Personal
get-my-db
Open app
More
Overview
Resources
Deploy
Dynos
Add-ons
Find more add-ons
Quickly add add-ons from Elements
Heroku Postgres
get-my-db
Provision
heroku.com Blogs Careers Documentation Support
Terms of Service Privacy Cookies © 2020 Salesforce.com

Plan name: Hobby Dev – Free

View add-on details in Elements Marketplace

By provisioning this add-on, I agree to the Terms of Service.

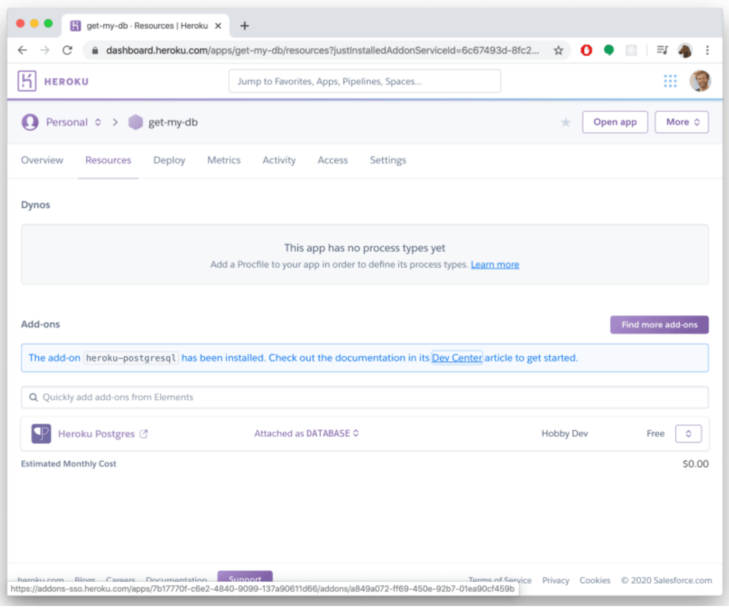
There are no add-ons for this app

You can add add-ons to this app and they will show here. Learn more

Estimated Monthly Cost $0.00

Step 6: Get the database credentials and connection URL

- Navigate to the Resources tab again
- Select Heroku Postgres



dashboard.heroku.com/apps/get-my-db/resources?justInstalledAddonServiceId=6c67493d-8fc2...

This app has no process types yet

Add a Procfile to your app in order to define its process types. Learn more

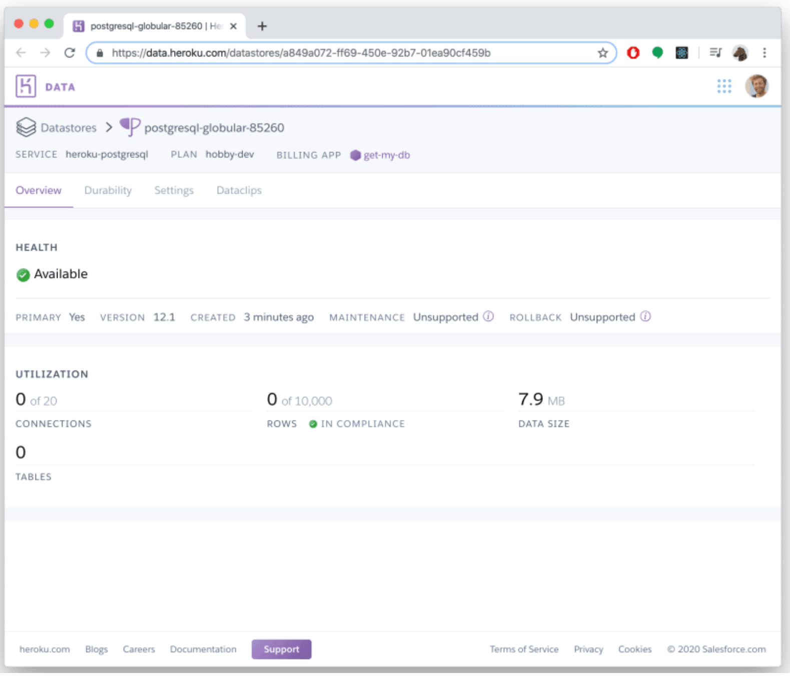
The add-on `heroku-postgresql` has been installed. Check out the documentation in its Dev Center article to get started.

Heroku Postgres	Attached as DATABASE	Hobby Dev	Free

Estimated Monthly Cost $0.00

https://addons-sso.heroku.com/apps/7b17770f-c6e2-4840-9099-137a90611d66/addons/a849a072-ff69-450e-92b7-01ea90cf459b

- Should look something like this



https://data.heroku.com/datastores/a849a072-ff69-450e-92b7-01ea90cf459b

SERVICE heroku-postgresql PLAN hobby-dev BILLING APP get-my-db

HEALTH

Available

PRIMARY Yes VERSION 12.1 CREATED 3 minutes ago MAINTENANCE Unsupported ROLLBACK Unsupported

UTILIZATION

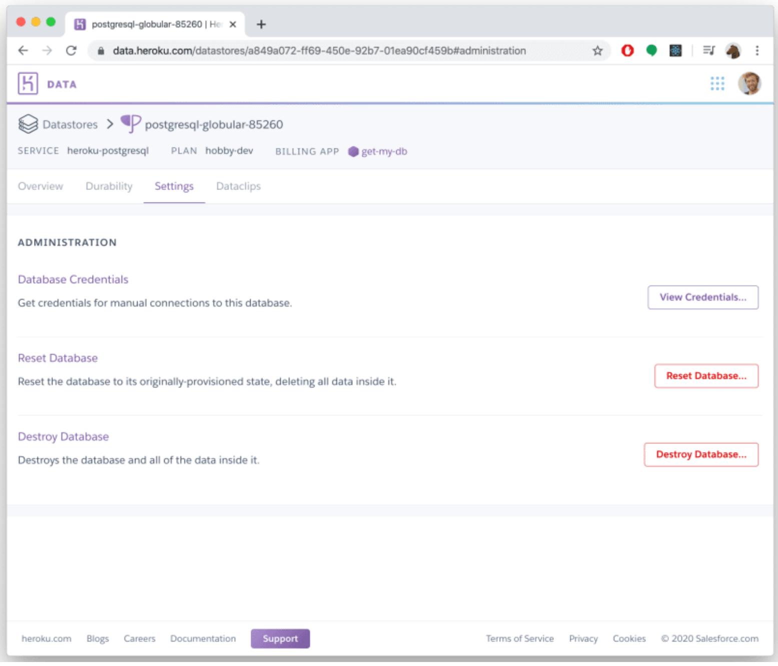
0 of 20 CONNECTIONS

0 of 10,000 ROWS IN COMPLIANCE

7.9 MB DATA SIZE

0 TABLES

- Select Settings Tab



postgresql-globular-85260 | He
data.heroku.com/datastores/a849a072-ff69-450e-92b7-01ea90cf459b#administration
DATA
Datastores > postgresql-globular-85260
SERVICE heroku-postgresql PLAN hobby-dev BILLING APP get-my-db
Overview
Durability
Settings
Dataclips
View Credentials...
Reset Database...
Destroy Database...
heroku.com Blogs Careers Documentation Support
Terms of Service Privacy Cookies © 2020 Salesforce.com

ADMINISTRATION

Database Credentials

Get credentials for manual connections to this database.

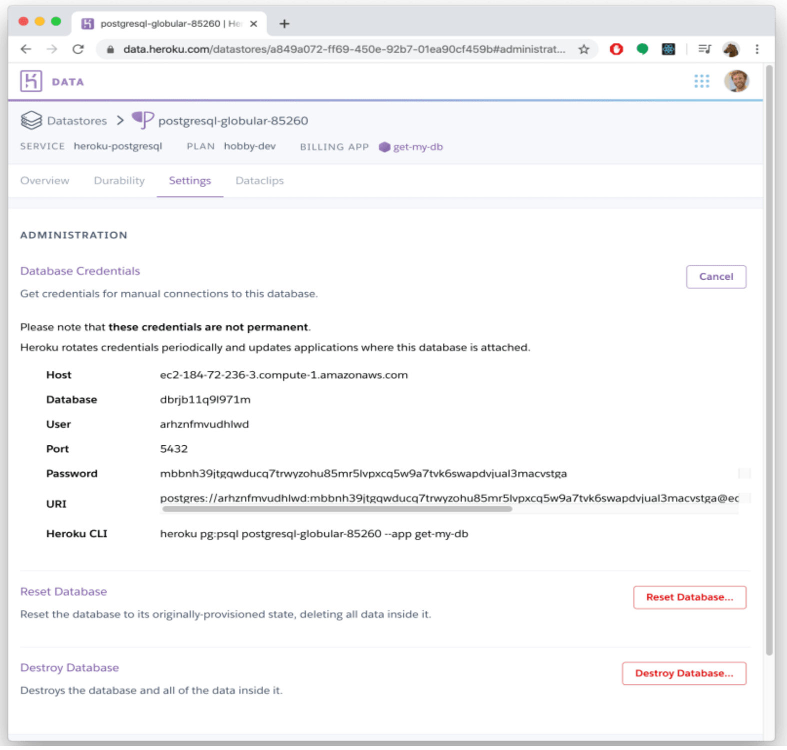
Reset Database

Reset the database to its originally-provisioned state, deleting all data inside it.

Destroy Database

Destroys the database and all of the data inside it.

- Click on View Credentials



postgresql-globular-85260 | He
data.heroku.com/datastores/a849a072-ff69-450e-92b7-01ea90cf459b#administrat...
DATA
Datastores > postgresql-globular-85260
SERVICE heroku-postgresql PLAN hobby-dev BILLING APP get-my-db
Overview
Durability
Settings
Dataclips
Cancel
Reset Database...
Destroy Database...

ADMINISTRATION

Database Credentials

Get credentials for manual connections to this database.

Please note that **these credentials are not permanent**.

Heroku rotates credentials periodically and updates applications where this database is attached.

Host	ec2-184-72-236-3.compute-1.amazonaws.com
Database	dbrjb11q9l971m
User	arhznfmvudhlwd
Port	5432
Password	mbbnh39jtgqwducq7trwyzohu85mr5lvpxcq5w9a7tvk6swapdvjual3macvstga
URI	postgres://arhznfmvudhlwd:mbbnh39jtgqwducq7trwyzohu85mr5lvpxcq5w9a7tvk6swapdvjual3macvstga@ec
Heroku CLI	heroku pg:psql postgresql-globular-85260 --app get-my-db

Reset Database

Reset the database to its originally-provisioned state, deleting all data inside it.

Destroy Database

Destroys the database and all of the data inside it.

- Note down the credentials in notepad or keep the tab open for the next steps.

Host:

Database:

User:

Port:

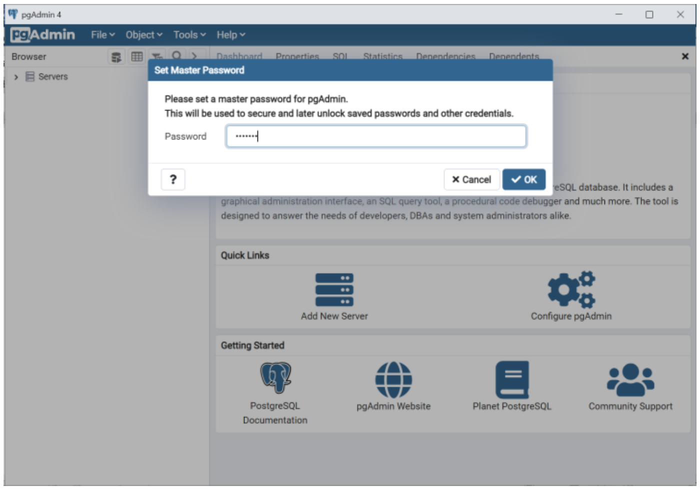
Password:

URI:

Heroku CLI:

Connect to the setup database from PG Admin

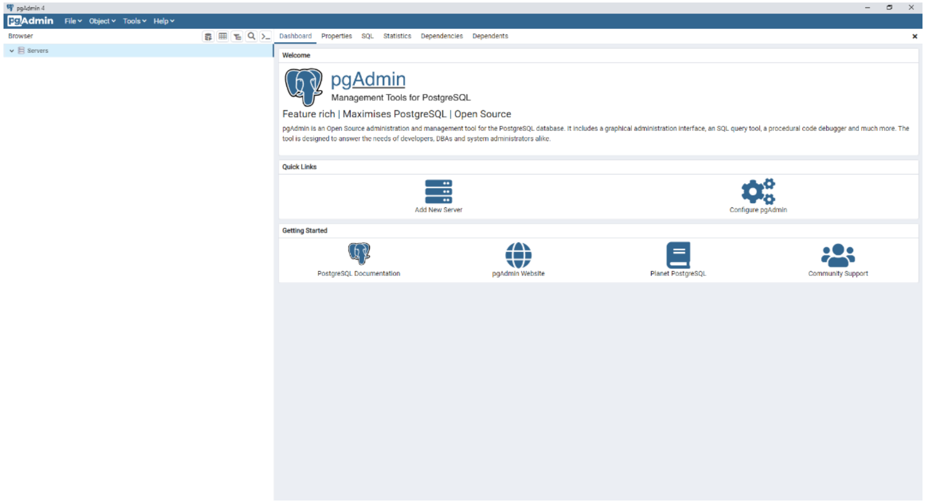
Step 7: Open pgAdmin and choose a master password.



Please set a master password for pgAdmin.
This will be used to secure and later unlock saved passwords and other credentials.

Password: ••••••

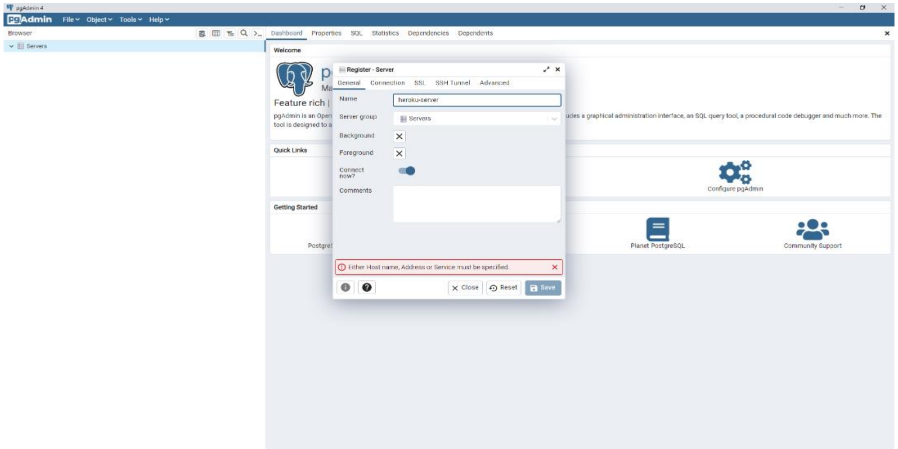
...eSQL database. It includes a graphical administration interface, an SQL query tool, a procedural code debugger and much more. The tool is designed to answer the needs of developers, DBAs and system administrators alike.



Feature rich | Maximises PostgreSQL | Open Source

pgAdmin is an Open Source administration and management tool for the PostgreSQL database. It includes a graphical administration interface, an SQL query tool, a procedural code debugger and much more. The tool is designed to answer the needs of developers, DBAs and system administrators alike.

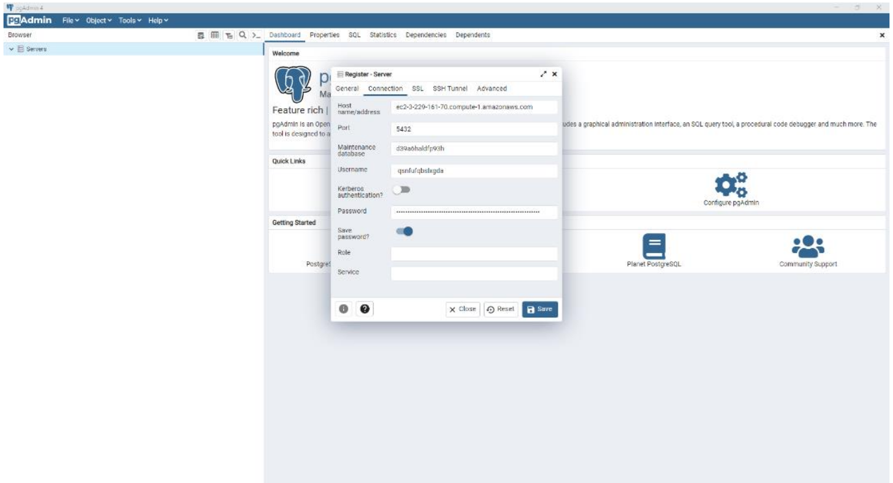
Step 8: Right-Click on Servers > Register > Server and enter a name of your choice



Name: heroku-server
Server group: Servers
Background: ☒
Foreground: ☒
Connect now?: ●
Comments:

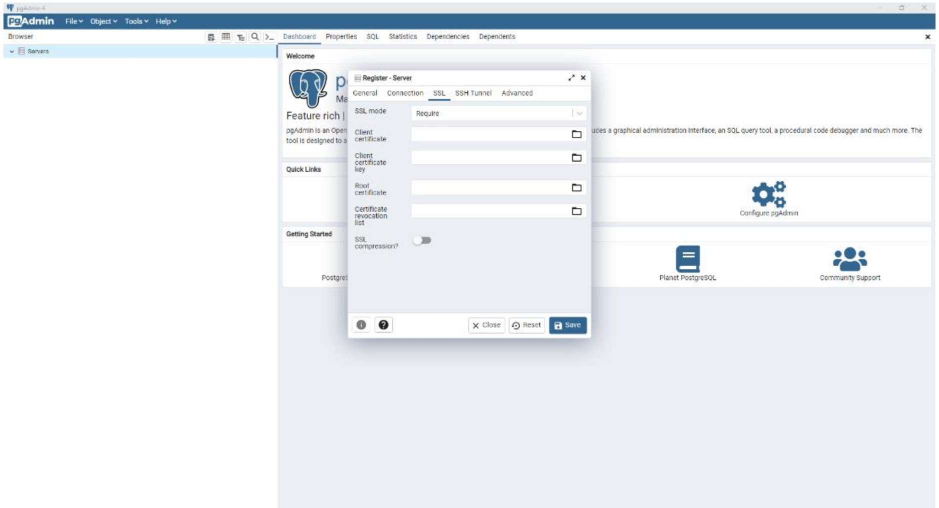
Either Host name, Address or Service must be specified.

Step 9: Navigate to the Connection Tab and enter the connection information collected at the end of Step 5



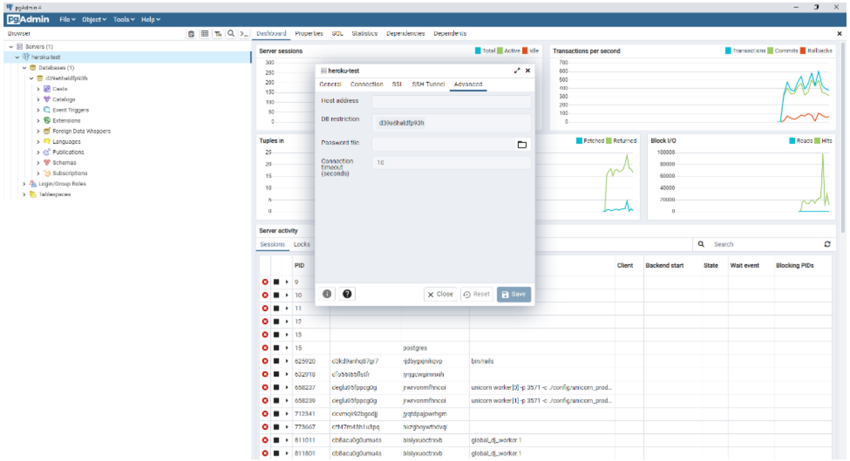
Host name/address: ec2-3-229-161-70.compute-1.amazonaws.com
Port: 5432
Maintenance database: d39a6haldfp93h
Username: qsnfufqbslxgda
Kerberos authentication?: ○
Password: ••
Save password?: ●
Role:
Service:

Step 10: Navigate to the SSL Tab and set SSL Mode as required



SSL mode: Require
Client certificate:
Client certificate key:
Root certificate:
Certificate revocation list:
SSL compression?: ○

Step 11: Navigate to the Advanced tab and set DB Restriction to the name of the database you previously copied at the end of step 5



heroku-test

General Connection SSL SSH Tunnel Advanced

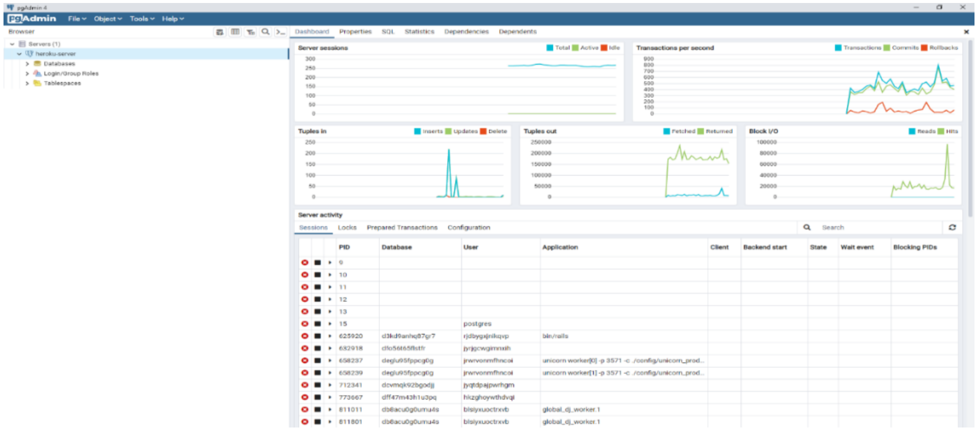
Host address:

DB restriction: d39a6haldfp93h

Password file:

Connection timeout (seconds): 10

Step 12: Click on Save to see a similar dashboard

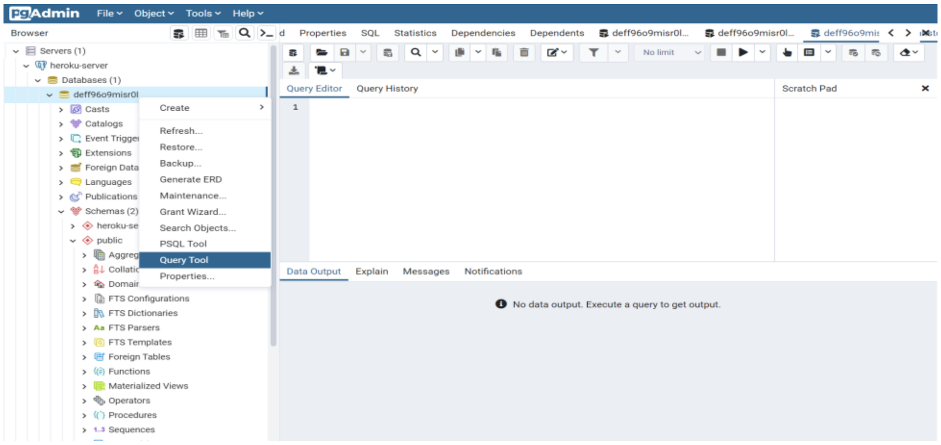


Server activity

PID	Database	User	Application	Client	Backend start	State	Wait event	Blocking PIDs
9								
10								
11								
12								
13								
15		postgres						
625920	d3kd9anhq87gr7	rjdbygxjnikqvp	bin/rails					
632918	dfo56t65flstfr	jyrjgcwgimnxih						
658237	deglu95fppcg0g	jrwrvonmfhncoi	unicorn worker[0] -p 3571 -c ./config/unicorn_prod...					
658239	deglu95fppcg0g	jrwrvonmfhncoi	unicorn worker[1] -p 3571 -c ./config/unicorn_prod...					
712341	dcvmqk92bgodjj	jyqtdpajpwrhgm						
773667	dff47m43h1u3pq	hkzghoywthdvqi						
811011	db6acu0g0umu4s	blsiyxuoctrxvb	global_dj_worker.1					
811801	db6acu0g0umu4s	blsiyxuoctrxvb	global_dj_worker.1					

Create a table using CREATE command

Step 1: Right click on the database click on Query Tool and type the queries.



No data output. Execute a query to get output.

Step 2: For our Database demonstration let us create TODOApp table. below are the command by which we can create the two tables:

```
CREATE TABLE TODOApp
(
   id  SERIAL PRIMARY KEY,
   title VARCHAR(50) NOT NULL,
   complete BOOLEAN,
   date_modified TIMESTAMP
);
```

BACKEND DEVELOPMENT

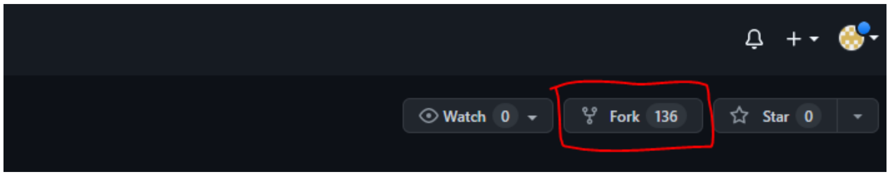
PREREQUISITES

Install your preferred IDE example Atom, VS Code, PyCharm

INSTALLING POSTMAN:

1. Go to: Postman (Download)
2. Select your OS and click on the Download option
3. Select and run the .exe file to install Postman.
4. -Refer to Postman Documentation for any queries
5. You can also use the web version of Postman (localhost requests will not work here)

FORKING THE REPOSITORY

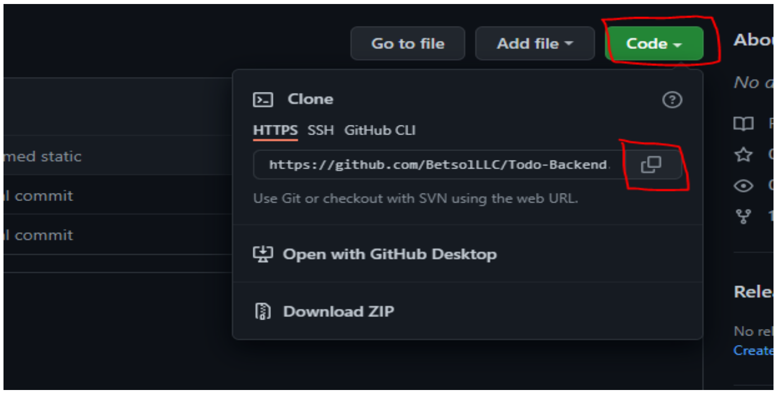


1. Click on the Fork Button on the Top-Right
2. Then go to yourUserName/Todo-Backend to clone the repository (Clone the code from the forked repository under your name. **NOT the original repository that you forked previously**).

CLONING THE REPOSITORY

Step 1: Go to YourUserName/Todo-Backend

Step 2: Click code and copy



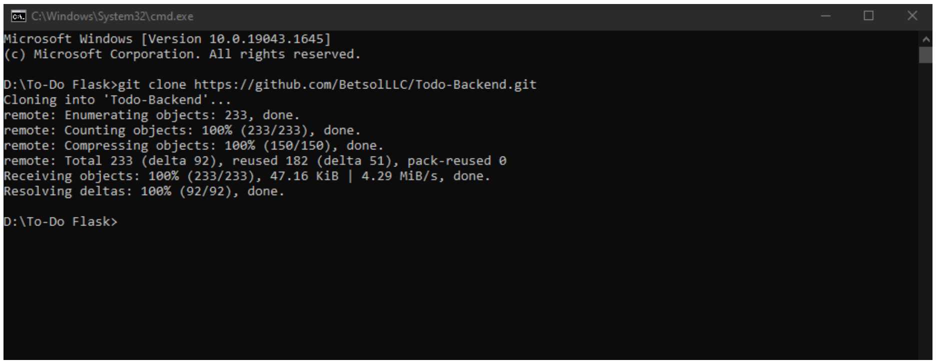
https://github.com/BetsolLLC/Todo-Backend

Use Git or checkout with SVN using the web URL.

Step3: Go to your desired folder preferably D drive and open the git-bash

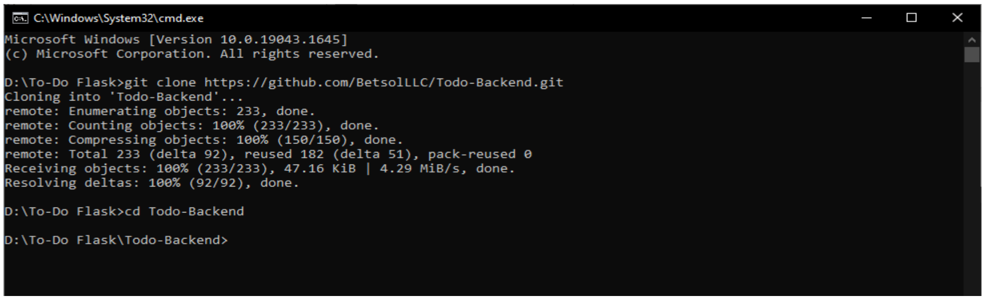
Step 4: Type the command

```
git clone <linkyoucopiedfromyourrepositary>
```



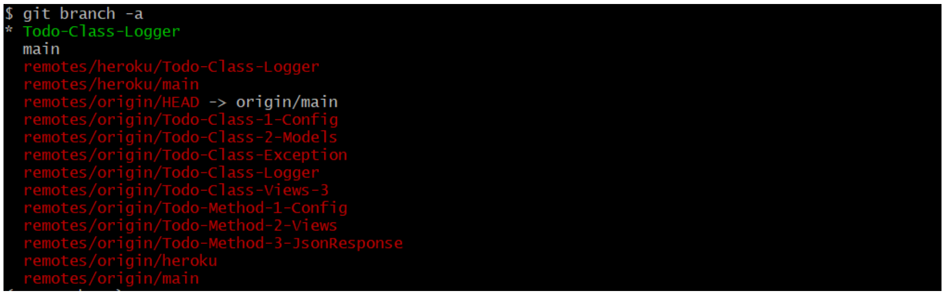
```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

D:\To-Do Flask>git clone https://github.com/BetsolLLC/Todo-Backend.git
Cloning into 'Todo-Backend'...
remote: Enumerating objects: 233, done.
remote: Counting objects: 100% (233/233), done.
remote: Compressing objects: 100% (150/150), done.
remote: Total 233 (delta 92), reused 182 (delta 51), pack-reused 0
Receiving objects: 100% (233/233), 47.16 KiB | 4.29 MiB/s, done.
Resolving deltas: 100% (92/92), done.

D:\To-Do Flask>
```

Step 5: Move into the folder using cd command



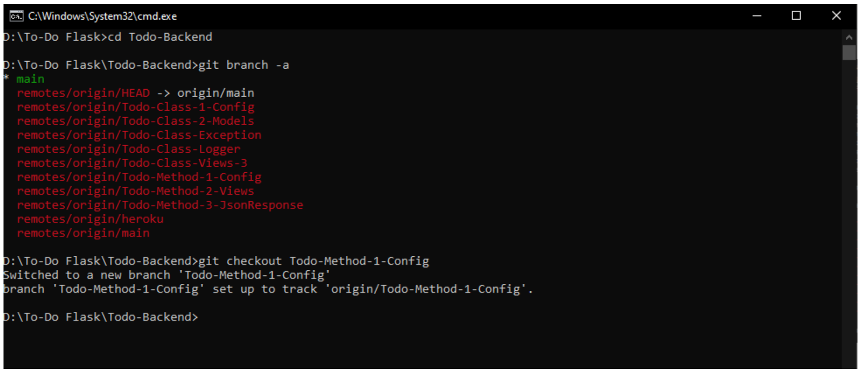
```
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

D:\To-Do Flask>git clone https://github.com/BetsolLLC/Todo-Backend.git
Cloning into 'Todo-Backend'...
remote: Enumerating objects: 233, done.
remote: Counting objects: 100% (233/233), done.
remote: Compressing objects: 100% (150/150), done.
remote: Total 233 (delta 92), reused 182 (delta 51), pack-reused 0
Receiving objects: 100% (233/233), 47.16 KiB | 4.29 MiB/s, done.
Resolving deltas: 100% (92/92), done.

D:\To-Do Flask>cd Todo-Backend

D:\To-Do Flask\Todo-Backend>
```

Step 6: Use `git branch -a` to see all branches



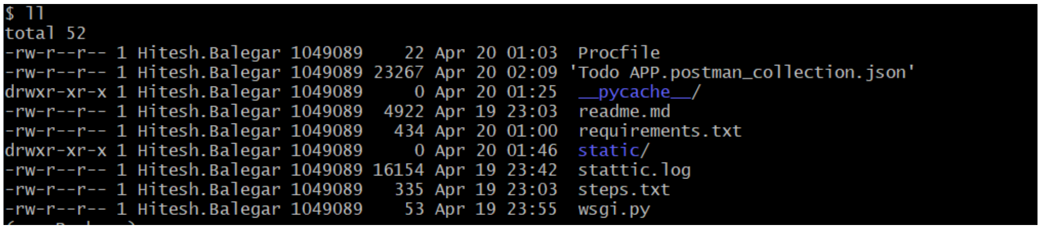
```
$ git branch -a
* Todo-Class-Logger
  main
  remotes/heroku/Todo-Class-Logger
  remotes/heroku/main
  remotes/origin/HEAD -> origin/main
  remotes/origin/Todo-Class-1-Config
  remotes/origin/Todo-Class-2-Models
  remotes/origin/Todo-Class-Exception
  remotes/origin/Todo-Class-Logger
  remotes/origin/Todo-Class-Views-3
  remotes/origin/Todo-Method-1-Config
  remotes/origin/Todo-Method-2-Views
  remotes/origin/Todo-Method-3-JsonResponse
  remotes/origin/heroku
  remotes/origin/main
```

Step 7: Use `git checkout Todo-Class-Logger` to change to the Todo-Class-Logger branch



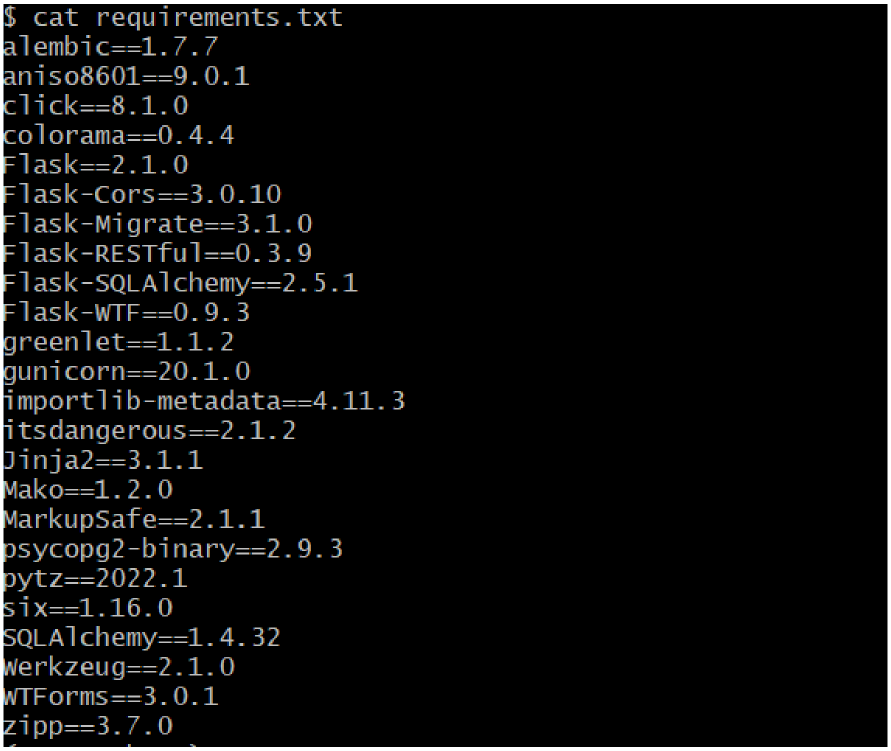
```
D:\To-Do Flask>cd Todo-Backend

D:\To-Do Flask\Todo-Backend>git branch -a
* main
  remotes/origin/HEAD -> origin/main
  remotes/origin/Todo-Class-1-Config
  remotes/origin/Todo-Class-2-Models
  remotes/origin/Todo-Class-Exception
  remotes/origin/Todo-Class-Logger
  remotes/origin/Todo-Class-Views-3
  remotes/origin/Todo-Method-1-Config
  remotes/origin/Todo-Method-2-Views
  remotes/origin/Todo-Method-3-JsonResponse
  remotes/origin/heroku
  remotes/origin/main

D:\To-Do Flask\Todo-Backend>git checkout Todo-Method-1-Config
Switched to a new branch 'Todo-Method-1-Config'
branch 'Todo-Method-1-Config' set up to track 'origin/Todo-Method-1-Config'.

D:\To-Do Flask\Todo-Backend>
```

Step 8: Once this is done, verify the files present in the directory to be same as the content given below.



```
$ ll
total 52
-rw-r--r-- 1 Hitesh.Balegar 1049089    22 Apr 20 01:03  Procfile
-rw-r--r-- 1 Hitesh.Balegar 1049089 23267 Apr 20 02:09 'Todo APP.postman_collection.json'
drwxr-xr-x 1 Hitesh.Balegar 1049089     0 Apr 20 01:25  __pycache__/
-rw-r--r-- 1 Hitesh.Balegar 1049089  4922 Apr 19 23:03  readme.md
-rw-r--r-- 1 Hitesh.Balegar 1049089   434 Apr 20 01:00  requirements.txt
drwxr-xr-x 1 Hitesh.Balegar 1049089     0 Apr 20 01:46  static/
-rw-r--r-- 1 Hitesh.Balegar 1049089 16154 Apr 19 23:42  stattic.log
-rw-r--r-- 1 Hitesh.Balegar 1049089   335 Apr 19 23:03  steps.txt
-rw-r--r-- 1 Hitesh.Balegar 1049089    53 Apr 19 23:55  wsgi.py
```

Step 9: Also verify the content present within the file cat requirements.txt to be same as given below.

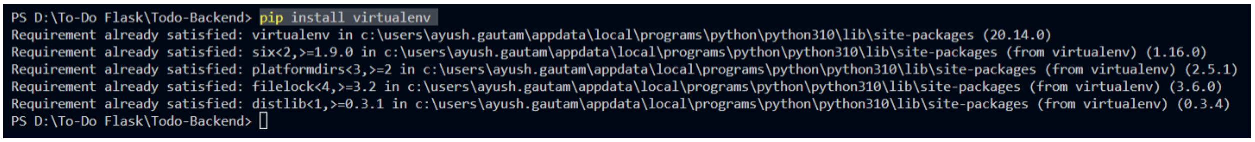


```
$ cat requirements.txt
alembic==1.7.7
aniso8601==9.0.1
click==8.1.0
colorama==0.4.4
Flask==2.1.0
Flask-Cors==3.0.10
Flask-Migrate==3.1.0
Flask-RESTful==0.3.9
Flask-SQLAlchemy==2.5.1
Flask-WTF==0.9.3
greenlet==1.1.2
gunicorn==20.1.0
importlib-metadata==4.11.3
itsdangerous==2.1.2
Jinja2==3.1.1
Mako==1.2.0
MarkupSafe==2.1.1
psycopg2-binary==2.9.3
pytz==2022.1
six==1.16.0
SQLAlchemy==1.4.32
Werkzeug==2.1.0
WTForms==3.0.1
zipp==3.7.0
```

Once verified, we are good to go. Hence forth the branch **"Todo-Class-Logger"** would be used to demonstrate **Class-based views approach** to a Flask application and if interested you can switch the branch **"Todo-Method-3-JsonResponse"** to view **Method-based approach**.

CREATING A VIRTUAL ENVIRONMENT

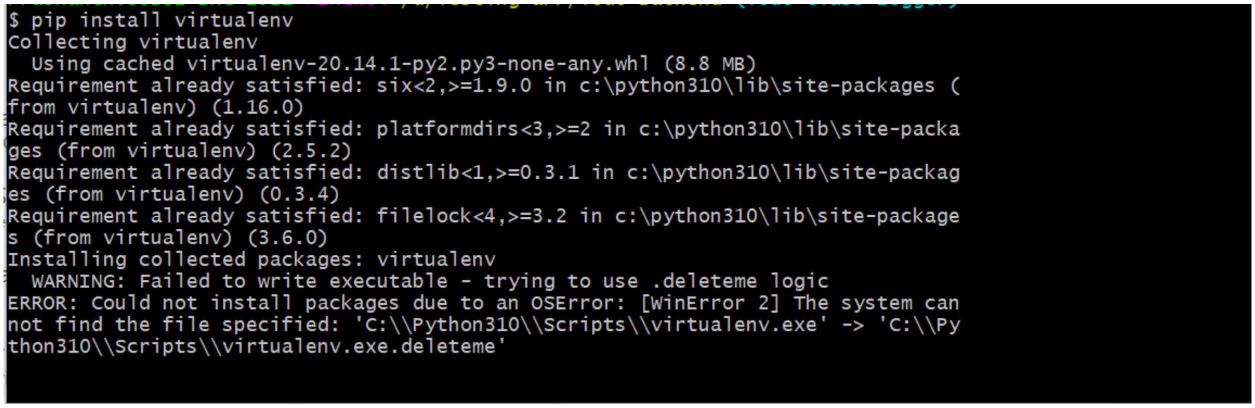
Step 1: Use pip install virtualenv to install virtualenv module



```
PS D:\To-Do Flask\Todo-Backend> pip install virtualenv
Requirement already satisfied: virtualenv in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (20.14.0)
Requirement already satisfied: six<2,>=1.9.0 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from virtualenv) (1.16.0)
Requirement already satisfied: platformdirs<3,>=2 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from virtualenv) (2.5.1)
Requirement already satisfied: filelock<4,>=3.2 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from virtualenv) (3.6.0)
Requirement already satisfied: distlib<1,>=0.3.1 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from virtualenv) (0.3.4)
PS D:\To-Do Flask\Todo-Backend> 
```

Note: If the above command fails with an error output similar to the one below. Then you can continue to step 4 directly without activating the environment. Just make sure to use pip instead of pipenv

Ex: `pipenv install -r requirements.txt` → `pip install – r requirements.txt`

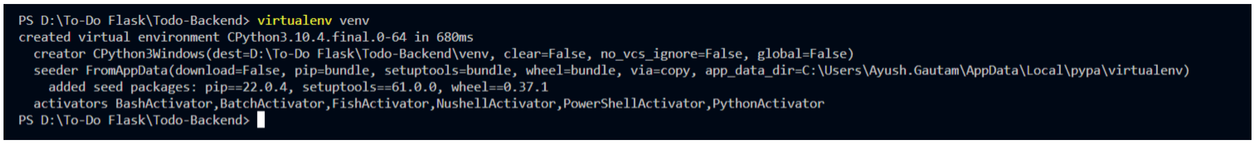


```
$ pip install virtualenv
Collecting virtualenv
  Using cached virtualenv-20.14.1-py2.py3-none-any.whl (8.8 MB)
Requirement already satisfied: six<2,>=1.9.0 in c:\python310\lib\site-packages (
from virtualenv) (1.16.0)
Requirement already satisfied: platformdirs<3,>=2 in c:\python310\lib\site-packa
ges (from virtualenv) (2.5.2)
Requirement already satisfied: distlib<1,>=0.3.1 in c:\python310\lib\site-packag
es (from virtualenv) (0.3.4)
Requirement already satisfied: filelock<4,>=3.2 in c:\python310\lib\site-package
s (from virtualenv) (3.6.0)
Installing collected packages: virtualenv
  WARNING: Failed to write executable - trying to use .deleteme logic
ERROR: Could not install packages due to an OSError: [WinError 2] The system can
not find the file specified: 'C:\\Python310\\Scripts\\virtualenv.exe' -> 'C:\\Py
thon310\\Scripts\\virtualenv.exe.deleteme'
```

Figure 1 shows one of the possible errors during the installation of virutalenv

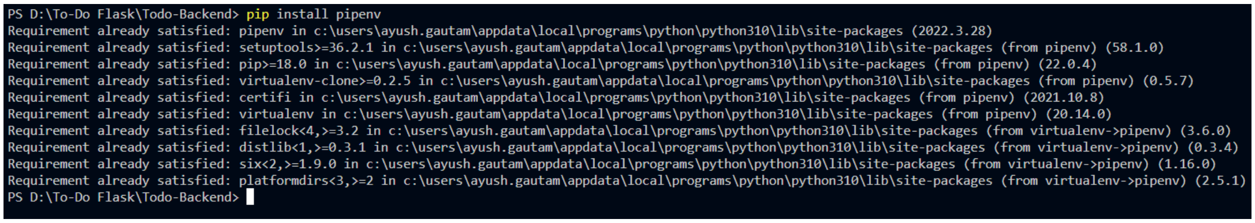
VISUAL STUDIO CODE

Step 1: Execute `virtualenv <yourVenvName>`



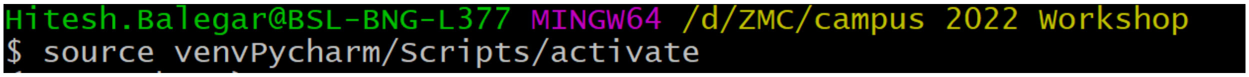
```
PS D:\To-Do Flask\Todo-Backend> virtualenv venv
created virtual environment CPython3.10.4.final.0-64 in 680ms
  creator CPython3Windows(dest=D:\To-Do Flask\Todo-Backend\venv, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=C:\Users\Ayush.Gautam\AppData\Local\pypa\virtualenv)
    added seed packages: pip==22.0.4, setuptools==61.0.0, wheel==0.37.1
  activators BashActivator,BatchActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
PS D:\To-Do Flask\Todo-Backend> █
```

Step 2: Type the command `pip install pipenv` to install pipenv module



```
PS D:\To-Do Flask\Todo-Backend> pip install pipenv
Requirement already satisfied: pipenv in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (2022.3.28)
Requirement already satisfied: setuptools>=36.2.1 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from pipenv) (58.1.0)
Requirement already satisfied: pip>=18.0 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from pipenv) (22.0.4)
Requirement already satisfied: virtualenv-clone>=0.2.5 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from pipenv) (0.5.7)
Requirement already satisfied: certifi in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from pipenv) (2021.10.8)
Requirement already satisfied: virtualenv in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from pipenv) (20.14.0)
Requirement already satisfied: filelock<4,>=3.2 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from virtualenv->pipenv) (3.6.0)
Requirement already satisfied: distlib<1,>=0.3.1 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from virtualenv->pipenv) (0.3.4)
Requirement already satisfied: six<2,>=1.9.0 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from virtualenv->pipenv) (1.16.0)
Requirement already satisfied: platformdirs<3,>=2 in c:\users\ayush.gautam\appdata\local\programs\python\python310\lib\site-packages (from virtualenv->pipenv) (2.5.1)
PS D:\To-Do Flask\Todo-Backend> █
```

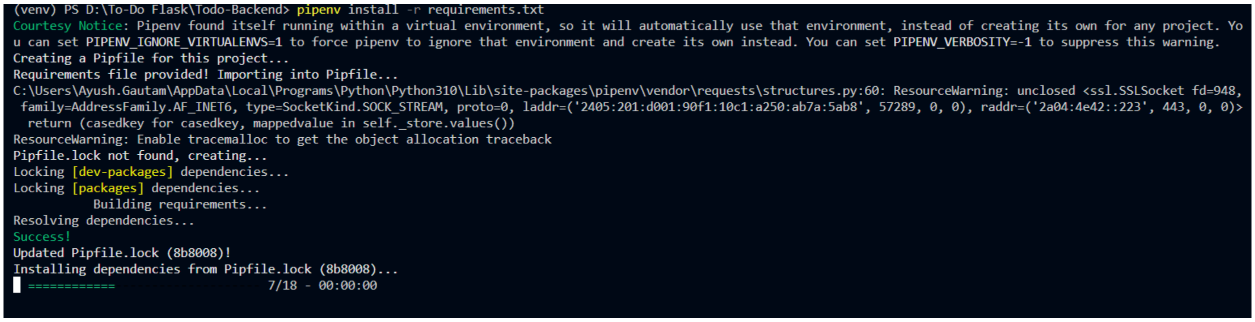
Step 3: Type `source venv/Scripts/activate` to activate a virtual environment



```
Hitesh.Balegar@BSL-BNG-L377 MINGW64 /d/ZMC/campus 2022 Workshop
$ source venvPycharm/Scripts/activate
```

Note: The args after `source "venvPycharm/Scripts/activate"` , has *venvPycharm* as the virtual env name , this can be different for your local setup depending on the name you give.

Step 4: Then `<pipenv install -r requirements.txt>` to install all the dependencies



```
(venv) PS D:\To-Do Flask\Todo-Backend> pipenv install -r requirements.txt
Courtesy Notice: Pipenv found itself running within a virtual environment, so it will automatically use that environment, instead of creating its own for any project. Yo
u can set PIPENV_IGNORE_VIRTUALENVS=1 to force pipenv to ignore that environment and create its own instead. You can set PIPENV_VERBOSITY=-1 to suppress this warning.
Creating a Pipfile for this project...
Requirements file provided! Importing into Pipfile...
C:\Users\Ayush.Gautam\AppData\Local\Programs\Python\Python310\Lib\site-packages\pipenv\vendor\requests\structures.py:60: ResourceWarning: unclosed <ssl.SSLSocket fd=948,
 family=AddressFamily.AF_INET6, type=SocketKind.SOCK_STREAM, proto=0, laddr=('2405:201:d001:90f1:10c1:a250:ab7a:5ab8', 57289, 0, 0), raddr=('2a04:4e42::223', 443, 0, 0)>
  return (casedkey for casedkey, mappedvalue in self._store.values())
ResourceWarning: Enable tracemalloc to get the object allocation traceback
Pipfile.lock not found, creating...
Locking [dev-packages] dependencies...
Locking [packages] dependencies...
           Building requirements...
Resolving dependencies...
Success!
Updated Pipfile.lock (8b8008)!
Installing dependencies from Pipfile.lock (8b8008)...
█ ============-------------------- 7/18 - 00:00:00
```

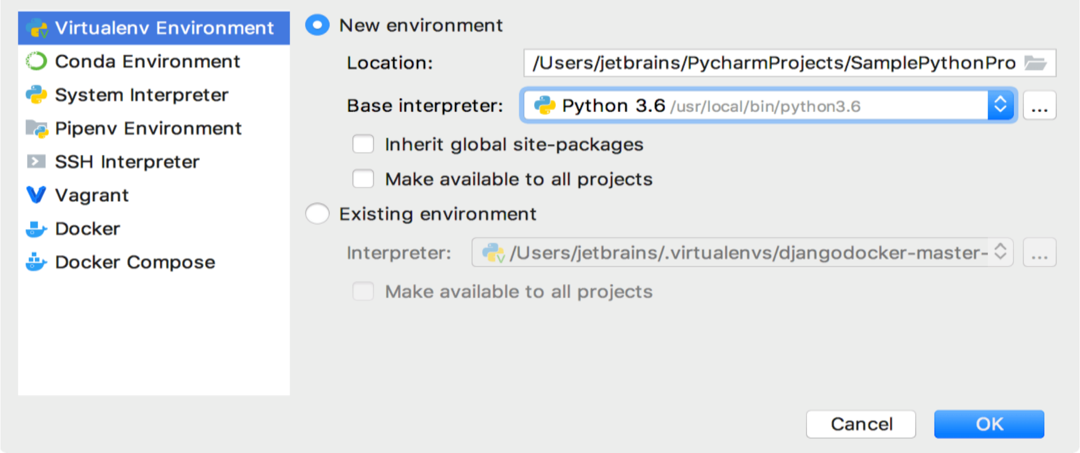
Note: For issues related to python setup not being in your **PATH** under your system environment variable you can click this document.

PYCHARM

Use one of the following:

- Click the Python Interpreter selector and choose Add Interpreter.
- Press Ctrl+Alt+S to open the project Settings/Preferences and go to Project <project name> | Python Interpreter. Click and select Add.

In the left-hand pane of the Add Python Interpreter dialog, select Virtualenv Environment. The following actions depend on whether the virtual environment existed before.



● New environment

Location: /Users/jetbrains/PycharmProjects/SamplePythonPro

Base interpreter: Python 3.6 /usr/local/bin/python3.6

□ Inherit global site-packages

□ Make available to all projects

○ Existing environment

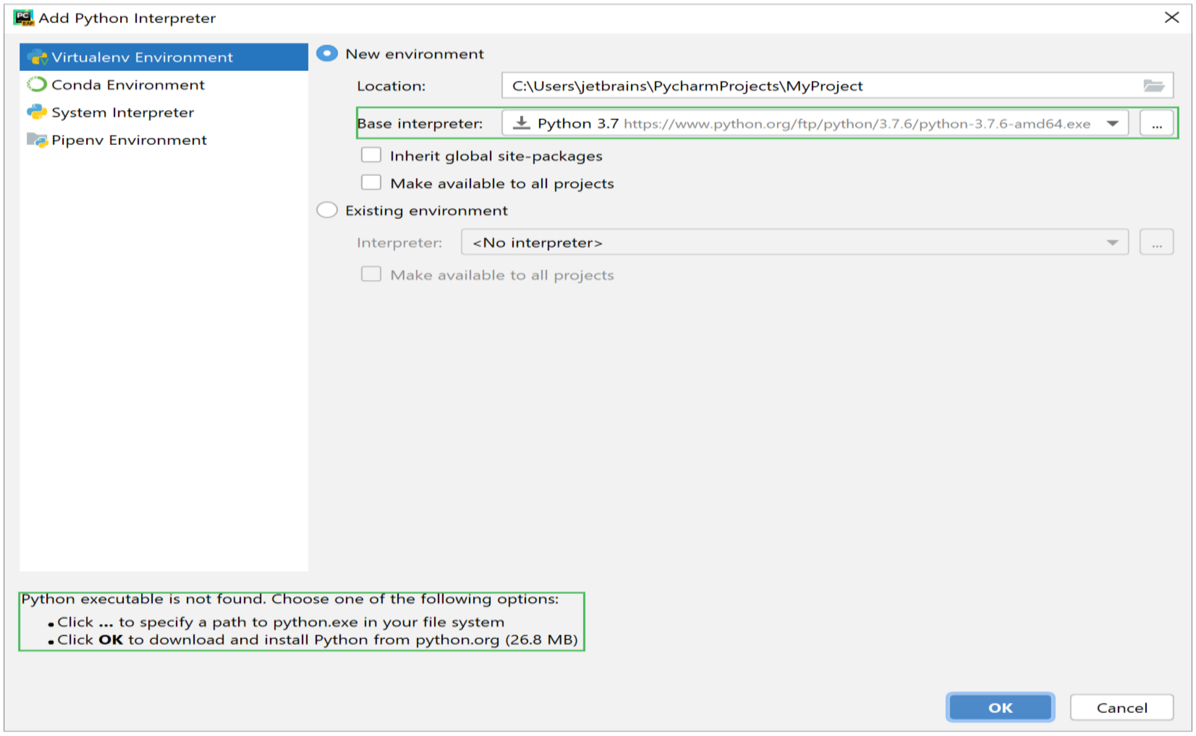
Interpreter: /Users/jetbrains/.virtualenvs/djangodocker-master-

□ Make available to all projects

If New Virtualenv is selected:

- Specify the location of the new virtual environment in the text field, or click … and find location in your file system. Note that the directory where the new virtual environment should be located, must be empty!
- Choose the base interpreter from the list, or click … and find a Python executable in your file system.

If PyCharm detects no Python on your machine, it provides two options: to download the latest Python versions from **python.org** or to specify a path to the Python executable (in case of non-standard installation).



Add Python Interpreter

● New environment

Location: C:\Users\jetbrains\PycharmProjects\MyProject

Base interpreter: Python 3.7 https://www.python.org/ftp/python/3.7.6/python-3.7.6-amd64.exe

□ Inherit global site-packages

□ Make available to all projects

○ Existing environment

Interpreter: <No interpreter>

□ Make available to all projects

Python executable is not found. Choose one of the following options:

- Click **...** to specify a path to python.exe in your file system
- Click **OK** to download and install Python from python.org (26.8 MB)

- Select the Inherit global site-packages checkbox if you want that all packages installed in the global Python on your machine to be added to the virtual environment you're going to create. This checkbox corresponds to the --system-site-packages option of the virtualenv tool.

If you select any of the existing virtual environments from the Interpreter list, it will be reused for the current project.

Finally Click OK to complete the task.

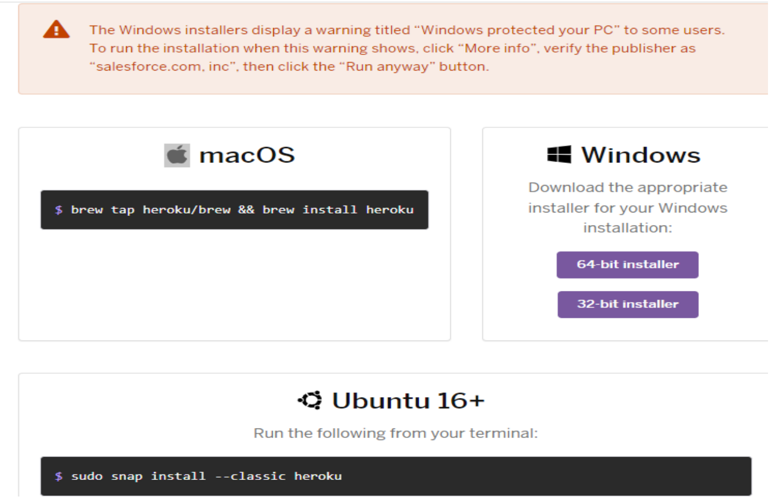
WORKING ON THE TO-DO APP

INSTALL HEROKU CLI

Once the above steps are done, we can proceed with deploying backend on Heroku. For this, follow the below steps:

1. Install Heroku CLI from https://devcenter.heroku.com/articles/heroku-cli using the appropriate method based on your OS.

2.Go for the **64-bit installer for Windows Operating System**.



The Windows installers display a warning titled "Windows protected your PC" to some users. To run the installation when this warning shows, click "More info", verify the publisher as "salesforce.com, inc", then click the "Run anyway" button.

macOS

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

Windows

Download the appropriate installer for your Windows installation:

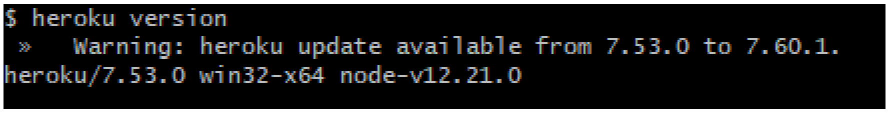
Ubuntu 16+

Run the following from your terminal:

```
$ sudo snap install --classic heroku
```

3.Verify the installation using command

```
heroku version
```



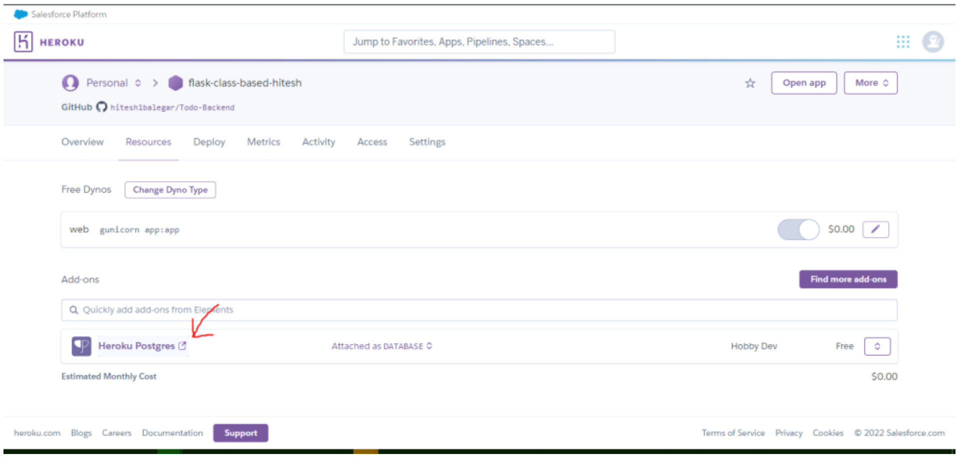
```
$ heroku version
 »   Warning: heroku update available from 7.53.0 to 7.60.1.
heroku/7.53.0 win32-x64 node-v12.21.0
```

Note: PS: you can ignore the warning if your version is >=0.53

MODIFY THE DATABASE URI

The URL for the projects under app.py in method based or __init__.py in class methods are the same URI that you receive from the "Heroku: Resource:YOUR_CUSTOM_postgres_app" with a small exception of adding

Step 1: Get to the "Resources" section from you created app on Heroku.

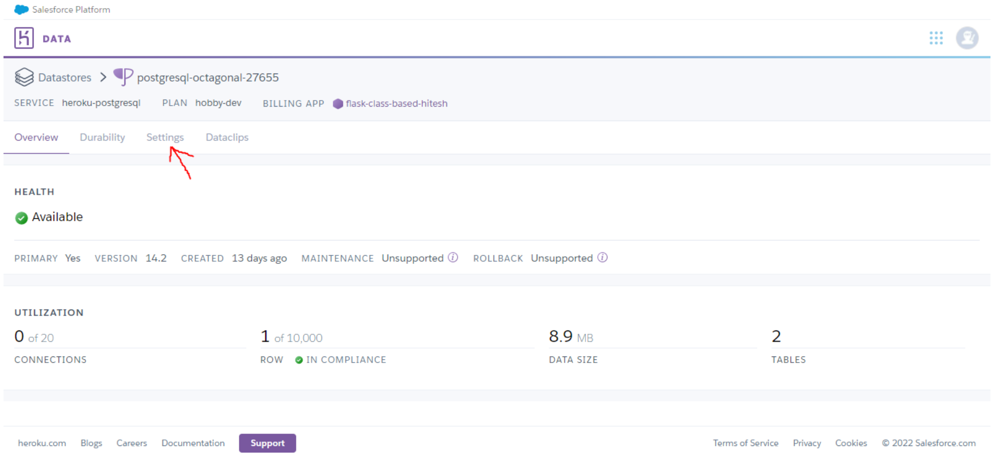


web gunicorn app:app $0.00

Heroku Postgres Attached as DATABASE Hobby Dev Free

Estimated Monthly Cost $0.00

Step 2: Once there click on the resource which would open a new tab.



SERVICE heroku-postgresql PLAN hobby-dev BILLING APP flask-class-based-hitesh

HEALTH

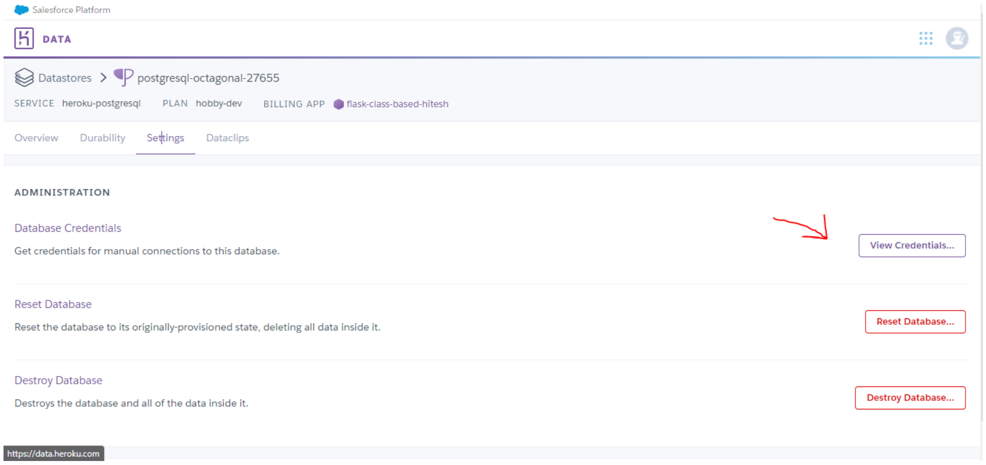
Available

PRIMARY Yes VERSION 14.2 CREATED 13 days ago MAINTENANCE Unsupported ⓘ ROLLBACK Unsupported ⓘ

UTILIZATION

0 of 20	1 of 10,000	8.9 MB	2
CONNECTIONS	ROW ● IN COMPLIANCE	DATA SIZE	TABLES

Step 3: Click on settings



SERVICE heroku-postgresql PLAN hobby-dev BILLING APP flask-class-based-hitesh

ADMINISTRATION

Database Credentials

Get credentials for manual connections to this database.

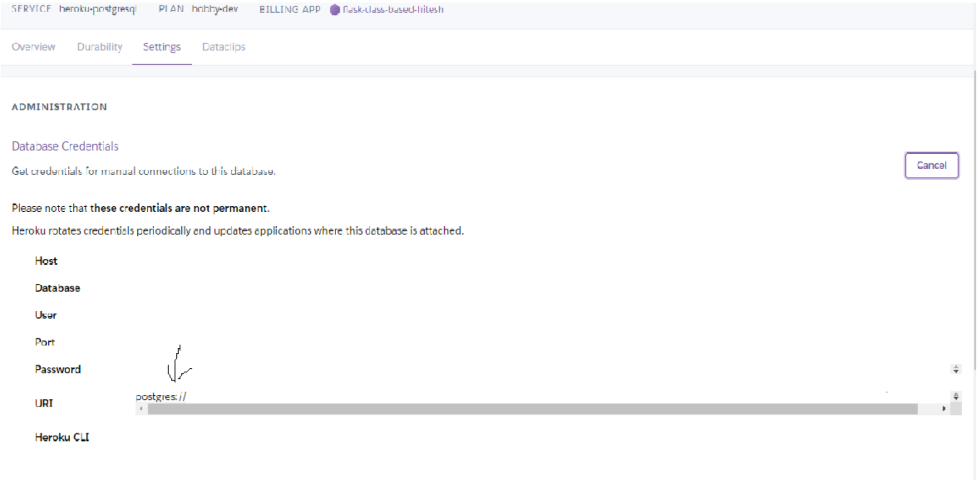
Reset Database

Reset the database to its originally-provisioned state, deleting all data inside it.

Destroy Database

Destroys the database and all of the data inside it.

Step 4: Once you click on the “View credentials” button, you would be directed to a new tab which would have all the Database URI required by the app to connect to the database present on Heroku



SERVICE heroku-postgresql
PLAN hobby-dev
BILLING APP flask-class-based-hitesh
Overview
Durability
Settings
Dataclips
Cancel

ADMINISTRATION

Database Credentials

Get credentials for manual connections to this database.

Please note that **these credentials are not permanent.**

Heroku rotates credentials periodically and updates applications where this database is attached.

Host

Database

User

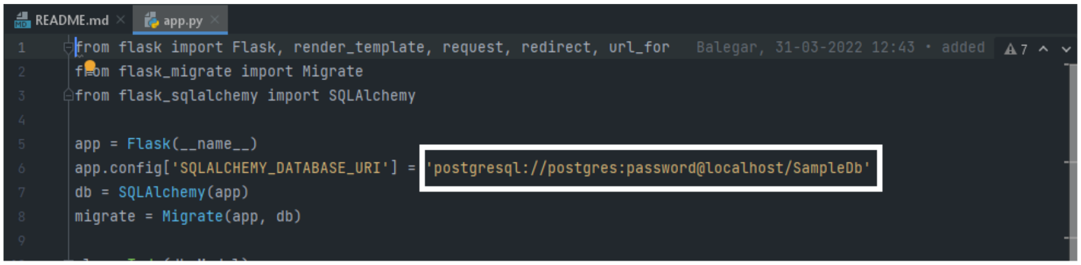
Port

Password

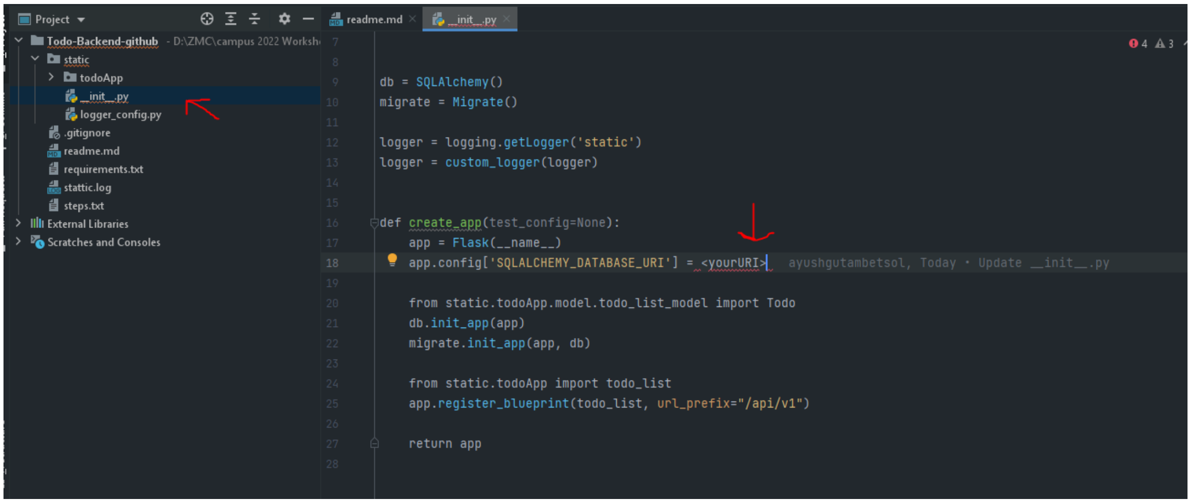
URI postgres://

Heroku CLI

Step 5: Once you have the URI which would in the format like "postgresql://__CREDENTIALS__", you need to replace the once present your app.py in method-based views and "__init__.py" in class-based views.



```
from flask import Flask, render_template, request, redirect, url_for
from flask_migrate import Migrate
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'postgresql://postgres:password@localhost/SampleDb'
db = SQLAlchemy(app)
migrate = Migrate(app, db)
```



```


db = SQLAlchemy()
migrate = Migrate()

logger = logging.getLogger('static')
logger = custom_logger(logger)


def create_app(test_config=None):
    app = Flask(__name__)
    app.config['SQLALCHEMY_DATABASE_URI'] = <yourURI>

    from static.todoApp.model.todo_list_model import Todo
    db.init_app(app)
    migrate.init_app(app, db)

    from static.todoApp import todo_list
    app.register_blueprint(todo_list, url_prefix="/api/v1")

    return app
```

Step 6: Change the URI from

postgres://user:password@localhost/DbName

to

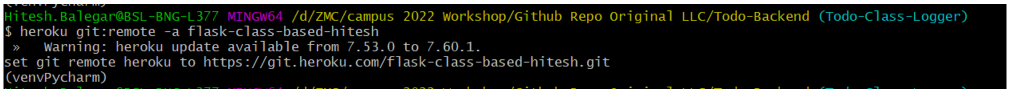
postgresql://user:password@localhost/DbName

DEPLOYING THE APP

Once you have the changes saved into the corresponding file. It's time for deployment, the procedure given here would be save for the class-based views or method-based-views.

Step 2: To deploy our app on Heroku now, attach it to the Heroku remote repository that was created for our Heroku app created recently. Use the command –

```
heroku git:remote -a $APP_NAME
```



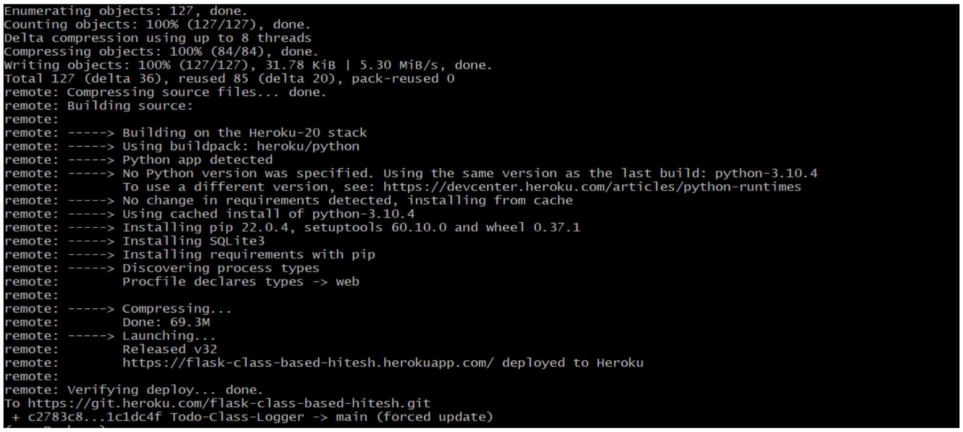
```
Hitesh.Balegar@BSL-BNG-L377 MINGW64 /d/ZMC/campus 2022 Workshop/Github Repo Original LLC/Todo-Backend (Todo-Class-Logger)
$ heroku git:remote -a flask-class-based-hitesh
 »   Warning: heroku update available from 7.53.0 to 7.60.1.
set git remote heroku to https://git.heroku.com/flask-class-based-hitesh.git
(venvPycharm)
```

Step 2: Commit the changes made to the respective file for the class-based views or method-based-views using

```
git add.
git commit -m "YOUR_MESSAGE"
```

Step 3: Push your changes to the remote branch and trigger a build

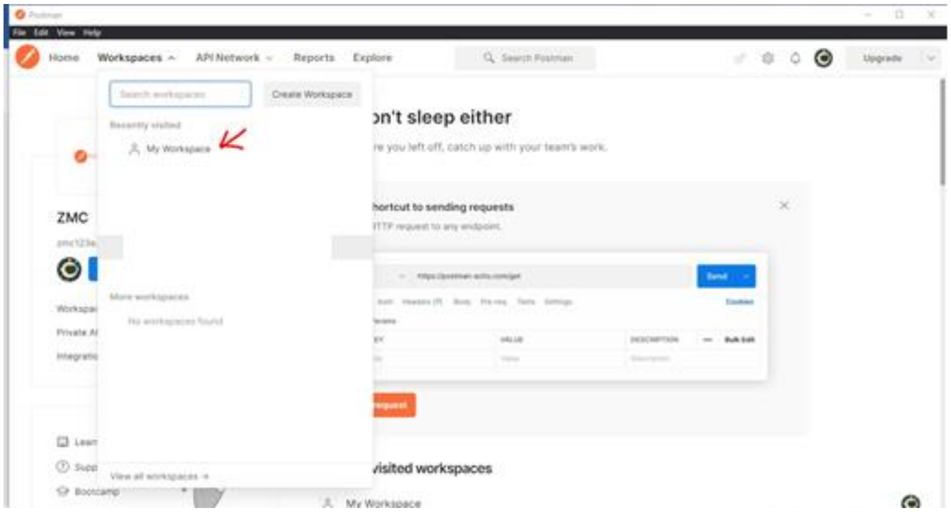
```
git push heroku Todo-Class-Logger:main
```



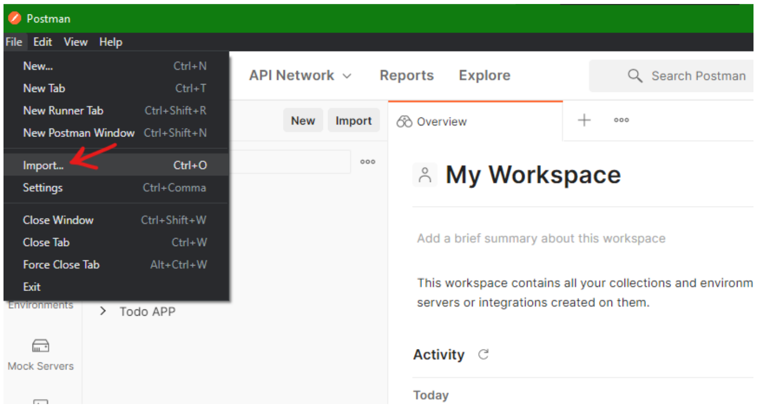
```
Enumerating objects: 127, done.
Counting objects: 100% (127/127), done.
Delta compression using up to 8 threads
Compressing objects: 100% (84/84), done.
Writing objects: 100% (127/127), 31.78 KiB | 5.30 MiB/s, done.
Total 127 (delta 36), reused 85 (delta 20), pack-reused 0
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Building on the Heroku-20 stack
remote: -----> Using buildpack: heroku/python
remote: -----> Python app detected
remote: -----> No Python version was specified. Using the same version as the last build: python-3.10.4
remote:        To use a different version, see: https://devcenter.heroku.com/articles/python-runtimes
remote: -----> No change in requirements detected, installing from cache
remote: -----> Using cached install of python-3.10.4
remote: -----> Installing pip 22.0.4, setuptools 60.10.0 and wheel 0.37.1
remote: -----> Installing SQLite3
remote: -----> Installing requirements with pip
remote: -----> Discovering process types
remote:        Procfile declares types -> web
remote:
remote: -----> Compressing...
remote:        Done: 69.3M
remote: -----> Launching...
remote:        Released v32
remote:        https://flask-class-based-hitesh.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/flask-class-based-hitesh.git
 + c2783c8...1c1dc4f Todo-Class-Logger -> main (forced update)
```

Step 4: Once this is done you can start using postman to hit and try out these URLs.

Step 5: Open you Postman and either go to workspace -> My Workspace or create a new workspace from the same option.

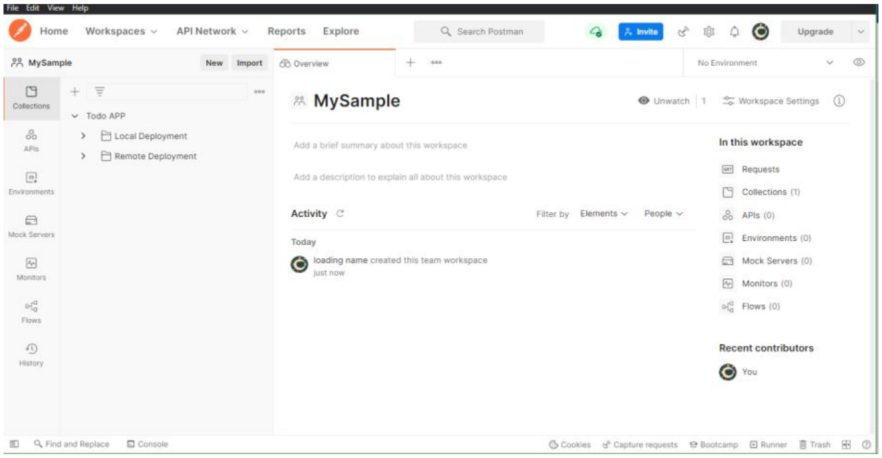


Step 6: Once selected click on file -> Import


Postman
File
Edit
View
Help
New... Ctrl+N
New Tab Ctrl+T
New Runner Tab Ctrl+Shift+R
New Postman Window Ctrl+Shift+N
Import... Ctrl+O
Settings Ctrl+Comma
Close Window Ctrl+Shift+W
Close Tab Ctrl+W
Force Close Tab Alt+Ctrl+W
Exit
API Network
Reports
Explore
Search Postman
New
Import
Overview
My Workspace
Add a brief summary about this workspace
This workspace contains all your collections and environm
servers or integrations created on them.
Activity
Today
Environments
Todo APP
Mock Servers

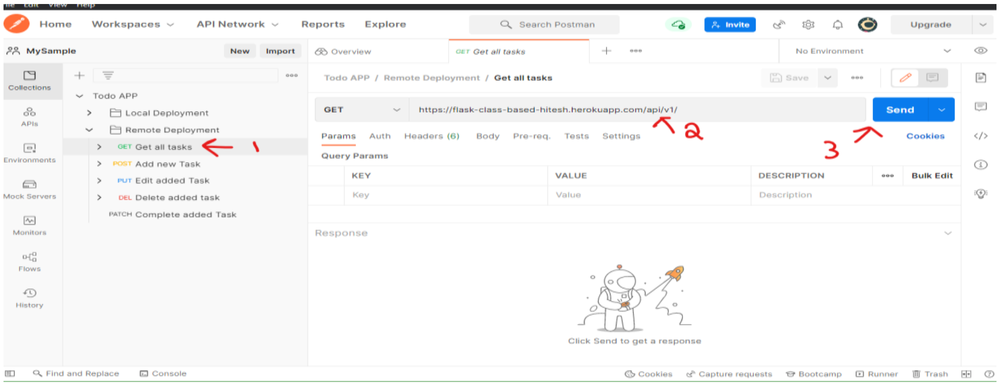
Step 7: You'll get a window to upload files, click on the upload button and upload the ***'Todo APP.postman_collection.json'*** which can be found from your present working directory.

Step 8: Once uploaded, you can probably see something like the figure below.

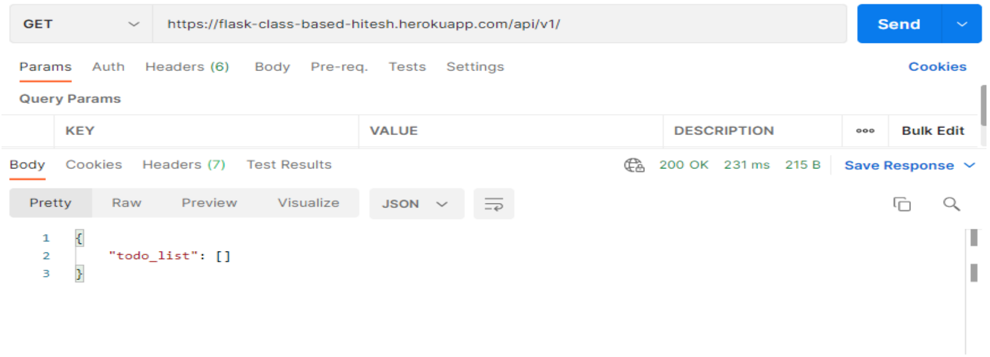


Step 9: From here you can open the Remote Deployment and click on the "Get All Tasks" collection to check out the get method for our App

1. Make sure to replace your application's URL in the request's URL section in POSTMAN.
2. Once done hit Send and see if the response if same as below.



GET https://flask-class-based-hitesh.herokuapp.com/api/v1/

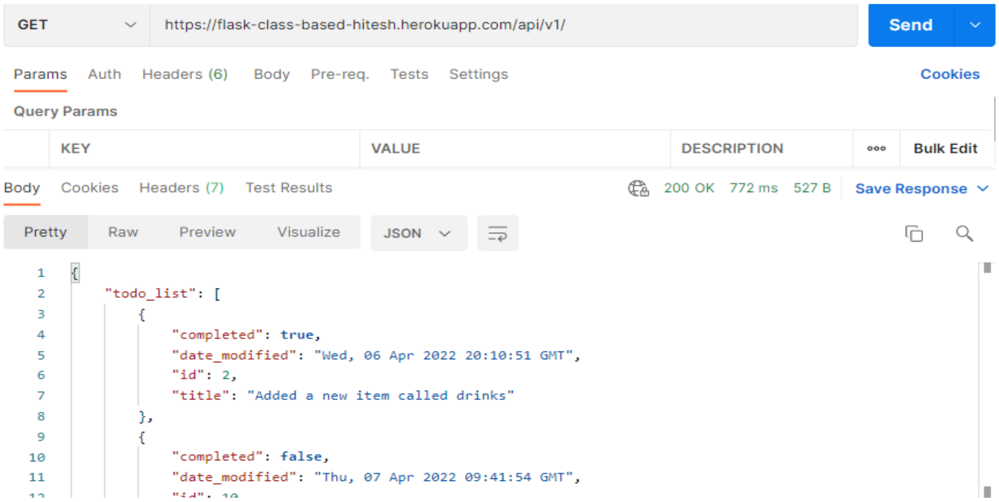


GET https://flask-class-based-hitesh.herokuapp.com/api/v1/

200 OK 231 ms 215 B

```
{
    "todo_list": []
}
```

Figure 2 Get response with no content body



GET https://flask-class-based-hitesh.herokuapp.com/api/v1/

200 OK 772 ms 527 B

```
{
    "todo_list": [
        {
            "completed": true,
            "date_modified": "Wed, 06 Apr 2022 20:10:51 GMT",
            "id": 2,
            "title": "Added a new item called drinks"
        },
        {
            "completed": false,
            "date_modified": "Thu, 07 Apr 2022 09:41:54 GMT",
            "id": 10
```

Figure 3 Get containing tasks already in the database

You can also explore the same examples under each request, which has already captured the sample output for the respective method.

From here you can go out and explore the various methods implemented on our App which are GET, POST, PUT and DELETE

EXERCISES:

E01 -> Implement PATCH Complete Item feature – to mark a "to do" item as complete.

E02 -> Implement a serializer for the DELETE method.

HINTS:

E01 - Please refer to the implementation of delete item for trying out patch item.

E02 - For complete item, refer the already implemented method POST.

FRONTEND DEVELOPMENT

PRE-REQUISITES

The following tools / modules must be installed before proceeding further:

1. NodeJS and NPM - https://nodejs.org/en/download/
2. Visual Studio Code i.e., VS Code - https://code.visualstudio.com/download
3. Heroku CLI - https://devcenter.heroku.com/articles/heroku-cli

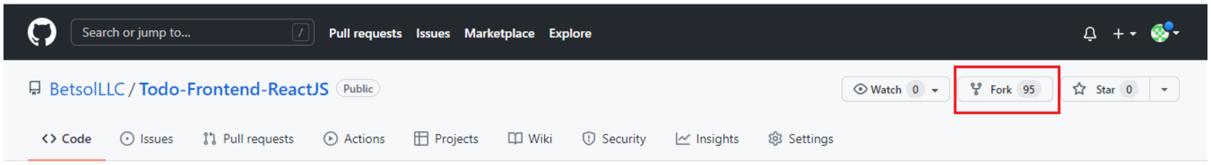
Note: Versions: **Node - 14.16.1 and npm - 6.14.12**

START WORKING ON THE TO DO APP

SETUP THE TEMPLATE CODE

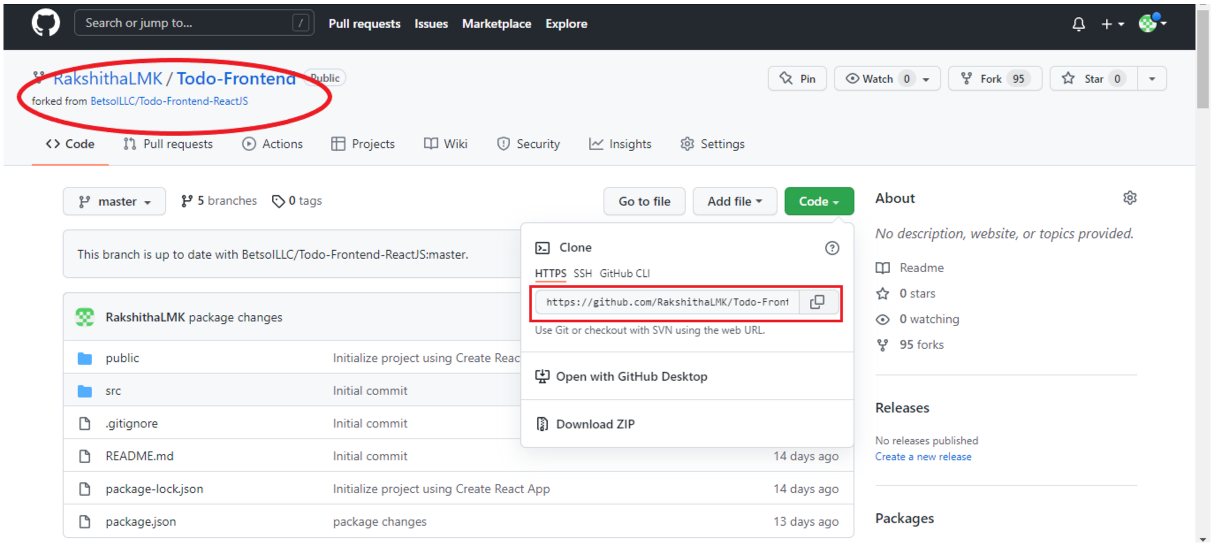
There is already a project template built and ready to use to start with the To Do App.

Step 1: Fork the repository from the given URL from GitHub - https://github.com/BetsolLLC/Todo-Frontend-ReactJS.git



Step 2: Then create a new folder in your system (preferably in D drive) and go into the folder.

Step 3: Then **clone the code from the forked repository under your name**. (NOT the original repository that you forked previously).



https://github.com/RakshithaLMK/Todo-Front

Use Git or checkout with SVN using the web URL.

This branch is up to date with BetsolLLC/Todo-Frontend-ReactJS:master.

No description, website, or topics provided.

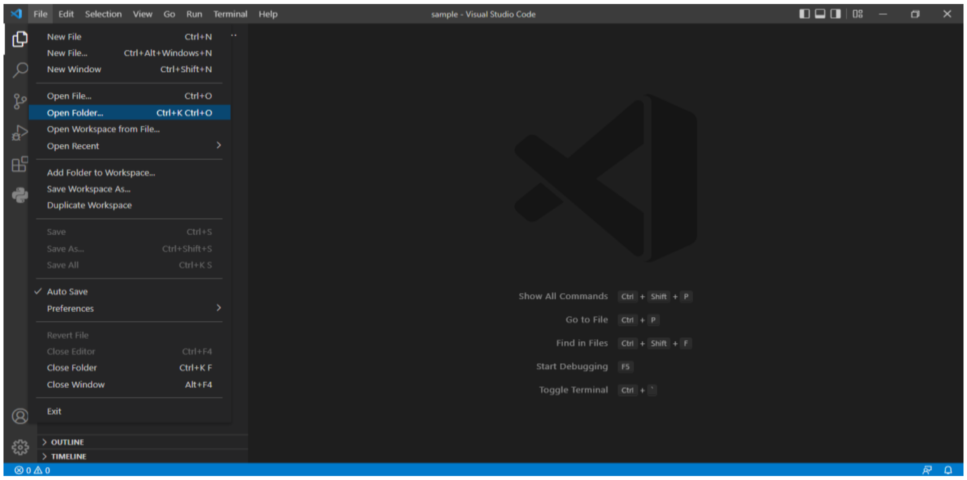
No releases published
Create a new release

RakshithaLMK package changes

public	Initialize project using Create Reac	
src	Initial commit	
.gitignore	Initial commit	
README.md	Initial commit	14 days ago
package-lock.json	Initialize project using Create React App	14 days ago
package.json	package changes	13 days ago

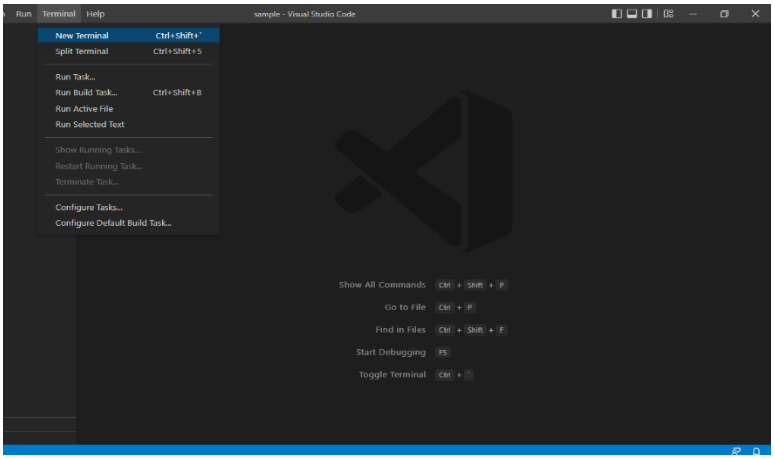
Step 4: Open VS Code.

Step 5: Go to File -> Open Folder option and select the “todo-frontend” folder.



Step 6: Then open a terminal here and **checkout**/switch to the branch **“app-lst”** using command

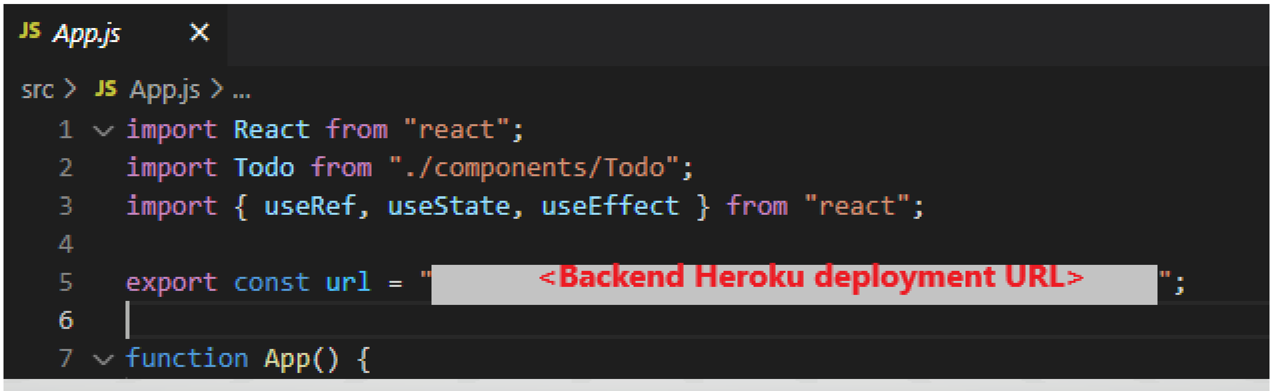
```
git checkout app-lst
```


Run
Terminal
Help
sample - Visual Studio Code
New Terminal Ctrl+Shift+`
Split Terminal Ctrl+Shift+5
Run Task...
Run Build Task... Ctrl+Shift+B
Run Active File
Run Selected Text
Show Running Tasks...
Restart Running Task...
Terminate Task...
Configure Tasks...
Configure Default Build Task...
Show All Commands Ctrl + Shift + P
Go to File Ctrl + P
Find in Files Ctrl + Shift + F
Start Debugging F5
Toggle Terminal Ctrl + `

Change the Endpoint URL in App.js

Step 1: Open the file src -> App.js.

Step 2: Change the value of the variable "url" to the backend URL that was deployed on Heroku.



App.js

src > App.js > ...

```
import React from "react";
import Todo from "./components/Todo";
import { useRef, useState, useEffect } from "react";

export const url = "<Backend Heroku deployment URL>";

function App() {
```

Step 3: Save the changes.

Step 4: Then execute the following commands to push the new changes to the repository.

```
git add .
git commit –m "changes made to the endpoint URL"
git push origin app-lst
```

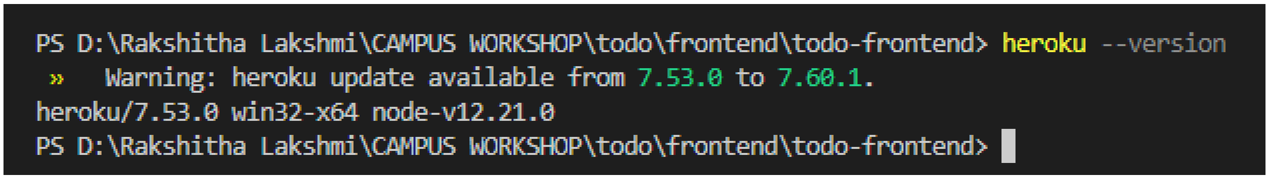
DEPLOY FRONTEND ON HEROKU

INSTALL HEROKU CLI (IF DONE ALREADY)

Once the above steps are done, we can proceed with deploying frontend on Heroku. For this, follow the below steps:

Assuming that Heroku CLI installation is done already, verify the installation using the command -

```
heroku version
```



```
PS D:\Rakshitha Lakshmi\CAMPUS WORKSHOP\todo\frontend\todo-frontend> heroku --version
 »   Warning: heroku update available from 7.53.0 to 7.60.1.
heroku/7.53.0 win32-x64 node-v12.21.0
PS D:\Rakshitha Lakshmi\CAMPUS WORKSHOP\todo\frontend\todo-frontend>
```

Note: If Heroku CLI is not installed, please follow the steps given during backend development for the same.

DEPLOY APP TO HEROKU

Step 1: Login to your Heroku account via CLI using the command –

```
heroku login -i
```

Here it prompts you to enter the following credentials –

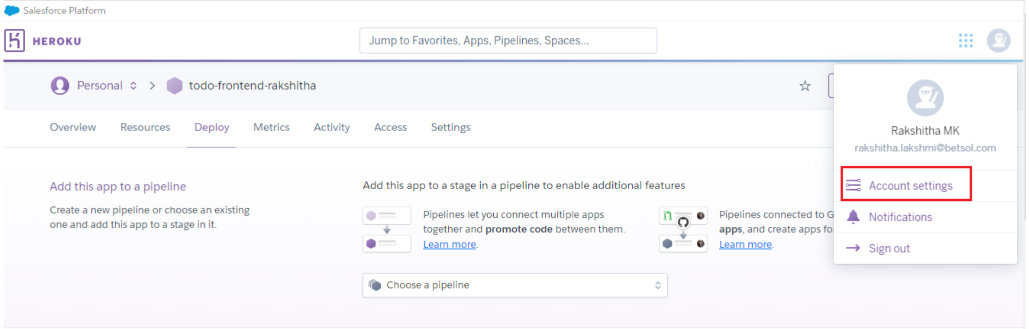
a. Email: <your Heroku login mail id”>

b. Password: <Your Heroku account authorization token for Heroku CLI (mentioned in detail below)>

   Note: Please use ‘right click’ to paste the password once u get it.

To get the authorization token,

Step 1: Go to Heroku Dashboard -> account settings



Add this app to a pipeline

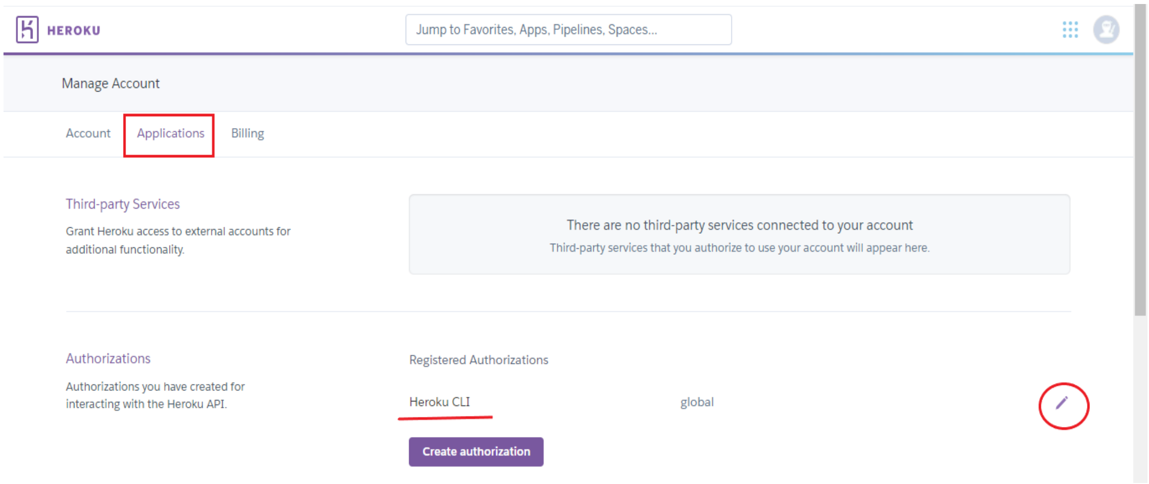
Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

Pipelines let you connect multiple apps together and **promote code** between them.

Pipelines connected to G **apps**, and create apps fo

Step 2: Then navigate to Applications tab and click on the ‘edit’ icon of the Heroku CLI Authorization.



Grant Heroku access to external accounts for additional functionality.

There are no third-party services connected to your account

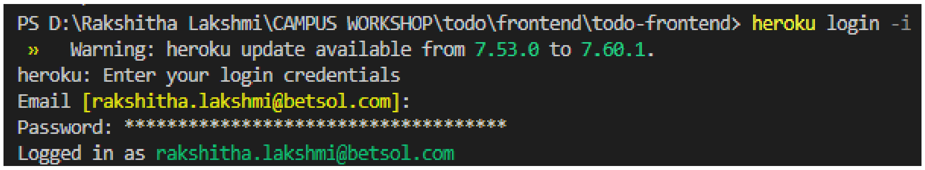
Third-party services that you authorize to use your account will appear here.

Authorizations you have created for interacting with the Heroku API.

Heroku CLI	global

Step 3: Then **copy the TOKEN** given there and that **would be the Password for login** via the command “`heroku login -i`” mentioned previously.

Step 4: Once done, it will show our login like below:

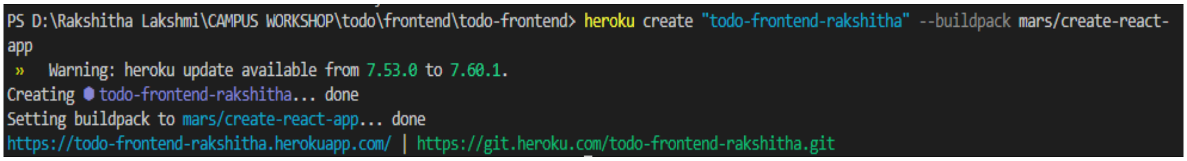


```
PS D:\Rakshitha Lakshmi\CAMPUS WORKSHOP\todo\frontend\todo-frontend> heroku login -i
 »   Warning: heroku update available from 7.53.0 to 7.60.1.
heroku: Enter your login credentials
Email [rakshitha.lakshmi@betsol.com]:
Password: ************************************
Logged in as rakshitha.lakshmi@betsol.com
```

Step 5: Then create a Heroku App using the command below: Here, we are using the build-pack suitable for ReactJS i.e., mars/create-react-app.

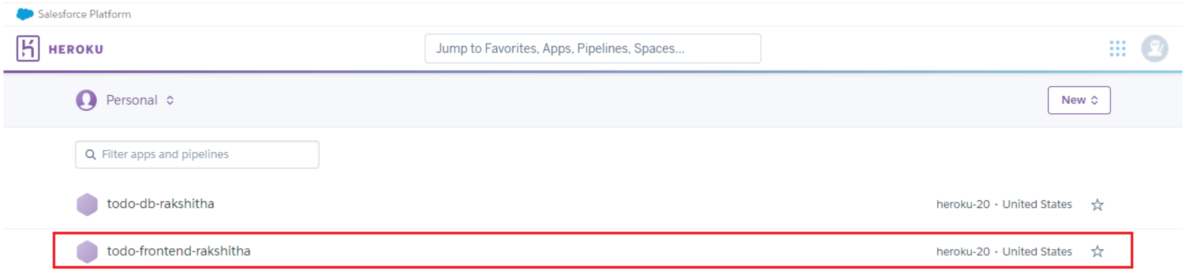
Note: The app name must start with a letter, end with a letter or digit and can only contain lowercase letters, digits, and dashes.

```
heroku create $APP_NAME --buildpack mars/create-react-app
```



```
PS D:\Rakshitha Lakshmi\CAMPUS WORKSHOP\todo\frontend\todo-frontend> heroku create "todo-frontend-rakshitha" --buildpack mars/create-react-
app
 »   Warning: heroku update available from 7.53.0 to 7.60.1.
Creating ⬢ todo-frontend-rakshitha... done
Setting buildpack to mars/create-react-app... done
https://todo-frontend-rakshitha.herokuapp.com/ | https://git.heroku.com/todo-frontend-rakshitha.git
```

Step 6: Once done, we can see the app under our personal apps section on Heroku. You can navigate to the URL https://dashboard.heroku.com/apps to check it.



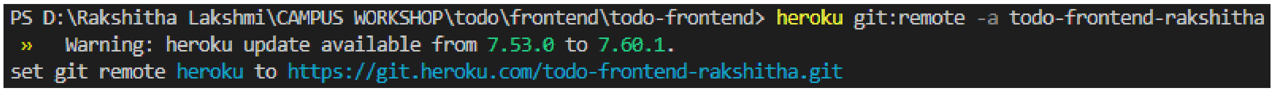
Salesforce Platform
HEROKU
Jump to Favorites, Apps, Pipelines, Spaces...
Personal
New
Filter apps and pipelines

todo-db-rakshitha heroku-20 · United States

todo-frontend-rakshitha heroku-20 · United States

Step 7: To deploy our app on Heroku now, attach it to the Heroku remote repository that was created for our Heroku app created recently. Use the command –

```
heroku git:remote -a $APP_NAME
```



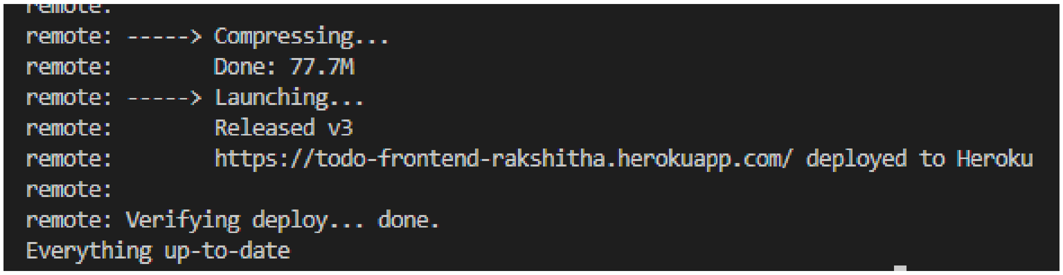
```
PS D:\Rakshitha Lakshmi\CAMPUS WORKSHOP\todo\frontend\todo-frontend> heroku git:remote -a todo-frontend-rakshitha
 »   Warning: heroku update available from 7.53.0 to 7.60.1.
set git remote heroku to https://git.heroku.com/todo-frontend-rakshitha.git
```

Step 8: Then do a git push to the heroku remote repository and the necessary branch i.e**, app-lst** using the command – Here, we deploy code to Heroku from a non-main branch i.e, app-lst.

```
git push heroku app-lst:main
```

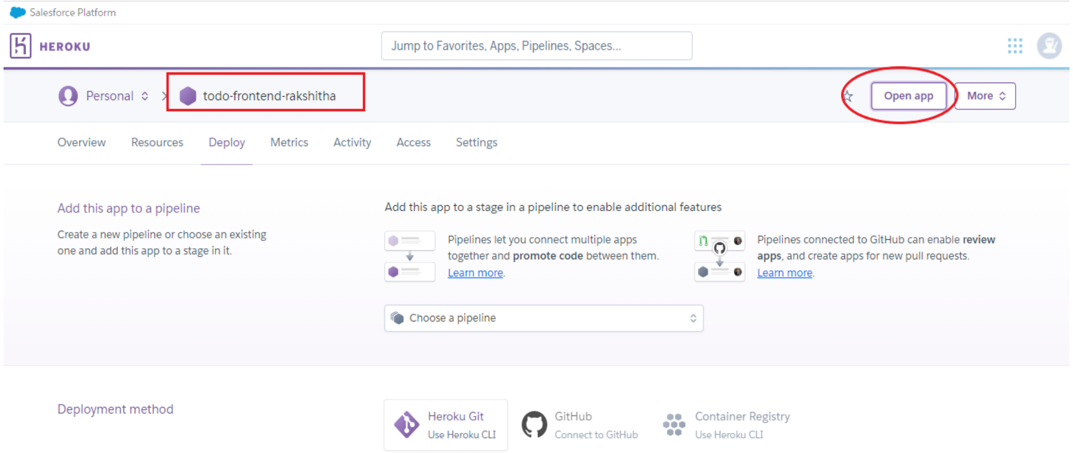
Note: Do **not to push the package.json.lock file** to the repository. This is because the build might fail due to inconsistencies in the lock files when it tries to download the dependencies.

Step 9: This deploys the App as well. Hence, we should see the output of the command as below:



```
remote:
remote: -----> Compressing...
remote:        Done: 77.7M
remote: -----> Launching...
remote:        Released v3
remote:        https://todo-frontend-rakshitha.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
Everything up-to-date
```

Step 10: Go to the heroku dashboard -> your frontend App and then click on “Open App” button.



Add this app to a pipeline

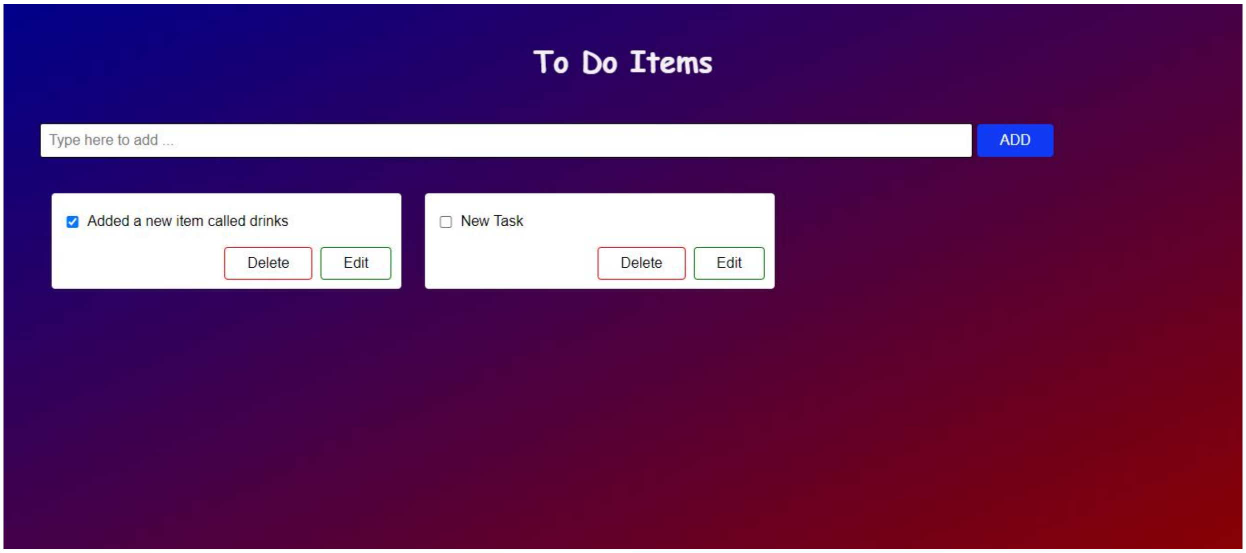
Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

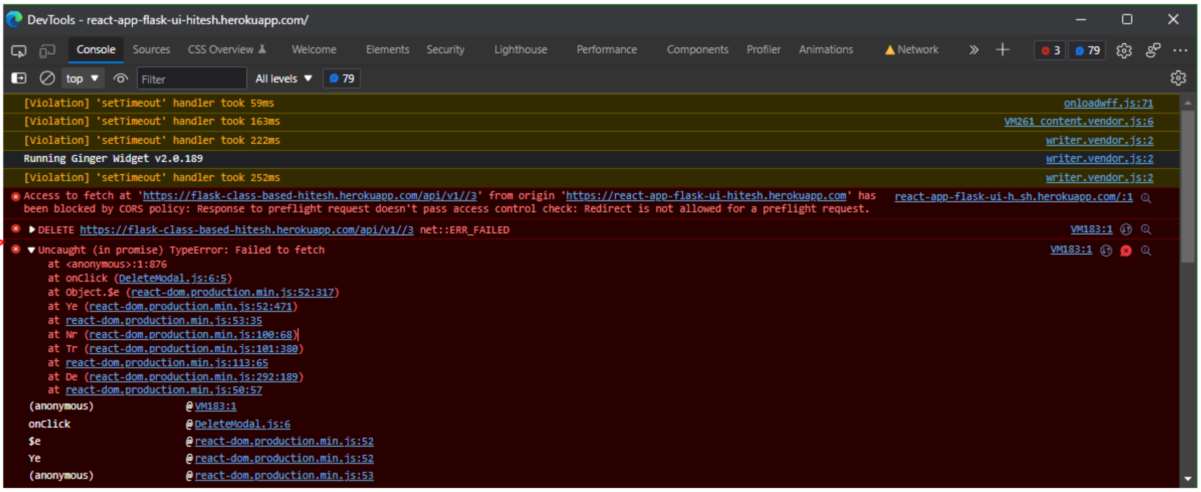
Pipelines let you connect multiple apps together and **promote code** between them. Learn more.

Pipelines connected to GitHub can enable **review apps**, and create apps for new pull requests. Learn more.

It will navigate to the respective URL of your Frontend App where you will be able to see the TO DO APP!



Note: 1. While you are on the react app tab , press F12 or right click->inspect and go to the console tab . Check if there is a CORS issue similar to the pic given below.



```
[Violation] 'setTimeout' handler took 59ms                                                onloadwff.js:71
[Violation] 'setTimeout' handler took 163ms                                    VM261 content.vendor.js:6
[Violation] 'setTimeout' handler took 222ms                                              writer.vendor.js:2
Running Ginger Widget v2.0.189                                                            writer.vendor.js:2
[Violation] 'setTimeout' handler took 252ms                                              writer.vendor.js:2
Access to fetch at 'https://flask-class-based-hitesh.herokuapp.com/api/v1//3' from origin 'https://react-app-flask-ui-hitesh.herokuapp.com' has   react-app-flask-ui-h…sh.herokuapp.com/:1
been blocked by CORS policy: Response to preflight request doesn't pass access control check: Redirect is not allowed for a preflight request.
▶ DELETE https://flask-class-based-hitesh.herokuapp.com/api/v1//3 net::ERR_FAILED        VM183:1
▼ Uncaught (in promise) TypeError: Failed to fetch                                        VM183:1
    at <anonymous>:1:876
    at onClick (DeleteModal.js:6:5)
    at Object.$e (react-dom.production.min.js:52:317)
    at Ye (react-dom.production.min.js:52:471)
    at react-dom.production.min.js:53:35
    at Nr (react-dom.production.min.js:100:68)
    at Tr (react-dom.production.min.js:101:380)
    at react-dom.production.min.js:113:65
    at De (react-dom.production.min.js:292:189)
    at react-dom.production.min.js:50:57
  (anonymous)             @ VM183:1
  onClick                 @ DeleteModal.js:6
  $e                      @ react-dom.production.min.js:52
  Ye                      @ react-dom.production.min.js:52
  (anonymous)             @ react-dom.production.min.js:53
```

In case there are CORS issues for the API requests, please start the chrome session without web security temporarily by executing this command in Command Prompt terminal:

```
"C:\Program Files\Google\Chrome\Application\chrome.exe" --disable-web-security --user-data-dir="C:\tmpChromeSession"
```

Then, navigate to URL where your frontend App is hosted.

Explore the existing functionalities such as Adding, deleting and listing the "To Do" items.

EXERCISES:

E01 -> Implement Complete Item feature – to mark a "to do" item as complete (checkbox).

E02 -> Implement Edit Item feature – to edit / update the title of the "to do" item.

Hints:

E01 - For complete item, the functionality is implemented from frontend ReactJS code **(refer file src -> components -> Todo.js)**, just that a **backend API should be brought up** to handle it properly.

E02 - Please refer to the implementation of delete item for trying out edit item.