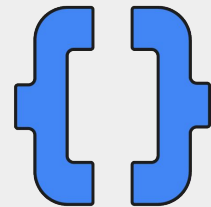




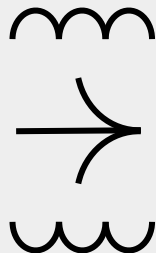
Google Developer Group
Sheridan College



Welcome Everyone

We're excited to have you here!

Please be seated. We will begin at 5:30 PM

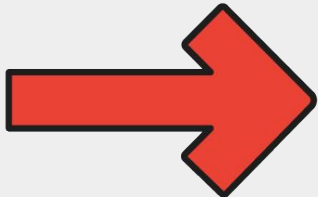




Google Developer Group
Sheridan College

How to Build a Full Stack App: GitHub + Flask + Next.js + BigQuery

Workshop A - George Farag



Prerequisites



Python

`python3 --version`

<https://www.python.org/downloads/>



Node.js

`node --version`

`npm --version`

<https://nodejs.org/en/download>



git

`git --version`

<https://git-scm.com/install/>

Tech Stack



GitHub

- Version Control



Flask

- Backend API



Next.js

- Frontend



Google Cloud BigQuery

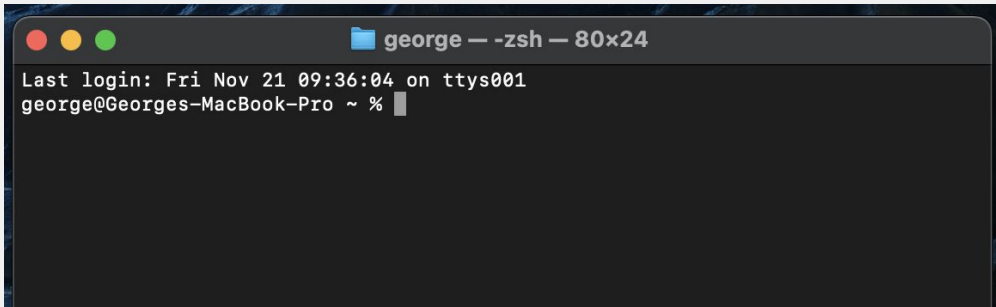
- Data Access

Step 1: Open Terminal

Open Terminal / Command Prompt on your laptop 🔍



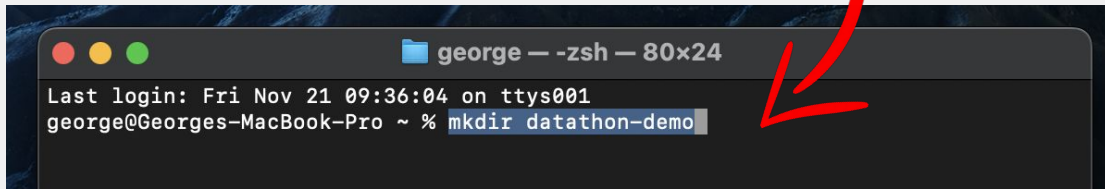
You should have this screen opened



Step 2: Create Directory

Type the following into your Terminal to create a new folder

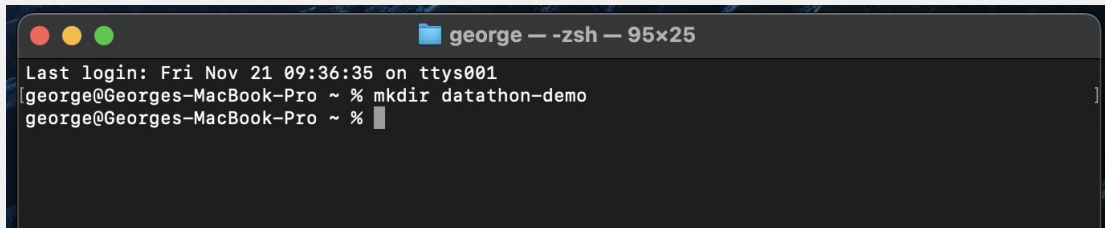
- `mkdir datathon-demo`



A terminal window titled "george — -zsh — 80x24". The window shows the output "Last login: Fri Nov 21 09:36:04 on ttys001" and the prompt "george@Georges-MacBook-Pro ~ %". The command "mkdir datathon-demo" is entered and highlighted with a blue selection box. A red arrow points from the command text above to the terminal window.

```
george@Georges-MacBook-Pro ~ % mkdir datathon-demo
```

Press enter



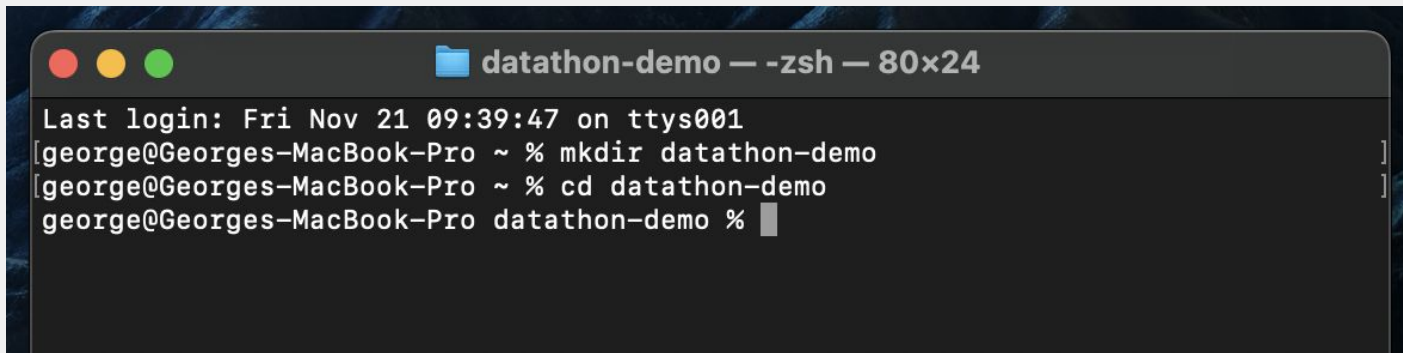
A terminal window titled "george — -zsh — 95x25". The window shows the output "Last login: Fri Nov 21 09:36:35 on ttys001" and the prompt "george@Georges-MacBook-Pro ~ %". The command "mkdir datathon-demo" is entered and followed by a new line, indicating it has been executed.

```
george@Georges-MacBook-Pro ~ % mkdir datathon-demo
george@Georges-MacBook-Pro ~ %
```

Step 3: Navigate into Folder

Now use 'change directory' (cd) to navigate into your new folder

- `cd datathon-demo`

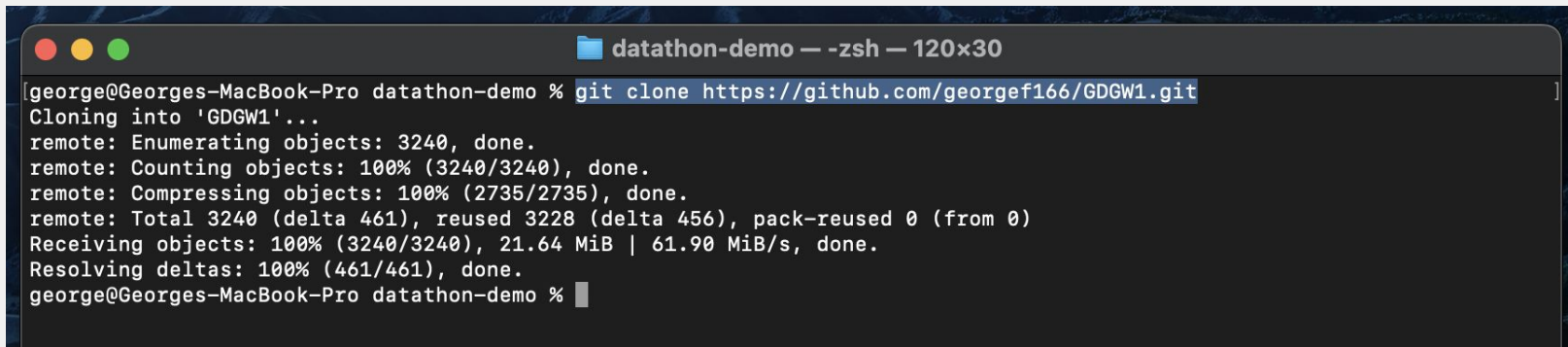
A screenshot of a macOS terminal window. The title bar shows a folder icon, the text 'datathon-demo', and '-zsh — 80x24'. The terminal content shows the following sequence: 'Last login: Fri Nov 21 09:39:47 on ttys001', 'george@Georges-MacBook-Pro ~ % mkdir datathon-demo', 'george@Georges-MacBook-Pro ~ % cd datathon-demo', and 'george@Georges-MacBook-Pro datathon-demo %' with a cursor. There are small vertical bracket-like artifacts on the right side of the terminal window.

```
Last login: Fri Nov 21 09:39:47 on ttys001
george@Georges-MacBook-Pro ~ % mkdir datathon-demo
george@Georges-MacBook-Pro ~ % cd datathon-demo
george@Georges-MacBook-Pro datathon-demo %
```

Step 4: Clone Repository

Use the following command to create a local copy of the repository

- `git clone https://github.com/georgef166/GDGW1.git`

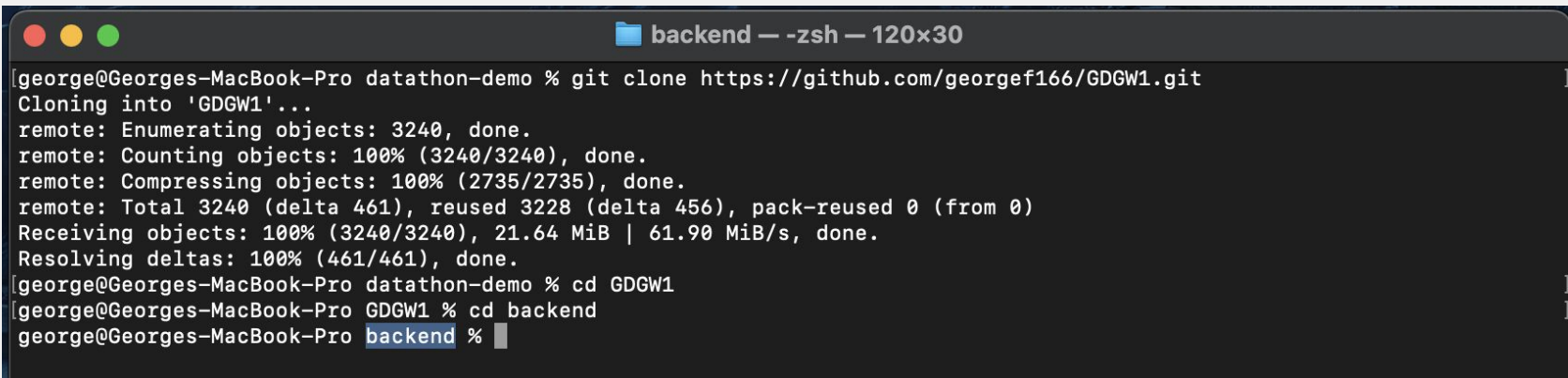
A terminal window titled "datathon-demo — -zsh — 120x30" is shown. The prompt is "george@Georges-MacBook-Pro datathon-demo %". The command "git clone https://github.com/georgef166/GDGW1.git" has been entered and executed. The output shows the cloning process: "Cloning into 'GDGW1'...", "remote: Enumerating objects: 3240, done.", "remote: Counting objects: 100% (3240/3240), done.", "remote: Compressing objects: 100% (2735/2735), done.", "remote: Total 3240 (delta 461), reused 3228 (delta 456), pack-reused 0 (from 0)", "Receiving objects: 100% (3240/3240), 21.64 MiB | 61.90 MiB/s, done.", "Resolving deltas: 100% (461/461), done.", and the prompt returns to "george@Georges-MacBook-Pro datathon-demo %".

```
george@Georges-MacBook-Pro datathon-demo % git clone https://github.com/georgef166/GDGW1.git
Cloning into 'GDGW1'...
remote: Enumerating objects: 3240, done.
remote: Counting objects: 100% (3240/3240), done.
remote: Compressing objects: 100% (2735/2735), done.
remote: Total 3240 (delta 461), reused 3228 (delta 456), pack-reused 0 (from 0)
Receiving objects: 100% (3240/3240), 21.64 MiB | 61.90 MiB/s, done.
Resolving deltas: 100% (461/461), done.
george@Georges-MacBook-Pro datathon-demo %
```


Step 5: Navigate to Backend

Use 'change directory' (cd) twice to navigate into the backend folder

- `cd GDGW1`
- `cd backend`

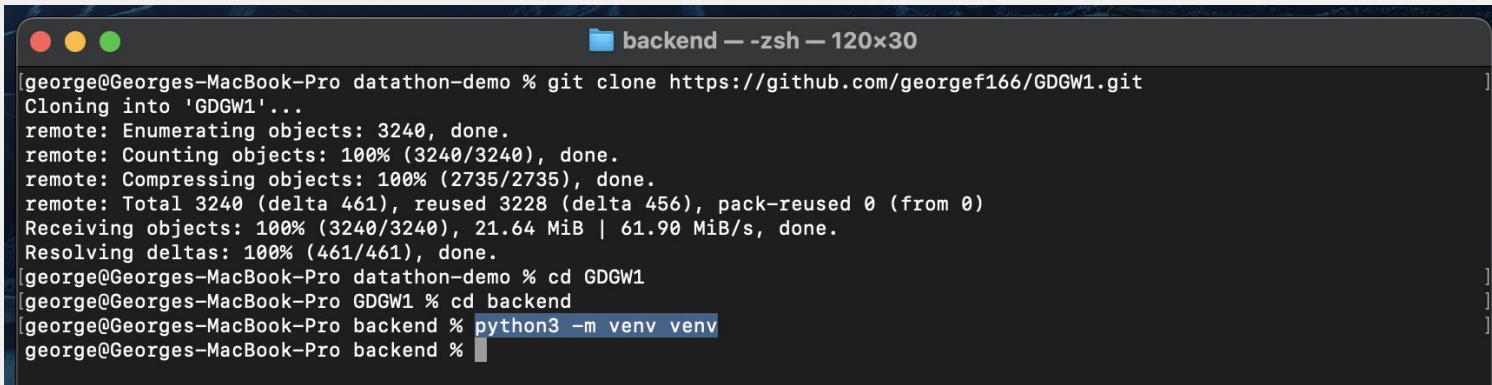


```
backend — -zsh — 120x30
[george@Georges-MacBook-Pro datathon-demo % git clone https://github.com/georgef166/GDGW1.git
Cloning into 'GDGW1'...
remote: Enumerating objects: 3240, done.
remote: Counting objects: 100% (3240/3240), done.
remote: Compressing objects: 100% (2735/2735), done.
remote: Total 3240 (delta 461), reused 3228 (delta 456), pack-reused 0 (from 0)
Receiving objects: 100% (3240/3240), 21.64 MiB | 61.90 MiB/s, done.
Resolving deltas: 100% (461/461), done.
[george@Georges-MacBook-Pro datathon-demo % cd GDGW1
[george@Georges-MacBook-Pro GDGW1 % cd backend
george@Georges-MacBook-Pro backend %
```

Step 6: Initialize VENV

Initialize your virtual environment

- `python3 -m venv venv`

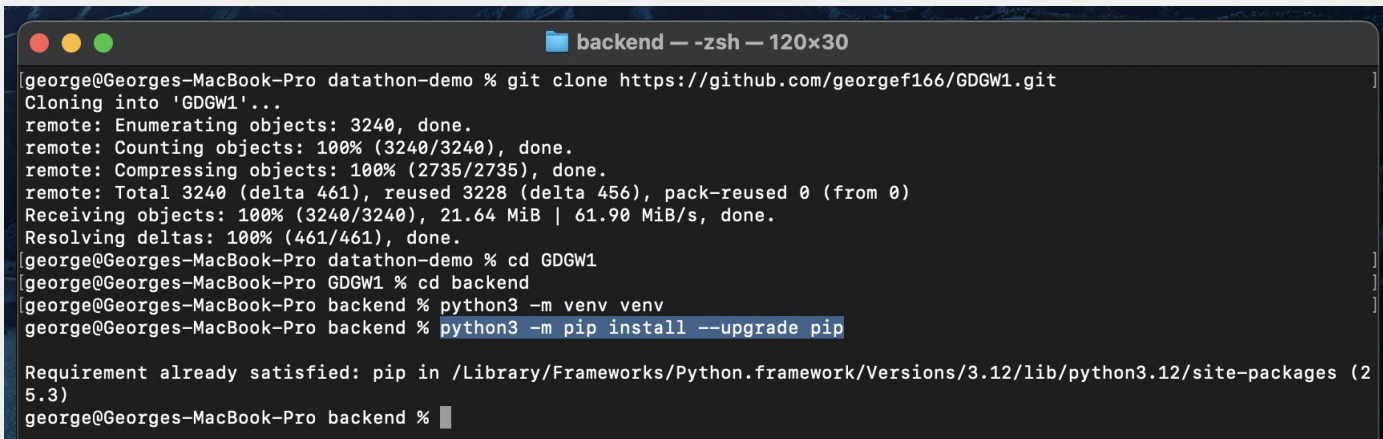


```
backend — zsh — 120x30
[george@Georges-MacBook-Pro datathon-demo % git clone https://github.com/georgef166/GDGW1.git]
Cloning into 'GDGW1'...
remote: Enumerating objects: 3240, done.
remote: Counting objects: 100% (3240/3240), done.
remote: Compressing objects: 100% (2735/2735), done.
remote: Total 3240 (delta 461), reused 3228 (delta 456), pack-reused 0 (from 0)
Receiving objects: 100% (3240/3240), 21.64 MiB | 61.90 MiB/s, done.
Resolving deltas: 100% (461/461), done.
[george@Georges-MacBook-Pro datathon-demo % cd GDGW1]
[george@Georges-MacBook-Pro GDGW1 % cd backend]
[george@Georges-MacBook-Pro backend % python3 -m venv venv]
[george@Georges-MacBook-Pro backend % ]
```

Step 7: Install/Update pip

Ensure you have the standard package manager for Python

- `python3 -m pip install --upgrade pip`

A terminal window titled 'backend — zsh — 120x30' on a macOS system. The user 'george' is in the directory 'datathon-demo'. The terminal shows the cloning of a repository, navigation to the 'backend' directory, and the execution of 'python3 -m pip install --upgrade pip'. The output indicates that pip is already satisfied at version 5.3.

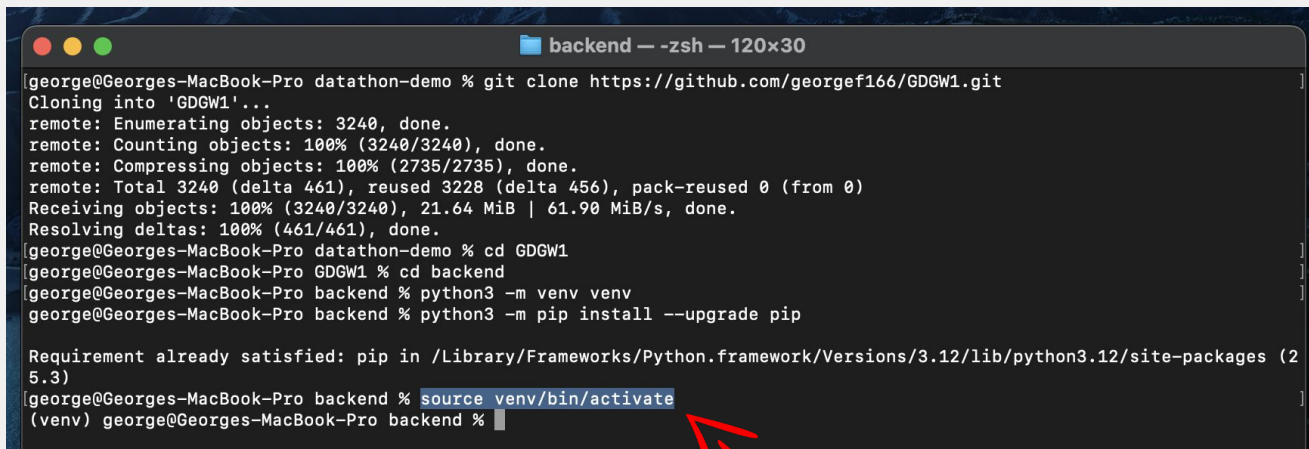
```
george@Georges-MacBook-Pro datathon-demo % git clone https://github.com/georgef166/GDGW1.git
Cloning into 'GDGW1'...
remote: Enumerating objects: 3240, done.
remote: Counting objects: 100% (3240/3240), done.
remote: Compressing objects: 100% (2735/2735), done.
remote: Total 3240 (delta 461), reused 3228 (delta 456), pack-reused 0 (from 0)
Receiving objects: 100% (3240/3240), 21.64 MiB | 61.90 MiB/s, done.
Resolving deltas: 100% (461/461), done.
george@Georges-MacBook-Pro datathon-demo % cd GDGW1
george@Georges-MacBook-Pro GDGW1 % cd backend
george@Georges-MacBook-Pro backend % python3 -m venv venv
george@Georges-MacBook-Pro backend % python3 -m pip install --upgrade pip

Requirement already satisfied: pip in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (2
5.3)
george@Georges-MacBook-Pro backend %
```

Step 7: Activate Virtual Env

Type the command based on your platform to activate the venv

- `source venv/bin/activate` (Mac/Linux)
- `.\venv\Scripts\activate.bat` (Windows)



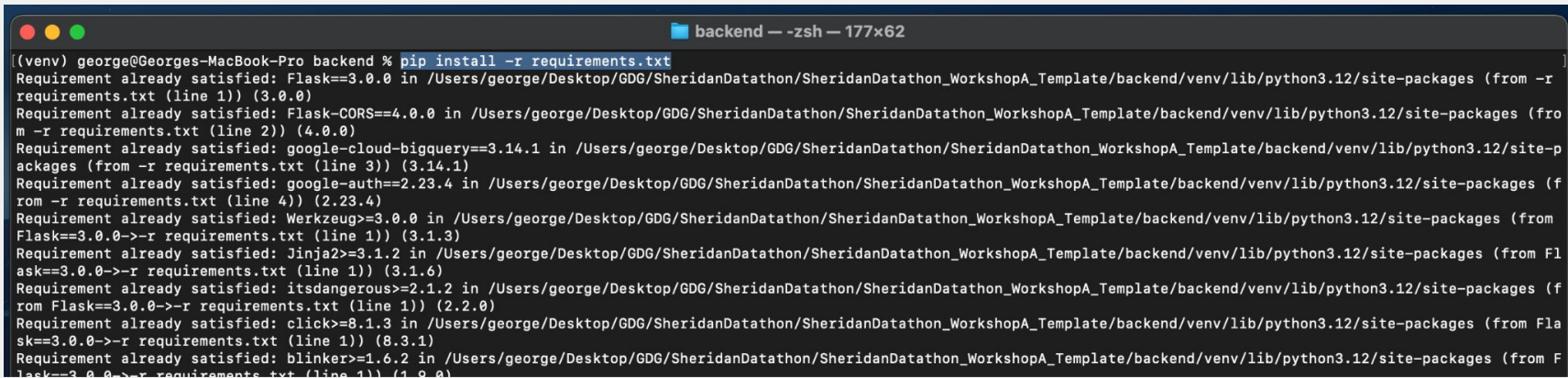
```
backend — zsh — 120x30
george@Georges-MacBook-Pro datathon-demo % git clone https://github.com/georgef166/GDGW1.git
Cloning into 'GDGW1'...
remote: Enumerating objects: 3240, done.
remote: Counting objects: 100% (3240/3240), done.
remote: Compressing objects: 100% (2735/2735), done.
remote: Total 3240 (delta 461), reused 3228 (delta 456), pack-reused 0 (from 0)
Receiving objects: 100% (3240/3240), 21.64 MiB | 61.90 MiB/s, done.
Resolving deltas: 100% (461/461), done.
george@Georges-MacBook-Pro datathon-demo % cd GDGW1
george@Georges-MacBook-Pro GDGW1 % cd backend
george@Georges-MacBook-Pro backend % python3 -m venv venv
george@Georges-MacBook-Pro backend % python3 -m pip install --upgrade pip

Requirement already satisfied: pip in /Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages (2
5.3)
george@Georges-MacBook-Pro backend % source venv/bin/activate
(venv) george@Georges-MacBook-Pro backend %
```

Step 8: Install Dependencies

Type the command based on your platform to activate the venv

- `pip install -r requirements.txt`

A screenshot of a macOS terminal window titled 'backend — zsh — 177x62'. The prompt is '(venv) george@Georges-MacBook-Pro backend %'. The command entered is 'pip install -r requirements.txt'. The output shows several requirements already satisfied, including Flask, Flask-CORS, google-cloud-bigquery, google-auth, Werkzeug, Jinja2, itsdangerous, click, and blinker, all with their respective versions and installation paths.

```
[(venv) george@Georges-MacBook-Pro backend % pip install -r requirements.txt
Requirement already satisfied: Flask==3.0.0 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 1)) (3.0.0)
Requirement already satisfied: Flask-CORS==4.0.0 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 2)) (4.0.0)
Requirement already satisfied: google-cloud-bigquery==3.14.1 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 3)) (3.14.1)
Requirement already satisfied: google-auth==2.23.4 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 4)) (2.23.4)
Requirement already satisfied: Werkzeug==3.0.0 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 5)) (3.1.3)
Requirement already satisfied: Jinja2==3.1.2 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 6)) (3.1.2)
Requirement already satisfied: itsdangerous==2.1.2 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 7)) (2.2.0)
Requirement already satisfied: click==8.1.3 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 8)) (8.3.1)
Requirement already satisfied: blinker==1.6.2 in /Users/george/Desktop/GDG/SheridanDatathon/SheridanDatathon_WorkshopA_Template/backend/venv/lib/python3.12/site-packages (from -r requirements.txt (line 9)) (1.8.0)
```

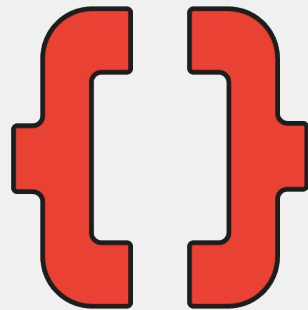
Open VS Code - app.py

```
from flask import Flask, jsonify
from flask_cors import CORS
from google.cloud import bigquery

app = Flask(__name__)
CORS(app)

@app.route("/api/bq")
def bigquery_test():
    client = bigquery.Client()
    query = """
    SELECT name, gender
    FROM `bigquery-public-data.usa_names.usa_1910_current`
    LIMIT 5"""
    rows = client.query(query).result()
    return {"data": [dict(r) for r in rows]}

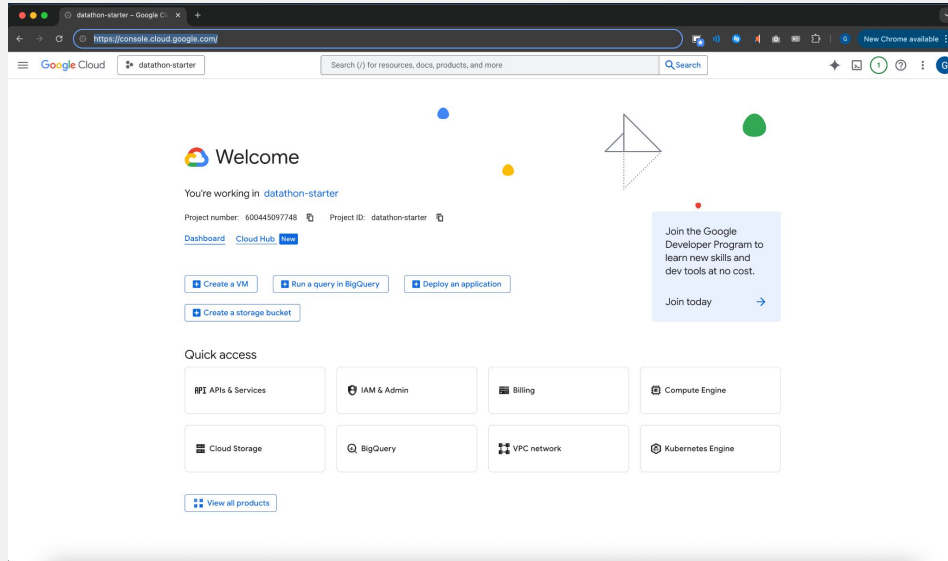
if __name__ == "__main__":
    app.run(port=5000, debug=True)
```



Step 9: Google Cloud Console

Navigate to the following link in your web browser (preferably Chrome)

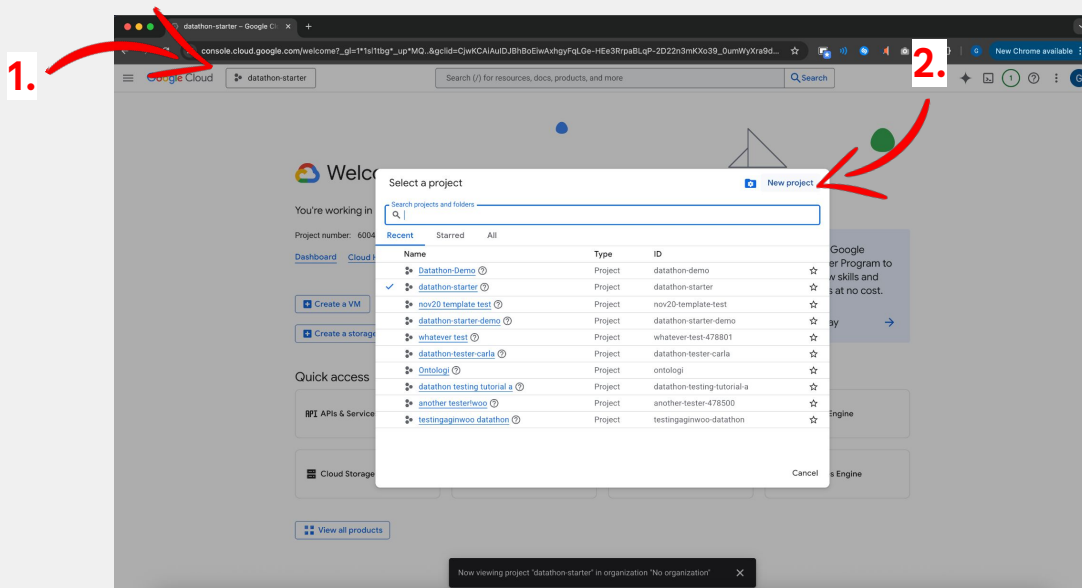
- console.cloud.google.com



Step 10: New Project

Press the New Project button in the top navbar

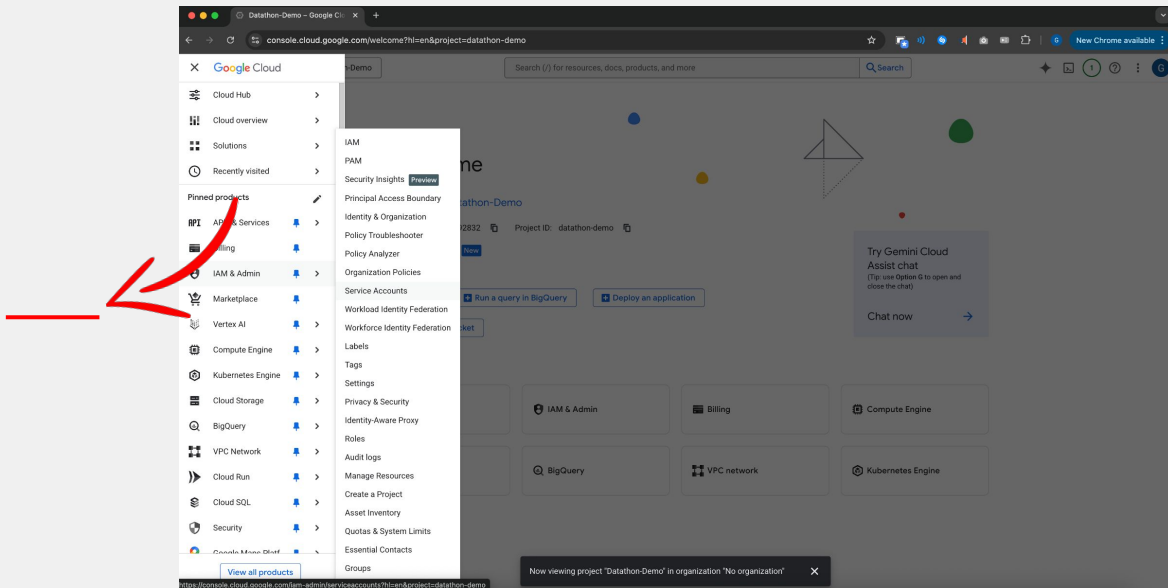
- Ensure the new project is selected upon creation, as by default it will not



Step 11: Service Account

Navigate to Service Accounts

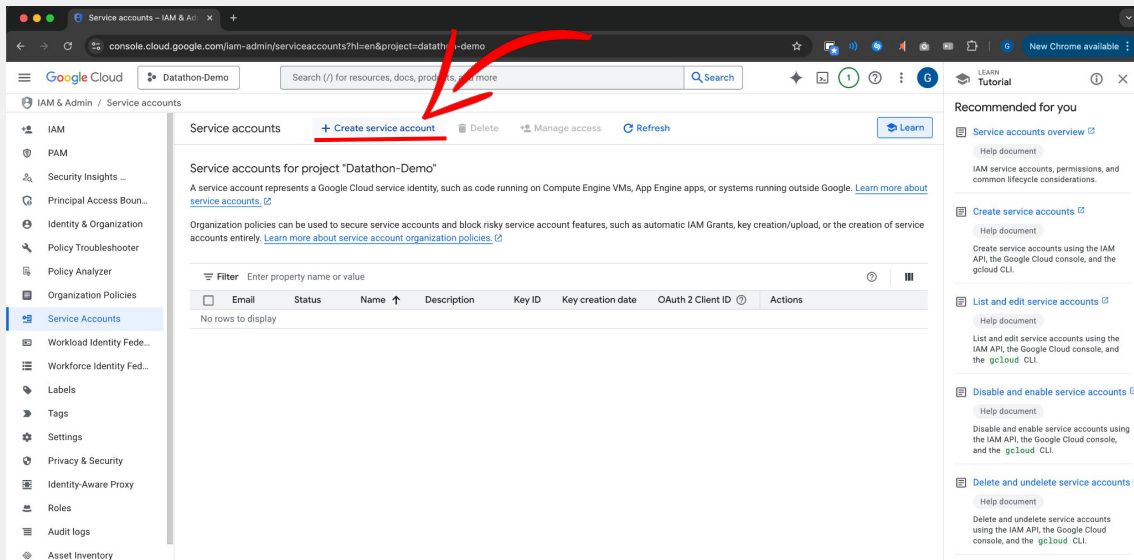
- IAM & Admin -> Service Accounts



Step 12: Create Account

Create a Service Account

- Service Accounts -> Create Service Account



Step 13: Account Permissions

1. **Create service account** -> Enter any name -> **Create and continue**
2. **Permissions** -> BigQuery User -> **Continue**
3. **Principals with access** -> *leave empty* -> **Done**

1 Create service account

Service account name

Display name for this service account

Service account ID *

Email address: datathondemo@datathon-demo.iam.gserviceaccount.com

Service account description

Describe what this service account will do

[Create and continue](#)

← Create service account

✓ Create service account

2 Permissions (optional)

Grant this service account access to Datathon-Demo so that it has permission to complete specific actions on the resources in your project. [Learn more](#)

Role

IAM condition (optional) [?](#)

[+ Add IAM condition](#)

When applied to a project, access to run queries, create datasets, read dataset metadata, and list tables, models and property graphs. When applied to a dataset, access to read dataset metadata and list tables, models, routines and property graphs within the dataset.

[+ Add another role](#)

[Help me choose roles](#)

[Continue](#)

← Create service account

✓ Create service account

✓ Permissions (optional)

3 Principals with access (optional)

Grant access to users or groups that need to perform actions as this service account. [Learn more](#)

Service account users role [?](#)

Grant users the permissions to deploy jobs and VMs with this service account

Service account admins role [?](#)

Grant users the permission to administer this service account

[Done](#) [Cancel](#)

Step 14: Generate a Key

Select Service Account

- Click on newly created Service Account

Service accounts + Create service account 🗑 Delete 👤 Manage access 🔄 Refresh 🎓 Learn

Service accounts for project "Datathon-Demo"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts.](#)

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies.](#)

Filter Enter property name or value

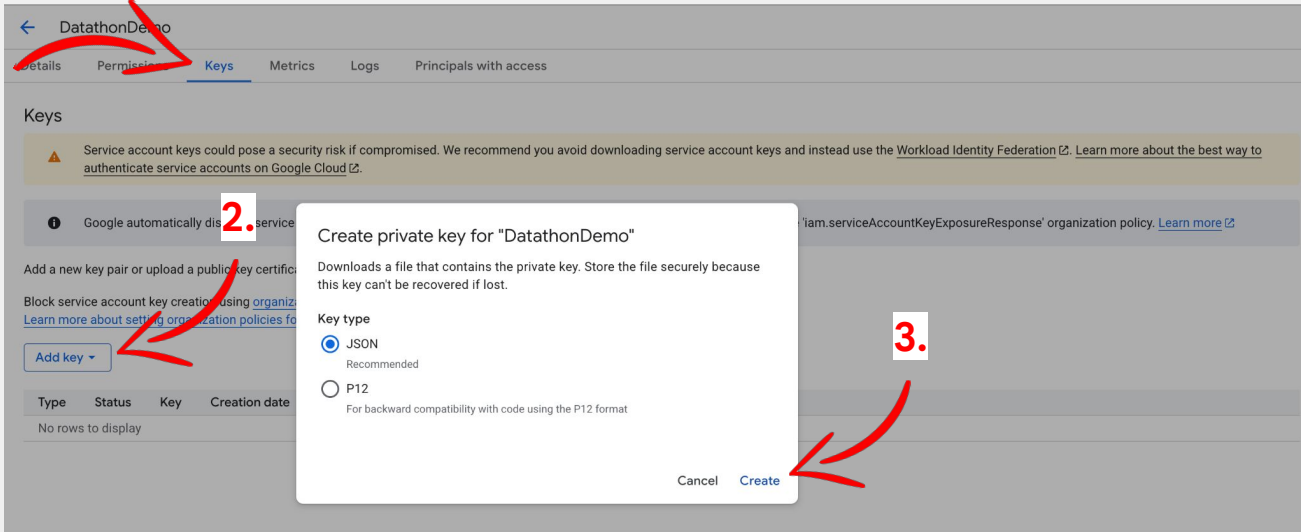
<input type="checkbox"/>	Email	Status	Name ↑	Description	Key ID	Key creation date	OAuth 2 Client ID ?	Actions
<input type="checkbox"/>	datathondemo@datathon-demo.iam.gserviceaccount.com	✔ Enabled	DatathonDemo		No keys		101943803108317800186	⋮

Step 15: Generate a Key

Add a JSON Key

- Keys -> Add Key -> Create New -> JSON -> Create

1.



2.

3.

Create private key for "DatathonDemo"

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

☒ JSON
Recommended

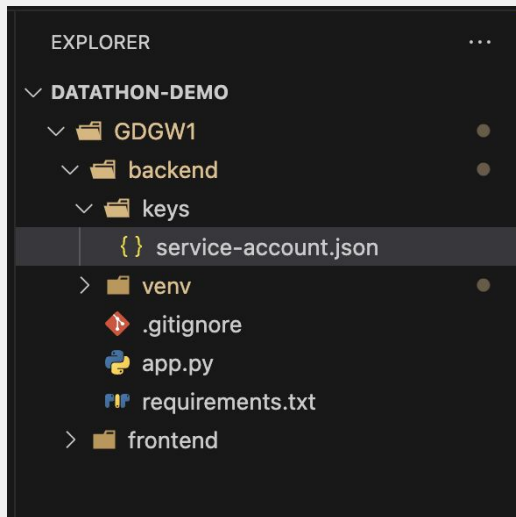
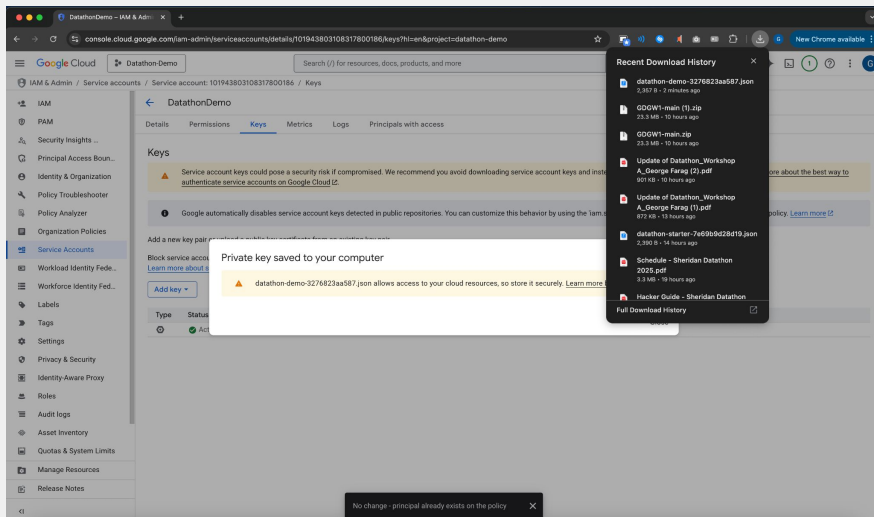
☐ P12
For backward compatibility with code using the P12 format

Cancel Create

Step 16: service-account.json

Add a JSON Key

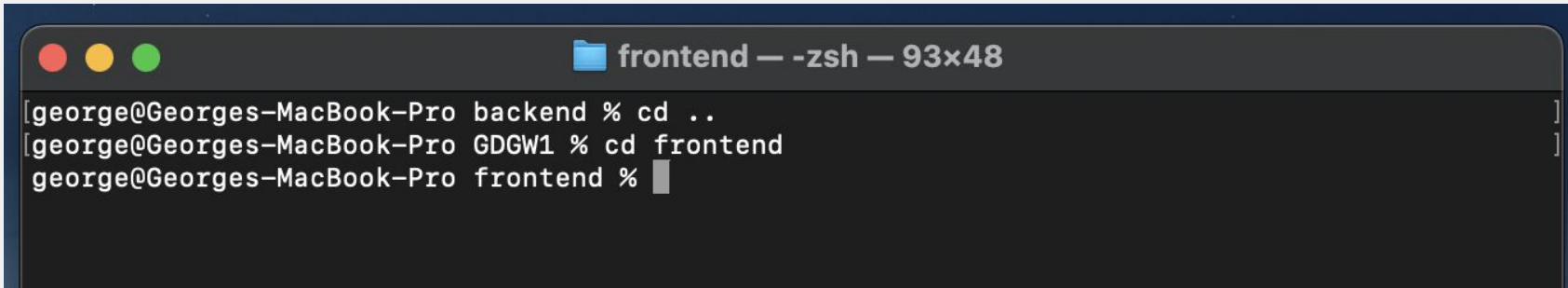
- Ensure JSON key was downloaded, and open the .json
- Copy and paste downloaded content in /backend/keys/service-account.json



Step 17: Navigate to Frontend

Use 'change directory' (cd) to navigate into the frontend folder

- `cd ..`
- `cd frontend`

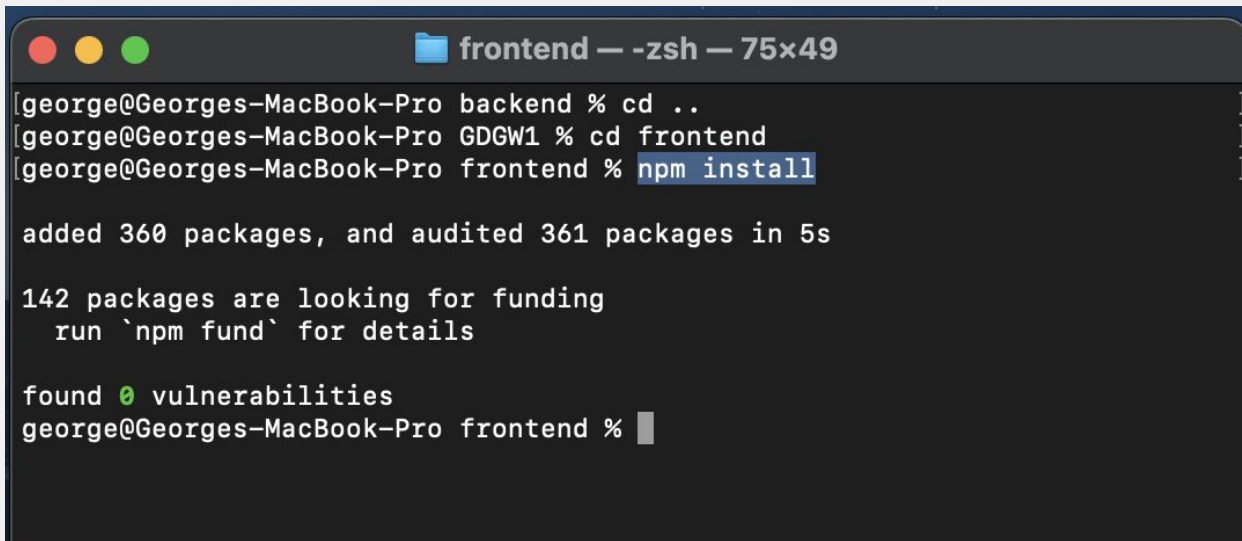
A terminal window titled 'frontend — -zsh — 93x48' with a folder icon. The window shows a sequence of commands and their outputs. The first command is 'cd ..' which changes the directory from 'backend' to 'GDGW1'. The second command is 'cd frontend' which changes the directory to 'frontend'. The prompt then shows 'frontend %' with a cursor.

```
[george@Georges-MacBook-Pro backend % cd ..  
[george@Georges-MacBook-Pro GDGW1 % cd frontend  
george@Georges-MacBook-Pro frontend %
```

Step 18: Node Packages

Install Required Node Packages with Node Package Manager

- `npm install`

A terminal window titled 'frontend — -zsh — 75x49' with three colored window control buttons (red, yellow, green) in the top-left corner. The terminal shows a sequence of commands and their output. The first command is 'cd ..' which changes the directory to 'backend'. The second command is 'cd frontend' which changes the directory to 'frontend'. The third command is 'npm install', which is highlighted with a blue selection box. The output of 'npm install' shows that 360 packages were added and 361 packages were audited in 5 seconds. It also mentions that 142 packages are looking for funding and provides a command to run 'npm fund' for details. Finally, it states that 0 vulnerabilities were found. The prompt 'george@Georges-MacBook-Pro frontend %' is visible at the bottom of the terminal window.

```
frontend — -zsh — 75x49
[george@Georges-MacBook-Pro backend % cd ..
[george@Georges-MacBook-Pro GDGW1 % cd frontend
[george@Georges-MacBook-Pro frontend % npm install

added 360 packages, and audited 361 packages in 5s

142 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
george@Georges-MacBook-Pro frontend %
```

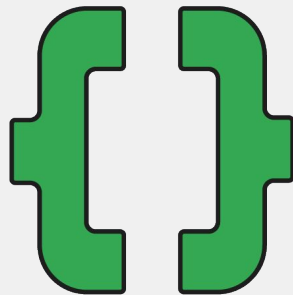

Open VS Code - page.tsx

```
from flask import Flask, jsonify
from flask_cors import CORS
from google.cloud import bigquery

app = Flask(__name__)
CORS(app)

@app.route("/api/bq")
def bigquery_test():
    client = bigquery.Client()
    query = """
    SELECT name, gender
    FROM `bigquery-public-data.usa_names.usa_1910_current`
    LIMIT 5"""
    rows = client.query(query).result()
    return {"data": [dict(r) for r in rows]}

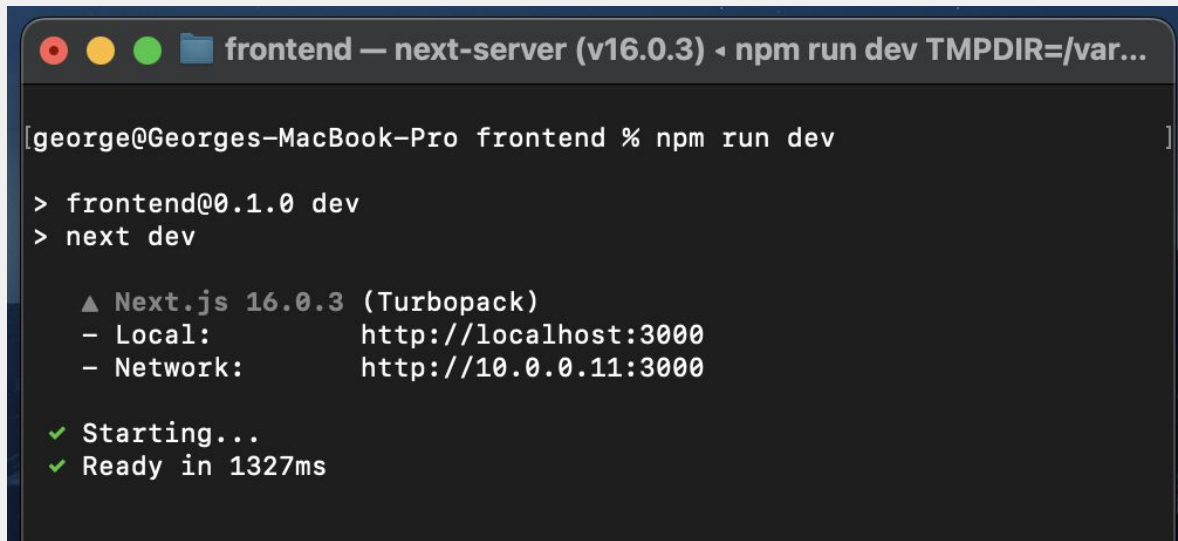
if __name__ == "__main__":
    app.run(port=5000, debug=True)
```



Step 19: Run the Frontend

Run the Frontend

- `npm run dev`



```
frontend — next-server (v16.0.3) ◀ npm run dev TMPDIR=/var...

[george@Georges-MacBook-Pro frontend % npm run dev]

> frontend@0.1.0 dev
> next dev

  ▲ Next.js 16.0.3 (Turbopack)
  - Local:      http://localhost:3000
  - Network:    http://10.0.0.11:3000

  ✓ Starting...
  ✓ Ready in 1327ms
```

Step 20: Run the Backend

Open New Terminal window → Navigate to Backend

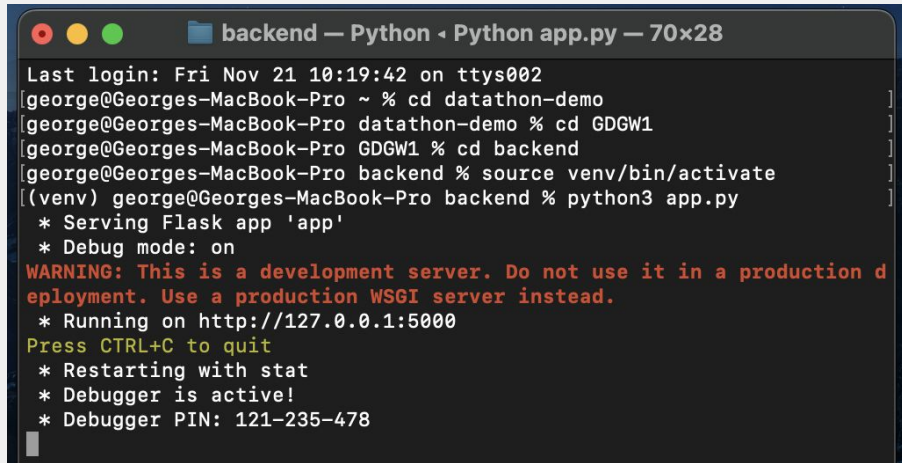
Activate venv → Run the Backend

Type the command based on your platform to activate the venv

- `source venv/bin/activate` (Mac/Linux)
- `.\venv\Scripts\activate.bat` (Windows)

Run the Backend

- `python3 app.py`

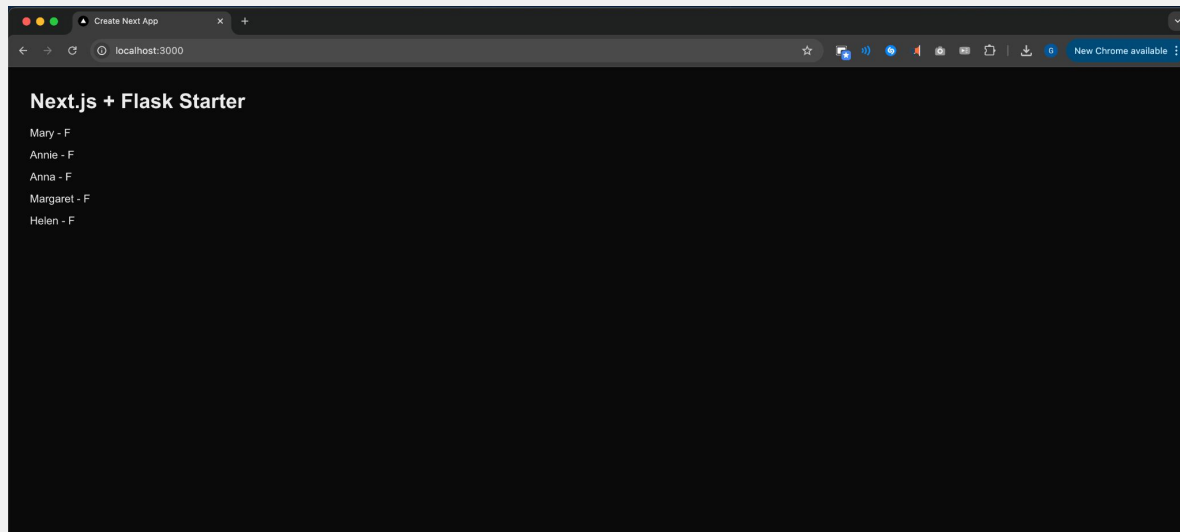


```
backend — Python < Python app.py — 70x28
Last login: Fri Nov 21 10:19:42 on ttys002
[george@Georges-MacBook-Pro ~ % cd datathon-demo ]
[george@Georges-MacBook-Pro datathon-demo % cd GDGW1 ]
[george@Georges-MacBook-Pro GDGW1 % cd backend ]
[george@Georges-MacBook-Pro backend % source venv/bin/activate ]
[(venv) george@Georges-MacBook-Pro backend % python3 app.py ]
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production d
ployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 121-235-478
```

Step 21: Open your WebApp

localhost:3000 - Default location where Next.js hosts the Frontend

- <http://localhost:3000>

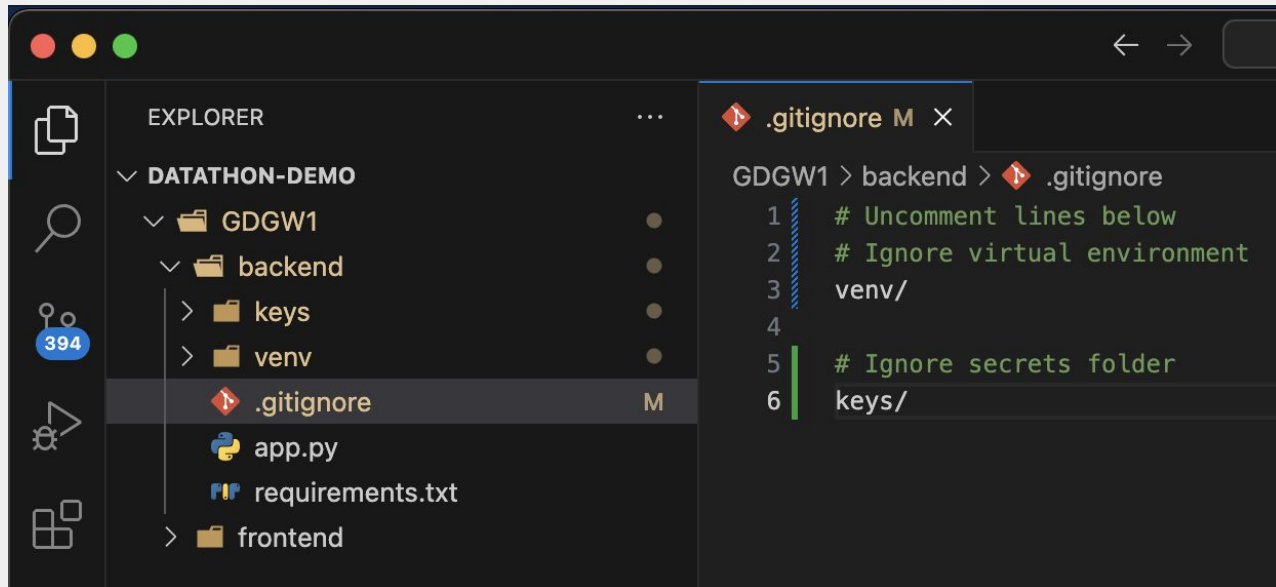


Step 22: .gitignore

Locate the .gitignore in /backend

Ensures you do not push anything you do not want public

- Select All
- cmd + /



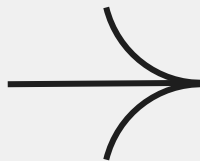
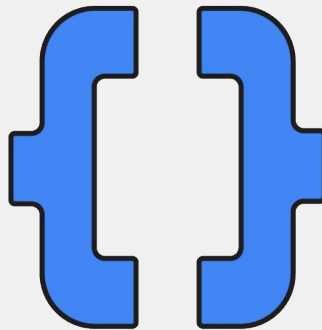
Recap

Flow

- User interacts with Next.js
- Next.js sends requests to Flask
- Flask authenticates using service account
- Flask sends SQL query to BigQuery
- BigQuery returns results
- Flask responds to frontend
- Frontend displays result

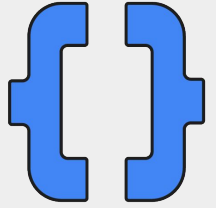
Execution

- Start Backend - Flask -> `venv -> python3 app.py`
- Start FrontEnd - Next.js -> `npm run dev`





Google Developer Group
Sheridan College



Thank You!

We hope you enjoyed the presentation!

