Baby Name App Project Instructions

Part 1: Install Ubuntu (Windows Subsystem for Linux – WSL)

1. Enable WSL and Required Features via Windows Features
   1. Open the Windows search bar
   2. Type “Turn Windows features on or off” and click on the matching result to open the Windows Features dialog
   3. Scroll through the list and check the boxes next to:
      1. “Windows Subsystem for Linux”
      2. “Virtual Machine Platform”
      3. “Hyper-V”
2. Click “OK” to apply the changes
3. Restart your computer if prompted
4. Download and Install Ubuntu on Windows
   1. You can download Ubuntu for free from the Microsoft Store
   2. If you have installed and uninstalled Umbuntu in the past, you may have to do the following in Powershell:
      1. Run: wsl --unregister ubuntu
      2. Run: wsl –install
5. Enter a Linux username and password. Don’t forget what they are!

Part 2. Create an Account in GitHub

1. Visit [GitHub.com](https://github.com/) and follow the sign-up process.

Part 3: Install VS Code (Visual Studio Code) and Enable Python, WSL and GitHub Repositories

1. Download VS Code
   1. Download and install VS Code from the [Visual Studio Code website] (<https://code.visualstudio.com/>)
2. Install the Python Extension for VS Code
   1. Open Extensions (`Ctrl+Shift+X`) in VS Code and install the Python extension provided by Microsoft.
3. Install Remote – WSL Extension
   1. In Extensions, search for and install the "Remote - WSL" extension.
4. Install GitHub Repositories Extension for Visual Studio Code
   1. In Extensions, search for "GitHub Repositories" by GitHub.
5. Sign in to GitHub
   1. After installing the GitHub Repositories extension, you'll need to sign in to your GitHub account:
   2. Open the Command Palette by pressing Ctrl+Shift+P
   3. Type GitHub: Sign in and select the command.
   4. You will be prompted to authenticate with GitHub; follow the instructions provided. This might involve opening a browser to complete the sign-in process.

Part 4: Fork the Baby-Name-App-Project-Files Repo to your personal GitHub Account

1. Navigate to <https://github.com/Chelsea-Myers/Baby-Name-App-Project-Files> and click “Fork”
   1. Forking a repository on GitHub involves creating a personal copy of someone else's project. This allows you to freely experiment with changes without affecting the original project.
   2. Copy the location of your fork by clicking into the new repo and clicking “code”, which will give you an html link to the location of your GitHub repo

Part 5: Clone the Repo

1. Open the Command Palette by pressing Ctrl+Shift+P and type “WSL: Open Folder in WSL”
   1. Navigate to the directory where you want to download the project files
2. Click Terminal at the top of VS code to open a new Terminal
3. Change the Terminal from PowerShell to WSL
   1. Click on the down arrow next to the + and change the terminal type from Powershell to WSL
4. Clone the Baby-Name-App repo
   1. Run: git clone <link to the repo you created above> It will look something like https://github.com/Chelsea-Myers/Baby-Name-App-Project-Files
   2. This step creates a Baby-Name-App-Project-Files directory with the files downloaded from GitHub.
5. You can verify that it exists in the Explorer menu on the left
6. Change directory into the Baby-Name-App-Project-Files folder by opening the Command Palette, typing “WSL: Open Folder in WSL” and clicking on the new project folder

Part 6: Set up a Virtual Environment

1. Install Prerequisites
   1. Ensure Python and pip are installed on your system:
      1. Run: sudo apt update
      2. Run: sudo apt install python3 python3-pip
2. Check the Python and pip versions to confirm successful installation:
   1. Run: python3 --version
   2. Run: pip3 --version
3. Create a Virtual Environment
   1. Navigate to the project directory where the virtual environment will be created by opening the folder in WSL
4. Use Python's built-in venv module to create the virtual environment:
   1. Run: python3 -m venv venv
   2. This creates a folder named venv within your project directory.
5. Activate the Virtual Environment
   1. Use the following command to activate the virtual environment: source venv/bin/activate
   2. After activation, you should see (venv) at the beginning of your terminal prompt. This indicates the virtual environment is active.
6. Install Project Dependencies
   1. Run: pip install -r requirements.txt
   2. You'll see pip begin downloading and installing the project dependencies
   3. This can take about 10 minutes

Part 7: Open and Edit the Project Files in VS Code

1. In the Explorer on the left hand side, you should see two folders: data and static, and several files including app.py, LICENSE, Procfile, README.md and requirements.txt.
2. Click on app.py
3. Under the comment
4. Under the comment   
   #Extract year and rank in year for the given name-sex combination:
   1. Create a subset of the babynames DataFrame called name\_subset where (babynames['name'] == name) & (babynames['sex'] == sex) Your code should look something like   
      name\_subset = babynames[<code to create subset>]
   2. Save the year feature as a list using   
      name\_years = name\_subset[‘year’].tolist()
   3. Save the rank feature as a list using   
      name\_ranks = name\_subset[‘rank\_in\_year’].tolist()

    # Extract year and rank in year for the given name-sex combination

    name\_subset = babynames[(babynames['name'] == name) & (babynames['sex'] == sex)]

    name\_years = name\_subset['year'].tolist()

    name\_ranks = name\_subset['rank\_in\_year'].tolist()

# Load babynames.csv into a pandas dataframe

babynames = pd.read\_csv('./data/babynames.csv', names=['sex', 'year', 'name', 'count'])

# Construct a column giving the rank within each year and sex for each name

#

# e.g. Mary is the #1 ranking name for girls in 1910

#      John is the #1 ranking name for boys in 1910

babynames['rank\_in\_year'] = babynames.groupby(['year', 'sex'])['count'].rank(ascending=False)

1. Save the file

Part 8: Commit the Changes to GitHub

1. Click on Source Control on the left side of VS Code.
   1. You should see a summary of the changes you have made to your repo.
   2. Enter a message and click Commit and Push.
   3. This makes a record of the changes you have made to your documents and makes those changes to the documents in your GitHub repo. There is a way to do this programmatically, but we’ve run into problems doing it because of – we think – the Full Sail firewall.

Part 9: Run the Flask app locally

1. Run the Flask app
   1. Run: flask --app app --debug run
   2. This command starts the Flask application using the built-in server and the app.py file
2. View the webpage
   1. Go to http://localhost:5000 in your browser

Part 10: Getting Your App Set up in PythonAnwhere

1. On your computer, delete the venv folder and zip the folder where the app files are.
2. Create a PythonAnywhere account. It will allow you to host one web app for free.
3. Click on Open Web Tab under All Web Apps
4. Click on Add New Web App
5. Agree to the domain name given by PythonAnywhere
6. Click on Manual Configuration
7. Click on Python 3.10 (or the most recent version of Python available)
8. Upload your app Files to PythonAnywhere
   1. Click Files
   2. Upload your zipped file.
   3. Click on Open Bash Console Here
   4. In the Bash console, type unzip Baby-Name-App-Files.zip
   5. This should unzip your files into the directory ‘home/yourusername/Baby-Name-App-Files’

Part 11: Setting up the Virtual Enviornment

1. Open a Bash Console in PythonAnywhere
2. Create the virtual environment (venv) directory
   1. Run: virtualenv venv
   2. This creates a directory named venv
   3. Verify that it exists by typing ls
3. Activate the virtual environment
   1. Run: source venv/bin/activate
   2. This will make a (venv) notation appear at the front of your command prompt
4. Install dependencies using requirements.txt
   1. Run: pip install -r requirements.txt
   2. You'll see pip begin downloading and installing the project dependencies
   3. This will take a few minutes

Part 12: Deploying the App

1. Click on “Web”
2. In the "Code" section, you're typically asked to specify the source code and the working directory:
   1. Source Code: This is the path to the directory where your PythonAnywhere web app's Python code lives. It will probably be something like ‘home/yourusername/Baby-Name-App-Files’
   2. Working Directory: This is often the same directory where your source code is. It's the directory PythonAnywhere will use as the current working directory when it starts your web app. It should be the same as above and is probably something like ‘‘home/yourusername/Baby-Name-App-Files’
3. In the “Virtualenv” section
   1. Virtualenv Path: This is the path to the virtual environment you've created for your project. Assuming you create the virtual environment in your home directory, the path you'd enter in the "Virtualenv" field on the web app setup page would be something like `home/yourusername/Baby-Name-App-Files/venv’.
4. Edit the WSGI Configuration file to contain only the text:

import sys

# Add your project directory to the sys.path

path = '/home/Chelsea/Baby-Name-App/' #The directory where your file apps are

if path not in sys.path:

sys.path.append(path)

# Set the 'application' variable to your Flask app

from app import app as application

**Everything else should be deleted or commented out.**

1. Click “Reload”
2. Navigate to and run your web app!