On-boarding tech stack for Python development.

The following is a highly suggested tools you need to install before the start of our work shop. The intent of this list is to keep consistent development environment that we can grown with.

If you are using a Mac, most of the tools are pre-installed. The following are to help Windows users to reach feature parity with the Mac environment.

- Git (Windows)
  - https://git-scm.com/download/win
  - Remember to select "Use Git and optional Unix tools from the Command Prompt", under the "Adjusting your PATH environment" dialog.
    - You will have access to most of the Unix utilities in your Command Prompt.

- Install Python (Mac or Windows)
  - https://www.python.org/downloads/
  - Minimum version to use is 3.7

- Install Python the virtual environment, `pipenv` (Mac or Windows).
  - From yoiur command prompt type the following:
    ```
    pip install pipenv
    ```
  - If you are prompted to upgrade your pip version, go ahead and do it.

- Install VSCode (Mac or Windows)
  - https://code.visualstudio.com/download
  - Once installed, click on the extensions icon on the the left bar to search and install the following extensions:
    - Python by Microsoft
    - SQL Server by Microsoft
    - Live Share by Microsoft
    - Bracket Pair Colorizer by Coenraads
    - Excel Viewer by GrapeCity
    - Json Editor by Nick DeMayo
    - REST Client by Huachao Mao
    - GitLens by Eric Amodio

- Slack (Mac or Windows)
  - https://slack.com/downloads/windows (My browser keeps pointing me to the windows download)
  - The workspace is: **cstuworkspace** .slack Get your login invite from Glen Qin.
  - This shall be the centralized communication platform. I suggets install it on your

phone.

- Install C/C++ compiler (Windows)
  - https://visualstudio.microsoft.com/thank-you-downloading-visual-studio/?sku=BuildTools&rel=16
  - https://www.itechtics.com/microsoft-visual-c-redistributable-versions-direct-download-links/#Microsoft_Visual_C_Redistributable_2019
  - Need it to compile the **`ujson`** package.

Version control with Git

Once you have installed the above tools, you are ready to work with the code in my personal repository. The following are done once.

Configure your commit profile.

```
git config --global user.name "First Last"
git config --global user.email "email@example.com"
```

Clone my repo down to you computer.

```
git clone https://github.com/elau1004/ETLite.git
```

Install the dependent libraries for the cloned project folder.

```
pipenv sync
```

The following are the basic commands that you will use on a regular basis.

Create your working branch. Never work on the **master** branch!

```
git checkout -b "branch name"   # FirstName-Task
```

Display the branches on your local repository.

```
git branch
```

Switch between working branch.

```
git checkout master
git checkout "branch name"
```

Get the status of the current branch.

```
git status
```

Pull the latest code down from the remote repository. Do this beginning of day!

```
git pull
```

Merge the code from a different branch onto your current branch. Do this beginning of day!

```
git merge master
```

Review differences between current and original file.

```
git diff "path to your file"
```

Stage your changes for commit into your local repo. Don't use wildcard or entire folder!

```
git add "path to your file"
```

Commit your changes. Do provide a good description!

```
git commit -m "Be descriptive"
```

Push your local repo back up to the remote repo.

```
git push
```

When things are working and you want to contribute to the project, request a pull of your contribution branch into the master branch.

https://github.com/elau1004/ETLite/pulls

Your submission shall be reviewed. Do NOT be offended if your pull request is rejected.

Review Process

Dos:

1. Look for Pythonic improvement.
2. Look for design flaws.
3. Funtionality
4. Security
5. Performance
6. Maintainability
7. Review within 24 hours of submission.
8. Initial discussions shall be private and direct between reviewer and submitter.
9. Do comment in the PR, if only your want to escalate the issue up a level for the community to discuss.
10. Understand the context of the review and not just the code differences between revisions.

Donts:

1. Nip pick the style and spelling. PEP-8 is the starting point with my over-writes.
2. Let a review hang for more than 48 hours.