*Florida International University*

*School of Computing and Information Sciences*

ASI PantherCentric 1.0 Installation Guide

**Team Members**:

Jake Lopez

Alfredo Santoyo

Rolando Moreno

Pricsila Cordova

**Product Owner**:

Myrian Herlle

**Mentor:**

Myrian Herlle

Francisco Ortega

**Instructor:**

Masoud Sadjadi

**Installation Guide for Development & Running with Docker**

To help facilitate development, some members have found it useful to use Docker to run the services necessary to operate the application. The following sections will be a step-by-step guide on how to set up Docker to both host and continue development on the project.

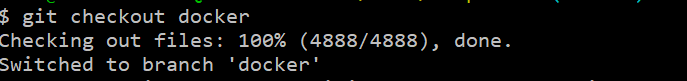
All the following steps are meant to be executed in your systems command line interface of choice. If you are using a Windows based operating system, please use either PowerShell or Git Bash which is included with the GitHub Windows tools.

**Pre-requisites**

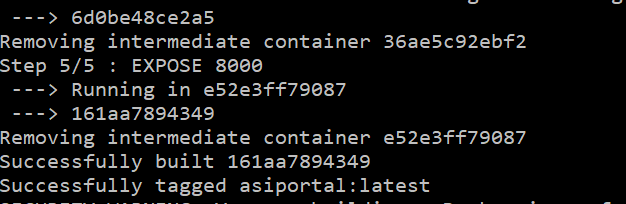
First, install docker from the docker website located at <https://www.docker.com>

**Docker set-up**

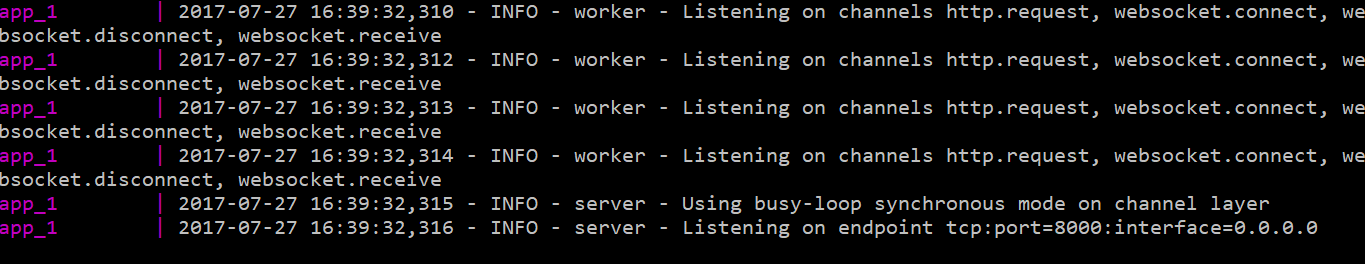
1. Checkout the latest version of the project from the GitHub repository. The latest version is maintained by the Academic Success Initiative (ASI), so you should request access from their internal development team.
2. Once you have the copy of the latest repository, make sure to check out the Docker compatible version by executing “git checkout docker” in the directory where you have the git repository stored.Your command line output should resemble that of **Figure 1**.
3. Next, you must build the necessary Docker images. The first step is to build the initial container which will host the Python code. To do this execute “docker build -t asiportal .” in your command line in the root directory of the repository. This command may take up to 10 minutes to complete running. Once completed, your command line output should be like **Figure 2**.
4. The next and last step is to build the other necessary images to run the application. To do this, execute “docker-compose up” in the root directory of the repository. This command may also take upwards of 10 minutes to complete running. Once completed, the output should look like **Figure 3**.
   1. Note that if you see some output like **Figure 4** The application will not finish setting up. You must enter “Control-C” to stop the message, then run “docker-compose up” again.
5. Once you see successful build output like that in Figure 3, you may now connect to the application by visiting the localhost:8000 in your web browser. You should be greeted by a screen like that in **Figure 5**.
   1. For the ease of development all passwords have been set to “password”. So, to log in as any user simply enter their username and “password” for password.



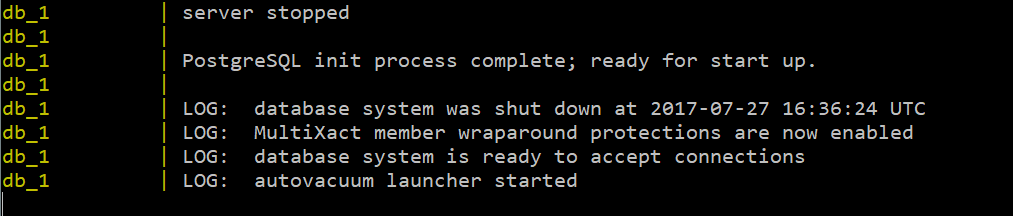
**Figure 1.** Result of executing “git checkout docker” in the command line.



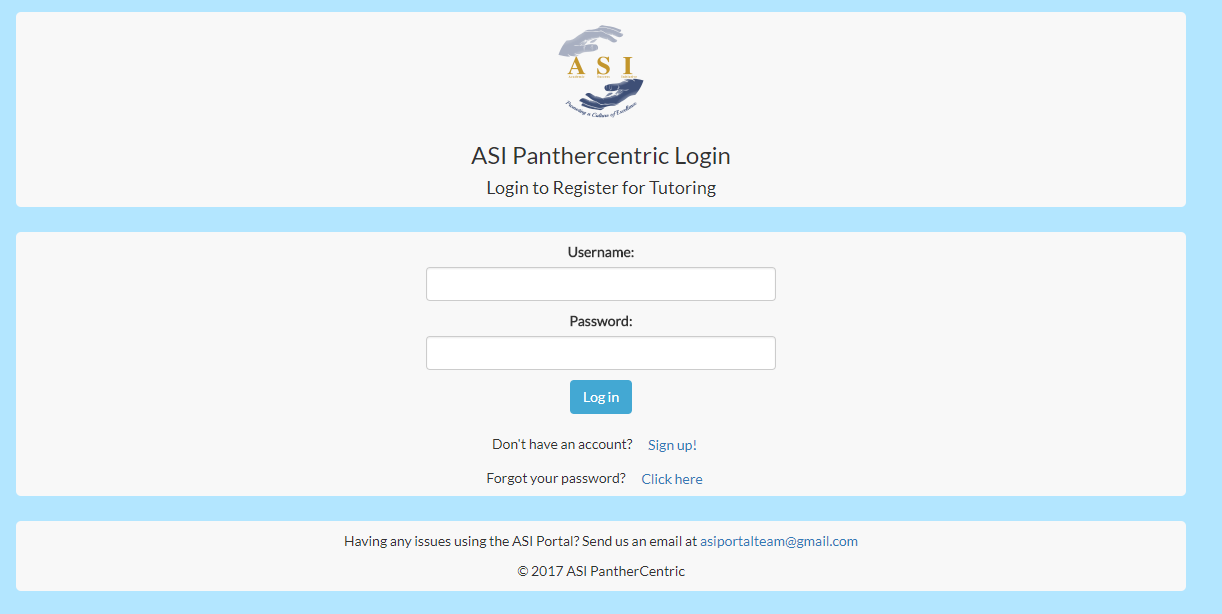
**Figure 2.** Result of successful completion of the “docker build -t asiportal .” command



**Figure 3.** Successful completion of “docker-compose up” command.

****

**Figure 4.** Common issue when first running “docker-compose up”. The process will hang and not complete the set-up of other images. To finish simply enter “Control-C” and run the “docker-compose up” command again.

****

**Figure 5.** The homepage of the ASI PantherCentric 1.0 project.

**Installation Without Docker**

In the case that you are unable to install Docker, or simply do not wish to install Docker, you are still able to get up and running with the installation of a few services. It is highly recommended that you do any development using either Linux or OS X if you choose this route of installation.

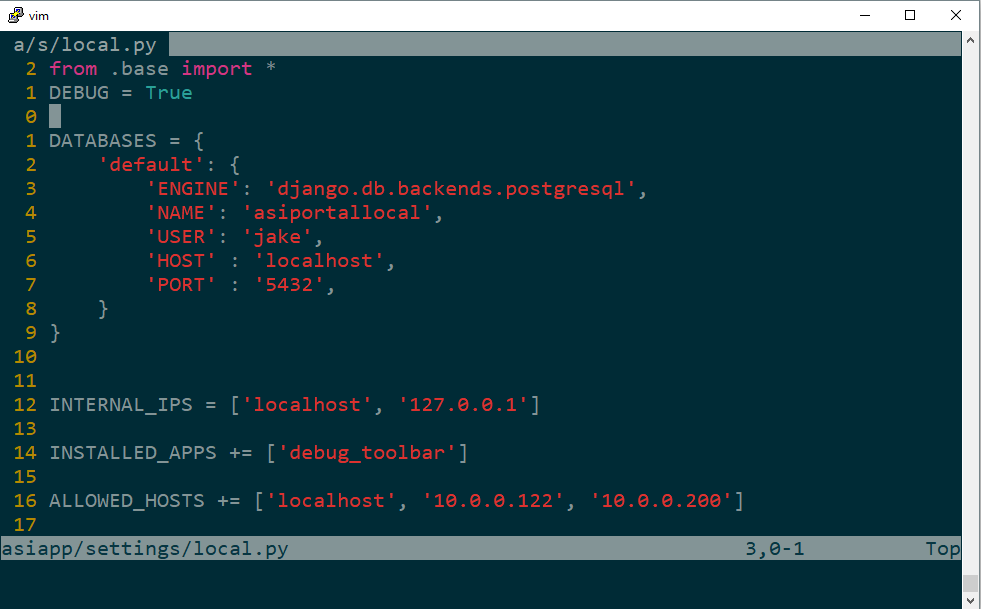
There are a few warnings that should be issued prior to going this route while working with Windows. The main warning is that Redis, a necessary service for the application, has no Windows installer. To install it you must compile Redis from source. While not impossible, it is a bit of an inconvenience. If you do not feel comfortable with this process then please consider either using Docker or a Linux virtual machine.

**Pre-requisites**

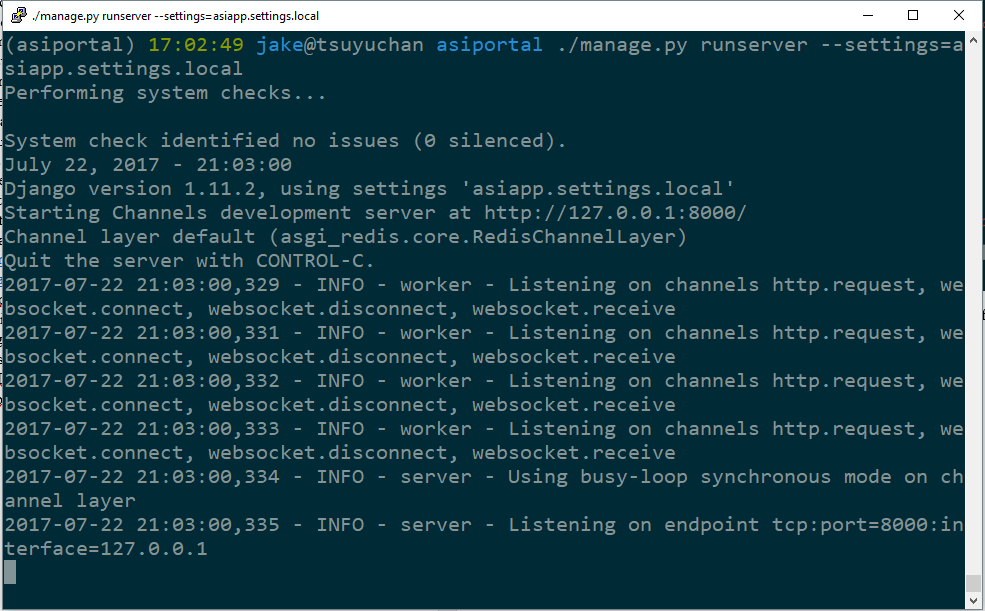
* Install Redis for your platform by visiting: <https://ww.redis.io>
  + If on a Linux distribution it is recommended to install this service using your distributions package manager.
* Install PostgreSQL for your platform by visiting: <https://www.postgresql.org>
  + If on a Linux distribution it is recommended to install this service using your distributions package manager.
  + When you create your database for the project, make sure to initialize it with the SQL file located in the repository root titled “init.sql”.
* Install Python (version 3.6 or higher) by visiting: <https://www.python.org>
  + If on a Linux distribution it is recommended to install this service using your distributions package manager.

**Setting Up**

1. Checkout the latest version of the project from the GitHub repository. The latest version is maintained by the Academic Success Initiative (ASI), so you should request access from their internal development team.
2. It is always recommended that you start a new branch for development. If you are working in a team and other members are using Docker, then checkout the docker branch.
3. Next, you will want to configure a settings file. A good example file to copy and adjust is located at: repository\_root/asiportal/asiapp/settings/local.py
4. For the most minimal amount of change to get the project running, please edit the HOST, PORT, USER, and NAME variables as shown in **Figure 6.**
5. Once you have made your changes, ensure that both PostgreSQL and Redis are running.
6. Next, you will need to install the Python packages necessary for this project. It is recommended to create a Python virtual environment to ensure there are no conflicts between any global packages you may have already installed. To learn more about how to accomplish this, visit <http://python-guide-pt-br.readthedocs.io/en/latest/dev/virtualenvs/>.
7. Navigate to the folder located at: repository\_root/asiportal
8. From here, execute the command “pip install -r requirements.txt”
9. This will install all necessary packages to run the application.
10. Once you have installed all packages, simply execute “python manage.py runserver –settings=asiapp.settings.YOUR\_SETTINGS\_FILE\_NAME” to run the application. This should be run from the repository\_root/asiportal directory. Correct output is shown in **Figure 7.**
11. You should now be able to navigate to <http://localhost:8000> and view the application in your web browser. **Figure 5** should resemble the page you see when you visit localhost:8000.

****

**Figure 6.** The settings file which should be manipulated to fit your Postgres configuration.



**Figure 7.** Successful output of running python manage.py runserver –setings=asiapp.settings.YOUR\_FILE\_NAME