SymCheck Installation Manual

About: What is SymCheck?

SymCheck is a self-diagnosing web application that allows users to upload a photo of a potential ailment he/she may have to a webpage, and then receive an estimated diagnosis on what that illness/ailment may be. The application itself utilizes a machine-learning algorithm for image classification with an accuracy of about 78%, and while it is in no way a complete *replacement* for physical medical diagnoses or care from trained medical professionals, it strives to enable users to save time and money by using the application to determine if a physical visit to a healthcare facility is, in fact, necessary.

Dependencies

At this point in time, the successful running of the complete SymCheck web application on a machine is via localhost, and requires the following environment be installed on the user's computer:

- **Python 3.5 – 3.8**
  - Please note that only *these specific* versions of Python work with the application as it is a dependency outlined in version 2.0 of TensorFlow (required for SymCheck's machine-learning-based algorithm of image classification)
  - The most recent version of Python, 3.9.0 is *not yet supported* by TensorFlow 2.0

- **Python pip 20.2.4**
  - This is a package-management system used to install and manage various software packages in Python
  - This is the most recent version of pip available, and will be used to install TensorFlow, and Pillow.

- **Django 3.1.3**
  - This is the most recent version of Django, a Python-based web framework used for the SymCheck application, available for use. This framework must also be installed on the user's computer in order to run the server used to access the SymCheck application via localhost.

- **Python Pillow 8.0.1**
  - Pillow is a user-friendly version of PIL, Python Imaging Library. This is another required install as it allows any images submitted to the SymCheck application's local media folder to be identified

- **TensorFlow 2.0**
  - This is the most recent version of TensorFlow available for use, TensorFlow 2.3.0, and must be installed to support the machine-learning-based classification of images for diagnosis

Installation on MacOSx

(for instructions on installation for Windows, skip to page 10)

***This installation guide assumes that you have Homebrew installed on your machine. If this is not the case, please go to https://brew.sh/ and follow the steps to install Homebrew, a user-friendly package manager and software installation system for MacOS and Linux.

Step 1: Install Python

Python is installed on all new Macs but it is an older version (Python 2.0) that is not supported by TensorFlow 2.0. Thus, we start by opening the Terminal app (you can do a spotlight search of Terminal if you are not sure where this application is located by pressing COMMAND + SPACE and typing in "Terminal"). Normally, we would install Python by simply entering the following command:

```
$ brew install python3
```

DO NOT USE THIS COMMAND. Because Homebrew is open-source, and by default installs the most up-to-date software, this will install Python 3.9.0 and we DO NOT WANT THIS as it will not work with TensorFlow 2.0. So instead, we use pyenv of

Homebrew which is essentially a collection of different versions of Python to install the version we want:

```
$ brew install pyenv
$ pyenv install <version>
```

For the purposes of this tutorial, we will be installing version 3.8.0 (insert the version in place of “<version>”.

```
$ pyenv local 3.8.0
```

The above command essentially sets the version that you desire. If you wish to list all available versions, simply use the command:

```
$ pyenv versions
```

You must also, however, update *PATH* to override the default version of Python on your machine using:

```
PATH="~/.pyenv/versions/3.8.0/bin:${PATH}"
```

For any additional information on installing a specific version of Python using Homebrew, go to https://gist.github.com/Bouke/11261620.

Now, in the event that you have **trouble switching between versions of Python** (this happens when a user might have many different versions of Python installed on his/her computer in different locations, we recommend removing these versions (don’t worry, you will, by default, not be able to remove the version of Python that comes with your Mac). To do this, you must first locate where you have Python 3 installed using the command:

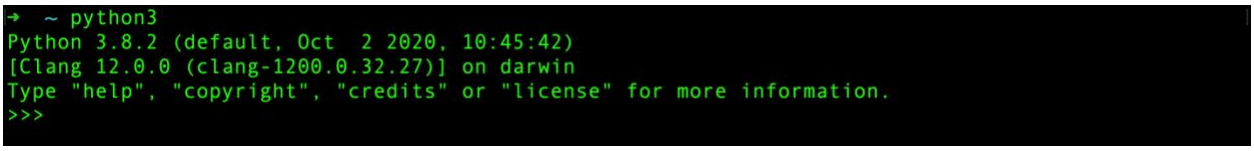
```
$ which python3
```

Now, take the paths of these locations and delete the Python installations using:

```
$ sudo rm -R <pythonDirectoryPath>
```

And now you can go back and install a specific version of Python as outlined above.

Now that you have installed Python 3 on your machine, test the installation by running Python. Note, we must type in “python3” whenever running Python, otherwise the system will default to running the factory version of Python 2. See the screenshot below to confirm that you have successfully installed a version of Python 3:



Great! Now exit Python3 by entering:

```
>>>quit()
```

And now we are ready to install the pip package manager.

Step 2: Install pip

Enter the following command to install pip:

```
$ sudo easy_install pip
```

Step 3: Install a virtual environment for Python and for Django

Type the following command in your terminal after installing pip:

```
$ sudo pip install virtualenv
```

Now we need to create a folder, to hold all the files needed for our virtual environment so first, make a directory for your virtual environment (you can name it anything you like, for the purposes of this tutorial, it has been named “pyvenv”), and then install the necessary folders as such:

```
$ mkdir pyvenv
$ virtualenv pyvenv
```

Now, go into the virtual environment folder with “cd” (change directory) and enter the following command to activate/start the virtual environment:

```
$ source bin/activate
```

You should now see the name of your virtual environment as the prompt in your terminal as such:

```
➜  ~ cd pyvenv
➜  pyvenv source bin/activate
(pyvenv) ➜  pyvenv
```

Note that you can exit the virtual environment at any time by simply typing the command “deactivate”, but for now, stay within this environment and within the pyvenv folder as well for the installation of Django.

Step 4: Install Django

While still within the virtual environment folder, enter the following command:

```
(pyvenv) $ sudo pip install django==3.1.3
```

You can then check the version using:

```
(pyvenv) $ python3 -m django –version
```

Great, you now have Django installed on your computer within your virtual environment. To verify the installation, you can create a new Django project using the following commands which create the skeleton of a Django project, enter the project folder, and run the Django server respectively:

```
(pyvenv) $ django-admin startproject sample
(pyvenv) $ cd sample
(pyvenv) $ python3 manage.py runserver
```

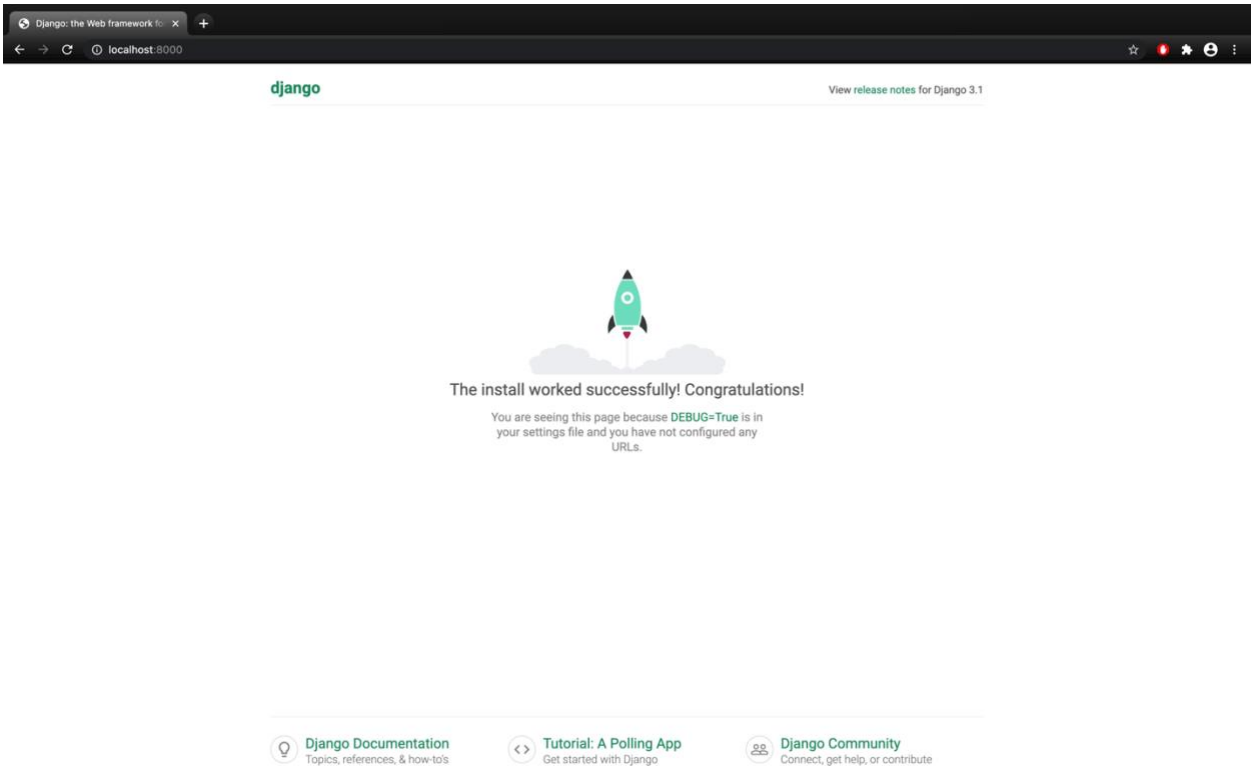
Once the server is running in your terminal as shown below:

```
(pyvenv) ➜  pyvenv django-admin startproject sample
(pyvenv) ➜  pyvenv cd sample
(pyvenv) ➜  sample python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes
, sessions.
Run 'python manage.py migrate' to apply them.
November 16, 2020 - 01:10:28
Django version 3.1.3, using settings 'sample.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Open your web browser of choice, and go to http://localhost:8000 or http://127.0.0.1:8000 (note the numeric address in the second URL is simply the IP of localhost). You should see the following page to confirm that you have installed Django successfully:



***To exit the server at any time, enter CONTROL + C into your terminal.

Step 5: Install TensorFlow

Now, that you have successfully installed Django, navigate back to your virtual environment folder (i.e. exit the sample project with “cd ..”) and enter the following command to install the latest version of TensorFlow:

```
(pyvenv) $ pip install --upgrade tensorflow
```

You can then test this installation with the following command:

```
(pyvenv) $ python -c "import tensorflow as
tf;print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
```

For more information on the installation of TensorFlow using pip, feel free to check out TensorFlow’s own installation page at: https://www.tensorflow.org/install/pip.

Step 6: Install Pillow

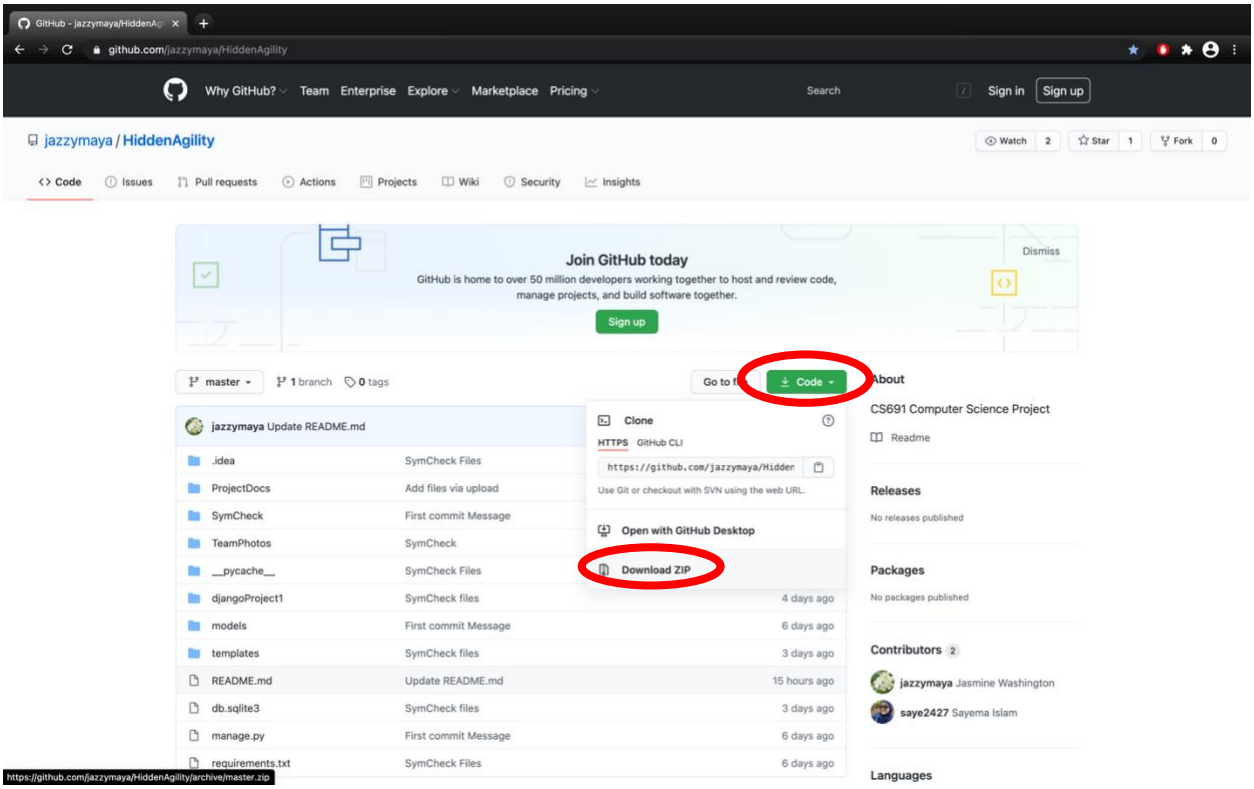
Lastly, to ensure that we do not receive any errors when submitting images through the SymCheck application server, use the following command:

```
(pyvenv) $ sudo pip install pillow
```

And there you have it, you are now ready to run the SymCheck application via localhost!

Step 7: Download SymCheck Files from GitHub

In order to run the SymCheck application, you will need to have a copy of the application's files on your local machine. This can be done by navigating to HiddenAgility's GitHub repository for SymCheck at: https://github.com/jazzymaya/HiddenAgility and clicking the green download button above the list of files, and clicking "Download ZIP" as shown below:

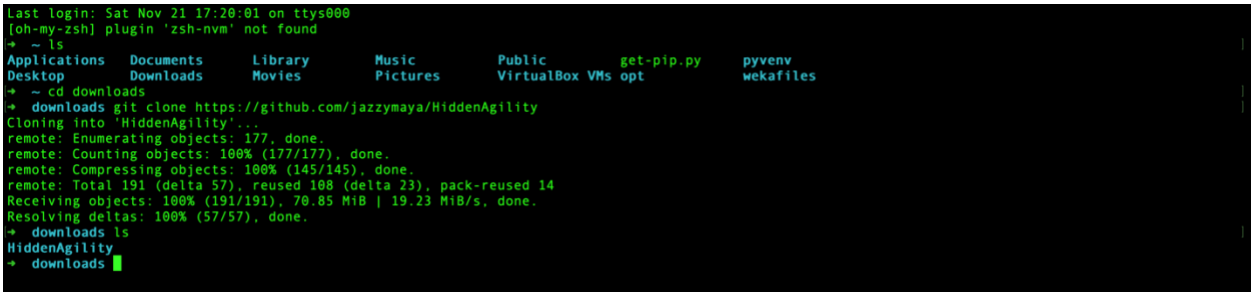


You can then unzip the downloaded folder, and voila! You now have the SymCheck files available on your computer.

Alternatively, you can pull a copy of the SymCheck code files from GitHub directly through your terminal using the following command:

```
$ git clone https://github.com/jazzymaya/HiddenAgility
```

This will save the SymCheck files to whatever directory you are in when executing said command:



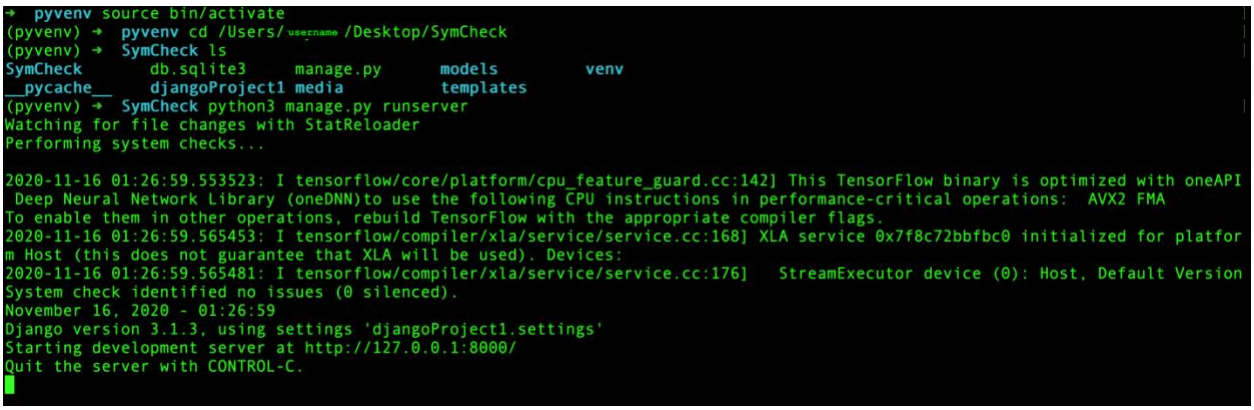
Now with the SymCheck files available on your local machine, we are ready to go forth with running the application!

Step 8: Running SymCheck via Localhost

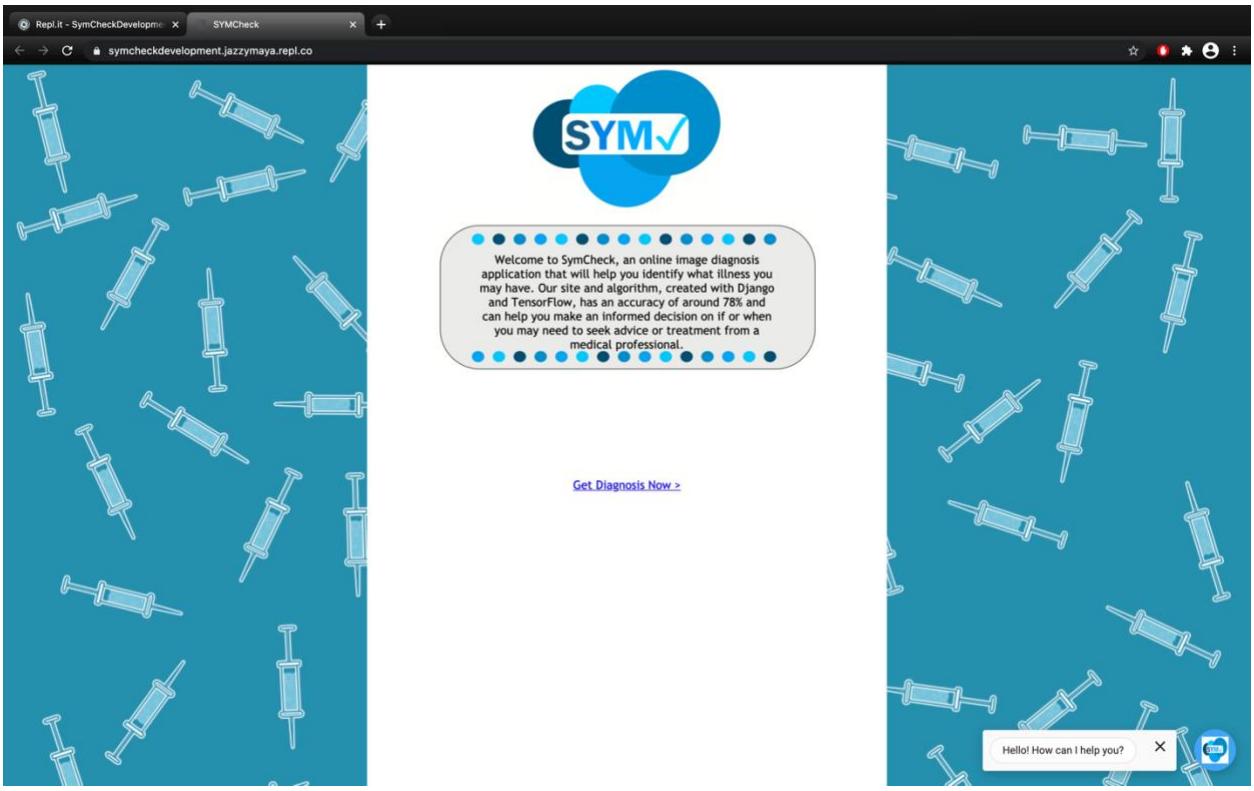
With your virtual environment activated, navigate to and enter your folder with all of SymCheck's files, and enter the command:

```
(pyvenv) $ python3 manage.py runserver
```

just as you did for the sample project:



Then navigate to http://localhost:8000 or http://127.0.0.1:8000 in your browser, and you should see the welcome page of SymCheck as shown on the next page:



Congratulations! You can now use the SymCheck web application on your Mac to receive your own image-based diagnosis!

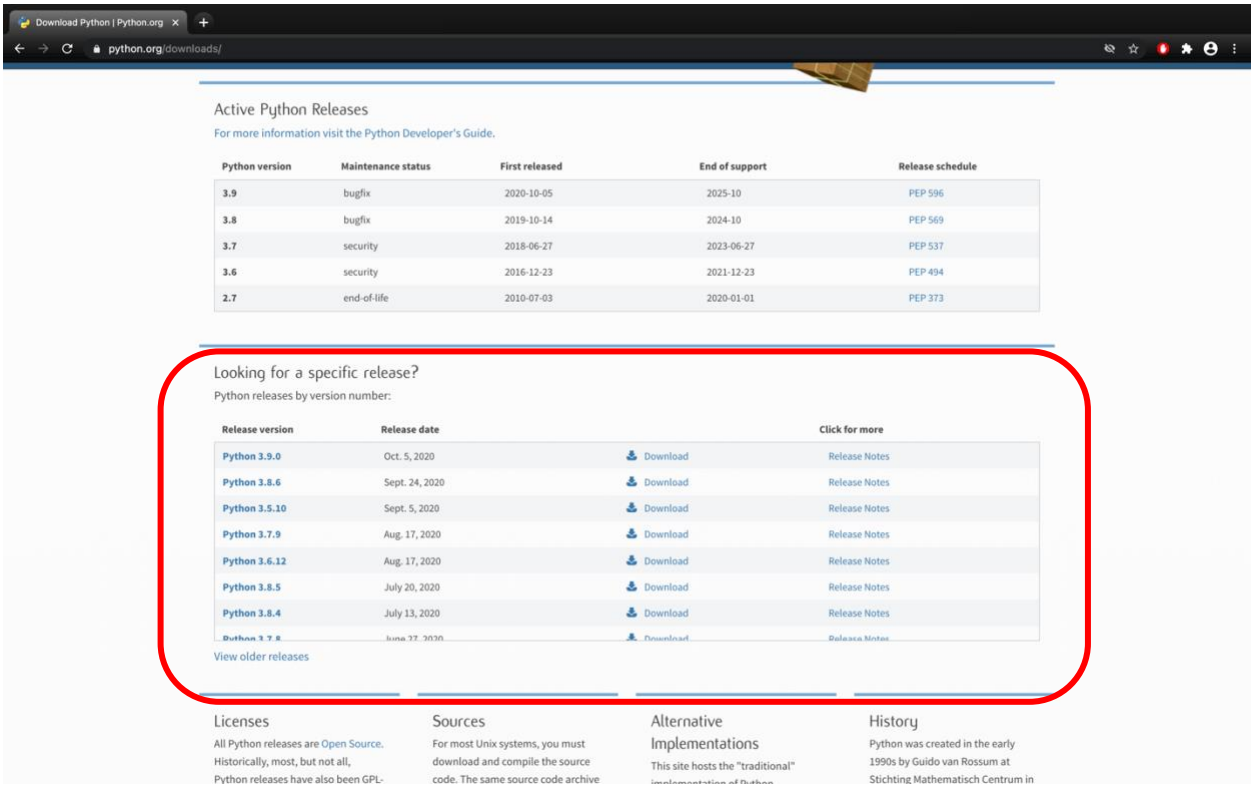
Check out our *SymCheck User Manual* next to learn more about how to navigate the website. And we hope our application serves you well in allowing you to make more informed decisions about your own care.

Installation on Windows

***Note: The setup of the environment for the SymCheck Application is more or less the same for Windows as it is for MacOSx, save the first step for Python 3 installation, and slightly different commands for command line vs. MacOSx's Terminal.

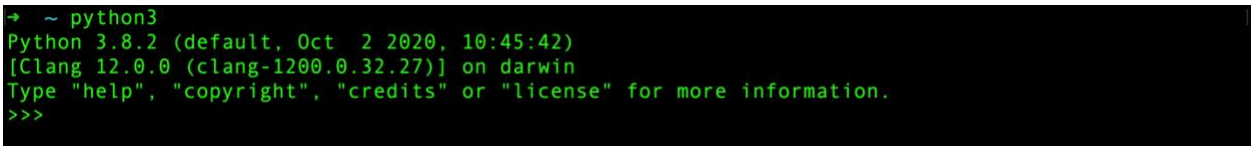
Step 1: Install Python

To install Python on your Windows machine, navigate to https://www.python.org/downloads/. Be sure to download the files for a version of Python *between 3.5 and 3.8* by scrolling down to where it says "Looking for a specific release?"



Download the files for Python 3 by clicking the Download link.
Remember Python 3.9.0 is not supported by the latest version of TensorFlow!

Once you have installed Python 3 on your machine, test the installation by running Python—simply type "python" into your command line as Mac users would type "python3". You should see the same command line prompt beginning with ">>>"



Great! Now exit Python by entering:

```
>>>quit()
```

And now we are ready to install the pip package manager.

Step 2: Install pip

Enter the following command to install pip:

```
> python -m pip install -U pip
```

Step 3: Install a virtual environment for Python and for Django

Type the following command in your terminal after installing pip:

```
> pip install virtualenv
```

Now we need to create a folder, to hold all the files needed for our virtual environment so first, make a directory for your virtual environment (you can name it anything you like, for the purposes of this tutorial, it has been named “pyvenv”), and then install the necessary folders as such:

```
> mkdir pyvenv
> virtualenv pyvenv
```

Now, go into the virtual environment folder with “cd” (change directory) and enter the following commands to activate/start the virtual environment:

```
> cd Scripts
> activate
```

You should now see the name of your virtual environment as the prompt in your terminal as such:

```
(pyvenv) [Folder_path]>
```

Note that you can exit the virtual environment at any time by simply typing the command “deactivate”, but for now, stay within this environment and within the pyvenv folder as well for the installation of Django.

Step 4: Install Django

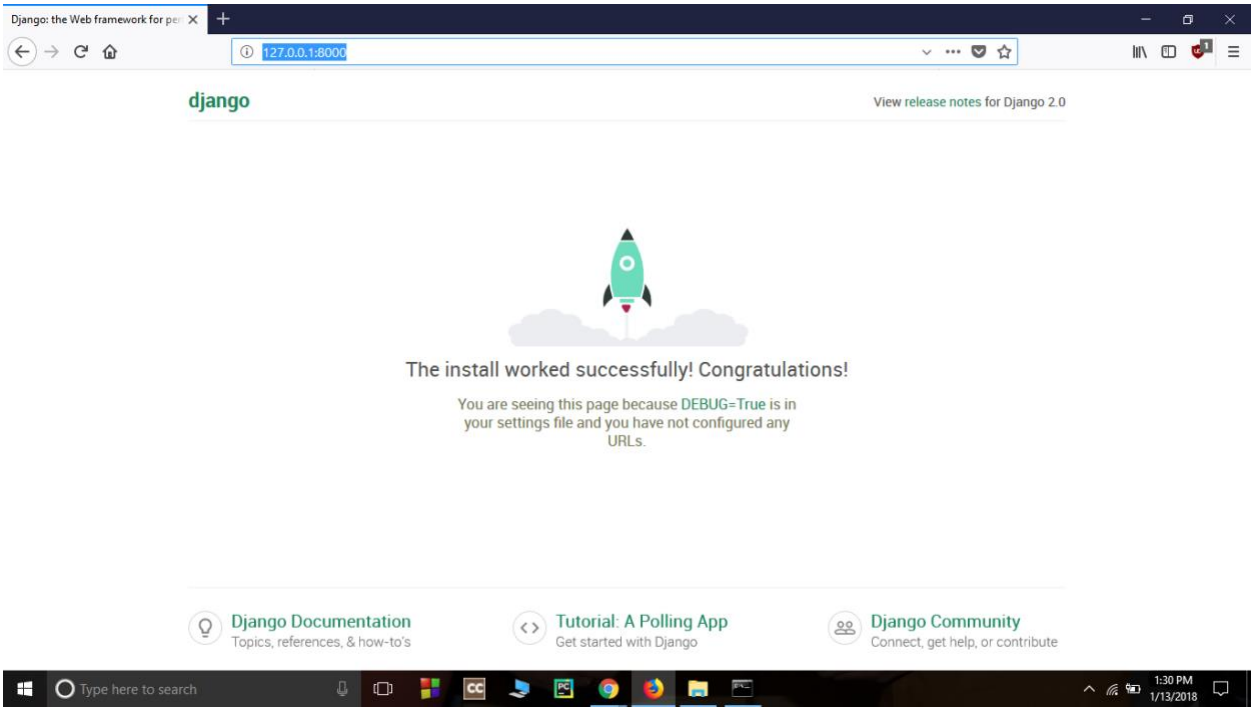
While still within the virtual environment folder, enter the following command:

```
(pyvenv) > pip install django
```

With this command, you should now have Django installed on your computer within your virtual environment. To verify the installation, you can create a new Django project using the following commands which create the skeleton of a Django project, enter the project folder, and run the Django server respectively:

```
(pyvenv) > django-admin startproject sample
(pyvenv) > cd sample
(pyvenv) > python manage.py runserver
```

Once the server is running in your command line, open your web browser of choice, and go to http://localhost:8000 or http://127.0.0.1:8000 (note the numeric address in the second URL is simply the IP of localhost). You should see the following page to confirm that you have installed Django successfully:



For more information on the installation of Django on a Windows machine, feel free to check out: https://www.geeksforgeeks.org/django-introduction-and-installation/.

Step 5: Install TensorFlow

Now, that you have successfully installed Django, navigate back to your virtual environment folder and enter the following command to install the latest version of TensorFlow:

```
(pyvenv) > pip install --upgrade tensorflow
```

You can then test this installation with the following command:

```
(pyvenv) > python -c "import tensorflow as
tf;print(tf.reduce_sum(tf.random.normal([1000, 1000])))"
```

For more information on the installation of TensorFlow using pip for various operating systems, feel free to check out TensorFlow's own installation page at: https://www.tensorflow.org/install/pip.

Step 6: Install Pillow

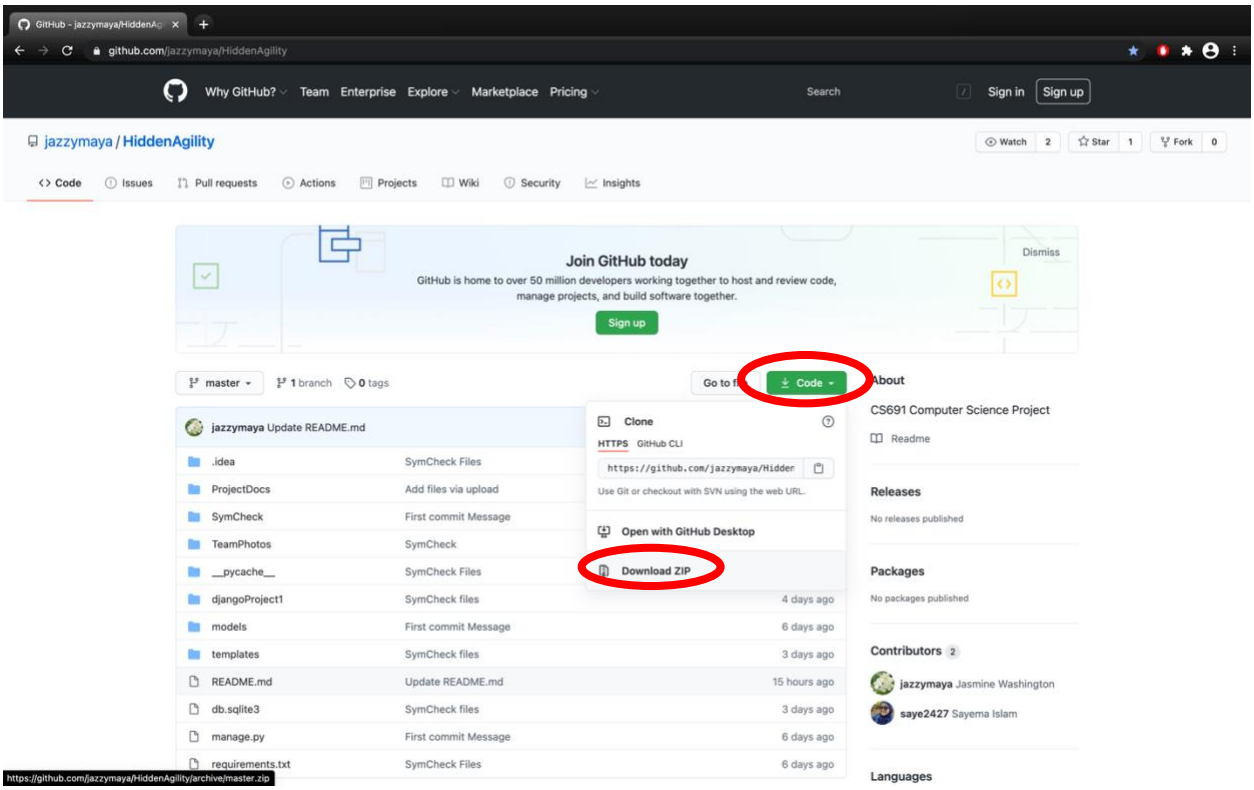
Lastly, to ensure that we do not receive any errors when submitting images through the SymCheck application server, use the following command:

```
(pyvenv) > pip install pillow
```

And there you have it, you are now ready to run the SymCheck application via localhost!

Step 7: Download SymCheck Files from GitHub

In order to run the SymCheck application, you will need to have a copy of the application's files on your local machine. This can be done by navigating to HiddenAgility's GitHub repository for SymCheck at: https://github.com/jazzymaya/HiddenAgility and clicking the green download button above the list of files, and clicking "Download ZIP" as shown below:



You can then unzip the downloaded folder, and voila! You now have the SymCheck files available on your computer.

Alternatively, you can pull a copy of the SymCheck code files from GitHub directly through your command line using the following command:

```
> git clone https://github.com/jazzymaya/HiddenAgility
```

This will save the SymCheck files to whatever directory you are in when executing said command:

```
Last login: Sat Nov 21 17:20:01 on ttys000
[oh-my-zsh] plugin 'zsh-nvm' not found
➜  ~ ls
Applications    Documents       Library         Music           Public          get-pip.py      pyvenv
Desktop         Downloads       Movies          Pictures        VirtualBox VMs  opt             wekafiles
➜  ~ cd downloads
➜  downloads git clone https://github.com/jazzymaya/HiddenAgility
Cloning into 'HiddenAgility'...
remote: Enumerating objects: 177, done.
remote: Counting objects: 100% (177/177), done.
remote: Compressing objects: 100% (145/145), done.
remote: Total 191 (delta 57), reused 108 (delta 23), pack-reused 14
Receiving objects: 100% (191/191), 70.85 MiB | 19.23 MiB/s, done.
Resolving deltas: 100% (57/57), done.
➜  downloads ls
HiddenAgility
➜  downloads
```

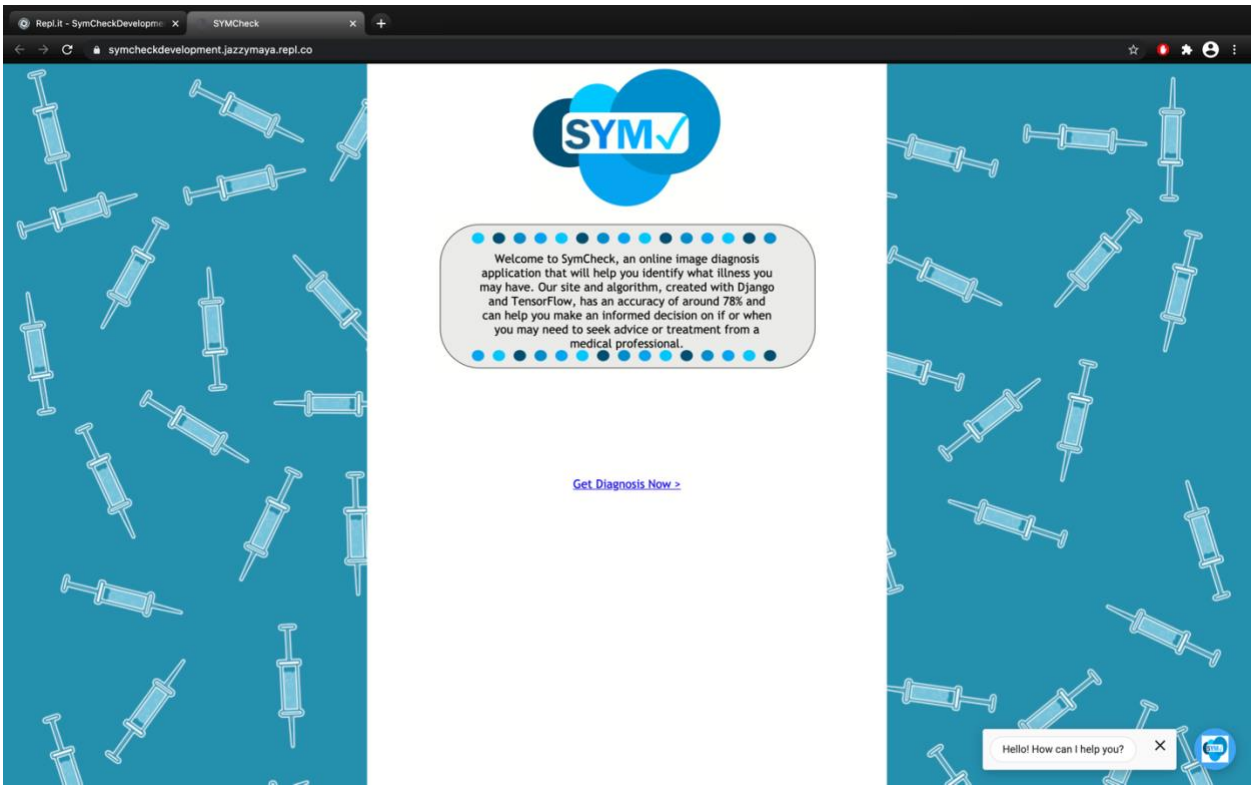
Now with the SymCheck files available on your local machine, we are ready to go forth with running the application!

Step 8: Running SymCheck via Localhost

With your virtual environment activated, navigate to and enter the folder with all of your SymCheck files are saved, and enter the command:

```
(pyvenv) > python manage.py runserver
```

just as you did for the sample project. Then navigate to http://localhost:8000 or http://127.0.0.1:8000 in your browser, and you should see the welcome page of SymCheck as shown on the next page:



Congratulations! You can now use the SymCheck web application on your Windows machine to receive your own image-based diagnosis!

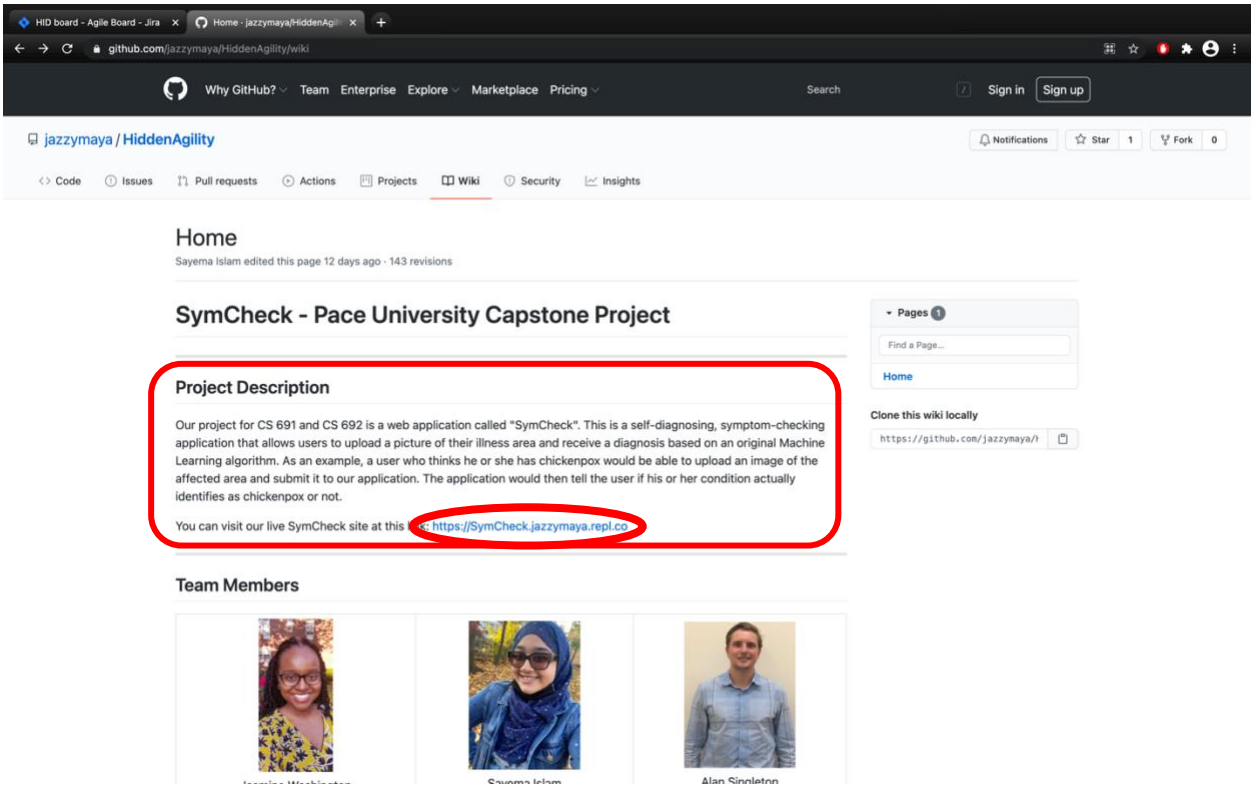
Check out our *SymCheck User Manual* next to learn more about how to navigate the website. And we hope our application serves you well in allowing you to make more informed decisions about your own care.

Accessing SymCheck via Cloud-Hosting – *Coming Soon!*

Updated 03/10/2021

At this point in time, our team is looking to work on hosting SymCheck on the cloud via Amazon Web Services or AWS, so that it can be accessed **without any installation** at: [*SymCheck cloud domain coming soon!!!*] or by simply navigating to the SymCheck wiki page on GitHub at: https://github.com/jazzymaya/HiddenAgility/wiki and clicking a hyperlink to the website under the "Project Description" section.

In the future, a link to the cloud-hosted SymCheck website will be accessible from said section as shown below:



***This screenshot is for sample purposes only, and will be updated to show a working link to a cloud-hosted SymCheck instance in a future sprint