Venter Testing and Deployment Learning

9th August 2019

By- Vedansh Kedia
Backend and Machine Learning Intern
CSE Department, IIT Bombay

Manual Testing

Learned the Testing and the procedure to be followed to perform so that the approach is optimised:

- Gather all the elementary features of the product
- List all the possible unique inputs that can be given
- List all the corresponding expected Output to each input
- Give the inputs to the feature, and observe the output
- Make a Note of Pass if Actual Output= Desired output else Fail

The format of Doc required to perform the above procedure is:

Index/fields	**Feature**	**Inputs Given**	**Expected Output**	**Actual Output**	**Result**
1	List of all elementary features of the product	List of all the possible unique inputs given to the Feature	List of expected output corresponding to each input given	List of Actual Output Received on giving the input to the feature	True : 'Actual Output = Expected Output' Else: Fail

VMware Testing

The basic requirement to test on VMware is to simulate how the product will perform on the Linux machine of the given configuration. This simulation is required because the servers on which the product is deployed are running on Linux.

Pre-requisites for testing on VM are:

- Knowledge of minimum configurations required to run the product.
- Requirement of VMware on the Machine

Steps involved in VMware Deployment Prior to testing are:

- Launch Terminal to perform all the operations over it.
- Pull the code from the GitHub repository to the local branch
- Enter into backend folder since manage.py file exists here.
- Start the Virtual Environment
- Install all the Packages mentioned in the requirement.txt file.
- Create the PostgreSQL Database and the User(in our case it prompted database already existed)
- Configure the settings from the settings.py file required for deployment. (Initially they are as per development)

    - Mention the allowed hosts
    - Check the Static_URL and Static_Root
- Make the Migrations and migrate
- Create the super user
- Collect Static
- Deactivate the virtual Environment
- Create the Systemd Socket and Service files for Gunicorn
- Check for the Gunicorn Socket File
- Testing Socket Activation
- Configure Nginx to Proxy Pass to Gunicorn
- Check the application is running

Now we can perform the Manual Testing as mentioned in the Initial Part of the document.

Tested the all the features including:

- SignUp, Login and Logout
- Going to the dashboard and check whether a file exist or not
- Checked the Upload file feature
- Checked the domain adding feature
- Checked keyword adding feature
- Checked the prediction
- Downloaded the report
- Checked the graph and downloaded
- Checked the Word-Cloud

Azure Deployment

Prerequisites: WinSCM and GitBash shoud be installed

Steps involved in Azure Deployment are:

- Launch GitBash
- Launch and configure the WinSCM
- Login to Azure
- Find the SSH to remote
- SSH to remote Venter on Azure using GitBash(Now we can operate on remote through our system)
- Create a new branch to pull the code
- Pull the Code from GitHub
- Create virtual environment
- Activate the virtual environment
- Install the packages in Requirements.txt
- Check for the database

- Add the static file to remote from local using WinSCM
- Add the MAX.bin file from Local to remote using WinSCM
- Reload the Daemon
- Restart the Gunicorn Server
- Check the status of Gunicorn to be “Active”
- Restart the reverse proxy server Nginx
- Check the status of Nginx to be “Active”
- Check the New Venter website running.

The above mentioned steps are well learned and Understood.

WinSCM is the software used to transfer files to remote system securely. This is needed because we can’t push Heavy files to GitHub.

Gunicorn and Nginx are the servers where Gunicorn is designed to connect with Django and Nginx is used to handle the Static files, i.e., user of application tend interact with Nginx and Gunicorn interact with Django framework.