Service Management – Systemd

If you want to create a service you have to define it in /etc/systemd/system/[filename].service

Example: bash script /usr/bin/project-mercury.sh -> /etc/systemd/system/project-mercury.service

[Unit] #to make sure that our service is started before another service

Description=Python Django for Project Mercury #not mandatory but good practice

Docummentation=http://www………….. #not mandatory but good practice

After=postgresql.service

[Service]

ExecStart(derivative used to run a command/app)= /bin/bash /usr/bin/project-mercury.sh

User=project\_mercury #using service account

Restart=on-failure #when and how to restart the service, another option Restart=always

RestartSec=10 #10 seconds restart interval

[Install] #to allow this service to be enabled during boot

Wantedby graphical.target

To run this service in background: sudo systemctl start project-mercury.service

To verify if the script is running in the background: systemctl status project.mercury.service

In order to modify the service you must first stop it: systemctl stop [service]

For the system to detect any changes done to the service file we have to reload the daemon:

**systemctl daemon-reload** and then we start the service **systemctl start project-mercury.service**

Systemctl is the main command used to manage services on a systemd managed server:

- manage system state

- start/stop/restart/reload

- enable/disable

- list and manage units

- list and update targets

Journalctl tool

- query systemd journal (contents of the systemd logging system called journal)

Commands:

systemctl start/stop [service]

systemctl reload [service] – used to reload the service without interrupting normal functionality

systemctl enable/disable – to enable/disable a service and make it persistent to cross reboot

systemctl status [service] -> States: active-service running/inactive-service stopped/transient states activating or deactivating/failed-crashed, error, timeout etc

systemctl daemon-reload – running this command after making changes to a service unit file reloads the system manager configuration and makes the systemd aware of the changes.

systemctl edit [service].service –-full – edit the service through this command in order to apply the changes made to the unit file immediately without running systemctl daemon-reload

**Systemctl to manage states**

systemctl get-default - see the current runlevel (target) of the system

systemctl set-default [target, ex: multi-user.target] – to change the current target to another

systemctl list-units --all – to list all the units that systemd has loaded or attempted to load (if you use it without --all flag it will print only the active units)

**journalctl** – is used when troubleshooting issues with systemd units as it checks the journal (log entries from all parts of the system) – journalctl used without flags prints all the log entries from the oldest to the newest entries

journalctl –b – prints the entries from the current boot

journalctl –u [unit, ex: [filename].service] – to see a specific unit