Automated Data Ingestion Pipeline & Docker Swarm Setup documentation

for

DiCRA

mistEO Private Limited

Submitted to

UNDP India

02.03.2023

TABLE OF CONTENTS

1. Installation using Docker Framework

1.1. Purpose

This document captures the steps involved in installing Docker and Setup Docker SWARM Cluster to install and configure DiCRA Data Automation Platform components.

The setup would primarily address 4 node production configuration - 2 nodes of Master and 2 nodes of Worker setup. The SWARM cluster is designed to scale up or scale down based on the workload requirements.

2. Installation Prerequisites

2.1. Hardware Requirements

The recommended hardware configuration for Installing DiCRA platform is Linux VMs with the below configurations:

VM Spec (Production) 4 Instances	
RAM	16 GB
vCPU	4 Core
Hard Disk	32 GB (System) + 100 GB (Application) – 2 mount points
OS	Ubuntu Linux

2.2. Software Requirements

For docker based installation of DiCRA components, we need the below software to be installed.

Software & Version	**Description**	**Installation File Information**
Docker version 18.0 or above	For Docker based deployment	Open Source Software

3. Pre-Deployment Steps

3.1. Docker Installation (with Internet Connection)

Before starting the installation of DiCRA components, Docker should be installed in all the nodes where DiCRA components are to be deployed. All the steps in this document should be executed by a user with Root or sudo privileges.

Login to each of the 4 Linux nodes and check if the docker is pre-installed and a supported version exists.

Verification Step - Type the below command in the terminal.

```
docker –version
```

It should output the version of the Docker installed. It should be above 18.0.

```
Docker version 20.10.16, build aa7e414
```

If the supported docker is not installed, then install docker using the steps mentioned in the docker setup instructions given in the below link or Refer Appendix as per the OS image you have selected for these VMs.

https://docs.docker.com/install/linux/docker-ce/ubuntu/#docker-ee-customers/

Note: The Docker Swarm needs to be started in one of the nodes (Designated Manager Node) and then all the other nodes (Designated Worker Nodes or Additional Manager nodes) should join the Swarm cluster before starting the deployment of the DiCRA Components.

3.2. Create Docker Swarm Manager

Login to the Manager Node (VM1) , or select the first VM as the master node in the production VM instances.

In the OS / Linux terminal type the below command: `#Initialize Swarm`

```
docker swarm init --advertise-addr <Manager Node IP Address>
```

Use the *"ifconfig"* command to get the IP address of the manager node. If multiple IP addresses found use an IP address which is accessible from all the other 3 Linux nodes. All 4 Linux nodes of the production server should be able to communicate with each other. Use "`ping <Manager IP>`" in all the 3 nodes to confirm the communication and connectivity.

Verification Step: `#Listing the nodes participating in the swarm`

```
docker node ls
```

4. Join Docker Swarm as WORKER

Tokens are required for the worker to join the swarm. To get the worker joining token, login to the Manager Swarm node (VM1) and in the terminal, type the below command:

```
docker swarm join-token -q worker
```

Then use the token given by the above command in the worker nodes to join the Swarm. Login to the Worker nodes (VM 2 & VM3) and in the terminal, type the below command:

```
docker swarm join --token <<Token from Manager Node>> <<Manager IP>>:2377
```

Verification Step: In the Manager Node (VM1) terminal, type the below command. `#Listing the nodes participating in the swarm`

```
docker node ls
```

5. Join Docker Swarm as MANAGER

Tokens are required for the manager to join the swarm. To get the manager joining token, login to the Manager Swarm node (VM1) and in the terminal, type the below command:

```
docker swarm join-token -q manager
```

Then use the token given by the above command to join the Swarm. Login to the Manager (VM4) node and type the below command in the terminal.

```
docker swarm join --token <<Token from Manager Node>> <<Manager IP>>:2377
```

Verification Step: In the Manager Node (VM1) terminal, type the below command. `# Listing the nodes participating in the swarm`

```
docker node ls
```

The above command should list 4 nodes, VM1 & VM4 would be listed as Manager / Leader nodes and VM2 & VM3 should be listed as worker nodes.

5.1. Docker Swarm Visualiser

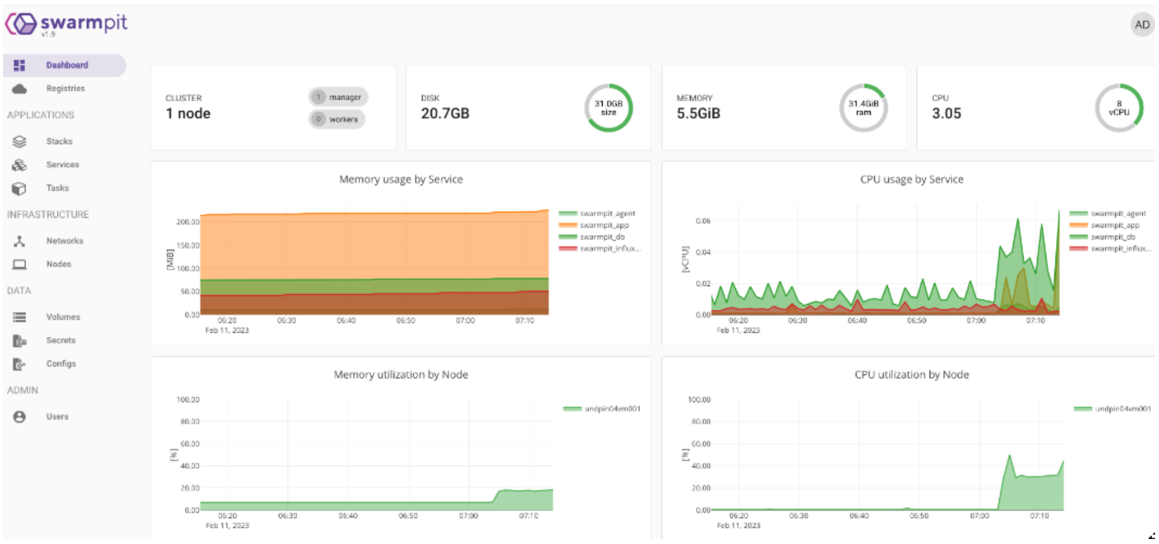
Install a SWARM visualizer to manage the containers with in swarm cluster

```
docker run -it --rm \

--name swarmpit-installer \

--volume /var/run/docker.sock:/var/run/docker.sock \

swarmpit/install:1.9
```

This will install swarmpit visualizer, provide a username and password for swarmpit access during the setup.

6. Verify the swarm visualiser

Access the Swarmpit web console at `http://localhost:888`.Use the admin credentials configured during the setup. Below screenshot is from the single node setup for development and testing.



7. Fail-over Scenarios

7.1. Manager Fail-over Scenario

All the Managers need to have access to the yml files and docker image files, so that they can restart any of the service. Ensure that all the relevant yml files are copied or the shared drive is mounted in both master nodes. Both master nodes should be configured to the same docker registry to pull the relevant container images.

7.2. Worker Node Fail-over scenario

If any of the worker nodes fails, then the manager node will take care of starting additional services in the currently available nodes based on the load distribution among them. No manual intervention is required in this case.

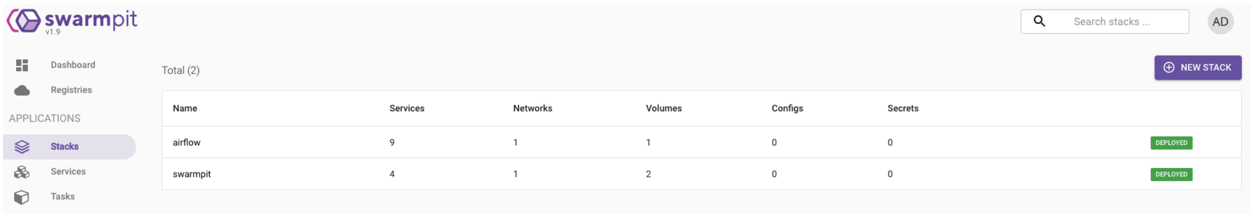
8. Airflow Build and Deployment

8.1. Airflow Services Setup and Configuration

All the required files are in the Github repository under the automation folder, also the same files are placed in the UNDP server /data/airflow.

Note:

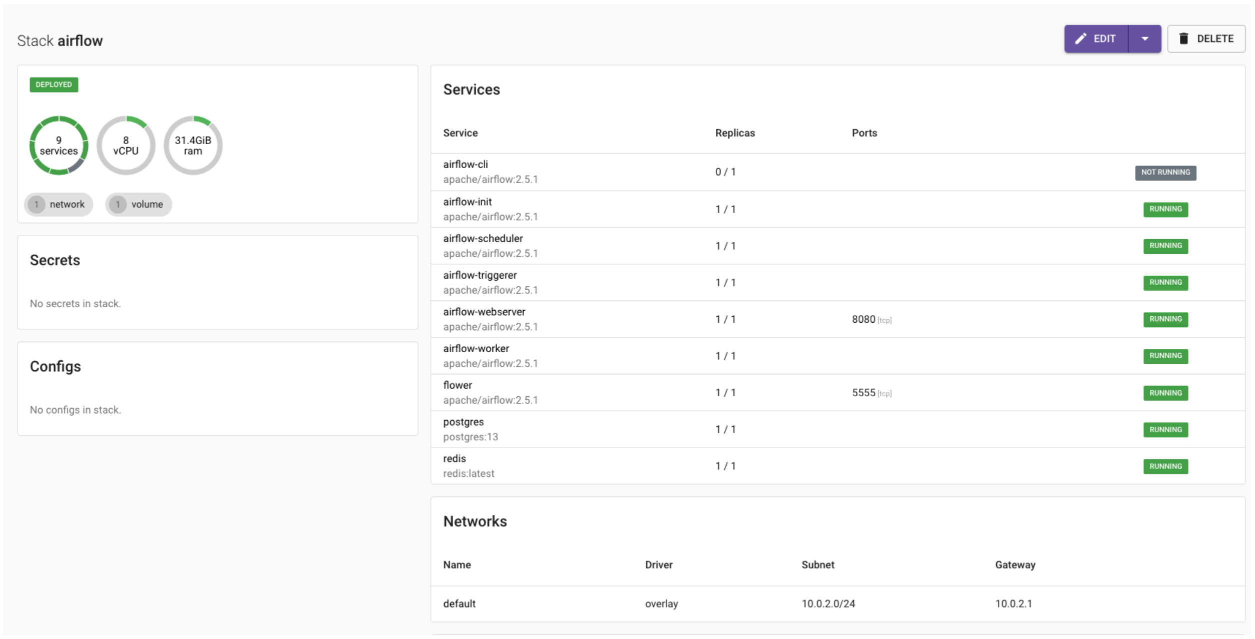
1. Ensure that requirements.txt file is up to date with all the required libraries.
2. Build the docker image required for DiCRA automation by running `build-undp.sh`. This process should be executed in all nodes
3. Create all the required folders for DiCRA automation by running `dsFolderSetup.sh`
4. Deploy the DiCRA automation stack in Docker Swarm by running `deployAirflow.sh`
5. Verify the services using Swarmpit console as shown below:



Total (2)

Name	Services	Networks	Volumes	Configs	Secrets	
airflow	9	1	1	0	0	DEPLOYED
swarmpit	4	1	2	0	0	DEPLOYED

Airflow Stack Deployed



Secrets

No secrets in stack.

Configs

No configs in stack.

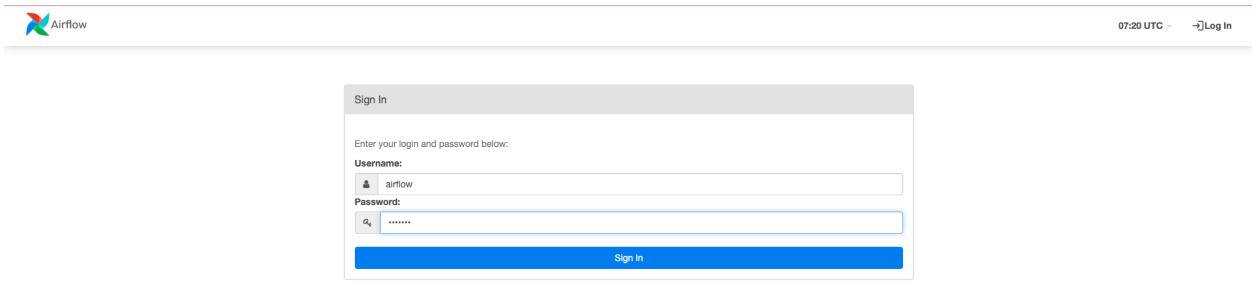
Services

Service	Replicas	Ports	
airflow-cli apache/airflow:2.5.1	0 / 1		NOT RUNNING
airflow-init apache/airflow:2.5.1	1 / 1		RUNNING
airflow-scheduler apache/airflow:2.5.1	1 / 1		RUNNING
airflow-triggerer apache/airflow:2.5.1	1 / 1		RUNNING
airflow-webserver apache/airflow:2.5.1	1 / 1	8080 [tcp]	RUNNING
airflow-worker apache/airflow:2.5.1	1 / 1		RUNNING
flower apache/airflow:2.5.1	1 / 1	5555 [tcp]	RUNNING
postgres postgres:13	1 / 1		RUNNING
redis redis:latest	1 / 1		RUNNING

Networks

Name	Driver	Subnet	Gateway
default	overlay	10.0.2.0/24	10.0.2.1

Airflow Services Running in the Stack



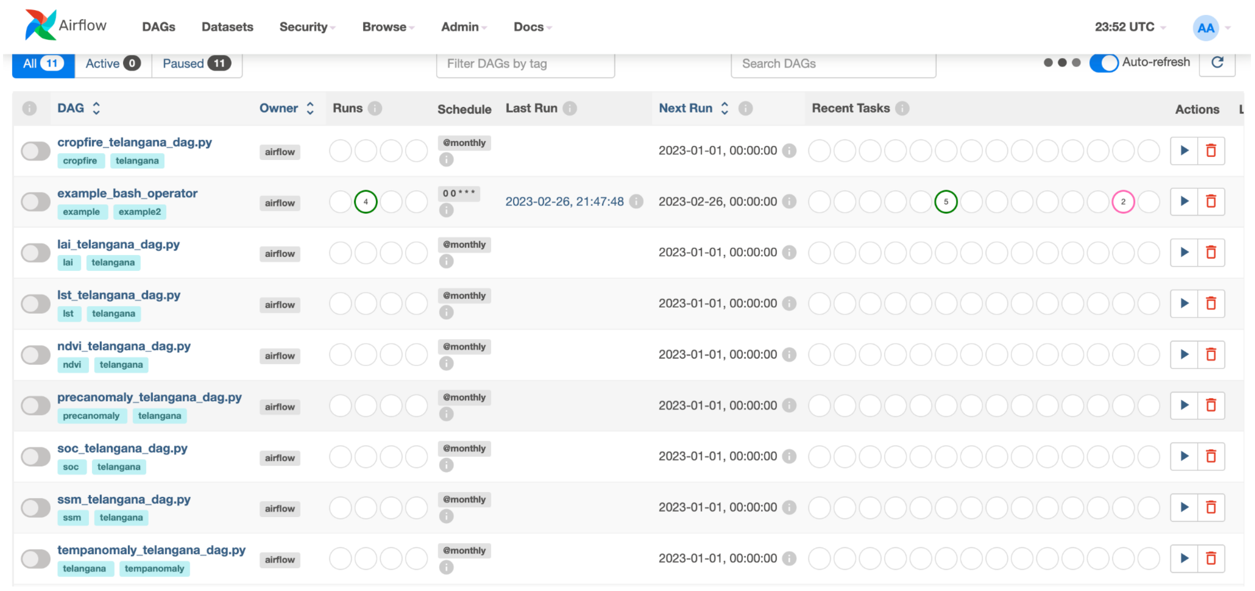
Sign In

Enter your login and password below:

Username: airflow

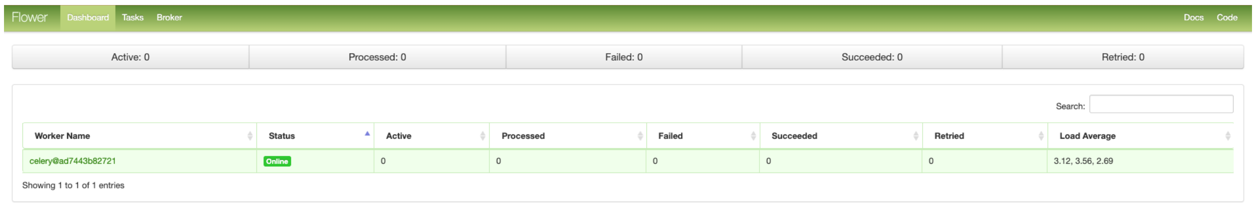
Password: •••••••

Login to Airflow Web Console



DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions
cropfire_telangana_dag.py (cropfire, telangana)	airflow		@monthly		2023-01-01, 00:00:00		
example_bash_operator (example, example2)	airflow	4	0 0 * * *	2023-02-26, 21:47:48	2023-02-26, 00:00:00	5, 2	
lai_telangana_dag.py (lai, telangana)	airflow		@monthly		2023-01-01, 00:00:00		
lst_telangana_dag.py (lst, telangana)	airflow		@monthly		2023-01-01, 00:00:00		
ndvi_telangana_dag.py (ndvi, telangana)	airflow		@monthly		2023-01-01, 00:00:00		
precanomaly_telangana_dag.py (precanomaly, telangana)	airflow		@monthly		2023-01-01, 00:00:00		
soc_telangana_dag.py (soc, telangana)	airflow		@monthly		2023-01-01, 00:00:00		
ssm_telangana_dag.py (ssm, telangana)	airflow		@monthly		2023-01-01, 00:00:00		
tempanomaly_telangana_dag.py (telangana, tempanomaly)	airflow		@monthly		2023-01-01, 00:00:00		

Verify Airflow Dicra DAGs



Active: 0 | Processed: 0 | Failed: 0 | Succeeded: 0 | Retried: 0

Worker Name	Status	Active	Processed	Failed	Succeeded	Retried	Load Average
celery@ad7443b82721	Online	0	0	0	0	0	3.12, 3.56, 2.69

Showing 1 to 1 of 1 entries

Airflow Cluster Monitor

The above setup of Airflow is using Docker SWARM technology and Celery Executor services. This supports scaling the services horizontally and vertically.

9. DiCRA Dataset and Folder Naming Convention

Folders

Primary Data Folders -> `<DSTYPE>/State/Stage`

Date Dependant Folder -> `<DSTYPE>/State/Stage/FileType/YYYYMMDD`

Publish Folders

`<DSTYPE>/State/Stage/FileType/`

Recommended Published Filename pattern

- `<DSType>_<StateCode>_<SpatialResolution>_YYYYMMDD_Random_<Stage>.<file_type>`
- Eg: NDVI_State_

Metadata by DataFile Type(JSON) should be maintained

User Generated / DataScientist manual data folder

```
UGC(DataScience)/<DSTYPE>/State/Stage/FileType
```

10. DiCRA Dataset Types

Dataset Type Code	**Description**
ndvi	NDVI
soc	SOC
cropfire	Crop Fire
tempanomaly	Temperature anomaly
precanomaly	Precipitation Anomaly
ssm	SSM
lst	LST
lai	LAI

11. DiCRA Dataset Types

State Code	**State Name**
TS	Telangana

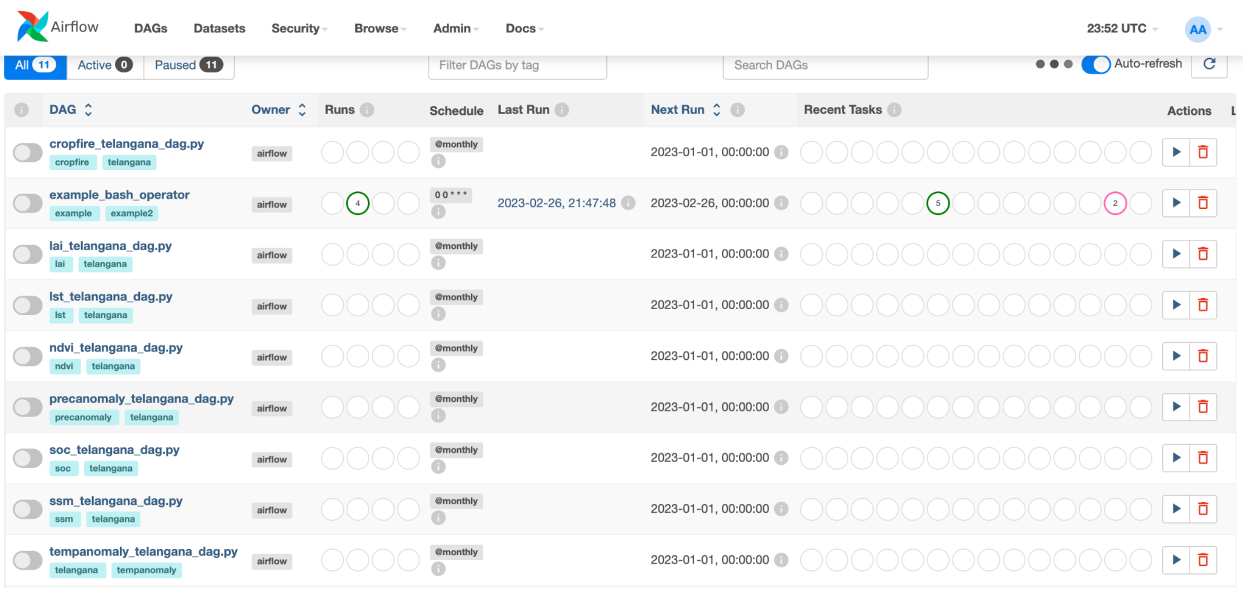
GJ	Gujarat
OD	Odisha
JH	Jharkhand
KL	Kerala
UP	Uttar Pradesh
MH	Maharashtra
UK	Uttarakhand

12. Configured Folder in DiCRA

Base Folder : /data/airflow/nfsdata

DataSet	**State**	**Subfolder**	**Description**
ndvi	Telangana	base	Folder to store reference or config files
		download	Folder to download the ndvi files from Google Earth Engine / Other sources
		process	Folder to store the processed files
		publish	Folder to store the published files
		publish/archive	Folder to store the archived files when a new file is published
		dppd	Folder to store the DPPD output

13. Automation pipelines in DiCRA Server



DAG	Owner	Runs	Schedule	Last Run	Next Run	Recent Tasks	Actions
cropfire_telangana_dag.py cropfire telangana	airflow		@monthly		2023-01-01, 00:00:00		
example_bash_operator example example2	airflow	4	0 0 * * *	2023-02-26, 21:47:48	2023-02-26, 00:00:00	5 2	
lai_telangana_dag.py lai telangana	airflow		@monthly		2023-01-01, 00:00:00		
lst_telangana_dag.py lst telangana	airflow		@monthly		2023-01-01, 00:00:00		
ndvi_telangana_dag.py ndvi telangana	airflow		@monthly		2023-01-01, 00:00:00		
precanomaly_telangana_dag.py precanomaly telangana	airflow		@monthly		2023-01-01, 00:00:00		
soc_telangana_dag.py soc telangana	airflow		@monthly		2023-01-01, 00:00:00		
ssm_telangana_dag.py ssm telangana	airflow		@monthly		2023-01-01, 00:00:00		
tempanomaly_telangana_dag.py telangana tempanomaly	airflow		@monthly		2023-01-01, 00:00:00		

14. DiCRA data ingestion programs - 1

The files are placed in a common scripts folder for the respective data scientist or data analyst to update. These are standard python scripts, but these scripts have to follow common standards and compatible python modules.

Ingestion Python File	File Description
ndvi-telangana-download.py	NDVI download script for the state Telangana
ndvi-telangana-process.py	NDVI processing script for the state Telangana
soc-telangana-download.py	SOC download script for the state Telangana
soc-telangana-process.py	SOC processing script for the state Telangana
cropfire-telangana-download.py	Crop Fire download script for the state Telangana
cropfire-telangana-process.py	Crop Fire processing script for the state Telangana
tempanomaly-telangana-download.py	Temperature Anomaly download script for the state Telangana

tempanomaly-telangana-process.py	Temperature Anomaly processing script for the state Telangana
tempanomaly-telangana-deviance.py	Temperature Anomaly Deviance processing script for the state Telangana
precanomaly-telangana-download.py	Precipitation download script for the state Telangana
precanomaly-telangana-process.py	Precipitation processing script for the state Telangana
precanomaly-telangana-deviance.py	Precipitation Anomaly Deviance processing script for the state Telangana

15. DiCRA data ingestion programs - 2

The files are placed in a common scripts folder for the respective data scientist or data analyst to update. These are standard python scripts, but these scripts have to follow common standards and compatible python modules.

Ingestion Python File	**File Description**
ssm-telangana-download.py	SSM download script for the state Telangana
ssm-telangana-process.py	SSM processing script for the state Telangana
lst-telangana-download.py	LST download script for the state Telangana
lst-telangana-process.py	LST processing script for the state Telangana
lai-telangana-download.py	LAI download script for the state Telangana
lai-telangana-process.py	LAI processing script for the state Telangana

- All users must use the same requirements.txt file for their python environment initialization
- Any new library used by the user should be included in the requirements.txt and it should be uploaded to the git repository for other developers to re-use and deploy to server
- All modules included in the requirements.txt must have a specific version number or min /max version number.
- The python version to be used for this is Python 3.x (Recommended 3.7)
- A governance process on the repository to ensure that all the developers are using the recommended version and supported libraries will be deployed in due course.