Microsoft Research
Summit 2022

FarmVibes.AI OSS Overview & Python Client

Bruno Silva <Microsoft Research>
Renato Luiz de Freitas Cunha <Microsoft Research>

FarmVibes.AI: requirements

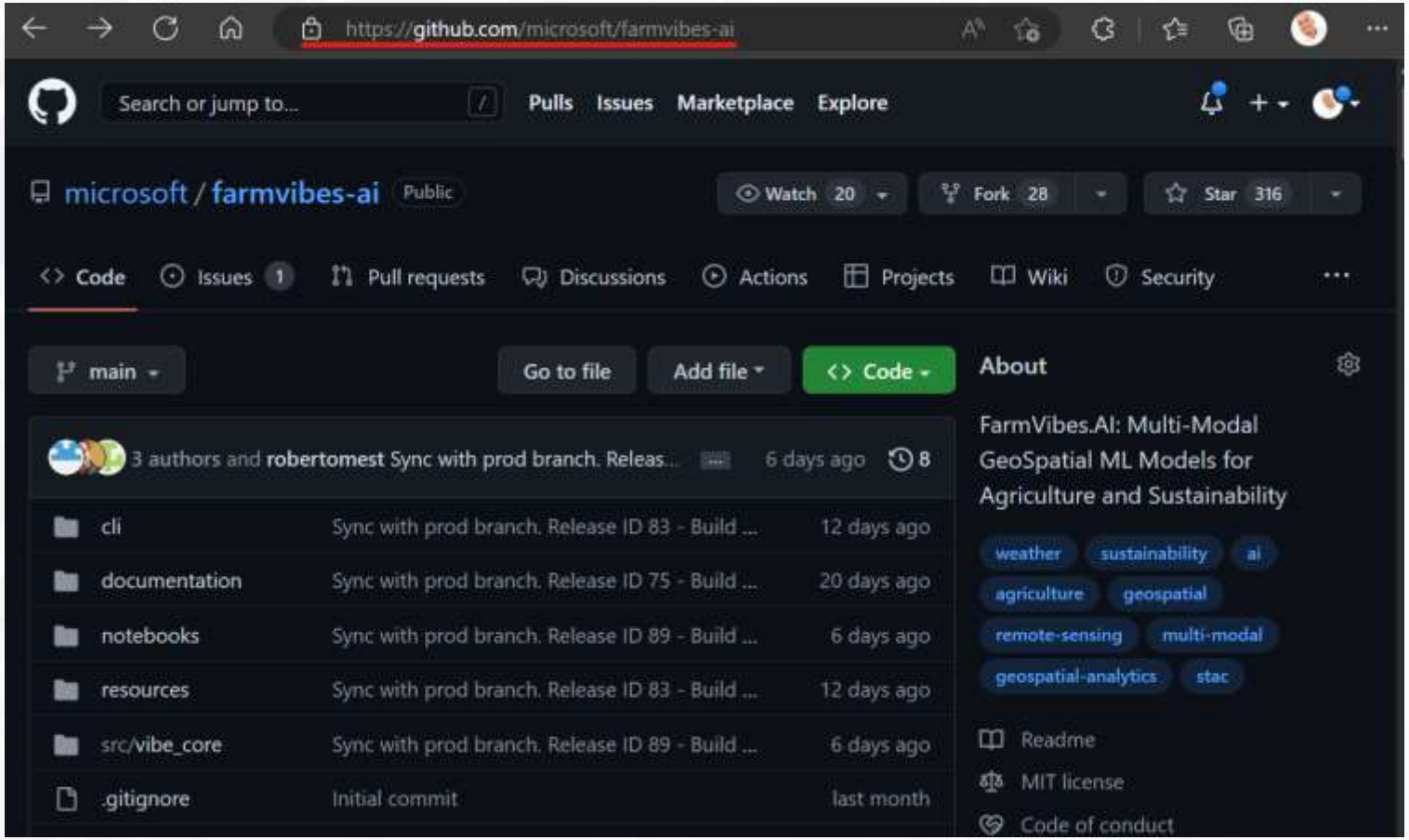
Hardware

- A Linux machine (Ubuntu 20.04+)
- Minimum 16GB of memory (32GB+ is recommended)
- 4+ CPU cores
- 512GB of storage (2 TB+, recommended)

Software

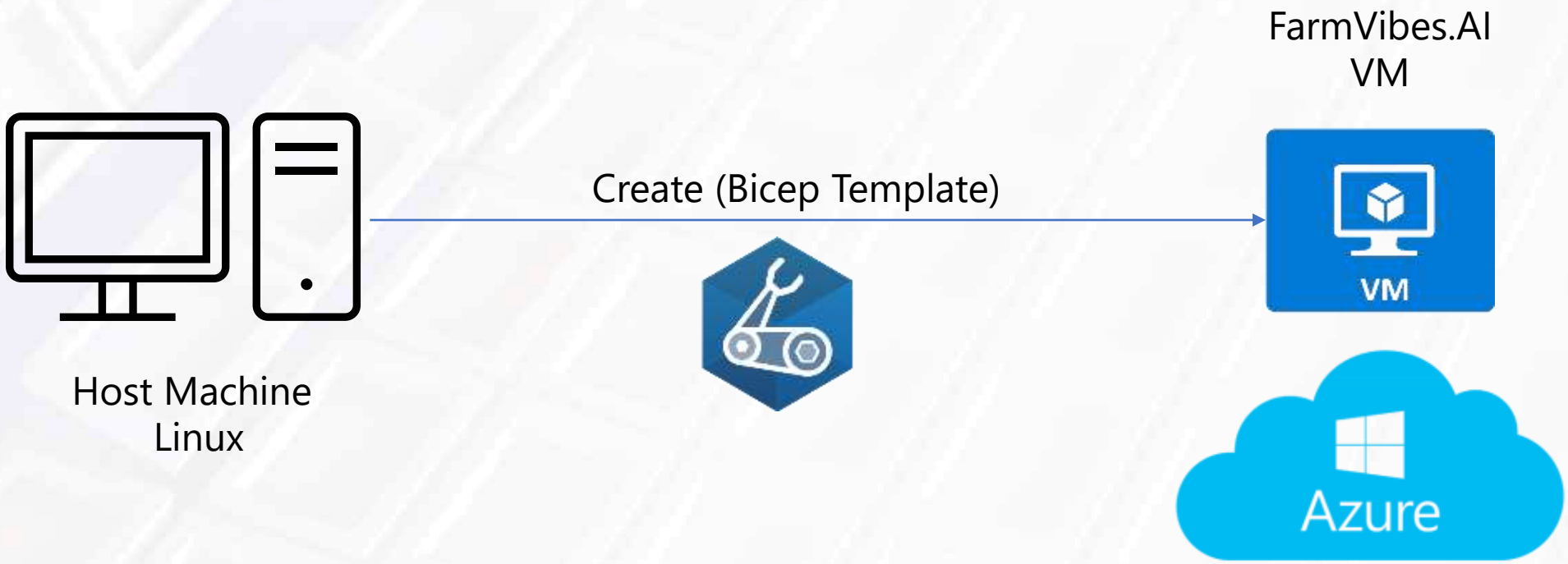
- Git to download the repository
- Docker. You should be able to run it without `sudo`.
- Curl. To download additional FarmVibes.AI required packages
- Python 3.8+. FarmVibes.AI provides a python client.

FarmVibes.AI Project



https://github.com/microsoft/farmvibes-ai

FarmVibes.AI Installation



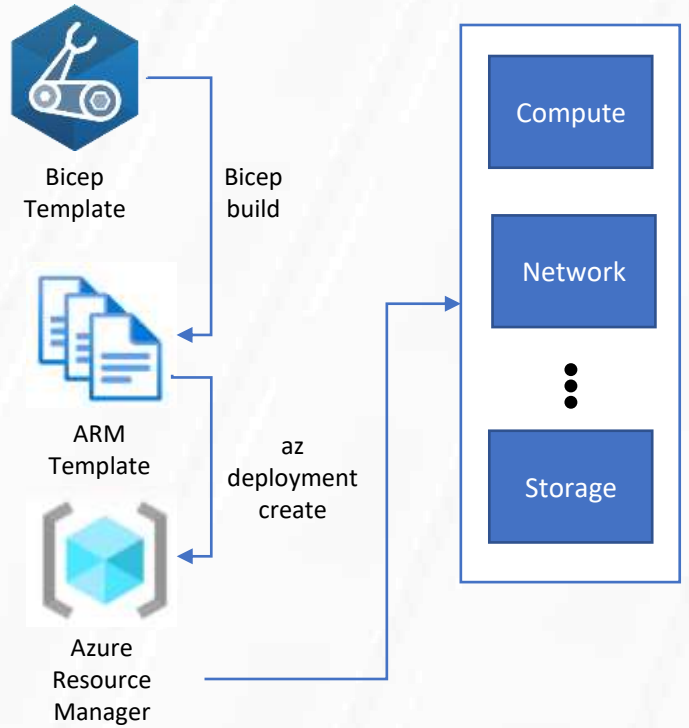
Microsoft

Microsoft Research
Summit 2022

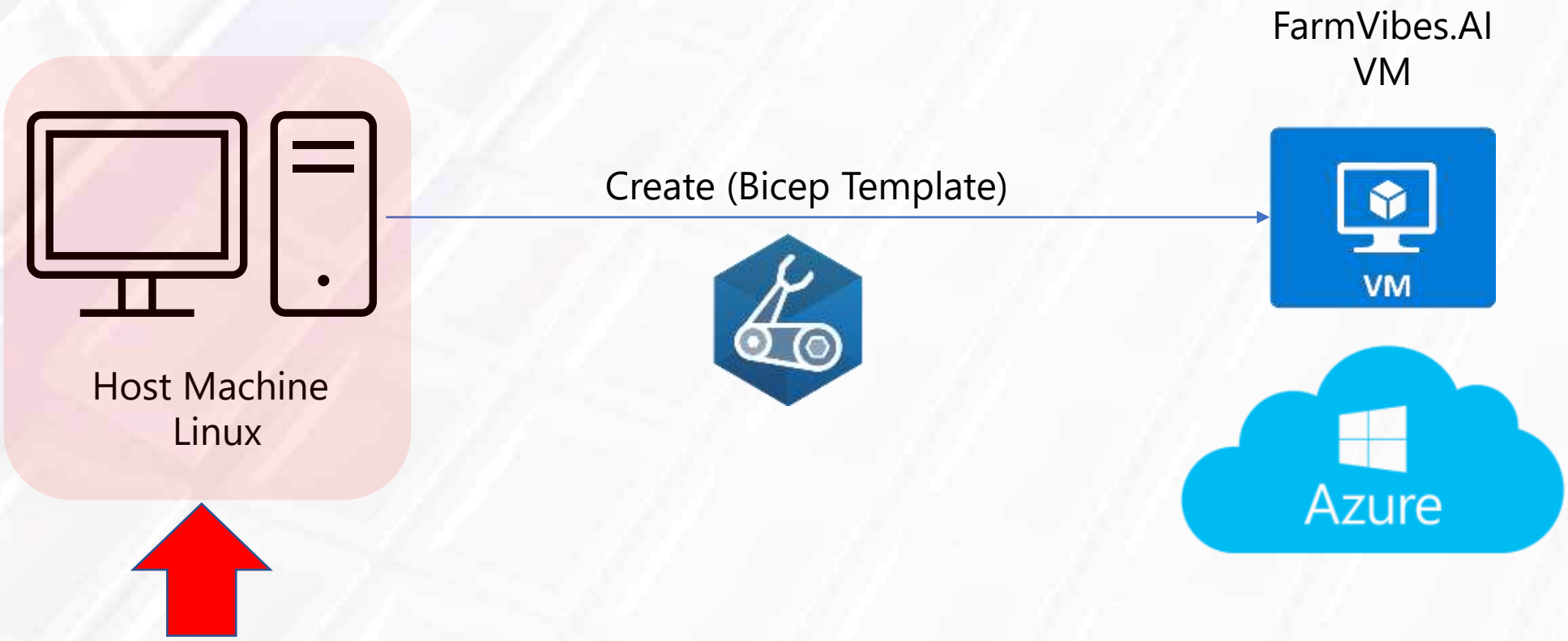
Deploy an Azure FarmVibes.AI VM

FarmVibes.AI VM using Azure Bicep

- Bicep is a declarative domain-specific language to deploy Azure resources.
- Support for all azure resources
- Simple syntax compared to ARM JSON templates
- Repeatable results. Idempotent behavior.
- Preview changes. You can perform what-if analysis before deploying resources.

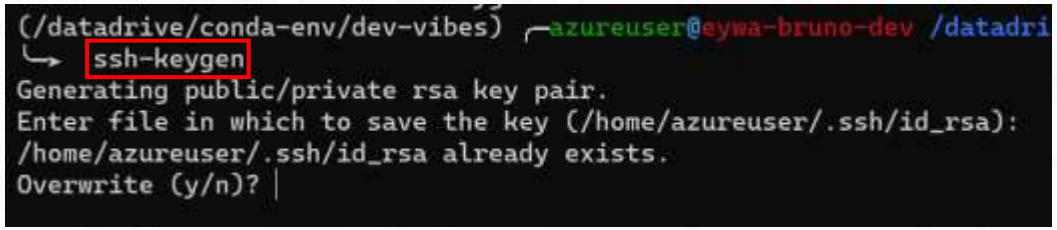


FarmVibes.AI Installation



FarmVibes Azure VM Creation requirements

- Linux computer or WSL on Windows
- Azure CLI installed and Azure Subscription
- A resource group in which you are a contributor
- You'll also need a SSH public key (ideally in `~/.ssh/id_rsa.pub`)



VM Creation Steps

1. Clone the repo https://github.com/microsoft/farmvibes-ai

```
azureuser@eywa-bruno-dev /datadrive/demo
  git clone https://github.com/microsoft/farmvibes-ai.git
Cloning into 'farmvibes-ai'...
remote: Enumerating objects: 199, done.
remote: Counting objects: 100% (199/199), done.
remote: Compressing objects: 100% (143/143), done.
remote: Total 199 (delta 66), reused 176 (delta 51), pack-reused 0
Receiving objects: 100% (199/199), 180.63 KiB | 30.10 MiB/s, done.
Resolving deltas: 100% (66/66), done.
```

2. Log into you azure account and follow the suggested steps

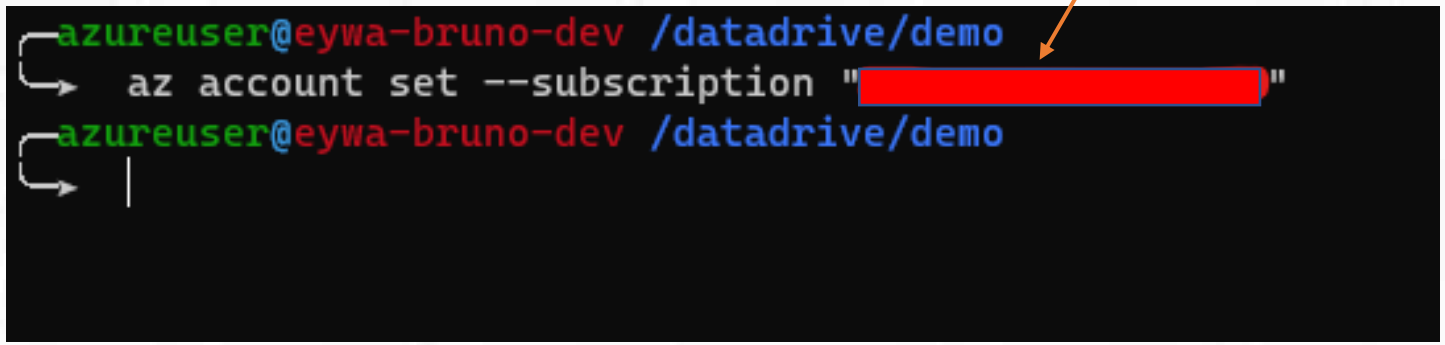
```
azureuser@eywa-bruno-dev /datadrive/demo
  az login                                                    130
To sign in, use a web browser to open the page https://microsoft.com/d
evicelogin and enter the code [redacted] to authenticate.
```

A device code will be generated for you.

VM Creation Steps

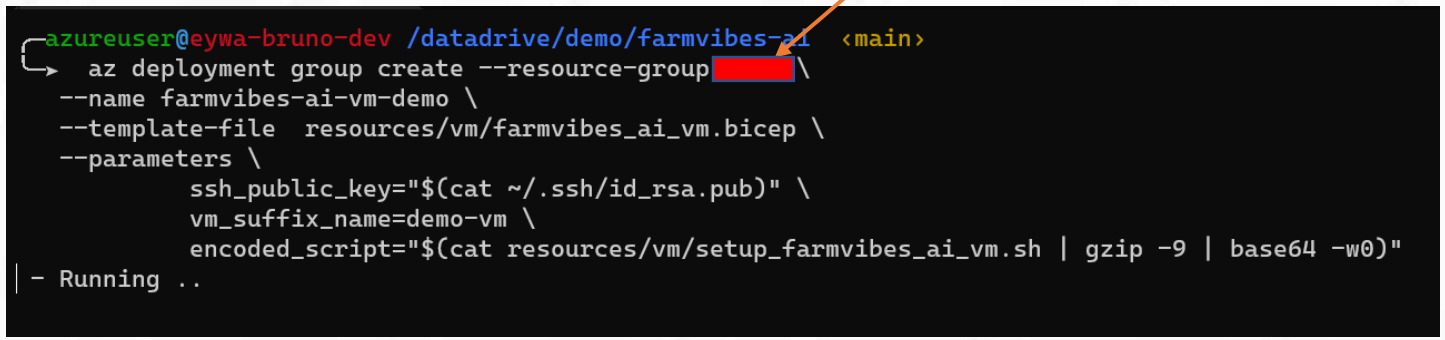
3. Select your subscription account.

Use your subscription name



4. Create the deployment

Use your resource group name



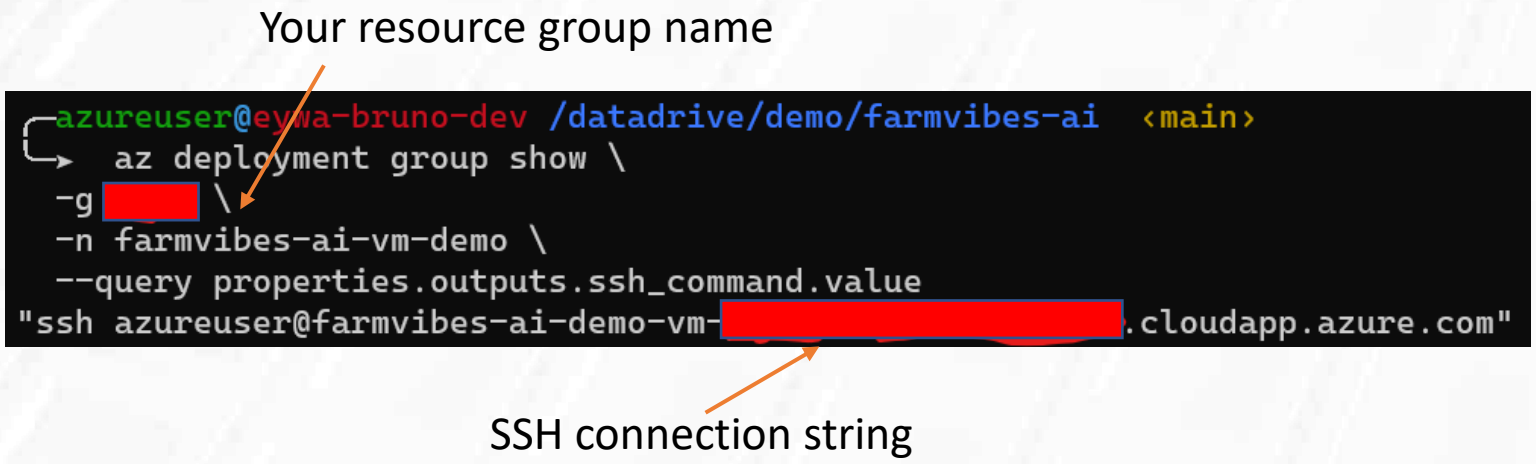
VM Creation Steps

```
azureuser@eywa-bruno-dev /datadrive/demo/farmvibes-ai  <main>
  az deployment group create --resource-group [redacted]
  --name farmvibes-ai-vm-demo \
  --template-file  resources/vm/farmvibes_ai_vm.bicep \
  --parameters \
          ssh_public_key="$(cat ~/.ssh/id_rsa.pub)" \
          vm_suffix_name=demo-vm \
          encoded_script="$(cat resources/vm/setup_farmvibes_ai_vm.sh | gzip -9 | base64 -w0)"
- Running ..
```

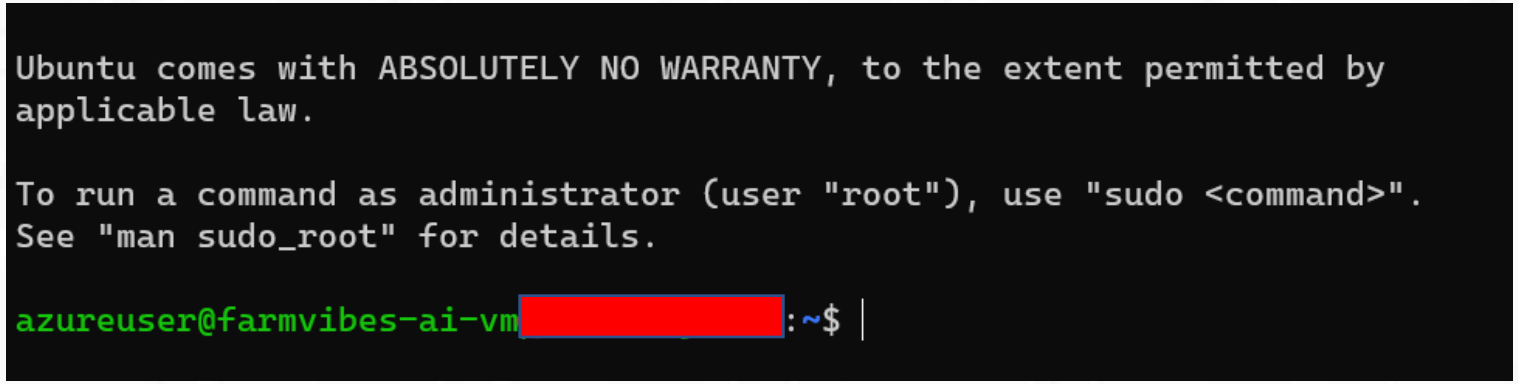
farmvibes-ai/VM-SETUP.md at main · microsoft/farmvibes-ai (github.com)

VM Creation Steps

5. A JSON describing the deployment is generated and you can check the output as follows.



6.Use the SSH connection string to log into the VM.



Example

```
└─➤  az deployment group create --resource-group eywa \
  --name farmvibes-ai-demo-vm-params \
  --template-file  resources/vm/farmvibes_ai_vm.bicep \
  --parameters \
          ssh_public_key="$(cat ~/.ssh/id_rsa.pub)" \
          vm_suffix_name=farmvibes-demo-vm-params \
          encoded_script="$(cat resources/vm/setup_farmvibes_ai_vm.sh | gzip -9 | base64 -w0)" \
          vm_size=Standard_DS4_v2 \
          location=eastus
| | Running ..
```

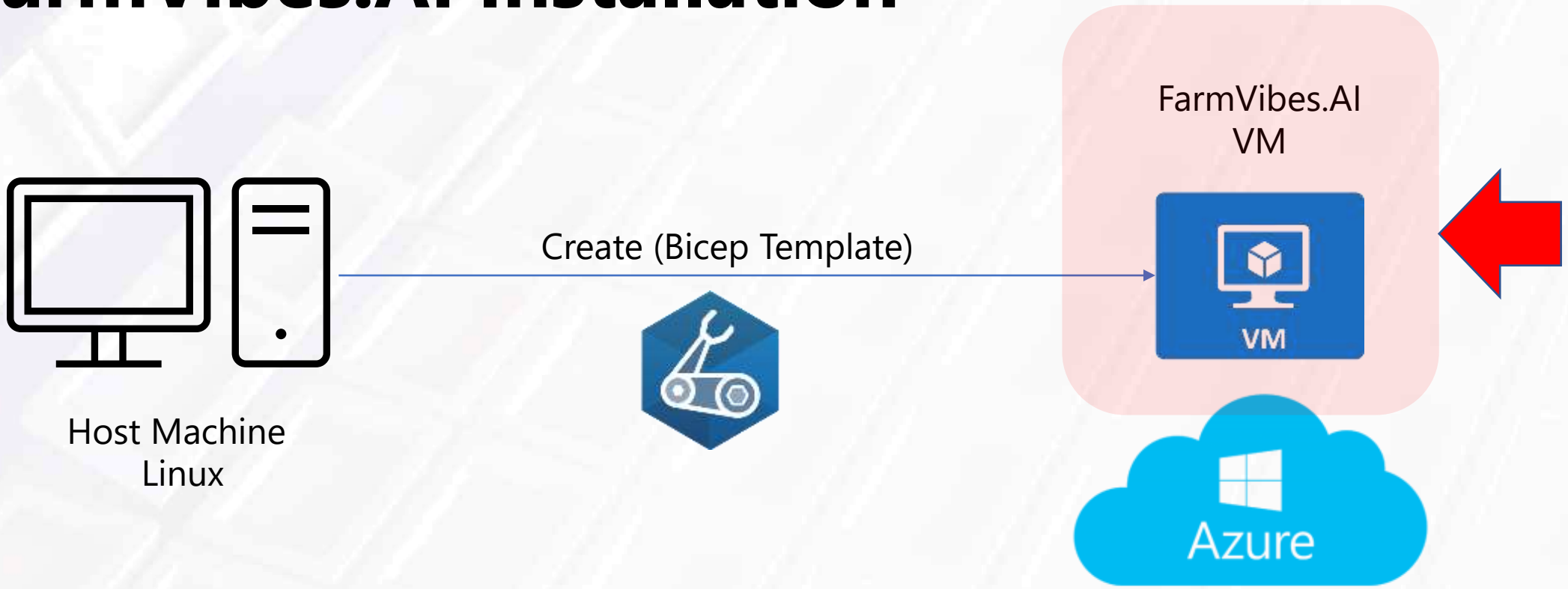
You can customize your VM by setting different parameters. Check the resources/vm/farmvibes_ai_vm.bicep file to see additional parameters.

Microsoft

Microsoft Research
Summit 2022

Installing FarmVibes.AI

FarmVibes.AI Installation



Install FarmVibes.AI – Clone the Project

- Once you connect to the VM, clone the project and enter the project folder.

```
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~$ git clone https://github.com/microsoft/farmvibes-ai.git
Cloning into 'farmvibes-ai'...
remote: Enumerating objects: 213, done.
remote: Counting objects: 100% (213/213), done.
remote: Compressing objects: 100% (147/147), done.
remote: Total 213 (delta 76), reused 190 (delta 60), pack-reused 0
Receiving objects: 100% (213/213), 953.77 KiB | 34.06 MiB/s, done.
Resolving deltas: 100% (76/76), done.
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~$ cd farmvibes-ai/
```

Install FarmVibes.AI – Pre-requisites

- Once the VM is deployed. You will have the pre-requisites to install FarmVibes.AI

- (Optional) If you do not want to use the FarmVibes.AI VM, you can use the following command to install the pre-requisites

```
bash ./resources/vm/setup_farmvibes_ai_vm.sh
```

- Make sure you have sudo permission on your computer and an Ubuntu installation

Install FarmVibes.AI – Pre-requisites check

```
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ curl www.microsoft.com > /dev/null
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1020  100  1020    0     0  31875      0 --:--:-- --:--:-- --:--:-- 31875
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ python -c "print('Hello FarmVibes.AI users😊')"
Hello FarmVibes.AI users😊
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

FarmVibes.AI management script

```
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ bash farmvibes-ai.sh -h
usage: farmvibes-ai.sh [-h] update-images|start|status|stop|destroy
       farmvibes-ai.sh [-h] add-secret <key> <value>
       farmvibes-ai.sh [-h] delete-secret <key>
       farmvibes-ai.sh [-h] setup
       farmvibes-ai.sh [-h] add-onnx <model-file>

Manages a Project FarmVibes.AI cluster in the local machine by configuring
a kubernetes cluster on top of docker.

Supported commands are:

    add-onnx       Adds onnx model to the FarmVibes.AI cluster
    add-secret     Adds a secret to an existing FarmVibes.AI cluster
    delete-secret  Deletes a secret in the FarmVibes.AI cluster
    destroy        Destroys the current FarmVibes.AI cluster
    restart        Restarts the services in the FarmVibes.AI cluster
    setup          Sets up a new cluster
    start          Starts the FarmVibes.AI cluster
    status         Shows the status of the current FarmVibes.AI cluster
    stop           Stops the FarmVibes.AI cluster
    update-images  Updates the services running in a current, existing cluster
```

Install FarmVibes.AI Cluster

```
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ bash farmvibes-ai.sh setup
Using /home/azureuser/.cache/farmvibes-ai as storage path.
Installing dapr CLI at /home/azureuser/.config/farmvibes-ai/dapr...
Installing minikube at /home/azureuser/.config/farmvibes-ai/minikube...
Installing kubectl at /home/azureuser/.config/farmvibes-ai/kubectl...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 42.9M  100 42.9M    0     0   175M      0 --:--:-- --:--:-- --:--:--  175M
Installing helm at /home/azureuser/.config/farmvibes-ai/helm...
```

```
Success!

FarmVibes.AI REST API is running at http://192.168.49.2:30000
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$
```

Check FarmVibes.AI Cluster Status

```
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ bash farmvibes-ai.sh status
|--------------|-----------|---------|--------------|------|---------|---------|-------|--------|
|   Profile    | VM Driver | Runtime |      IP      | Port | Version | Status  | Nodes | Active |
|--------------|-----------|---------|--------------|------|---------|---------|-------|--------|
| farmvibes-ai | docker    | docker  | 192.168.49.2 | 8443 | v1.24.3 | Running |     1 |        |
|--------------|-----------|---------|--------------|------|---------|---------|-------|--------|
FarmVibes.AI REST API is running at http://192.168.49.2:30000
```

Check FarmVibes.AI Installation

Install vibe_core

```
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ pip install ./src/vibe_core
Processing ./src/vibe_core
Collecting hydra-zen~=0.7
  Downloading hydra_zen-0.8.0-py3-none-any.whl (67 kB)
     |████████████████████████████████| 67 kB 1.3 MB/s
Collecting jsonschema~=4.6
  Downloading jsonschema-4.17.0-py3-none-any.whl (83 kB)
     |████████████████████████████████| 83 kB 492 kB/s
Collecting pydantic~=1.8.2
  Downloading pydantic-1.8.2-cp38-cp38-manylinux2014_x86_64.whl (13.7 MB)
     |████████████████████████████████| 13.7 MB 68.5 MB/s
Collecting pystac~=1.6.0
  Downloading pystac-1.6.1-py3-none-any.whl (146 kB)
     |████████████████████████████████| 146 kB 82.9 MB/s
Collecting requests>=2.27
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
     |████████████████████████████████| 62 kB 770 kB/s
```

Check FarmVibes.AI Installation

Run hello world workflow!

```
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$ python -m vibe_core.farmvibes_ai_hello_world
INFO:__main__:Successfully obtained a FarmVibes.AI client (addr=http://192.168.49.2:30000)
INFO:__main__:available workflows: ['helloworld', 'farm_ai/land_degradation/landsat_ndvi_trend', 'farm_ai/land_degradation/ndvi_linea
r_trend', 'farm_ai/agriculture/ndvi_summary', 'farm_ai/agriculture/methane_index', 'farm_ai/agriculture/change_detection', 'farm_ai/a
griculture/weed_detection', 'farm_ai/agriculture/emergence_summary', 'farm_ai/agriculture/canopy_cover', 'farm_ai/land_cover_mapping/
conservation_practices', 'farm_ai/carbon_local/carbon_whatif', 'ml/driveway_detection', 'ml/crop_segmentation', 'ml/dataset_generatio
n/datagen_crop_segmentation', 'data_processing/merge/match_merge_to_ref', 'data_processing/index/index', 'data_processing/timeseries/
timeseries_aggregation', 'data_processing/timeseries/timeseries_masked_aggregation', 'data_processing/threshold/threshold_raster', 'd
ata_processing/linear_trend/chunked_linear_trend', 'data_processing/gradient/raster_gradient', 'data_processing/clip/clip', 'data_pro
cessing/outlier/detect_outlier', 'data_ingestion/osm_road_geometries', 'data_ingestion/soil/usda', 'data_ingestion/soil/soilgrids', '
data_ingestion/user_data/ingest_raster', 'data_ingestion/user_data/ingest_geometry', 'data_ingestion/cdl/download_cdl', 'data_ingesti
on/airbus/airbus_download', 'data_ingestion/airbus/airbus_price', 'data_ingestion/weather/download_era5', 'data_ingestion/weather/dow
nload_chirps', 'data_ingestion/weather/get_ambient_weather', 'data_ingestion/weather/get_forecast', 'data_ingestion/naip/download_nai
p', 'data_ingestion/sentinel1/preprocess_s1', 'data_ingestion/landsat/preprocess_landsat', 'data_ingestion/sentinel2/cloud_ensemble',
 'data_ingestion/sentinel2/improve_cloud_mask', 'data_ingestion/sentinel2/preprocess_s2_improved_masks', 'data_ingestion/sentinel2/pr
eprocess_s2', 'data_ingestion/sentinel2/preprocess_s2_ensemble_masks', 'data_ingestion/sentinel2/improve_cloud_mask_ensemble', 'data_
ingestion/spaceeye/spaceeye_preprocess', 'data_ingestion/spaceeye/spaceeye', 'data_ingestion/spaceeye/spaceeye_preprocess_ensemble',
'data_ingestion/spaceeye/spaceeye_interpolation', 'data_ingestion/dem/download_dem']
INFO:__main__:Running helloworld workflow...
INFO:__main__:Successfully executed helloworld workflow. Result 'VibeWorkflowRun'(id='22776bdb-0a08-48d0-b0ad-ba171c196c7a', name='te
st_hello', workflow='helloworld', status='done')
azureuser@farmvibes-ai-vmphdbdlb5ye3b2:~/farmvibes-ai$
```

Status Done.

Microsoft

Microsoft Research
Summit 2022

FarmVibes.AI Python Client, REST API & Workflows

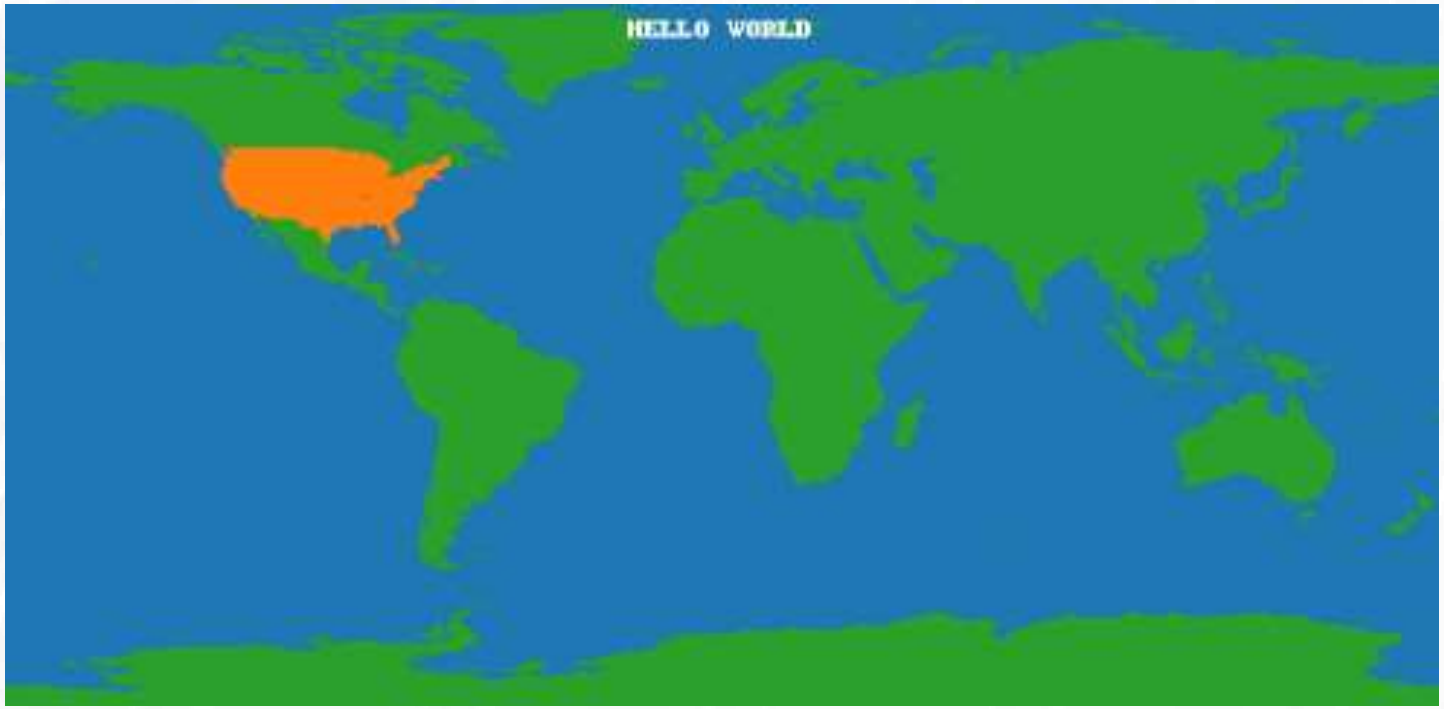
Bruno Silva <Microsoft Research>
Renato Luiz de Freitas Cunha <Microsoft Research>

Checking the sanity of the installation

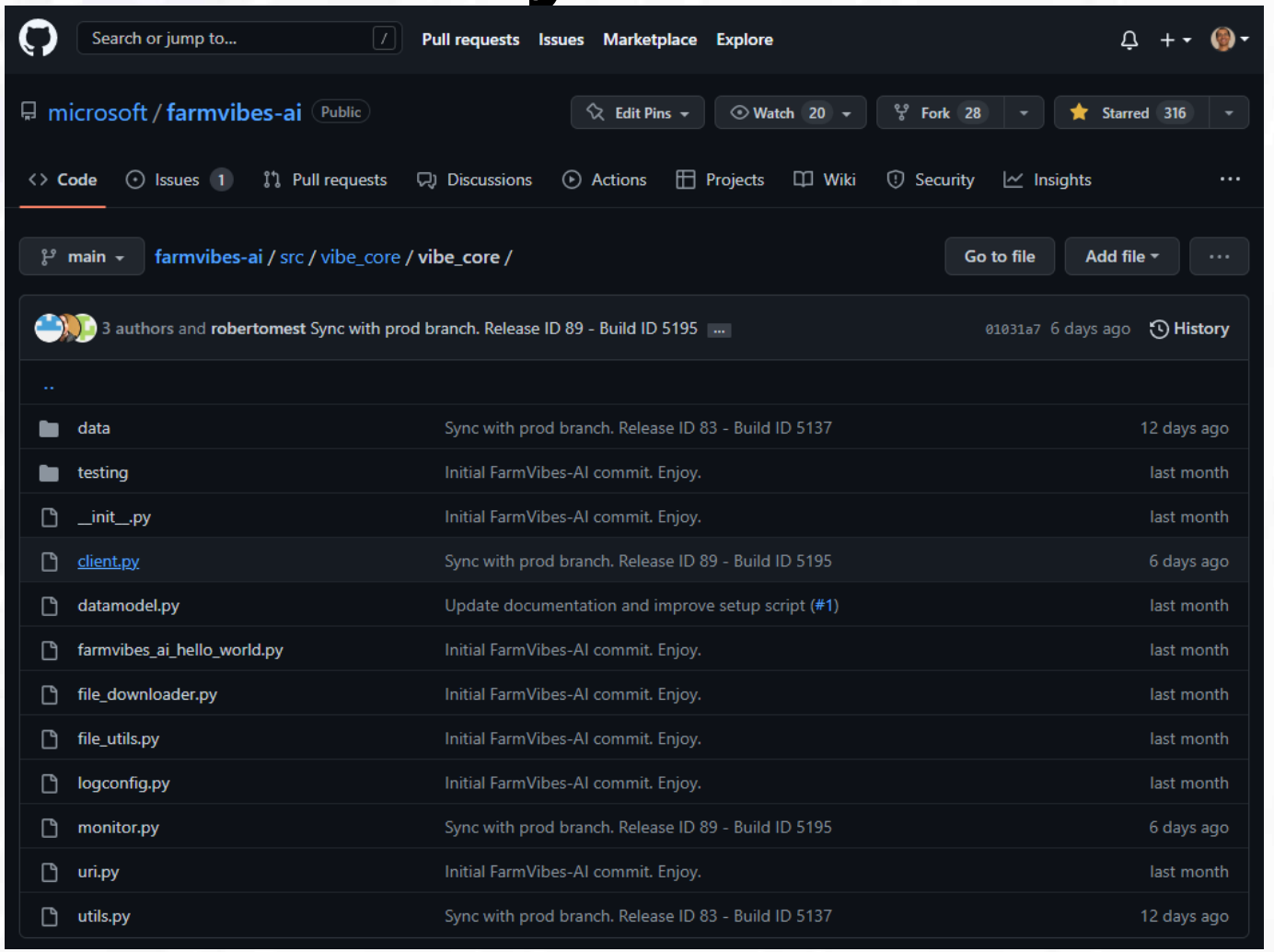
Running the hello world workflow

```
python -m vibe_core.farmvibes_ai_hello_world
```

Hello World Output



The FarmVibes.AI Python client

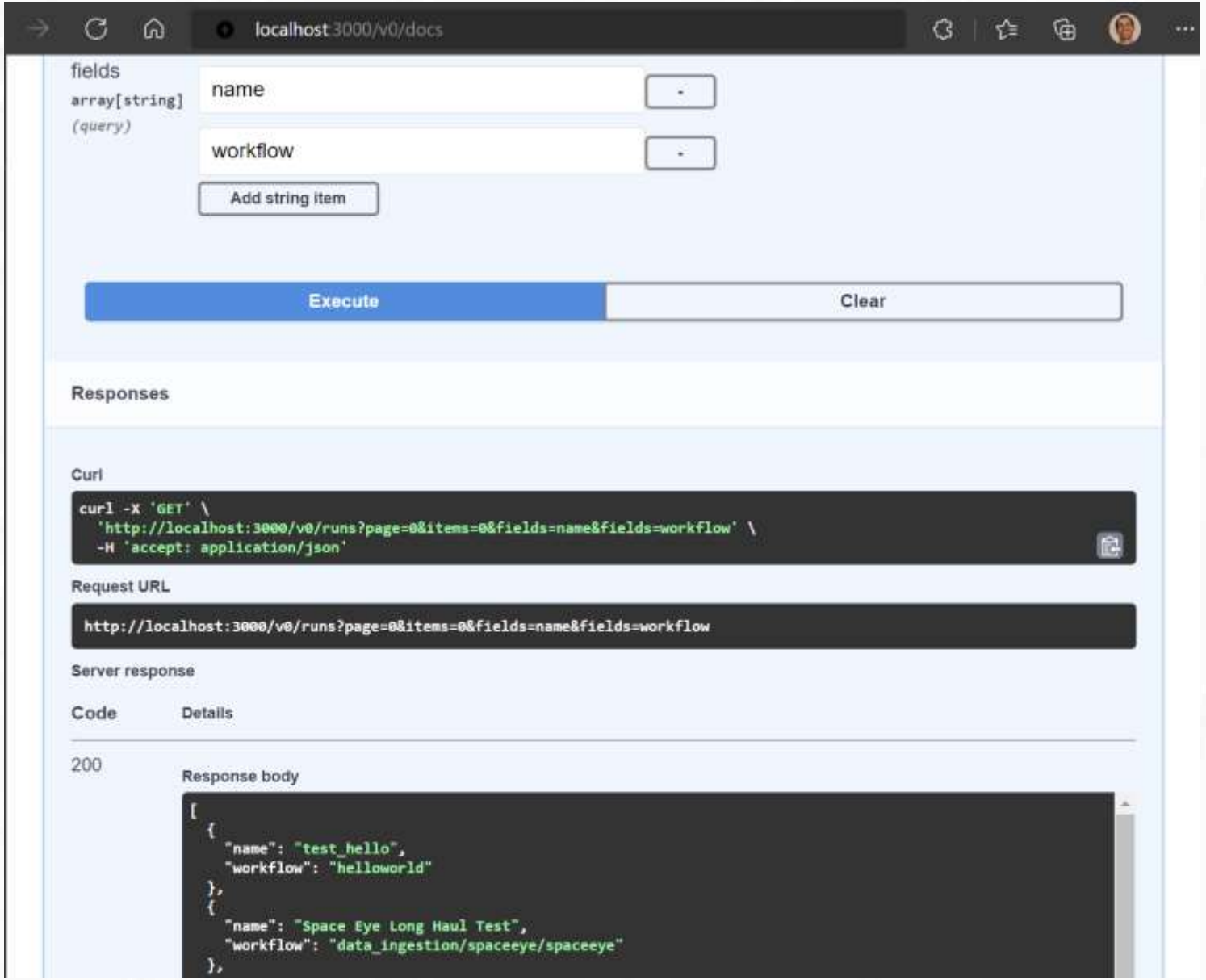


FarmVibes.AI / TerraVibes REST API

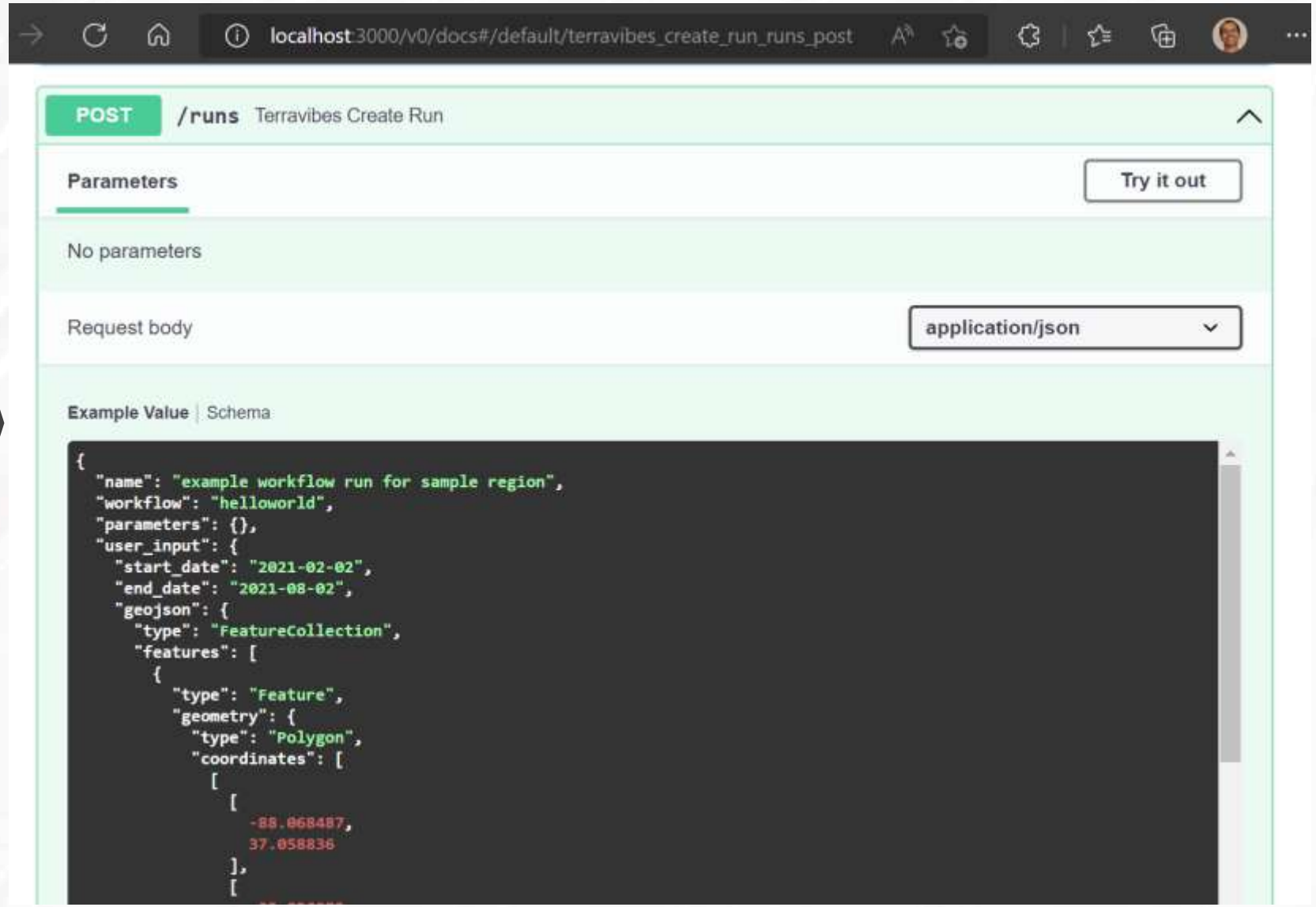
`http://192.168.49.2:30000/v0/docs`

GET	`/`	Terravibes Root
GET	`/workflows`	Terravibes List Workflows
GET	`/runs`	Terravibes List Runs
POST	`/runs`	Terravibes Create Run
GET	`/runs/{run_id}`	Terravibes Describe Run
POST	`/runs/{run_id}/cancel`	Terravibes Cancel Run

Querying workflow runs in the API



Requesting new runs with the API



Client Features

- All REST API operations on workflow runs:
  - Submission
  - Cancelation
  - Description
  - Monitoring
- Listing workflows
- Workflow documentation
- Custom workflow submission

Workflows

- A mapping with the keys:
  - Name
  - Sources
  - Sinks
  - Edges
  - Tasks
    - Ops
    - Workflows
  - Description

- Which describes
  - A static computational graph
    - Data fed into *sources*
    - Computation happens at nodes (*tasks*)
    - With *typed edges*
    - Output gathered from *sinks*
    - Parameterizable
    - Self-documenting

Anatomy of a workflow

- TerraVibes (FarmVibes.AI's engine) works with ***workflows***
  - Computational graphs for geospatial processing

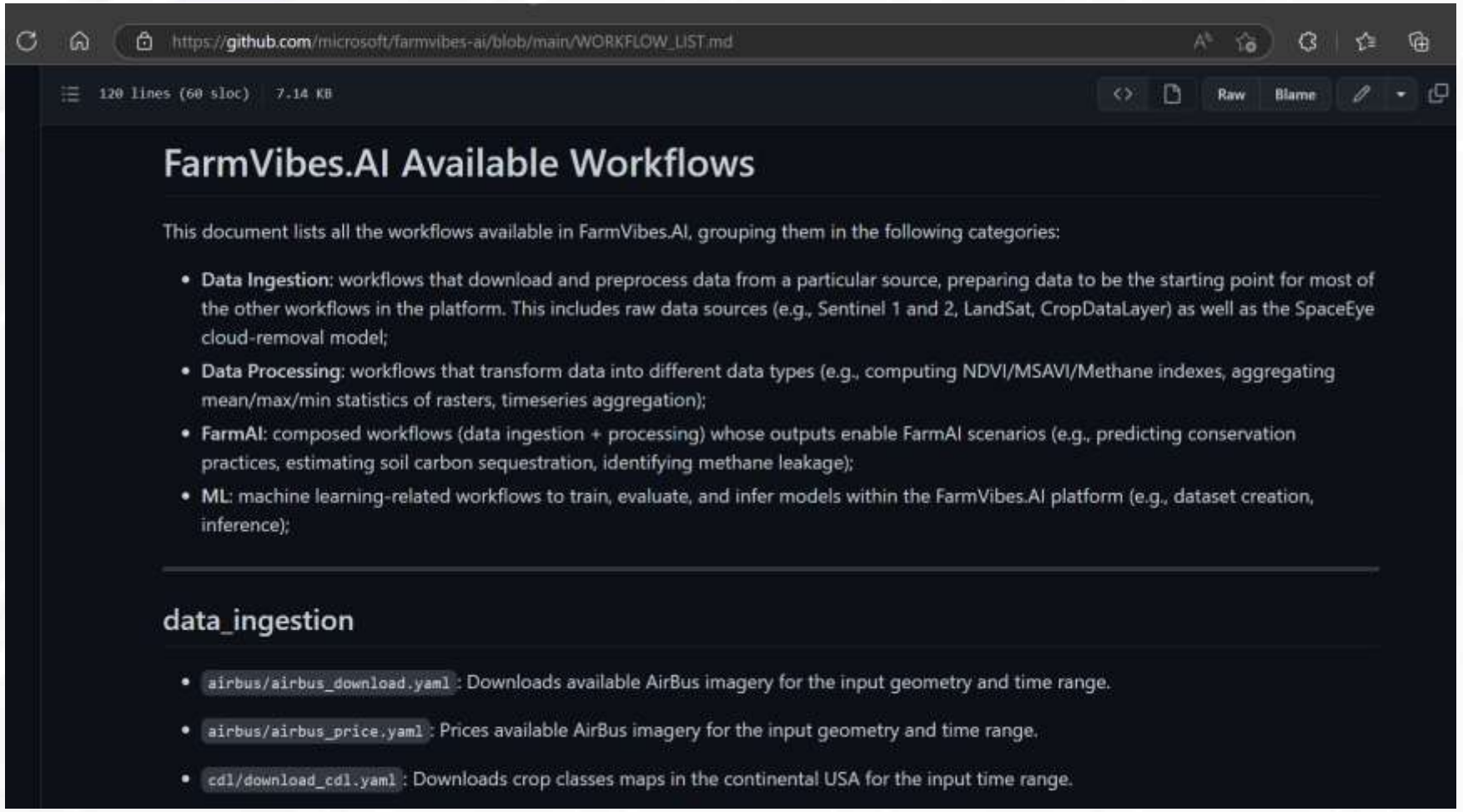
```yaml
name: example_workflow
sources:



sinks:


parameters:

tasks:




edges:
```

Anatomy of a Workflow: Documentation

- Keep documentation and definition close together

```
>>> from vibe_core.client import get_default_vibe_client
>>> client = get_default_vibe_client()
>>> hello = [w for w in client.list_workflows() if 'hello' in w][0]
>>> client.document_workflow(hello)
Workflow: helloworld

Description:
    Hello world! Small test workflow that generates an image of the Earth with countries that
    intersect with the input geometry highlighted in orange.

Sources:
    - user_input (vibe_core.data.core_types.DataVibe): Input geometry.

Sinks:
    - raster (vibe_core.data.rasters.Raster): Raster with highlighted countries.

Tasks:
    - hello
```

Documentation on github



Anatomy of a Workflow: Sources

- Inputs to the workflow
- Usually, a GeoJSON + Time Range
- Provided by the user at time of submission

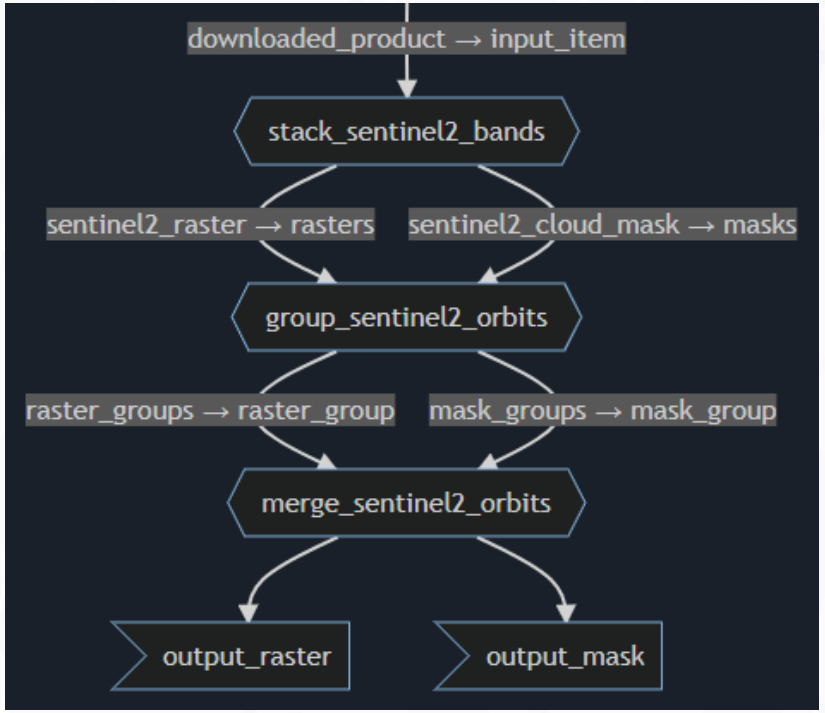
- Example:

Sample User Input

```
"user_input": {
  "start_date": "2021-02-02",
  "end_date": "2021-08-02",
  "geojson": {
    "type": "FeatureCollection",
    "features": [
      {
        "type": "Feature",
        "geometry": {
          "type": "Point",
          "coordinates": [
            [[0, 0], [0, 1], [1, 1], [1, 0], [0, 0]]
          ]
        }
      }
    ]
  }
}
```

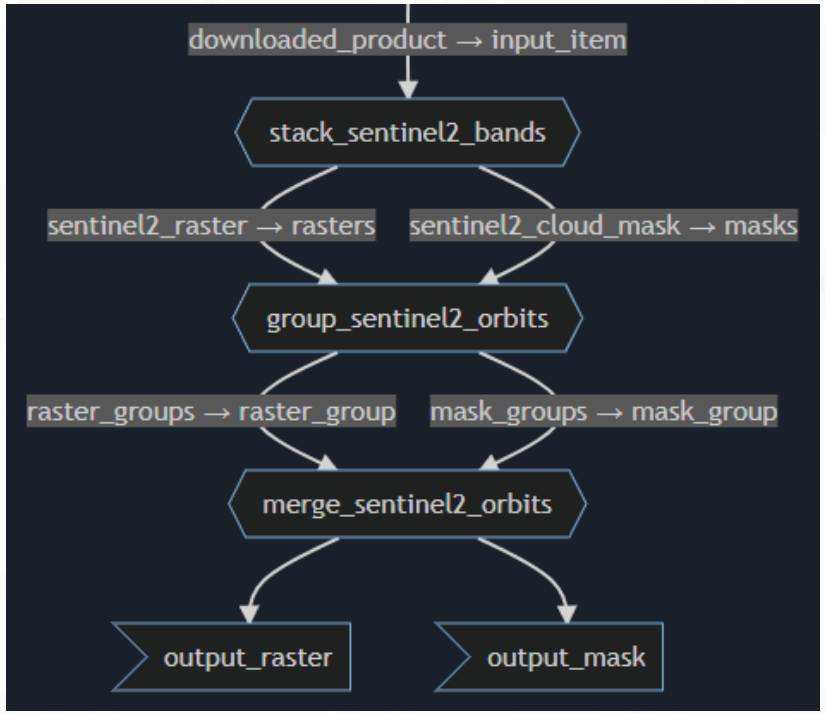
Anatomy of a Workflow: Tasks

- Which tasks FarmVibes.AI should run as part of the workflow
- A dictionary with task names as keys, and task definitions as values:
  - Ops
    - The most basic computation unit in FarmVibes.AI
  - Workflows
    - Allows composability



Anatomy of a Workflow: Edges (Typed)

- Edges: Connections between task output and input ports
- Have typing information
  - Validated before execution



Anatomy of a Workflow: Sinks

- Sinks define what the output of a workflow will be
- Collected from task outputs
- Available at REST API / Storage

Sample Workflow Output


```
{
  "name": "test_hello",
  "workflow": "helloworld",
  "output": {
    "raster": [
      {
        "assets": {
          "3efe308a-c35f-4b49-b46d-f415e15393b5": {
            "href": "/home/azureuser/.cache/farmvibes-
ai/data/assets/3efe308a-c35f-4b49-b46d-f415e15393b5/3efe308a-
c35f-4b49-b46d-f415e15393b5.tif",
            "type": "image/tiff"
          }
        },
        ...
        "terravibes_data_type": "Raster"
      }
    ]
  }
}
```

Anatomy of a Workflow: Parameters

- Customize behavior in runs, define secrets

```
name: ndvi_summary
sources:
  user_input:
    - s2.user_input
    - summary_timeseries.input_geometry
sinks:
  timeseries: summary_timeseries.timeseries
parameters:
  pc_key:
description:
  parameters:
    pc_key: Planetary computer API key.
  short_description:
    Calculates NDVI statistics (mean, standard deviation, maximum and minimum) for the input
    geometry and time range.
  long_description:
    The workflow retrieves the relevant Sentinel-2 products with Planetary Computer (PC) API,
    forwards them to a cloud detection model and combines the predicted cloud mask to the mask
    obtained from the product. The workflow computes the NDVI for each available tile and date,
    summarizing each with the mean, standard deviation, maximum and minimum values for the regions
```

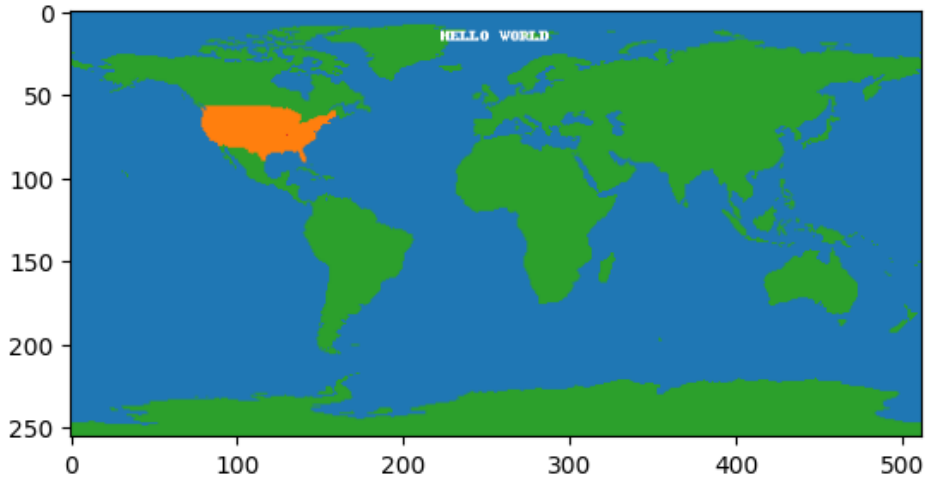
```
farmvibes-ai.sh add-secret <key> <value>

Adds a secret with a key <key> and value <value>.

Secrets are used in ops with the @SECRET parameter. For example,
"@SECRET(eywa-secrets, pc-sub-key)" in which "pc-sub-key" is the key and
eywa-secrets is the key-vault. Key-vaults are only required for an Azure
instalation. For this local farmvibes.ai instalation, the key-vault can be any
non-empty string.
```

The “Hello, World!” workflow definition

```yaml
name: helloworld
sources:
  user_input:
    - hello.user_input
sinks:
  raster: hello.raster
tasks:
  hello:
    op: helloworld
description:
  short_description: Hello world!
  long_description:
    Small test workflow that generates an image of the Earth with countries that intersect with the

    input geometry highlighted in orange.
  sources:
    user_input: Input geometry.
  sinks:
    raster: Raster with highlighted countries.
```

The “Hello, World!” workflow output

```
In [1]: %matplotlib inline
        import matplotlib.image as mpimg
        import matplotlib.pyplot as plt
        from vibe_core.client import get_default_vibe_client

In [2]: client = get_default_vibe_client()
        hello = client.get_run_by_id(client.list_runs()[0])

In [3]: world = mpimg.imread(hello.output['raster'][0].assets[0].path_or_url)

In [4]: plt.imshow(world)

Out[4]: <matplotlib.image.AxesImage at 0x7f20cf177d30>
```



Finding workflow outputs on disk

- All op outputs are cached

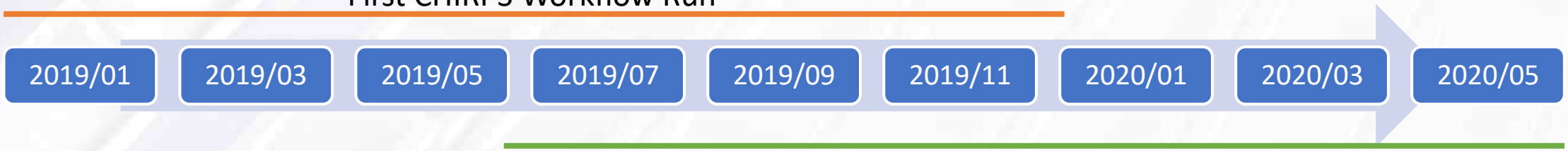
```
→  ~ ls ~/.cache/farmvibes-ai/data/stac/
compute_cloud_prob          group_sentinel2_orbits      select_necessary_coverage_items
compute_shadow_prob         helloworld                  stack_sentinel2_bands
download_sentinel2_from_pc  list_sentinel2_products_pc
group_s2_tile_sequence      merge_sentinel2_orbits
```

- And saved in the STAC format

```
→  ~ tree ~/.cache/farmvibes-ai/data/stac/helloworld
/home/azureuser/.cache/farmvibes-ai/data/stac/helloworld
└── 5aeb900db2668ed7571400062032da49ea1ccb3415bd3d4290a8542bf9c9e8d9
    ├── catalog.json
    └── raster
        ├── 989bbf0c-f75b-4751-84b6-cfba166c72c0
        │   └── 989bbf0c-f75b-4751-84b6-cfba166c72c0.json
        └── collection.json

3 directories, 3 files
```

Example leveraging the cache

First CHIRPS Workflow Run



Second CHIRPS Workflow Run

Time Range: 2019-01-01 00:00:00, 2020-01-01 00:00:00

FarmVibes.AI data_ingestion/weather/download_chirps
Run name: CHIRPS Download Test - 2019
Run id: 52a4fc4d-390f-4dfd-a7dd-3d1caf4c3b58

Task Name	Status	Start Time	End Time	Duration
output_handler	done	2022/11/09 16:31:43	2022/11/09 16:31:43	00:00:00
download_chirps	done	2022/11/09 16:26:49	2022/11/09 16:31:43	00:04:54
list_chirps	done	2022/11/09 16:26:42	2022/11/09 16:26:49	00:00:06

Last update: 2022/11/09 19:01:46

Time Range: 2019-06-01 00:00:00, 2020-06-01 00:00:00

FarmVibes.AI data_ingestion/weather/download_chirps
Run name: CHIRPS Download Test - Mid 2019 -> Mid 2020
Run id: 81d8b6c9-39db-4f54-a0d4-0d2451227a39

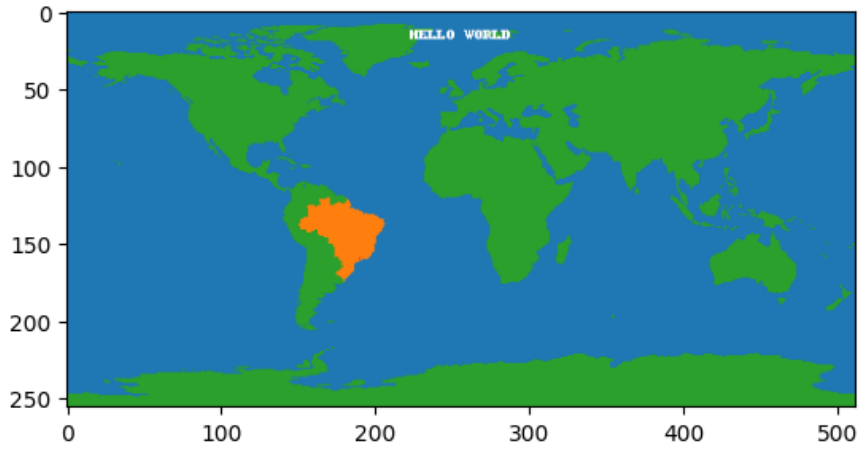
Task Name	Status	Start Time	End Time	Duration
output_handler	done	2022/11/09 17:04:40	2022/11/09 17:04:40	00:00:00
download_chirps	done	2022/11/09 17:02:30	2022/11/09 17:04:40	00:02:10
list_chirps	done	2022/11/09 17:02:19	2022/11/09 17:02:30	00:00:10

Last update: 2022/11/09 19:01:48

Submitting custom workflows

```yaml
name: custom_hello_world
tasks:
  hi:
    op: helloworld
sources:
  in:
    - hi.user_input
sinks:
  world:
    hi.raster
```

```
In [13]: hi = client.get_run_by_id(client.list_runs()[-1])
         hi.block_until_complete()

Out[13]: 'VibeWorkflowRun'(id='110bcbca-af2e-4e0e-b119-86bcf590c63e', name='custom', workflow='{'name': 'custom_hello_world', 'tasks': {'hi': {'op': 'helloworld'}}, 'sources': {'in': ['hi.user_input']}, 'sinks': {'world': 'hi.raster'}, 'default_parameters': {}, 'parameters': {}}', status='done')

In [14]: another_world = mpimg.imread(hi.output['world'][0].assets[0].path_or_url)
         plt.imshow(another_world)

Out[14]: <matplotlib.image.AxesImage at 0x7f30affb35b0>
```



Task Nam				
output_h				
hi	done	2022/11/08 20:55:46	2022/11/08 20:59:26	00:03:39

Last update: 2022/11/08 21:09:35

Microsoft Research
Summit 2022

FarmVibes.AI OSS Overview & Python Client

Stay in touch:

@renatolfc
@in/brsilvarec

github.com/microsoft/farmvibes-ai