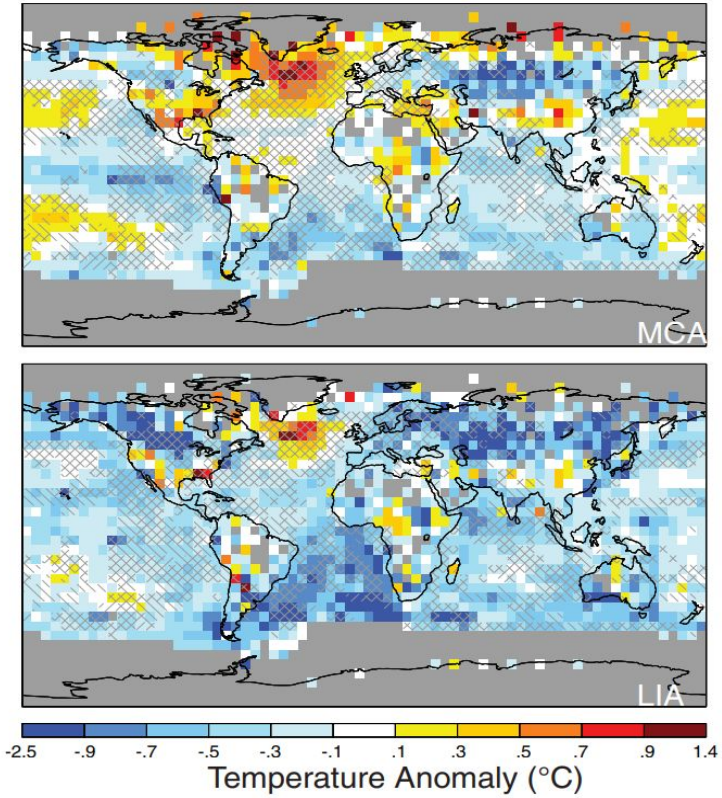
CS Capstone: Fossilized Controller

Fossilized Containers: Emily Ramirez Serrano, Jeremy Klein, Jadon Fowler, and Mumbi Mbuthia

Background

- Paleoclimatology is the study of past climates
- Climate Reconstructions show models of climate over time
- Combating climate change
  - Better understand the past for the future



Our Client

Dr. Nicholas McKay

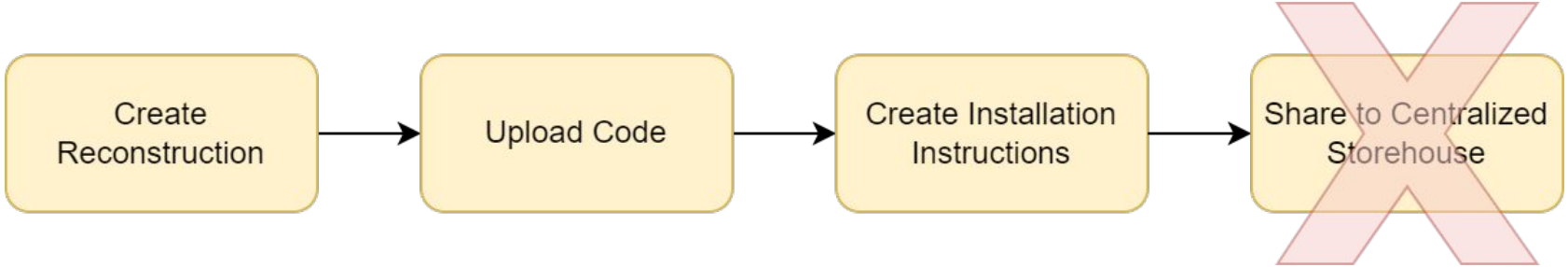
Associate Professor at NAU

Paleoclimate Dynamics Laboratory

Problem Statement

No Centralized Storehouse

- Difficult to share climate reconstructions to others.
- Difficult to find climate reconstructions online



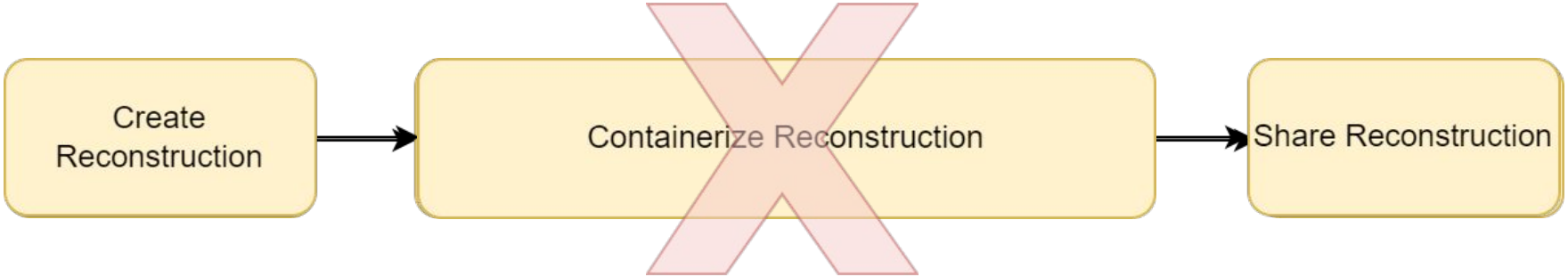
Problem Statement

Difficult Setup

- New environment and dependencies for each climate reconstruction
- Time consuming installation process

Containerization

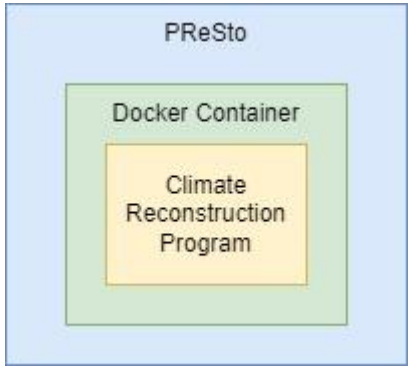
- A way to package software
- Not every scientist knows how to containerize their code



Solution Overview

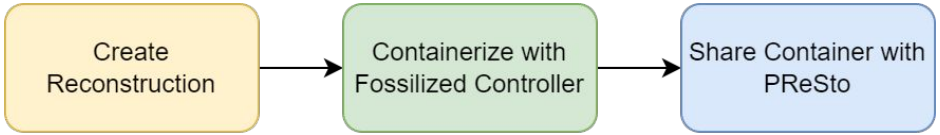
PReSto

- Paleoclimate Reconstruction Storehouse



Docker Container

- Removes assumptions of installed libraries and operating system

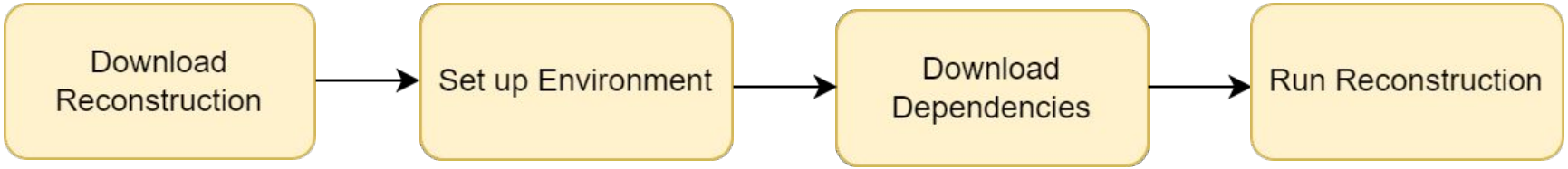


Fossilized Controller

- User Interface for building containers

Improved Workflow Use Case

- Download containers from centralized repositories
- Run climate models with custom parameters
- Automate running multiple different configurations
- Save hours of set up time



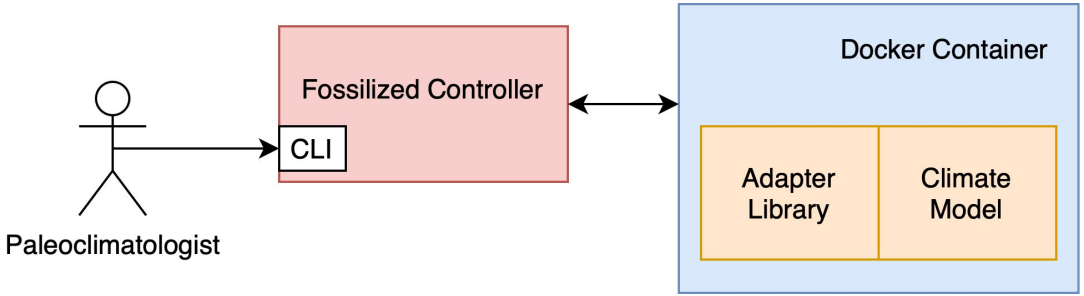
Requirements / Specs Review

- *Simple* to containerize and run climate models
- Language agnostic
- Quick integration for projects written in Python and R

```
# using Dr. McKay's Temp12k project as an example
~/projects $ cd ./Temp12k/

# guide the user through the creation process for the Dockerfile & other
metadata, creating prompts like "Are you using R? [Y/n]: "
~/projects/Temp12k $ presto create --maybe --some --flags --here
Are you using R? [Y/n]: Y
Creating PReSto Project ...

# on the user's computer, they can run the PReSto (Docker) container with
~/projects/Temp12k $ presto run --some --other --optional --flags
Running PReSto Project Temp12k ...

# now let's use it on monsoon
~/projects/Temp12k $ ssh jado@monsoon.nau.edu

# assume the presto controller & docker are installed on monsoon already
# also assume I've already uploaded my version of temp12k to Docker Hub
jado@monsoon.nau.edu:~ $ presto pull jado/temp12k

# now that the docker image has been pulled to the server, I can run it
# "input.json" may contain params / file locations sent to the HTTP server in
the container
jado@monsoon.nau.edu:~ $ presto run jado/temp12k --input input.json
```

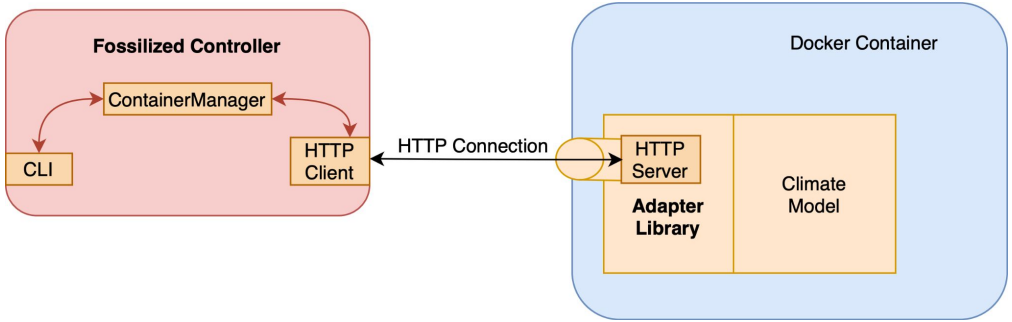
Architecture Overview

- Fossilized Controller contains Command Line Interface
- Controller communicates with Docker Containers
- Adapter Library for Scientists to use



Implementation Review

- Fossilized Controller is written in Python, using *Click* and *Docker SDK*
- Controller & containers connect over HTTP
- Adapter Libraries are written in Python and R



Prototype Review

Containerize, Run and Share

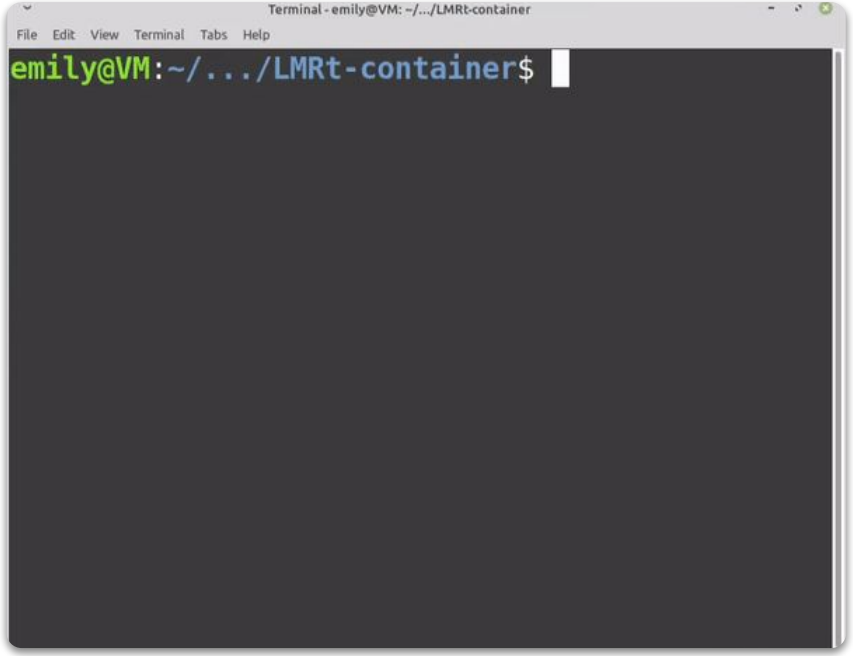
Prototype Review

```python
def lmrt_wrapper(adapter):
    # preprocessing
    print("\n======== Preprocessing ========\n")

    # ===Adapter work starts here===
    files = adapter.get_files()
    config = files['configs']
    parameters = adapter.get_parameters()
    print(parameters)

    # grabbing the specific parameter and saving it
    recon_param = parameters['recon_iterations']
    figure_type = parameters['figure_type']
    # ===Adapter work ends here===

    print(config)
```

Set up the climate reconstructions

- Make sure they add our adapter libraries to their main climate code

Prototype Review



Presto create

- Answer the prompt about their reconstruction and view Dockerfile

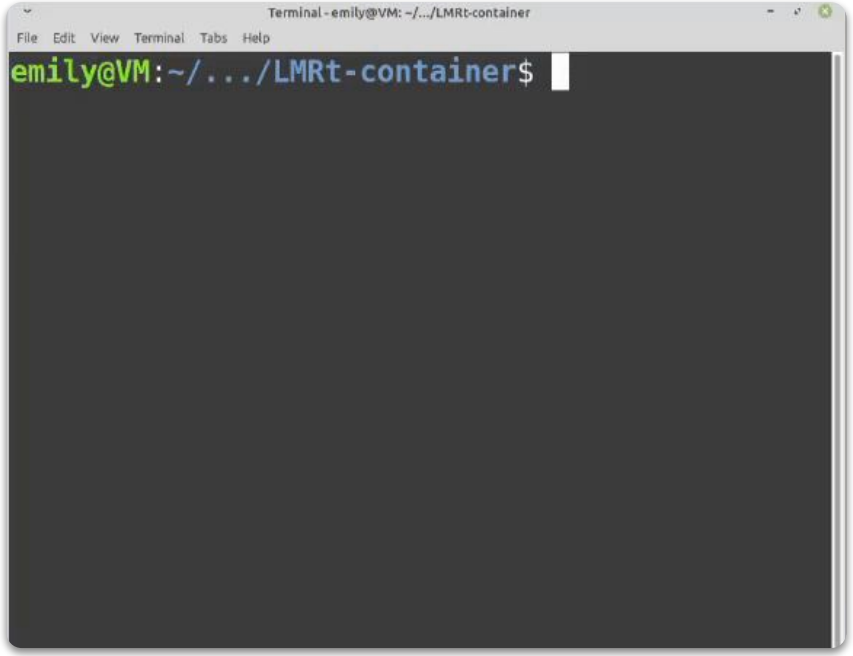
Prototype Review

```
FROM continuumio/anaconda3

RUN conda update -n base -c defaults conda

# setup conda environment
COPY presto_environment.yml .
RUN conda env create -f presto_environment.yml
RUN echo "conda activate presto_container" >> ~/.bashrc
SHELL ["/bin/bash", "--login", "-c"]
RUN conda activate presto_container

# copy all files to the root directory of the container
COPY . /

# run the command in the context of the environment we made
CMD conda run --no-capture-output -n \
    presto_container python3 main.py
```

Dockerfile

- Instructions to create the container

Prototype Review



Presto build

- Builds the image based on the Dockerfile

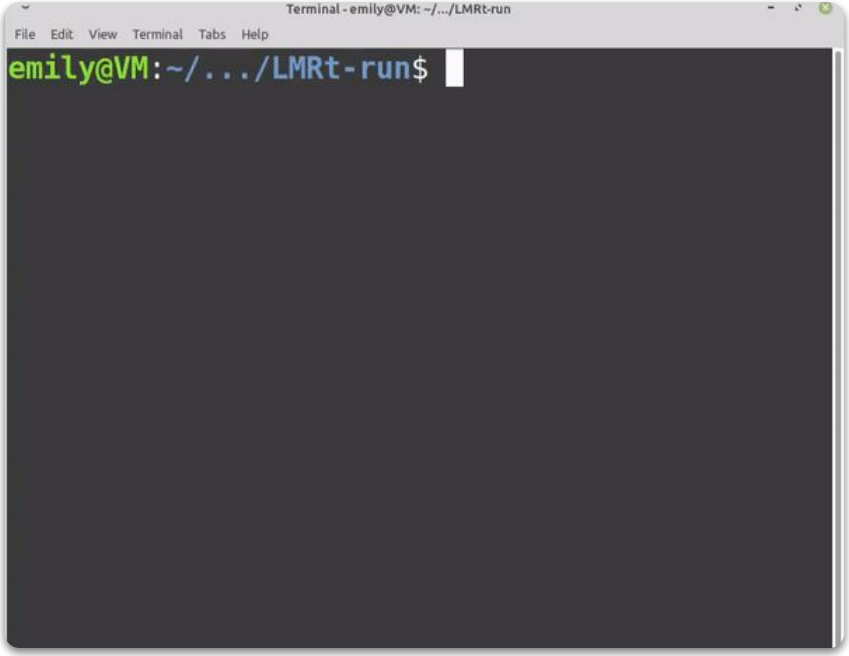
Prototype Review

```
{
    "parameters": {
        "recon_iterations": 1,
        "figure_type": "graph",
        "job_dirpath": "./recon"
    },
    "inputs": {
        "configs": "configs.yml"
    }
}
~
~
```

Metadata

- File that allows the user to communicate with the container

Prototype Review



Presto run

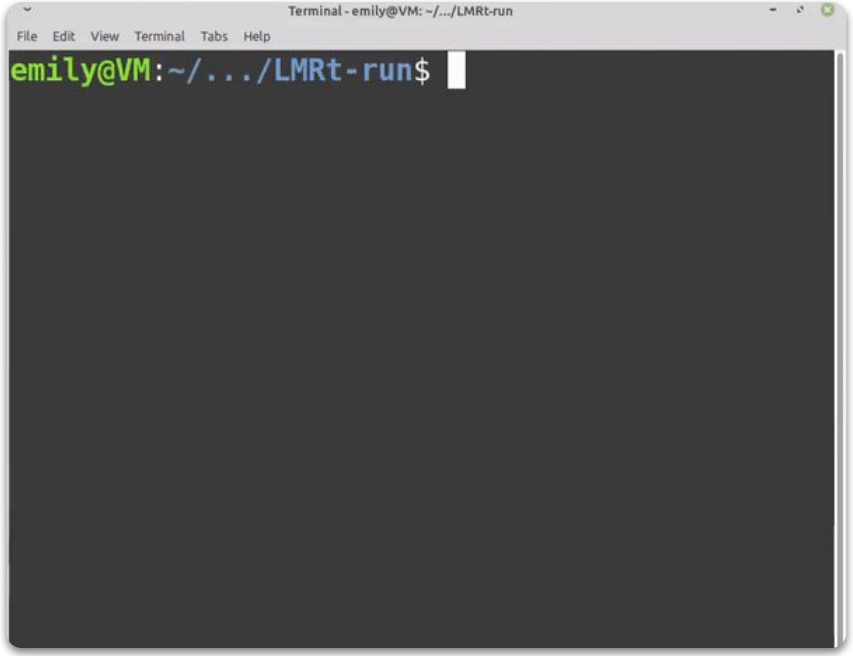
- Allows user to run the climate reconstruction container

Prototype Review

```
emily@VM:~/.../LMRt-run$ presto run lmr
Running the container...
{'Status': 'running', 'Running': True, 'Paused'
: False, 'Restarting': False, 'OOMKilled': Fals
e, 'Dead': False, 'Pid': 13631, 'ExitCode': 0,
'Error': '', 'StartedAt': '2022-04-17T01:55:11.
927166716Z', 'FinishedAt': '0001-01-01T00:00:00
Z'}
```

```
emily@VM:~/.../LMRt-run$ docker logs --follow $
(docker ps -q)
 * Serving Flask app 'adapter' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do no
t use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
```

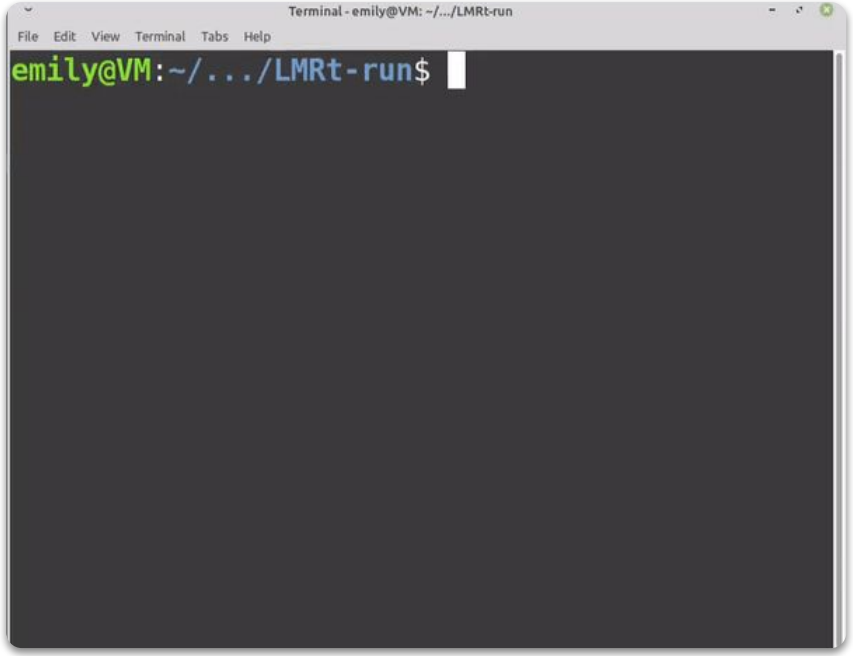
Prototype Review



Output data

- Results are returned in a zip file

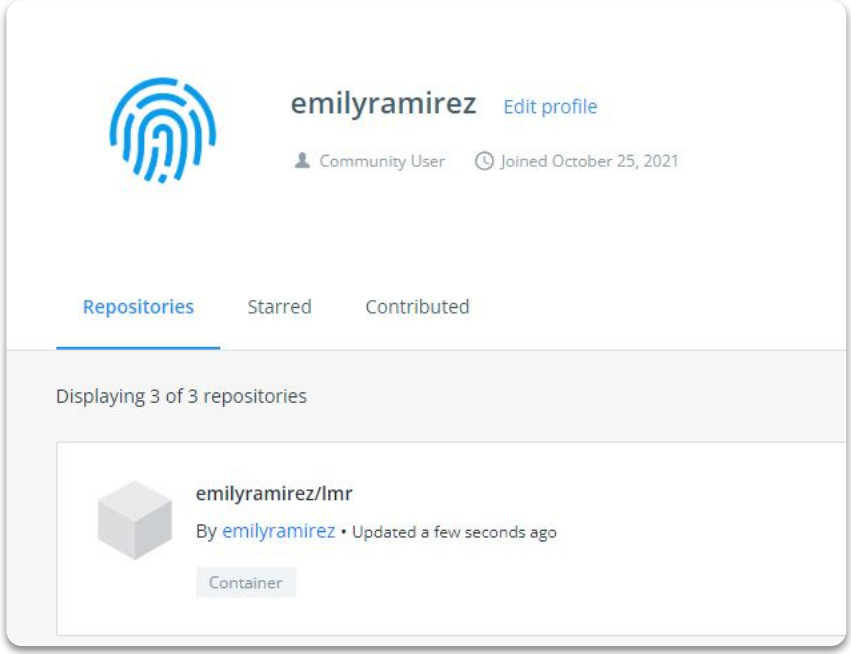
Prototype Review



Presto upload

- Upload to Docker Hub

Prototype Review

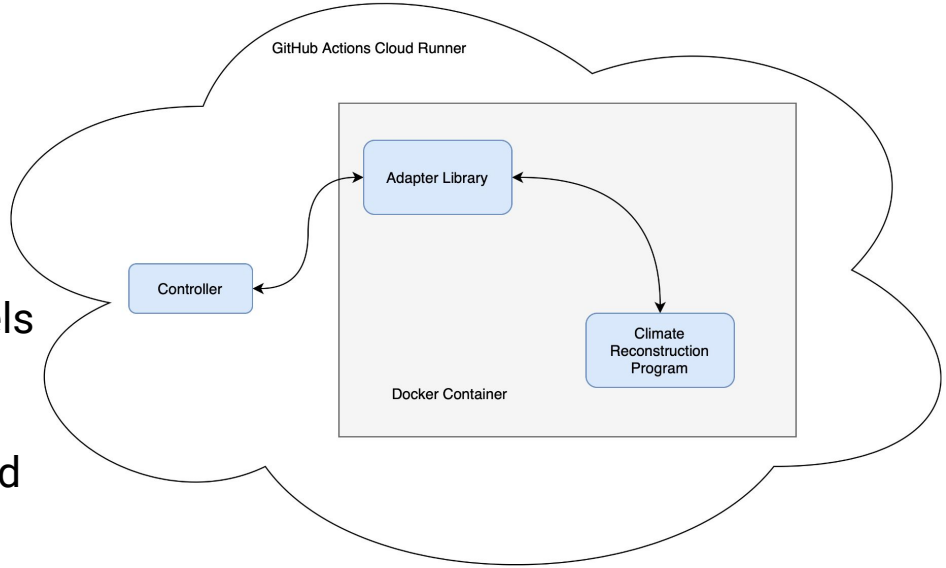


Presto upload

- Upload to Docker Hub

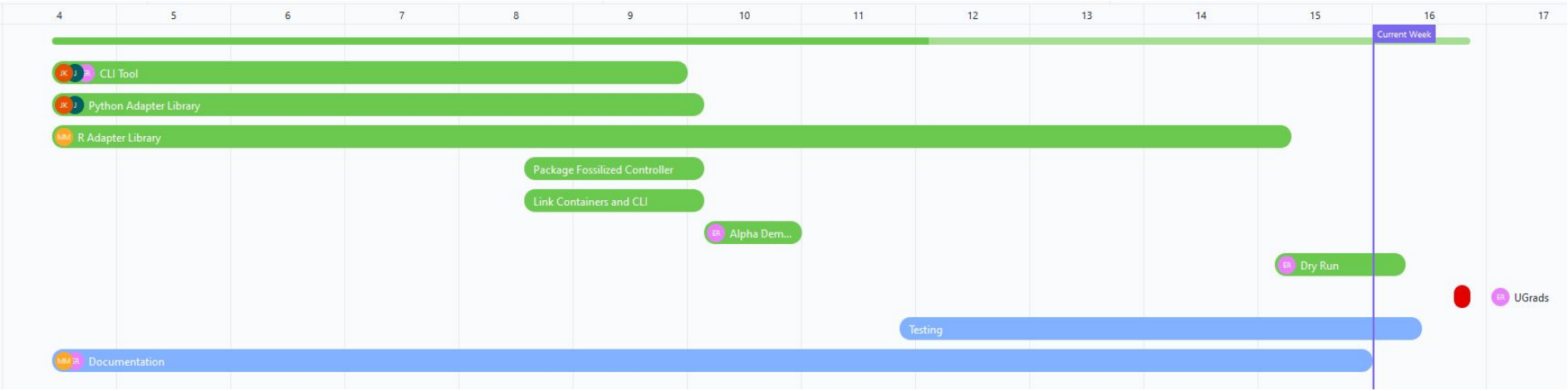
Testing Plan

- **Unit Tests** for Controller and Adapter Libraries
- **Integration Tests** run full climate models
- **Usability Tests** of prototyped Command Line Interface



Challenges and Resolutions

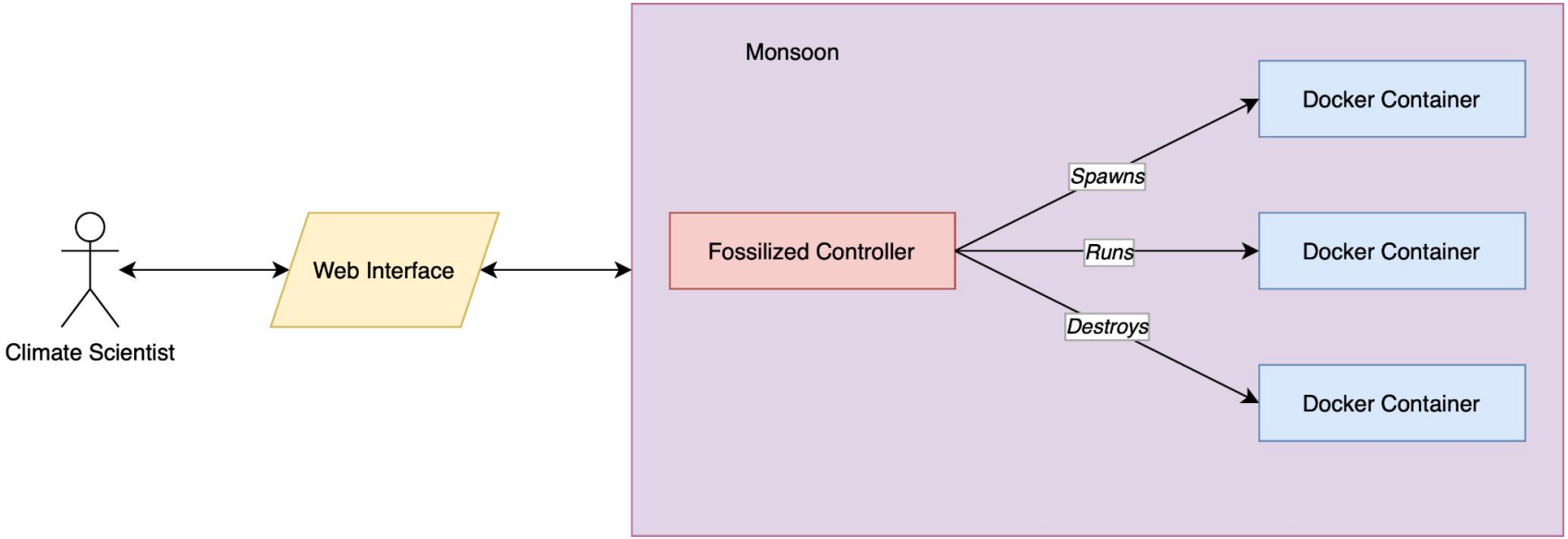
Risk	Difficulty	Solution
Live Logs for Time-Consuming Docker Processes (i.e. Building a Docker Image and Running a Docker Container)	**High**	1. **Build Image: Return Logs After Build is Finished** 2. **Run Container: Access Separate Terminal**
R Adapter Implementation	**Moderate**	**Research and Preparation**
Amount of Public, Testable Reconstructions	**Low**	**Thorough Usability Testing**


4
5
6
7
8
9
10
11
12
13
14
15
16
17
Current Week
CLI Tool
Python Adapter Library
R Adapter Library
Package Fossilized Controller
Link Containers and CLI
Alpha Dem...
Dry Run
UGrads
Testing
Documentation

Schedule

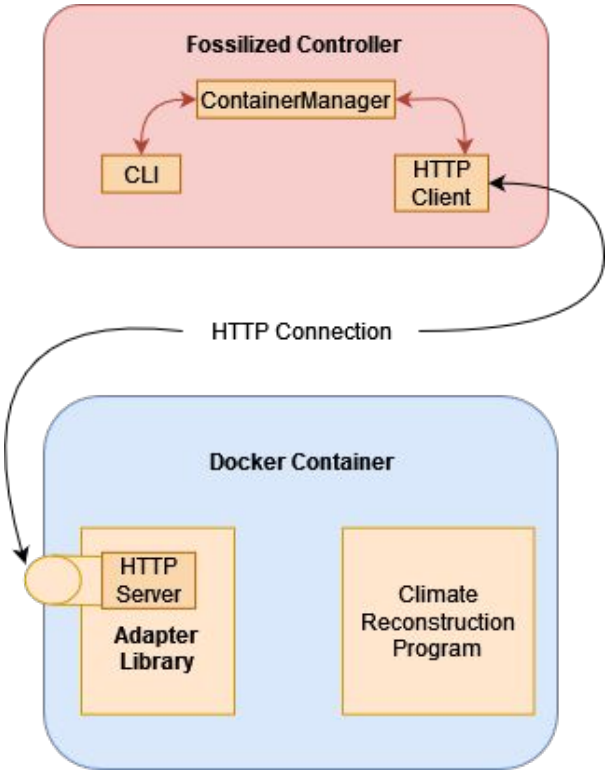
Future Work

Building a web interface for running different variations of model parameters.



Conclusion

- Paleoclimatology and model sharing
- What is The Fossilized Controller
- Where We Are Now
- Next:
  - Refine and test modules
  - Update documentation



Thank You