

Deploy and Monitor ML Pipelines with Open Source and Free Tools

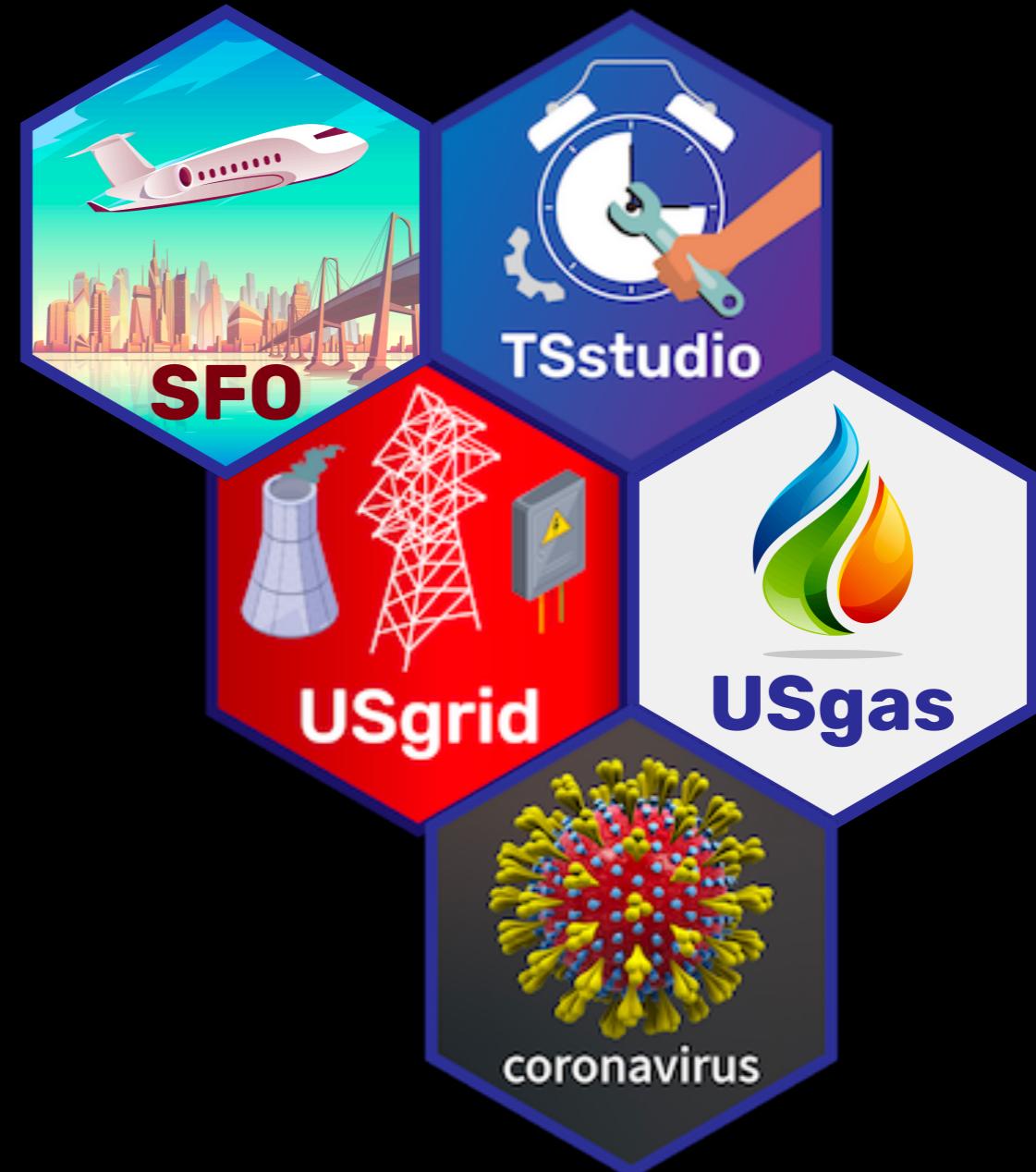
AI_Dev Conference

Rami Krispin, June 19th, 2024



About Me

- Senior Manager
- Forecasting
- MLOps
- Open Source
- Author



About Me

- Senior Manager
- Forecasting
- MLOps
- Open Source
- Author

Tutorials

vscode-python
A Tutorial for Setting Python Development Environment with VScode and Docker
● Shell ⭐ 781 ⚡ 69

vscode-python-template Template
A template for a dockerized Python development environment for VScode
● JavaScript ⭐ 83 ⚡ 19

ollama-poc
Getting started with Ollama for Python - a short tutorial for setting up Ollama for Python
● Python ⭐ 80 ⚡ 11

Introduction-to-Docker
(WIP) Getting started with Docker - An introduction to Docker with data science and engineering applications
⭐ 124 ⚡ 11

shinylive-r
A guide for deploying Shinylive R application into Github Pages
● R ⭐ 135 ⚡ 22

vscode-r
A Tutorial for Setting R Development Environment with VScode, Dev Containers, and Docker
● R ⭐ 224 ⚡ 23

vscode-r-template Template
A template for a dockerized R development environment for VScode
● R ⭐ 10 ⚡ 3

lang2sql
A tutorial for setting an SQL code generator with the OpenAI API
● Jupyter Notebook ⭐ 238 ⚡ 33

deploy-flex-actions
Deploying flexdashboard on Github Pages with Docker and Github Actions
● HTML ⭐ 210 ⚡ 28

shinylive
A guide for deploying Shinylive Python application into Github Pages
● HTML ⭐ 129 ⚡ 18

About Me

- Senior Manager
- Forecasting
- MLOps
- Open Source
- Author

Add a note...

Rami Krispin in Towards Data Science

Setting A Dockerized Python Environment — The Hard Way

This post will review different methods to run a dockerized Python environment from the command line (CLI). Am I...

Feb 13 543 6

...

Add a note...

Rami Krispin

Setting a Dockerized Python Development Environment Template

In this post, we will review how to set, with a few simple steps, a dockerized Python development environment with VScode and...

Jan 13 116 2

...

Add a note...

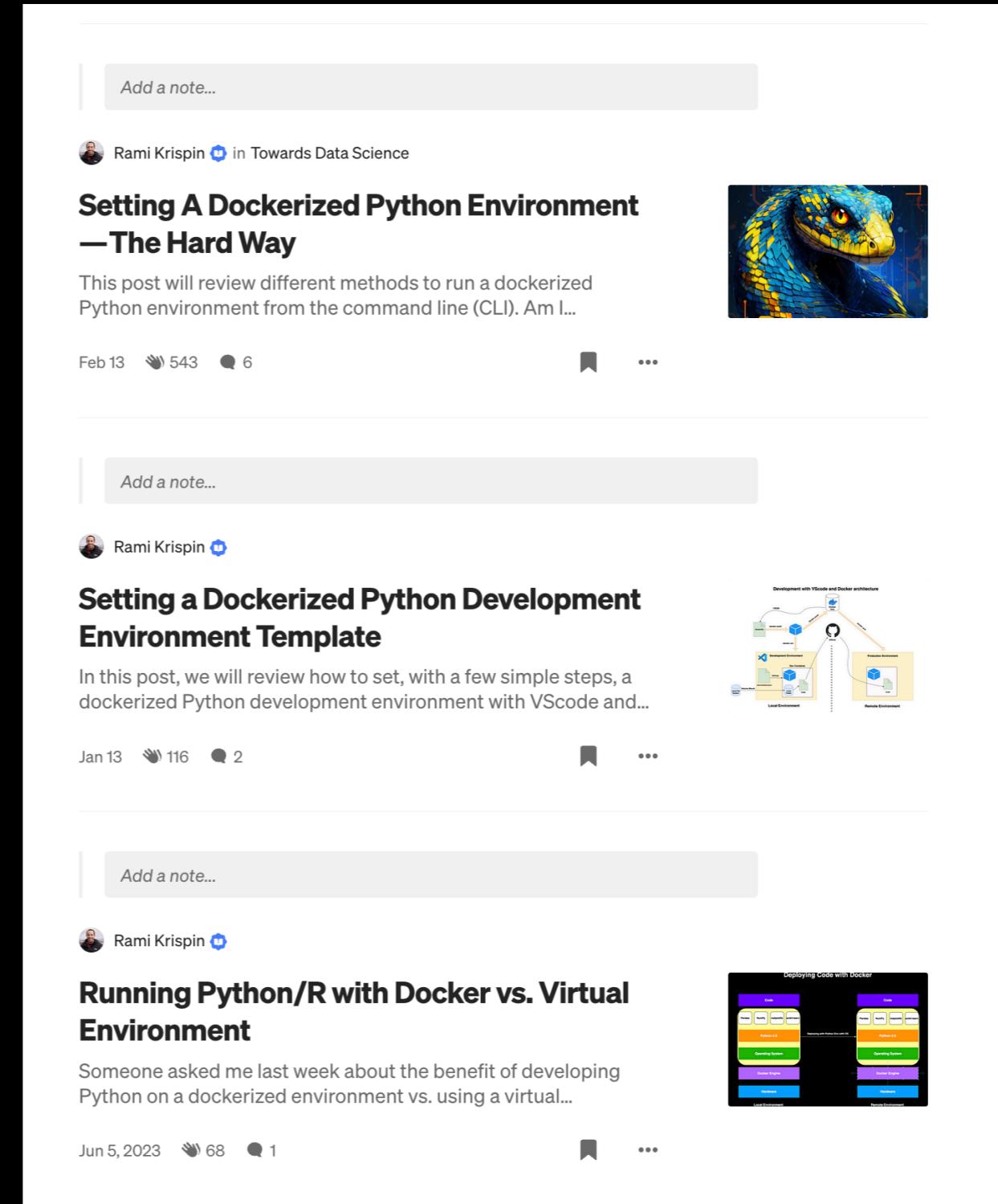
Rami Krispin

Running Python/R with Docker vs. Virtual Environment

Someone asked me last week about the benefit of developing Python on a dockerized environment vs. using a virtual...

Jun 5, 2023 68 1

...



About Me

- Senior Manager
- Forecasting
- MLOps
- Open Source
- Author

The screenshot shows a LinkedIn Learning course page titled "Data Pipeline Automation with GitHub Actions Using R and Python". The main video player features a smiling man named Rami Krispin, described as a "Senior manager of data science and engineering" with "decades of experience in working with data". The video progress bar indicates it's at 0:40 / 0:46. The left sidebar displays the course contents, which include an introduction and several sections on EIA API, GET request structure, and various data querying methods using R and Python. The right sidebar provides options for Overview, Q&A, Notebook, Transcript, Instructor details (Rami Krispin), related courses, and exercise files on GitHub.

About Me

- Senior Manager
- **Forecasting**
- **MLOps**
- **Open Source**
- Author

[https://github.com/RamiKrispin/
ai-dev-2024-ml-workshop](https://github.com/RamiKrispin/ai-dev-2024-ml-workshop)

Poll

Are You Familiar with?

- Docker
- GitHub Actions
- Quarto
- MLflow
- Python/R

Docker

[Docs](#) [Get support](#) [Contact sales](#)



Products ▾

Developers ▾

Pricing

Support

Blog

Company ▾



[Sign In](#)

[Get started](#)



Docker Builds: Now Lightning Fast

Announcing Docker Build Cloud general availability

[Discover Docker Build Cloud](#)

[What is Docker?](#)

**Accelerate how you build,
share, and run applications**

GitHub Actions



Search or jump to...

Pull requests Issues Codespaces Marketplace Explore



Features Actions Packages Security Codespaces Copilot Code review Search Issues Discussions

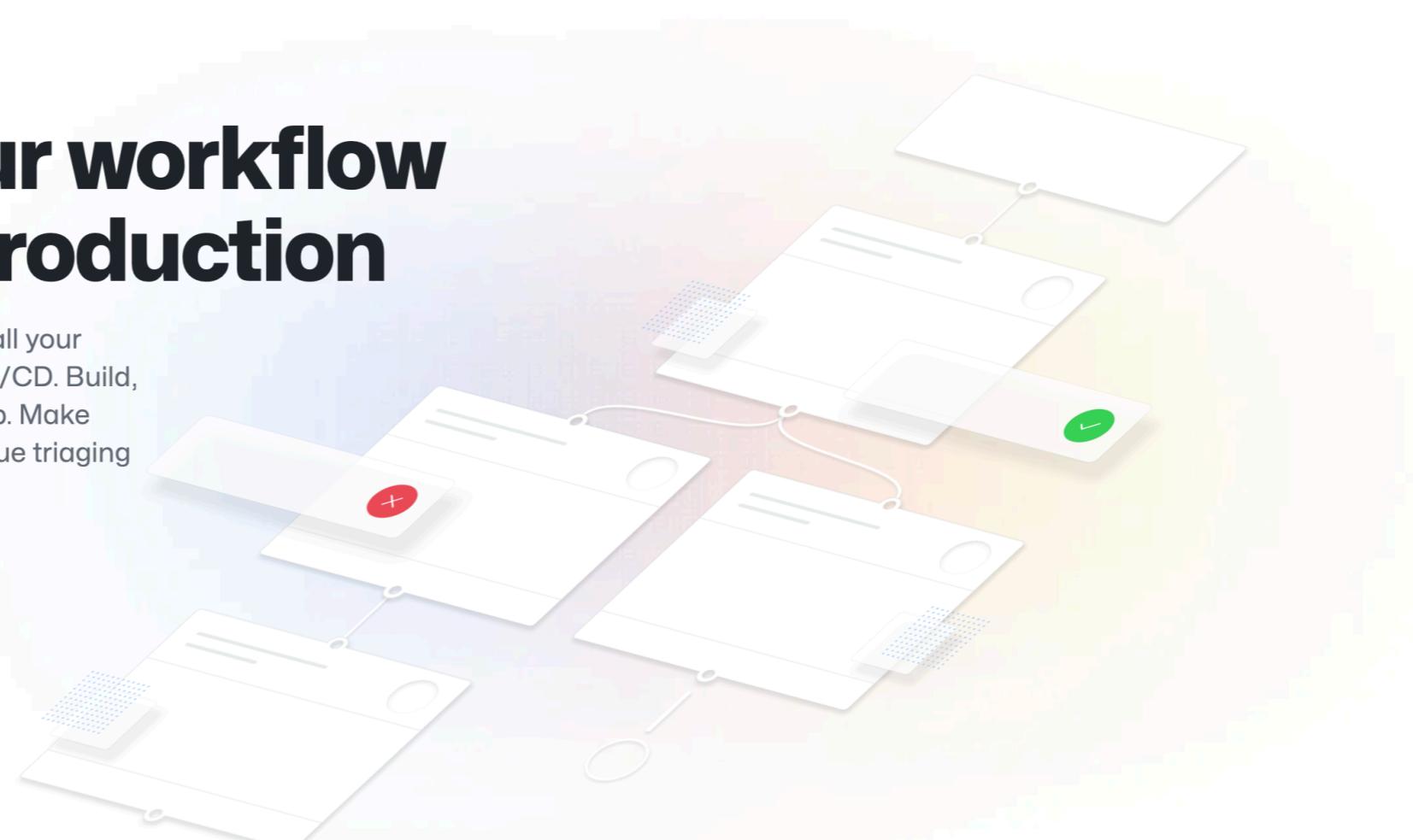


Automate your workflow from idea to production

GitHub Actions makes it easy to automate all your software workflows, now with world-class CI/CD. Build, test, and deploy your code right from GitHub. Make code reviews, branch management, and issue triaging work the way you want.

[Get started with Actions >](#)

Questions? [Contact Sales >](#)



Quarto

Hello, Quarto

Python

R

Julia

Observable

Combine Jupyter notebooks with flexible options to produce production quality output in a wide variety of formats. Author using traditional notebook UIs or with a plain text markdown representation of notebooks.

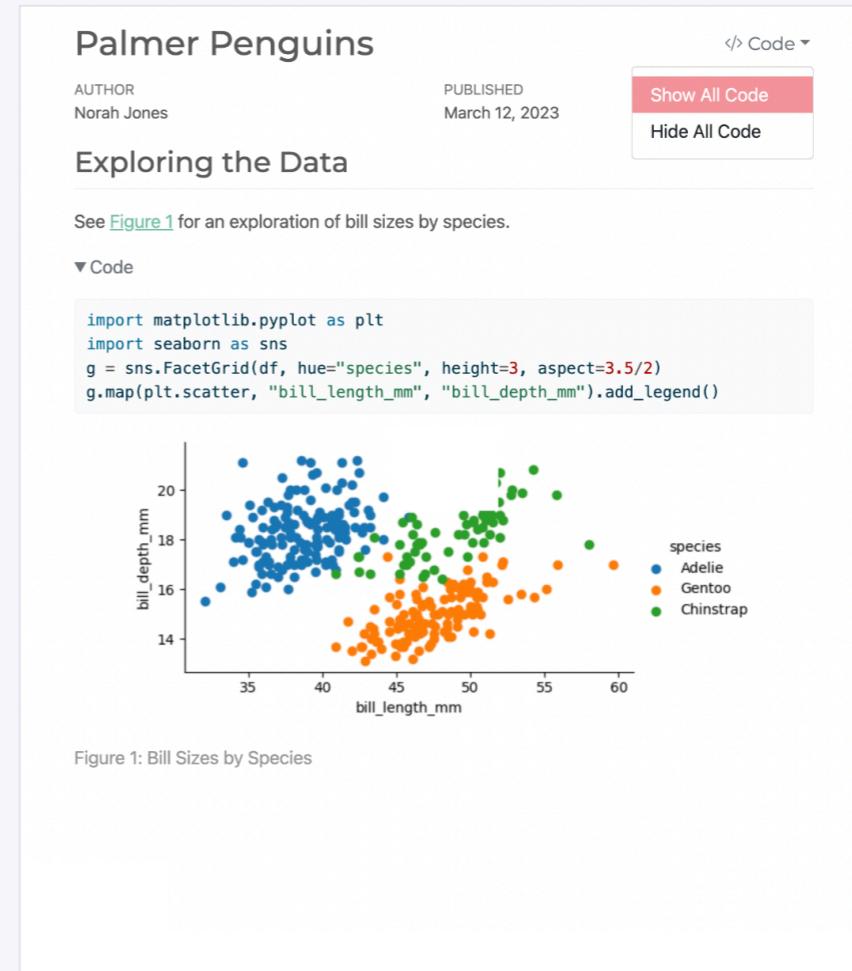
The screenshot shows a Jupyter Notebook interface with the title "Palmer Penguins". It contains the following content:

- A code cell with the following configuration:

```
author: Norah Jones
format:
  html:
    code-tools: true
    code-fold: true
```
- A code cell with the following Python code:

```
[3]: #| echo: false
import pandas as pd
df = pd.read_csv("palmer-penguins.csv")
```
- A section titled "Exploring the Data" with the instruction: "See @fig-bill-sizes for an exploration of bill sizes by species."
- A code cell with the following Python code:

```
[6]: #| label: fig-bill-sizes
#| fig-cap: Bill Sizes by Species
import matplotlib.pyplot as plt
import seaborn as sns
g = sns.FacetGrid(df, hue="species", height=3, aspect=3.5/1.5)
g.map(plt.scatter, "bill_length_mm", "bill_depth_mm").add_legend()
```
- A scatter plot showing bill length (mm) on the x-axis (ranging from 35 to 60) versus bill depth (mm) on the y-axis (ranging from 14 to 20). The data points are colored by species: Adelie (blue), Gentoo (orange), and Chinstrap (green).



Dynamic Documents

Generate dynamic output using Python, R, Julia,

Beautiful Publications

Publish high-quality articles, reports,

Scientific Markdown

Pandoc markdown has excellent support for

MLflow

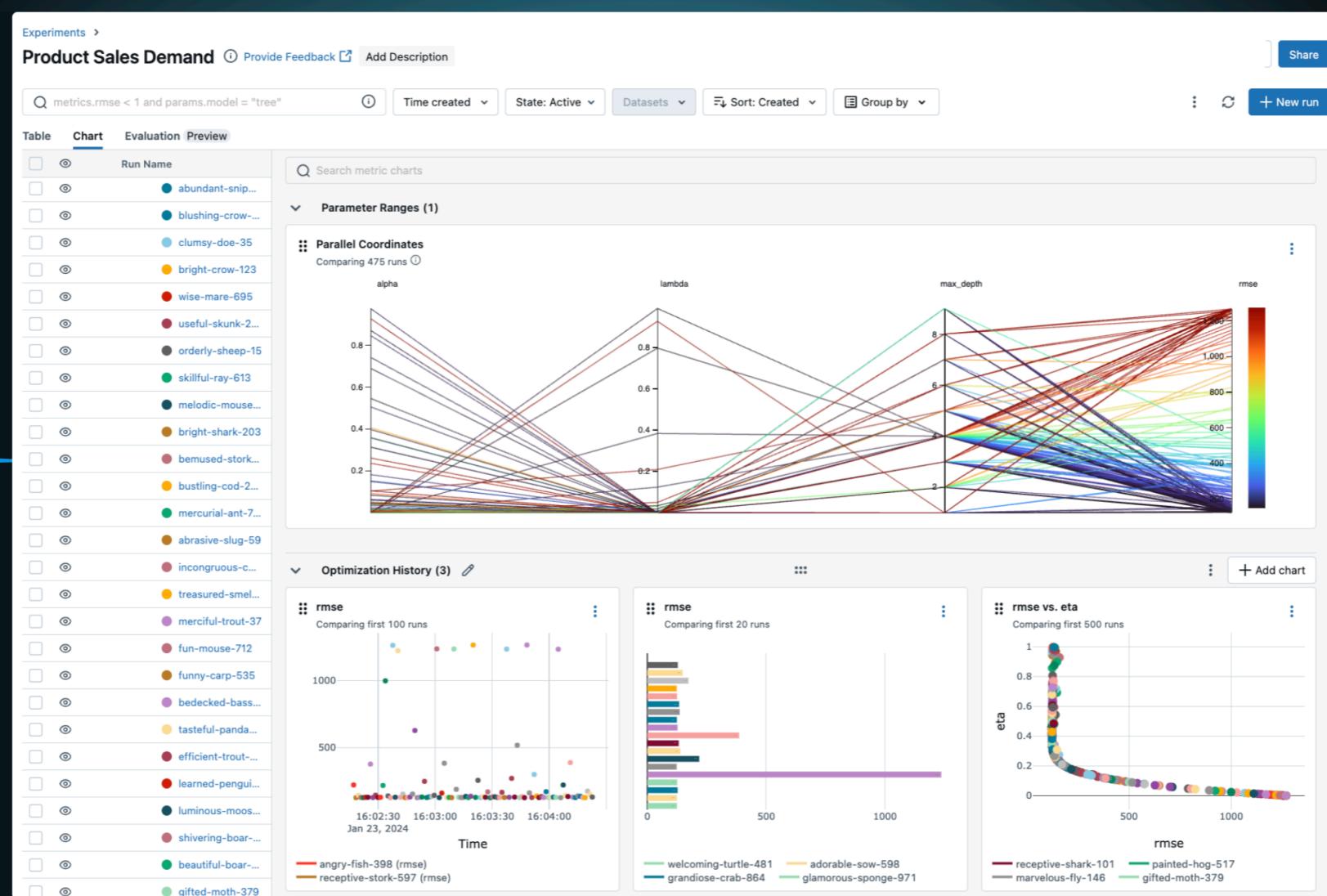
mlflow

Blog Releases Docs Contribute ↗ Ambassador Program

Get Started

Get Started

v2.14.0 See the latest release →



Alternatives Tools

Agenda

Motivation

Workflow

Design

Demo

Motivation

February 2020

Chinese stocks plunged 8% as coronavirus fears took hold. It's the worst day in years

By Laura He, CNN Business

Updated 9:15 AM EST, Mon February 3, 2020



10:32 p.m. ET, February 2, 2020

The best photos from the game



Source:
CNN

Kyle Terada/USA Today Sports

Jennifer Lopez's epic Super Bowl flag coat was custom Versace



By [Sandra Gonzalez](#), CNN

Updated 6:58 AM EST, Mon February 3, 2020

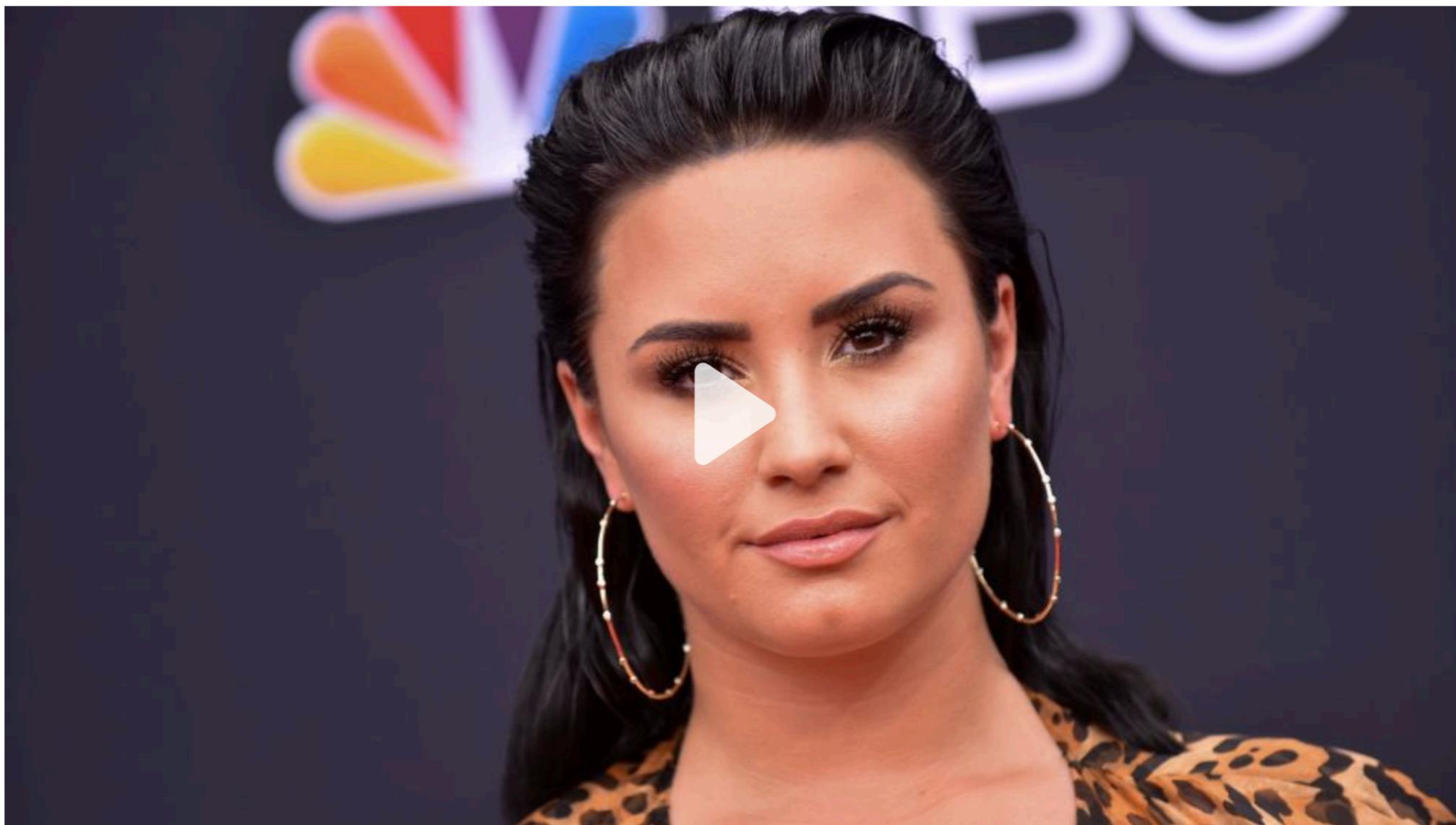


Demi Lovato rocks the National Anthem at Super Bowl LIV



By [Lisa Respers France, CNN](#)

Updated 11:27 PM EST, Sun February 2, 2020



Trump and Pelosi haven't spoken in months



By [Manu Raju](#), Senior Congressional Correspondent

Updated 8:49 AM EST, Mon February 3, 2020



'No reason for Americans to panic': White House seeks to calm fears over coronavirus

By Chadelis Duster, CNN

Updated 6:21 PM EST, Sun February 2, 2020





EVERYBODY PANIC!



 **Rami Krispin** @Rami_Krispin · Feb 8

Does anyone aware of a public dataset of the #coronavirus by case over time (e.g., time, city, country, lon, lat, etc...)?

2 1 1 1 1

 **Rami Krispin** @Rami_Krispin · Feb 12

I packaged the data behind the Johns Hopkins University #CoronavirusOutbreak dashboard into a #rstats #tidy format dataset package. Thanks to Johns Hopkins University Center for Systems Science and Engineering (JHU CCSE) for making the data public!



RamiKrispin/coronavirus
The coronavirus dataset. Contribute to RamiKrispin/coronavirus development by creating ...
[github.com](https://github.com/RamiKrispin/coronavirus)

36 110 1 1

 **Rami Krispin** @Rami_Krispin · Feb 24

(1/n)The coronavirus R dataset package is now available on CRAN (v0.1.0). The package provides a #tidy format for the data behind the Johns Hopkins University Center for Systems Science and Engineering dashboard:
ramikrispin.github.io/coronavirus/
#rstats, #coronavirus, #data



The 2019 Novel Coronavirus COVID-19 (2019-nCo...
Provides a daily summary of the Coronavirus (COVID-19) cases by state/province. Data source: ...
[ramikrispin.github.io](https://ramikrispin.github.io/coronavirus/)

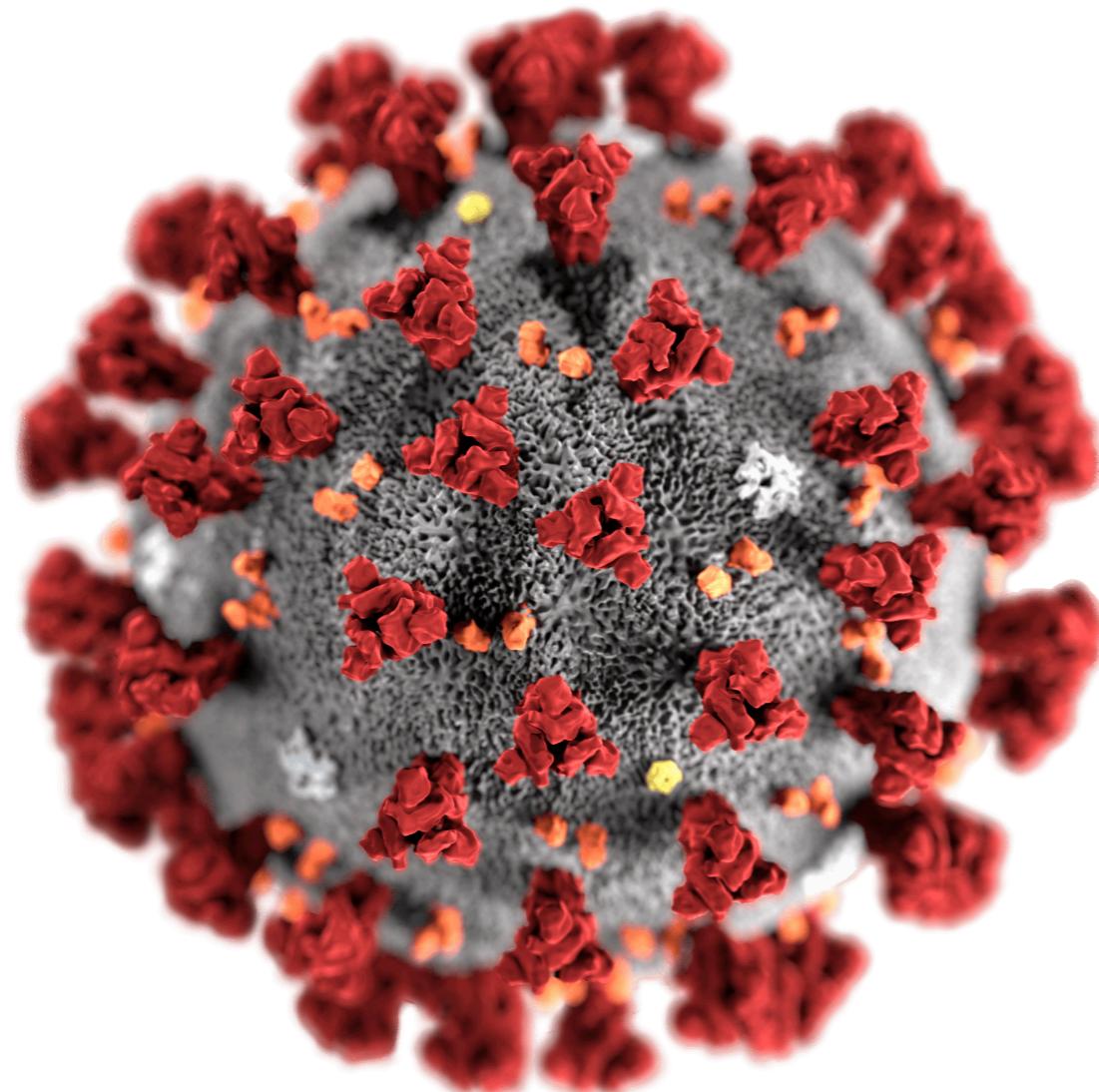
2 17 21 1 1

coronavirus

The coronavirus package provides a tidy format for the COVID-19 dataset collected by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University. The dataset includes daily new and death cases between January 2020 and March 2023 and recovery cases until August 2022.

More details available [here](#), and a `csv` format of the package dataset available [here](#)

Data source: <https://github.com/CSSEGISandData/COVID-19>



Source: Centers for Disease Control and Prevention's Public Health Image Library



Links

[View on CRAN](#)

[Browse source code](#)

[Report a bug](#)

License

[Full license](#)

[MIT + file LICENSE](#)

Citation

[Citing coronavirus](#)

Developers

Rami Krispin

Author, maintainer

Jarrett Byrnes

Author 

Dev status



 Data Pipeline passing

 CRAN 0.4.1

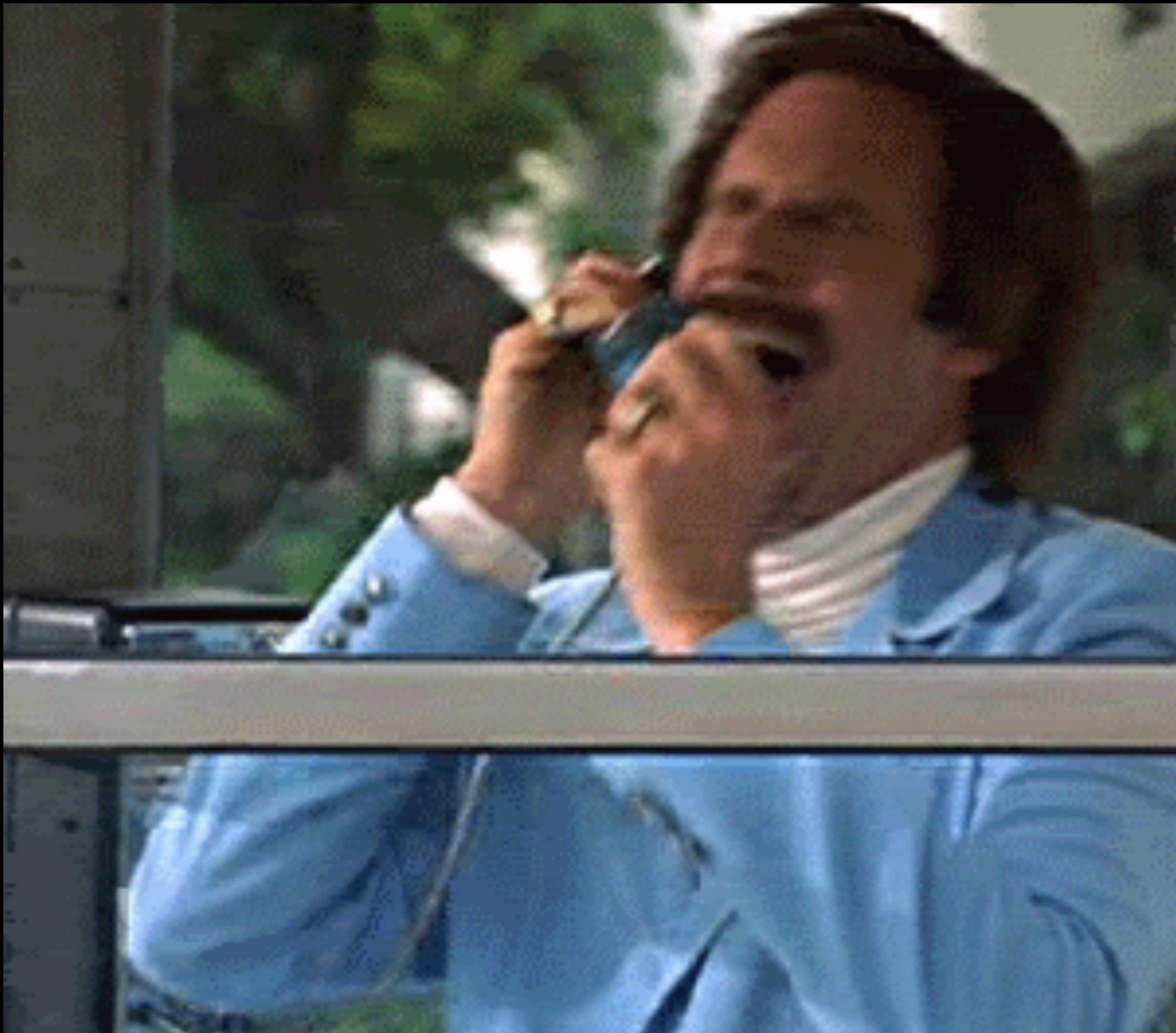
 lifecycle stable

 License MIT

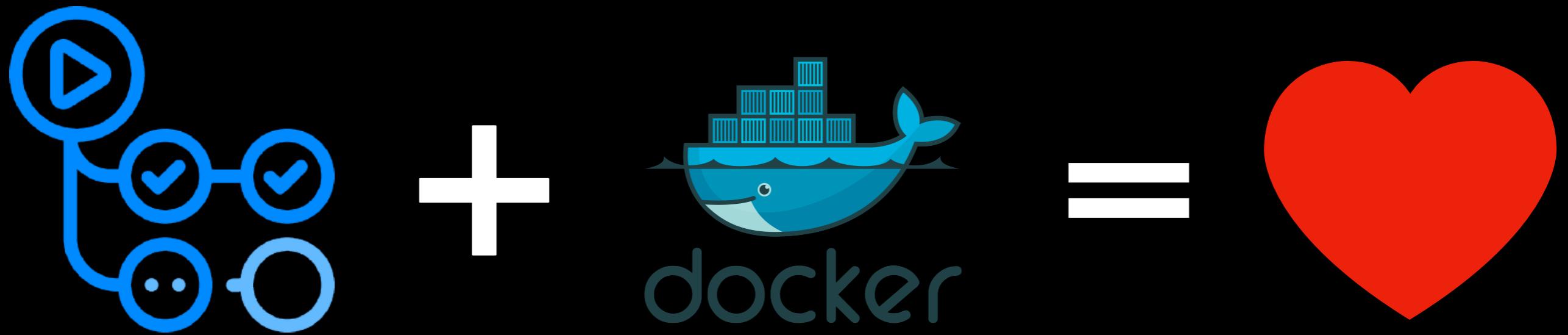
 last commit march

 downloads 74K

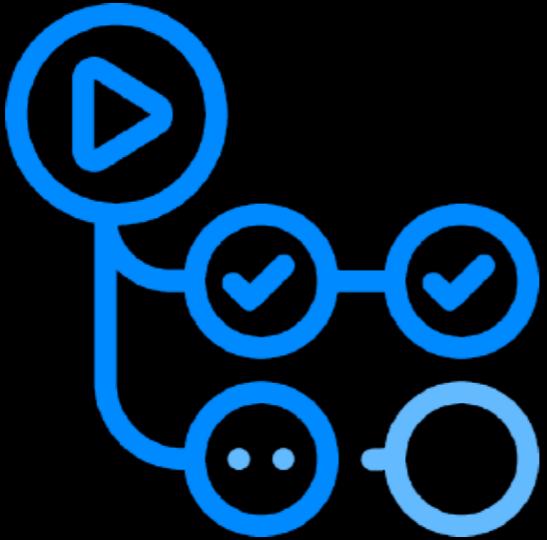
- Connecting through Power BI** ✉ 1
#14 by jimmyd7377 was closed on Mar 20, 2020
- Error when devtools::install_github("RamiKrispin/coronavirus")** ✉ 1
#13 by Danielchui was closed on Mar 20, 2020
- Error in Hubei data for 3/11/2020** ✉ 2
#12 by tcarleton was closed on Mar 13, 2020
- Country name in standard format** ✉ 12
#11 by shubhrampandey was closed on Jul 1, 2020
- 0 Case Vectors (non-essential)** ✉ 2
#10 by j4yr0u93 was closed on Mar 13, 2020
- How do u make it realtime and auto update** ✉ 3
#9 by navmedvideos was closed on Apr 25, 2020
- Negative values were found in the package** ✉ 7
#7 by ddong63 was closed on Mar 10, 2020
- Update package locally** ✉ 2
#6 by shubhrampandey was closed on Mar 13, 2020
- Negative case values** ✉ 2
#5 by j4yr0u93 was closed on Mar 10, 2020
- Covid-19** ✉ 3
#4 by acgerstein was closed on Mar 6, 2020
- Adding a country filter to the dashboard** ✉ 3
#3 by Agusum was closed on Mar 8, 2020
- " Azerbaijan" Has Space in the Name** ✉ 1
#2 by cannin was closed on Feb 29, 2020
- is there any plan to automate data update?** ✉ 16
#1 by statklee was closed on Apr 25, 2020



Actions + Docker

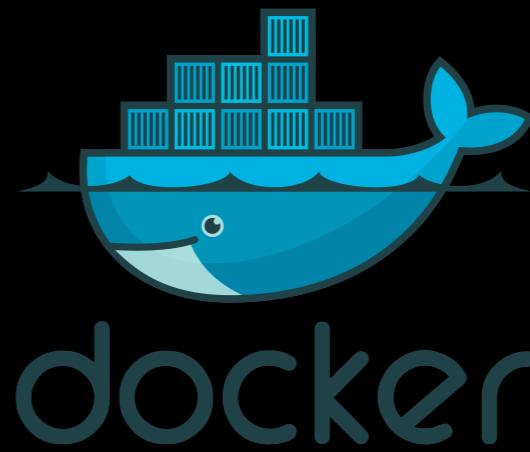


Actions + Docker



GitHub Actions

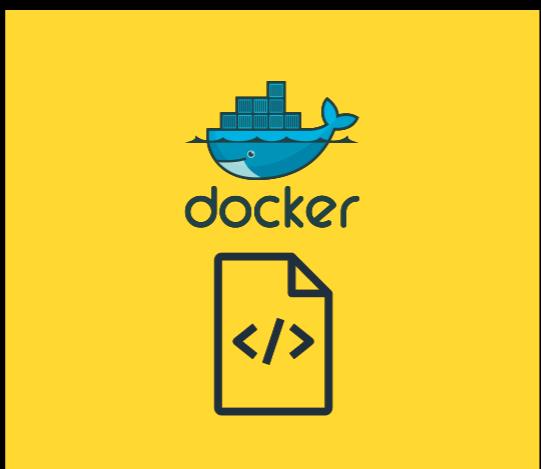
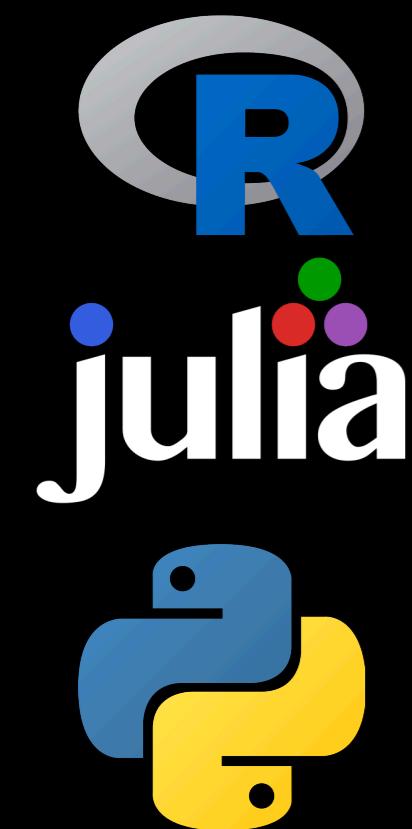
- Scheduler
- CI/CD



Docker

- Container
- CI/CD

Docker In A Nutshell

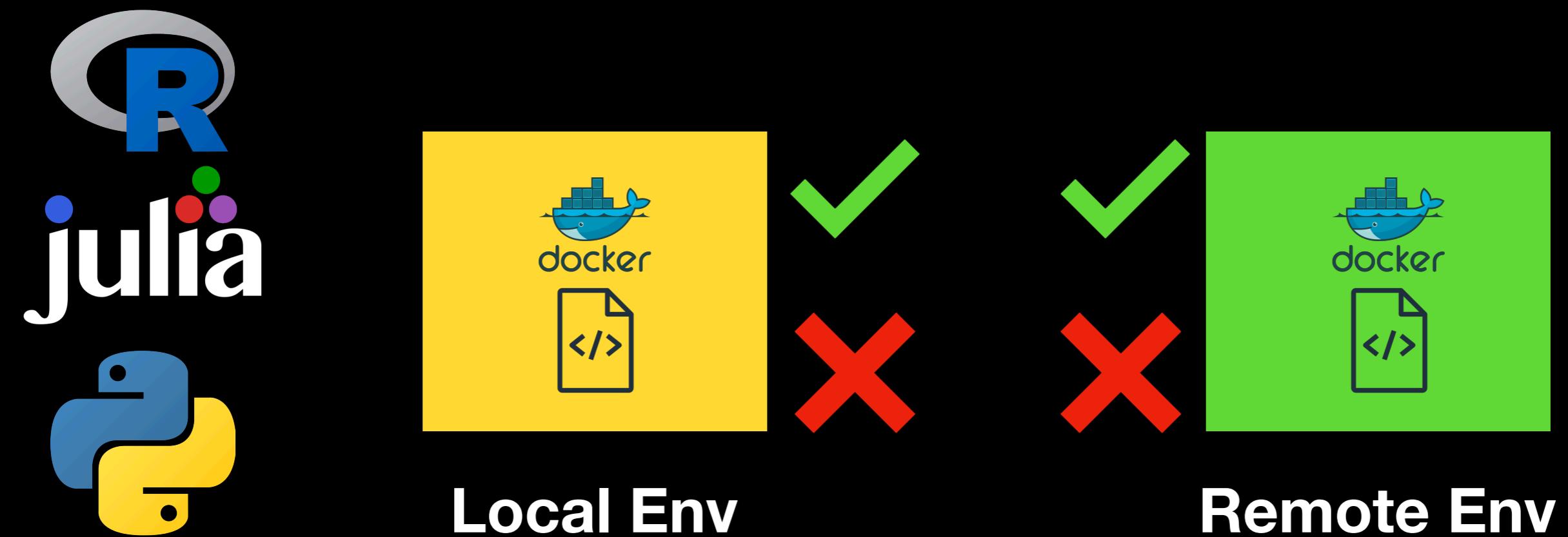


Local Env



Remote Env

Docker In A Nutshell



What Can You Build?

CI/CD & Automation

The screenshot shows the GitHub Actions interface for the repository `RamiKrispin / eia-poc`. The `Actions` tab is selected. On the left, a sidebar lists actions: `Data Refresh`, `pages-build-deployment`, `Management`, `Caches`, `Deployments`, and `Runners`. The main area displays the `All workflows` section with the heading `All workflow runs` and a count of `1,488 workflow runs`. A search bar at the top right says `Filter workflow runs`. The table lists workflow runs with columns for `Event`, `Status`, `Branch`, and `Actor`. The runs are categorized by workflow name, such as `pages build and deployment` and `Data Refresh`, and are timestamped from 30 minutes ago to 10 hours ago.

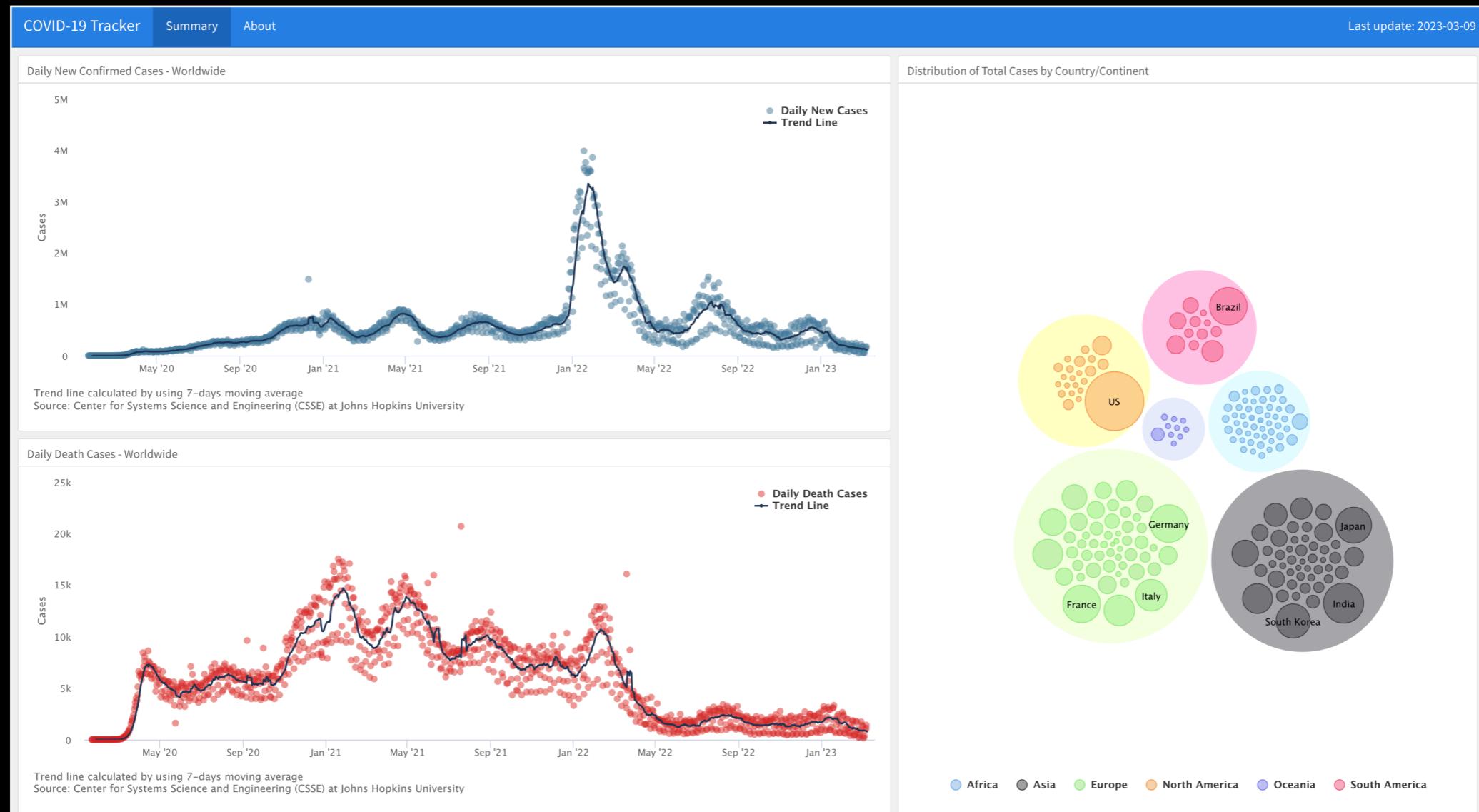
Event	Status	Branch	Actor
30 minutes ago	Success	main	...
32 minutes ago	Success	main	...
2 hours ago	Success	main	...
2 hours ago	Success	main	...
3 hours ago	Success	main	...
3 hours ago	Success	main	...
4 hours ago	Success	main	...
4 hours ago	Success	main	...
10 hours ago	Success	main	...
10 hours ago	Success	main	...

Code Development and Testing

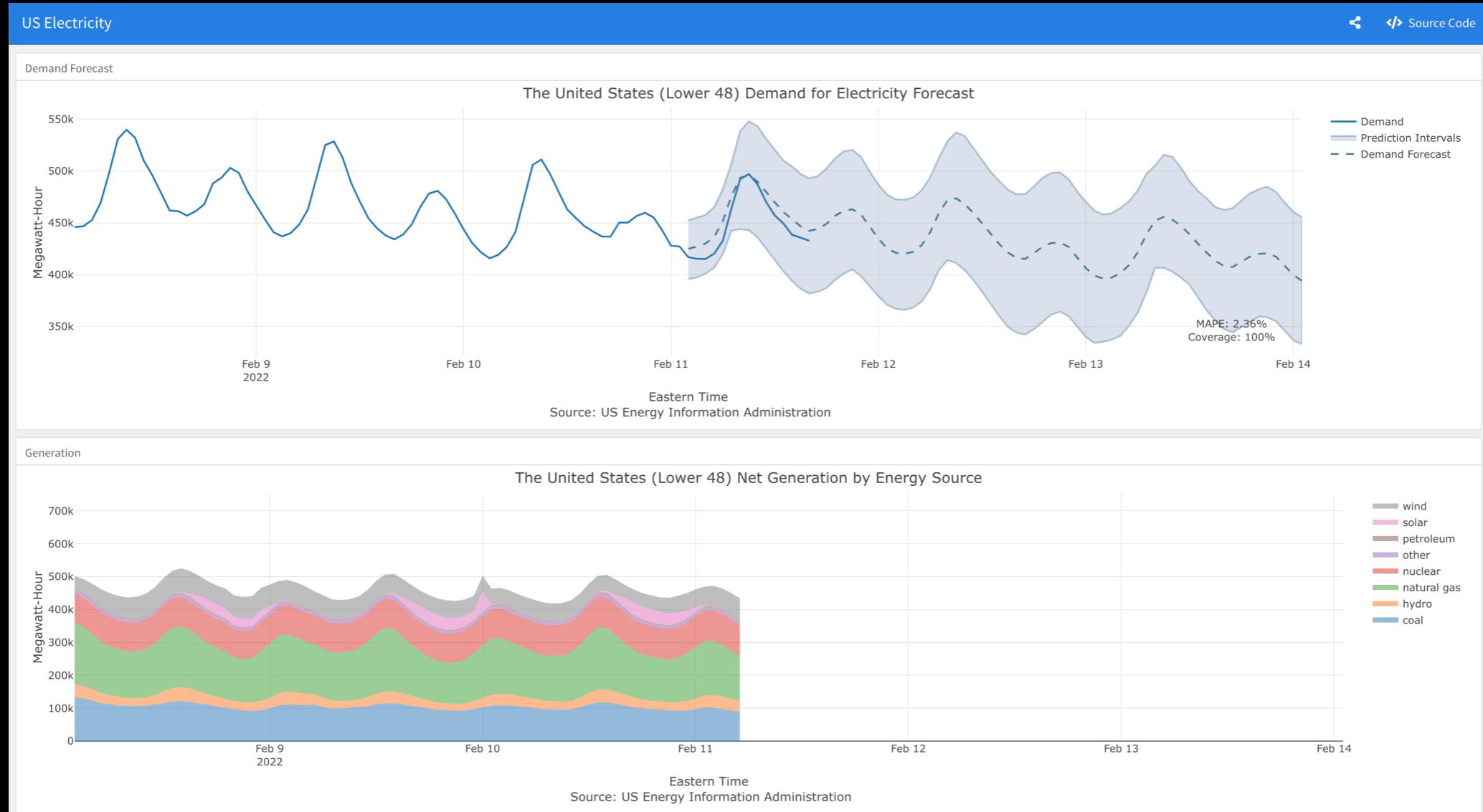
The screenshot shows a GitHub repository interface for the 'coronavirus' repository owned by 'RamiKrispin'. The 'Code' tab is selected. On the left, the file tree shows the directory structure, including '.github/workflows/main.yml' which is currently selected. The main pane displays the content of 'main.yml':

```
1  on: [push, pull_request]
2
3  name: R CMD
4  jobs:
5    R-CMD-check:
6      name: R CMD check
7      runs-on: ubuntu-18.04
8      container:
9        image: docker.io/rkrispin/coronavirus:prod.0.3.31
10     steps:
11       - name: checkout_repo
12         uses: actions/checkout@v2
13       - name: Check
14         run: Rscript -e "rcmdcheck::rcmdcheck(args = '--no-manual', error_on = 'error')"
```

Data Automation



ML Applications

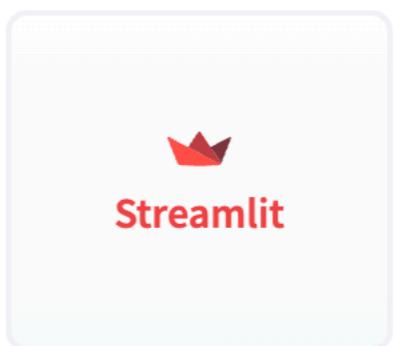


Hugging Face

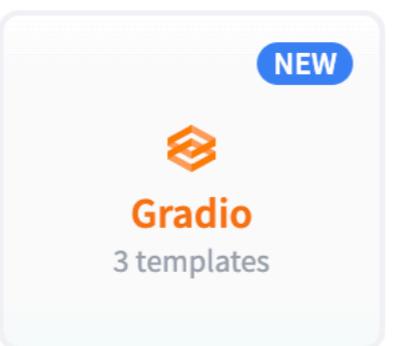


Select the Space SDK

You can choose between Streamlit, Gradio and Static for your Space. Or pick Docker to host any other app.



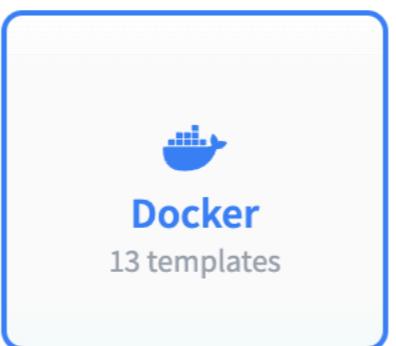
Streamlit



Gradio

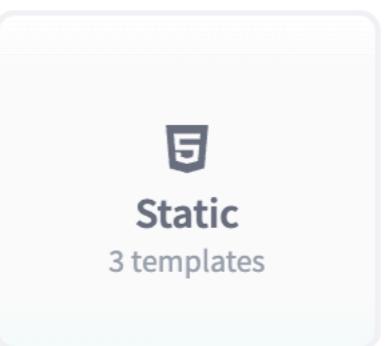
3 templates

NEW



Docker

13 templates



Static

3 templates

Choose a Docker template:



Blank



JupyterLab



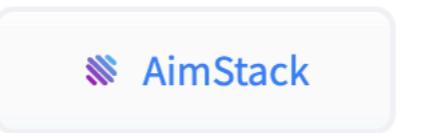
Argilla



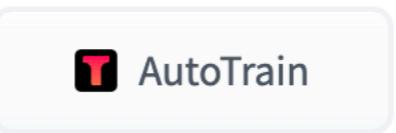
Livebook



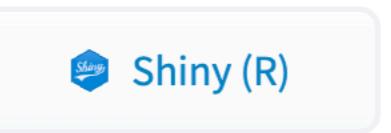
LabelStudio



AimStack



AutoTrain



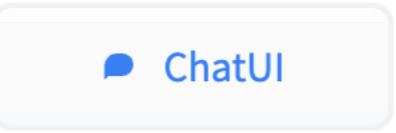
Shiny (R)



Shiny (Python)



ZenML



ChatUI



Panel



Giskard



Quarto

Open and Free To Use

Limitations

Limitations

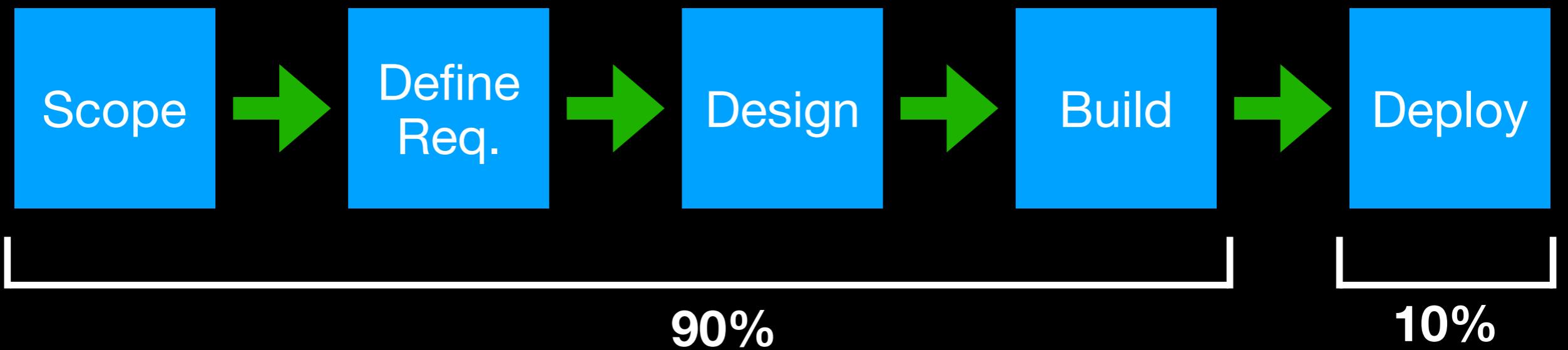
- Not secure
- Compute power
- Availability
- Data storage

Deploy and Monitor ML and Data Pipelines with GitHub Actions

Building Pipeline - Workflow



Building Pipeline - Workflow

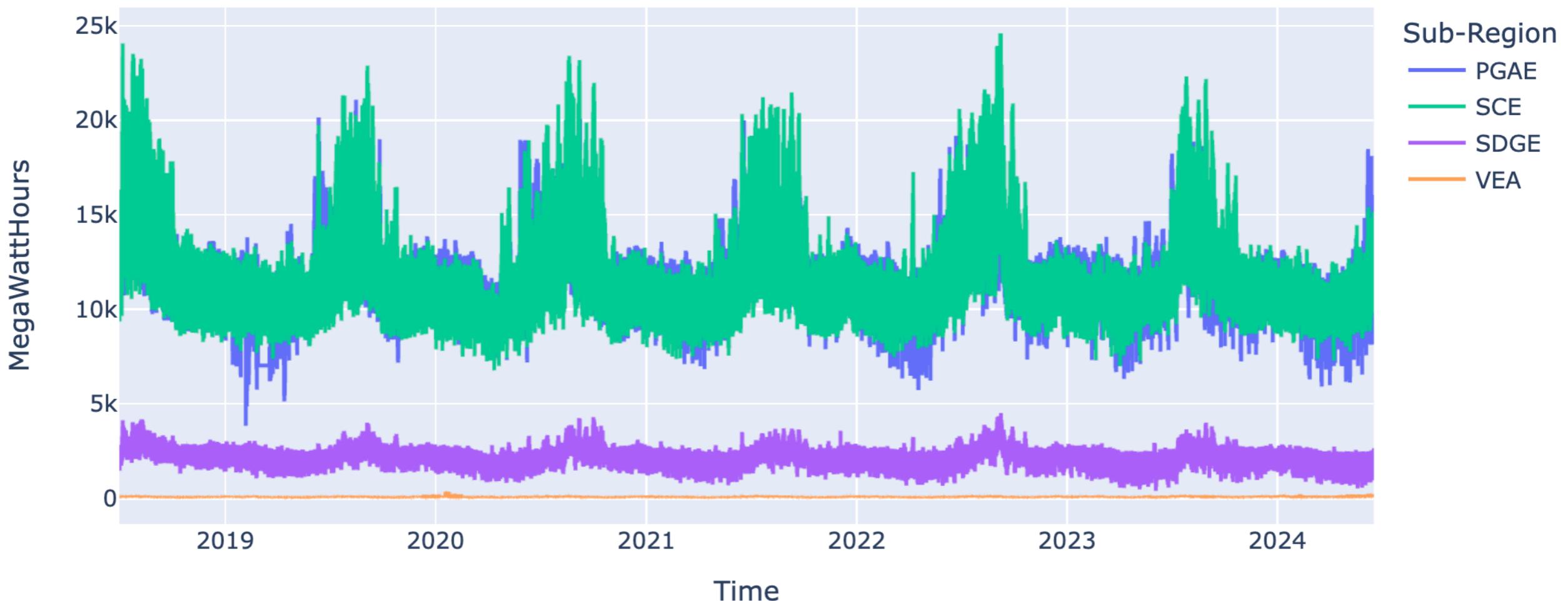


Scope

- Forecast the hourly demand for electricity in California by sub-region
- This includes the following four providers:
 - Pacific Gas and Electric (PGAE)
 - Southern California Edison (SCE)
 - San Diego Gas and Electric (SDGE)
 - Valley Electric Association (VEA)
- Refresh the data every hour and the forecast every 24 hours

Scope

California Hourly Demand By Operating Provider



Requirements

- Environment - Docker image
- Orchestration - GitHub Actions
- Data pipeline - keep the data up-to-date
- ML pipeline - refresh the forecast
- Dashboard - GitHub Pages

Data Source - EIA API

The screenshot shows the official website of the U.S. Energy Information Administration (EIA). The header features the EIA logo and the tagline "Independent Statistics and Analysis U.S. Energy Information Administration". Navigation links include "+ Tools", "+ Learn About Energy", "+ News", "+ Sources & Uses", "+ Topics", and "+ Geography". A search bar with the placeholder "Search eia.gov" and a magnifying glass icon is also present. To the right of the search bar are social media icons for X and Facebook.

Short-Term Energy Outlook June 2024
Energy projections for supply, demand, and prices >

What's New

- Refinery Capacity Report**
Jun 14, 2024
- Wholesale Electricity Market Data**
Jun 13, 2024
- Short-Term Energy Outlook**
Jun 11, 2024
- [More >](#)

Today in Energy > Posted June 18, 2024

U.S. refinery capacity increased 2% in 2023 >

Operable atmospheric crude oil distillation capacity, our primary measure of refinery capacity in the United States, increased by 2%, or 324,000 barrels per day (b/d), in 2023, according to our recently published *Refinery Capacity Report*. [More >](#)

U.S. refinery atmospheric distillation capacity on Jan 1 (2017–2024)
million barrels per day

Year	Capacity (million barrels per day)
2017	18.0
2018	18.0
2019	18.2
2020	18.2
2021	17.8
2022	17.8
2023	17.8
2024	18.0

idle operating

Data Highlights

- WTI crude oil futures price**
6/17/2024: NA/barrel
NA from week earlier
NA from year earlier
- Natural gas futures price**
6/17/2024: NA/MMBtu
NA from week earlier
NA from year earlier
- Retail gasoline price**
6/17/2024: \$3.435/gal
▲ \$0.006 from week earlier
▼ \$0.142 from year earlier
- Retail diesel price**
6/17/2024: \$3.735/gal
▲ \$0.077 from week earlier
▼ \$0.080 from year earlier
- Weekly coal production**

Data Source - EIA API



+ Sources & Uses | + Topics | + Geography

Search eia.gov

SUBMIT RESET

API URL: `https://api.eia.gov/v2/electricity/rto/region-sub-ba-data/data/?frequency=hourly&data[0]=value&facets[parent][]=CISO&facets[subba][]=PGAE&facets[subba][]=SCE&facets[subba][]=SDGE&facets[subba][]=VEA&sort[0][column]=period&sort[0][direction]=desc&offset=0&length=5000`

METHOD: GET

SERIES DESCRIPTION: Hourly demand by balancing authority subregion. Source: Form EIA-930
Product: Hourly Electric Grid Monitor

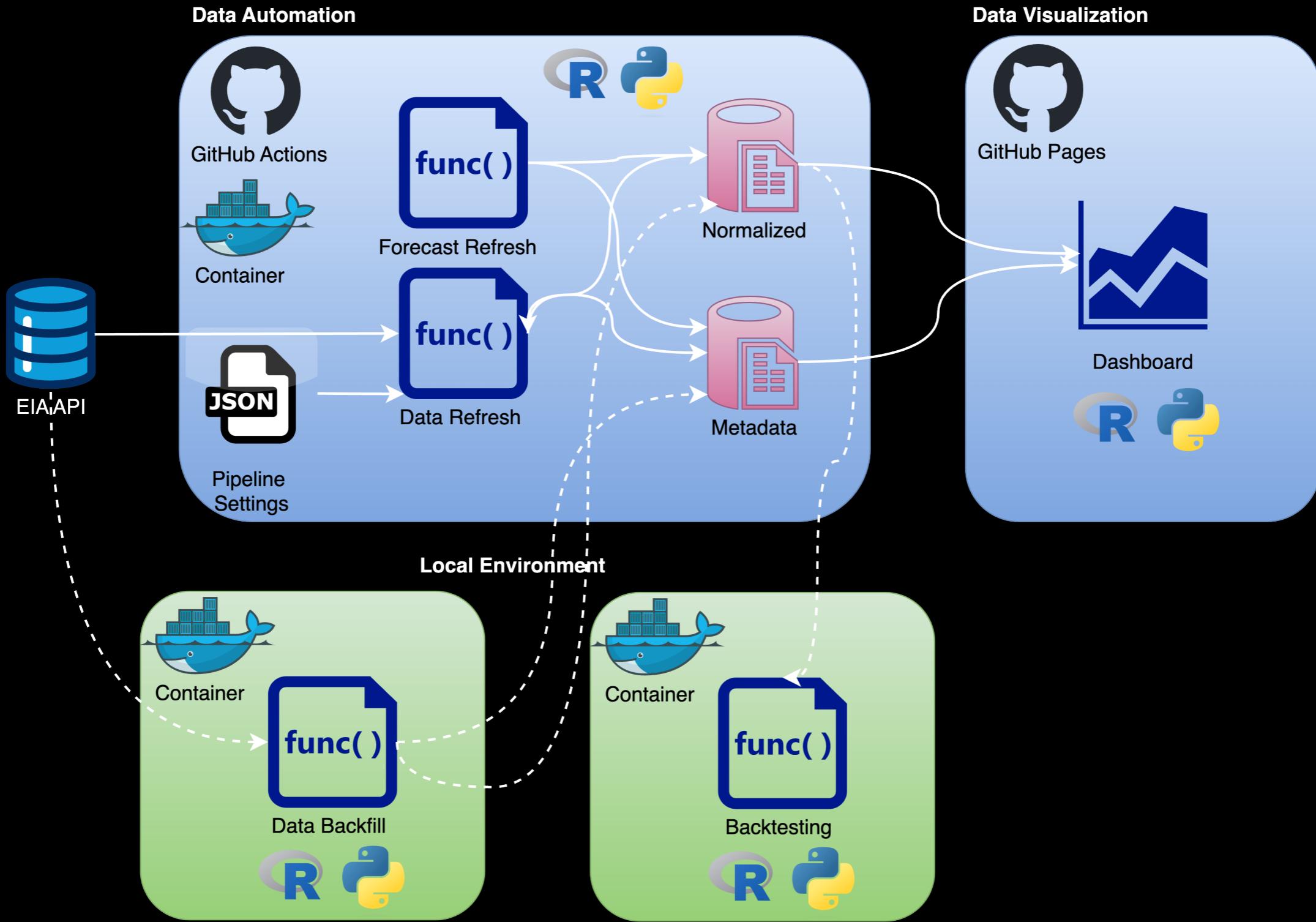
Need help using our API? Read our [API Documentation](#) to learn more.

HEADER:

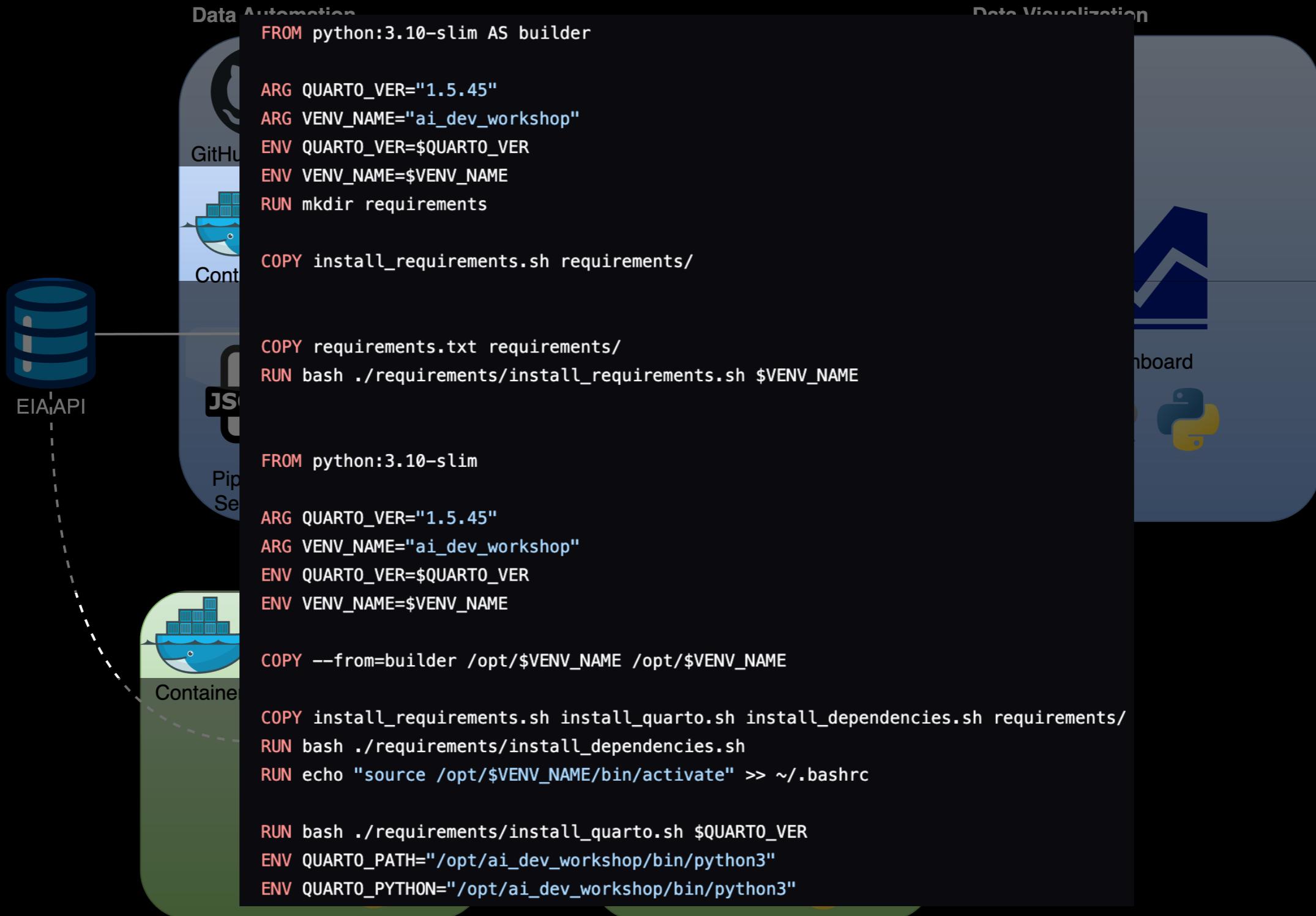
```
X-Params: {  
    "frequency": "hourly",  
    "data": [  
        "value"  
    ],  
    "facets": {  
        "parent": [  
            "CISO"  
        ],  
        "subba": [  
            "PGAE",  
            "SCE",  
            "SDGE",  
            "VEA"  
        ]  
    },  
    "start": null,  
    "end": null,  
    "sort": [  
        {  
            "column": "period",  
            "direction": "desc"  
        }  
    ],  
    "offset": 0,  
    "length": 5000  
}
```

Design

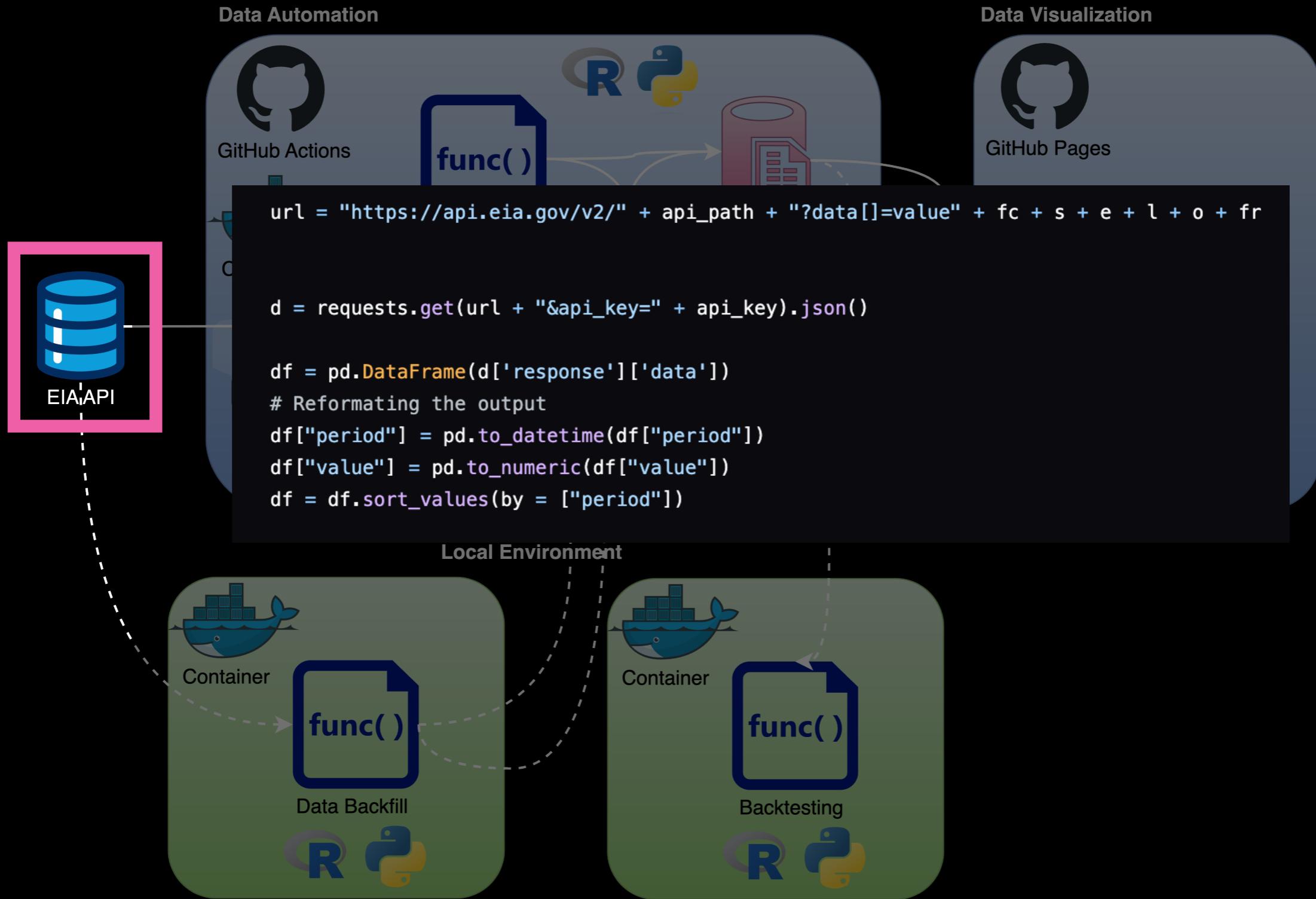
Pipeline Design



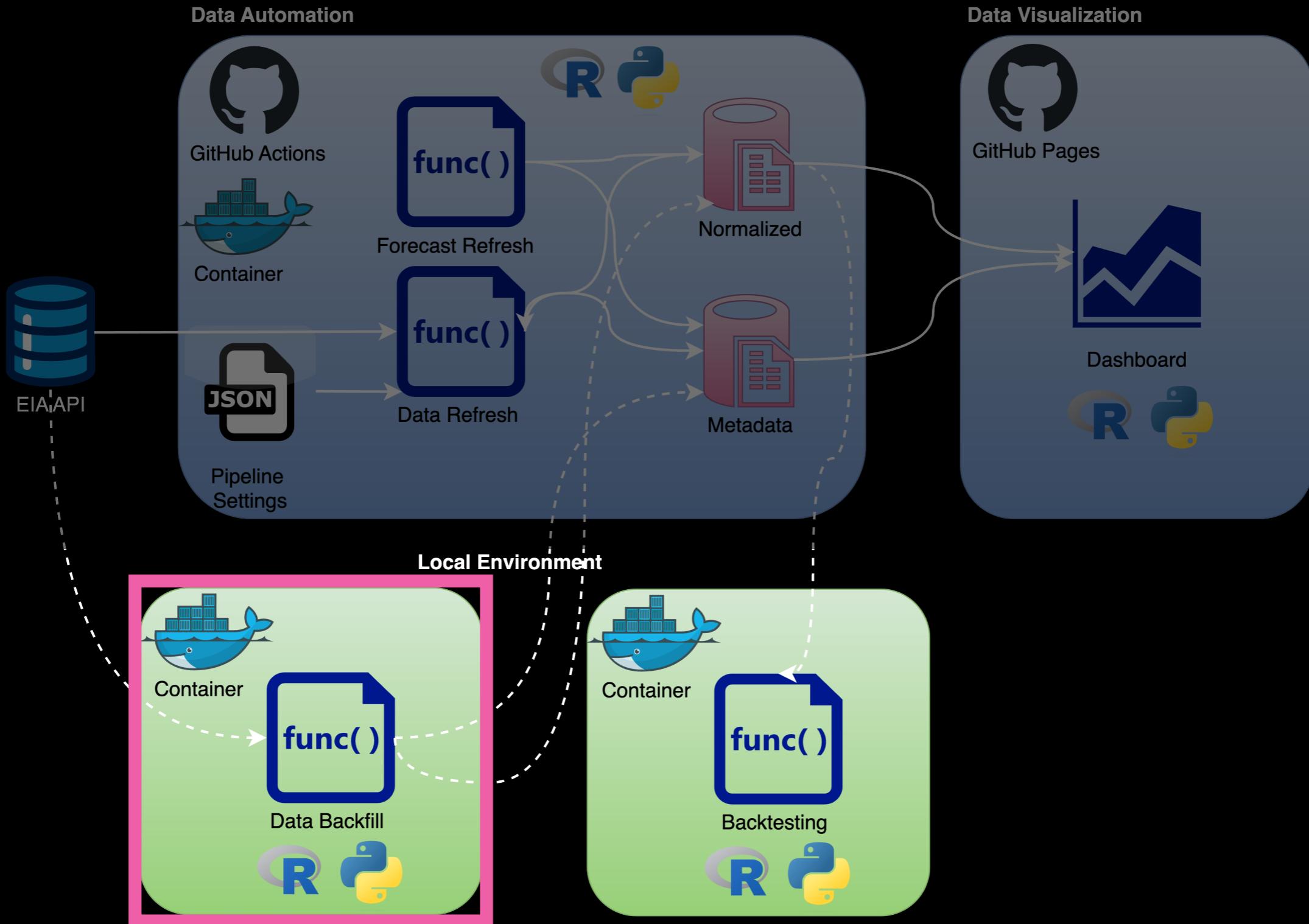
Pipeline Design



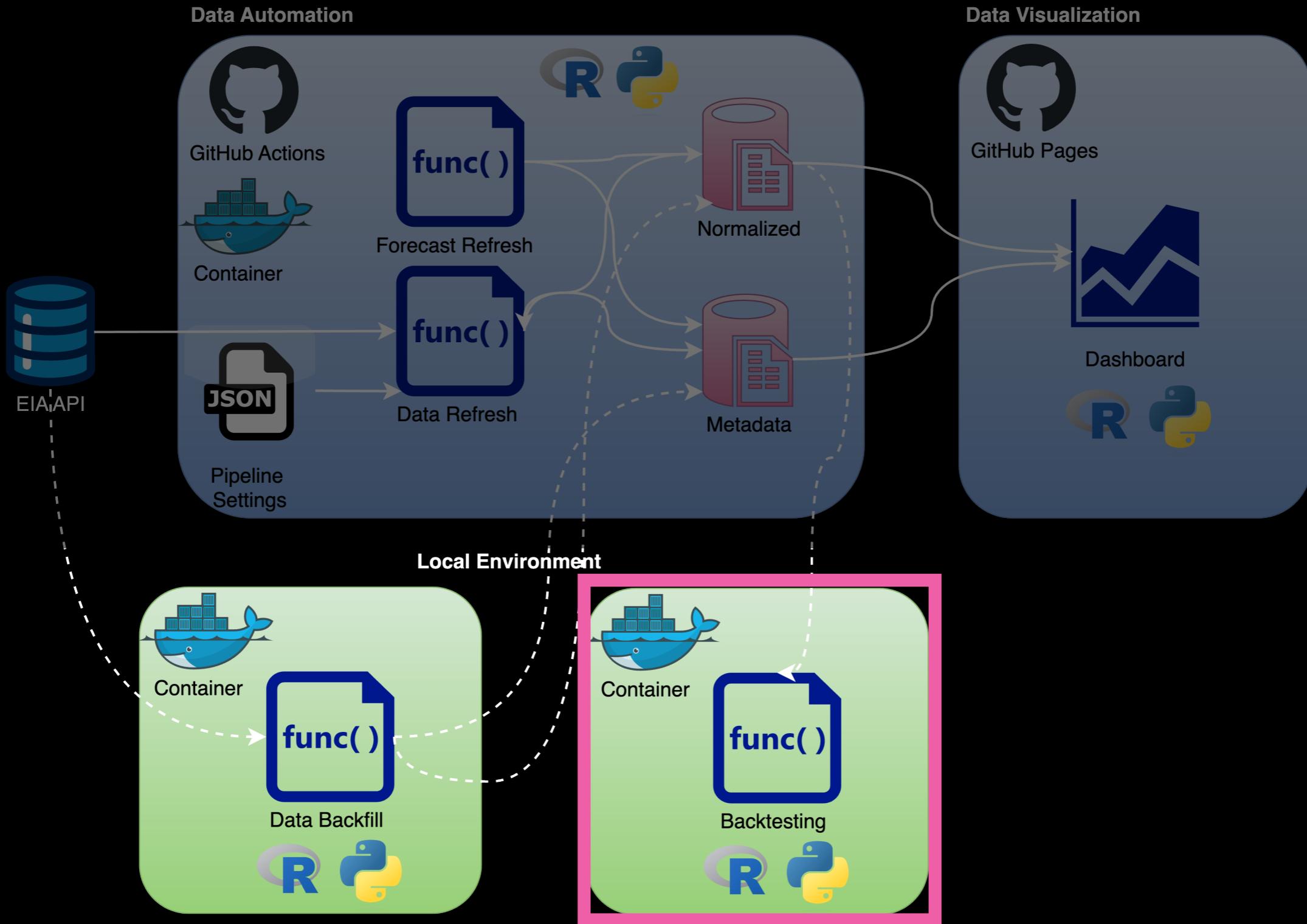
Pipeline Design



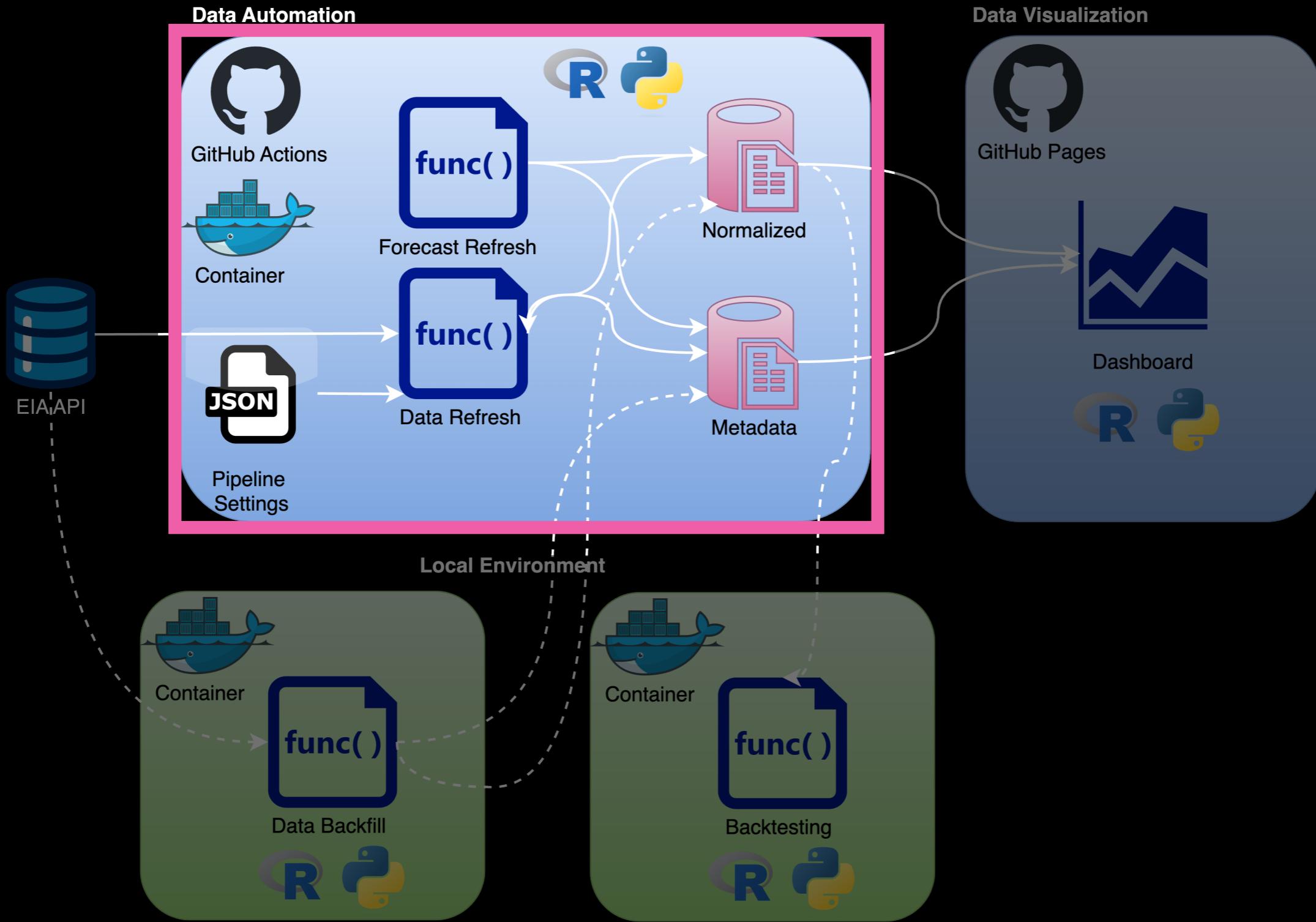
Pipeline Design



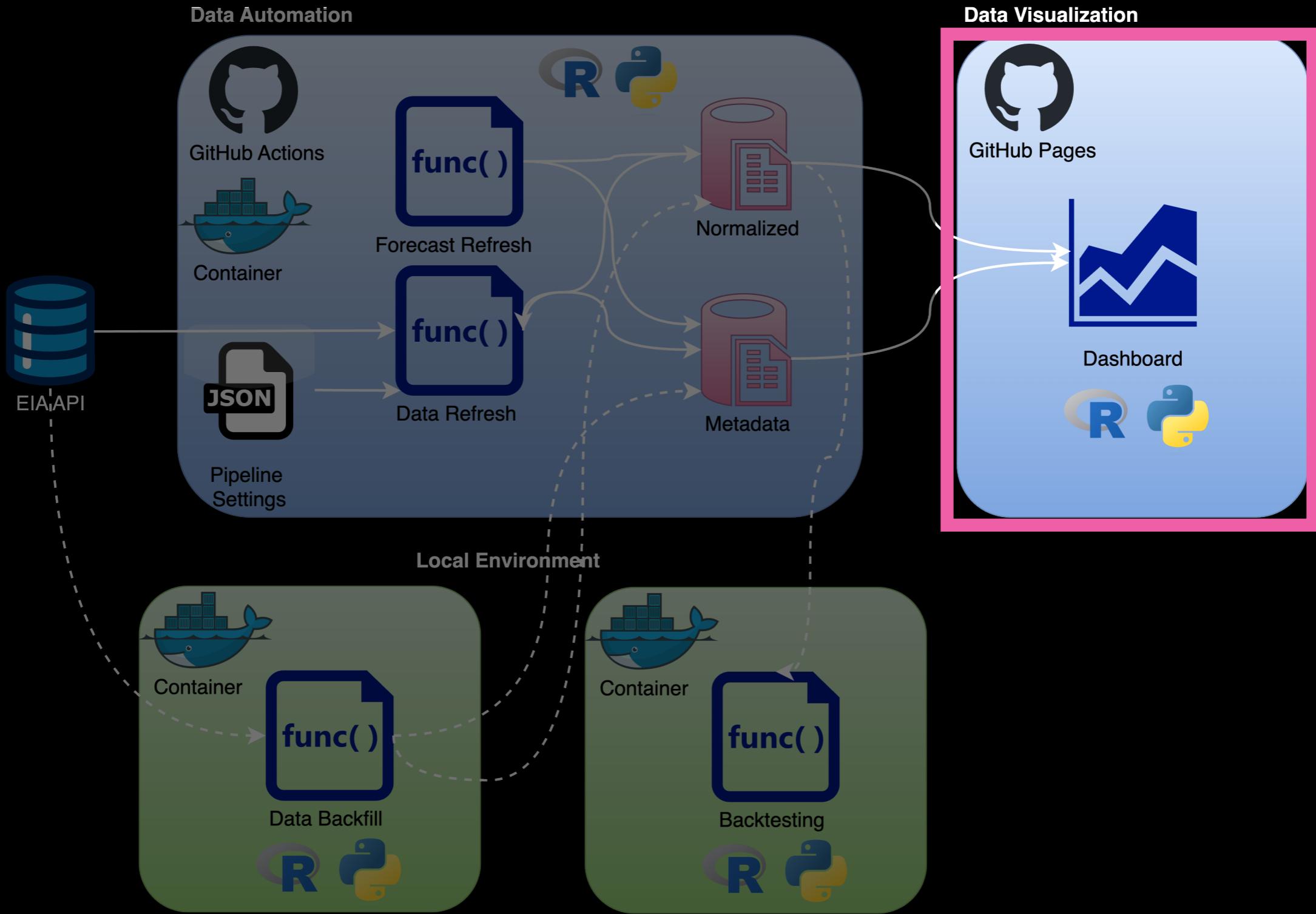
Pipeline Design



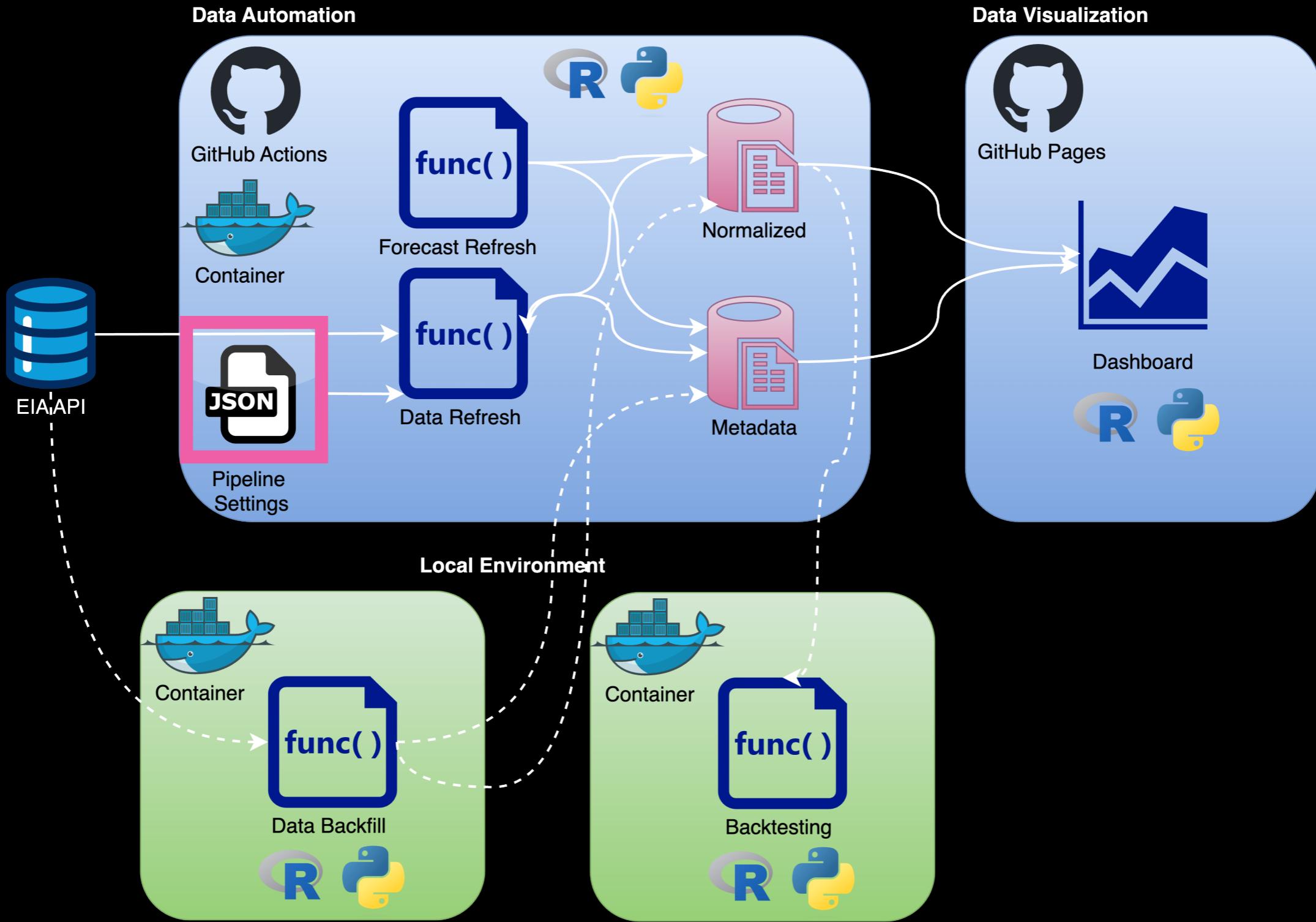
Pipeline Design



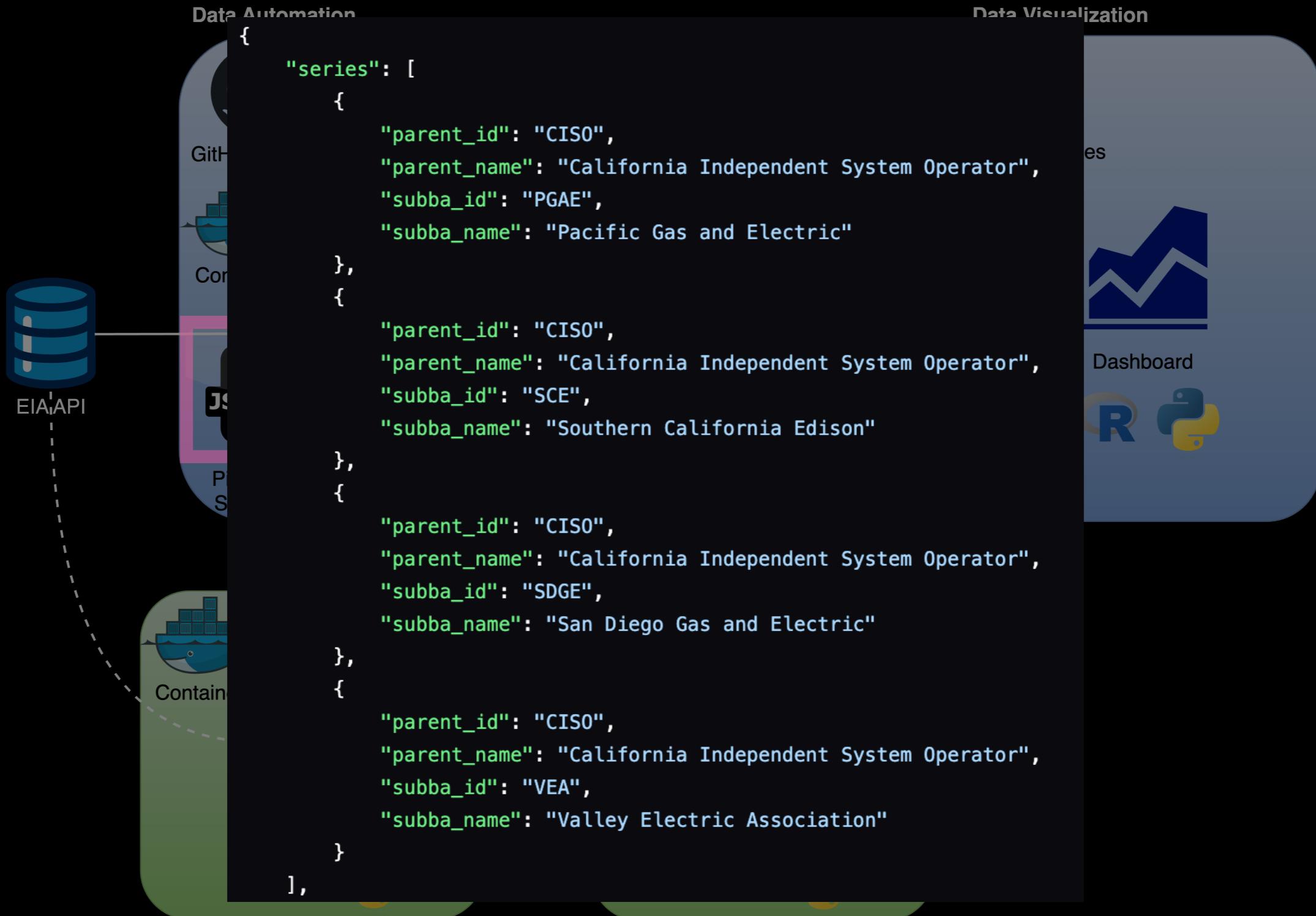
Pipeline Design



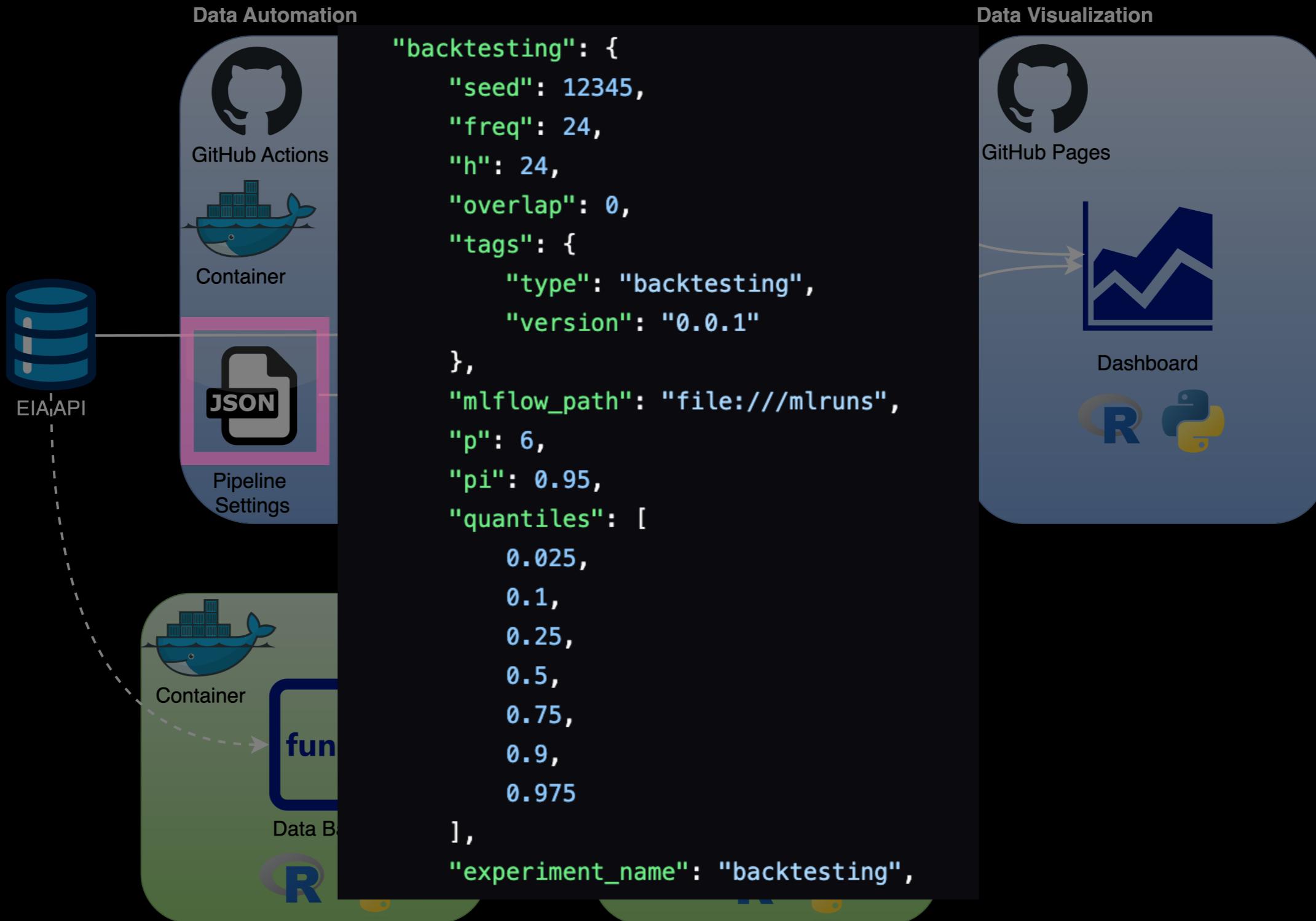
Pipeline Design



Pipeline Design



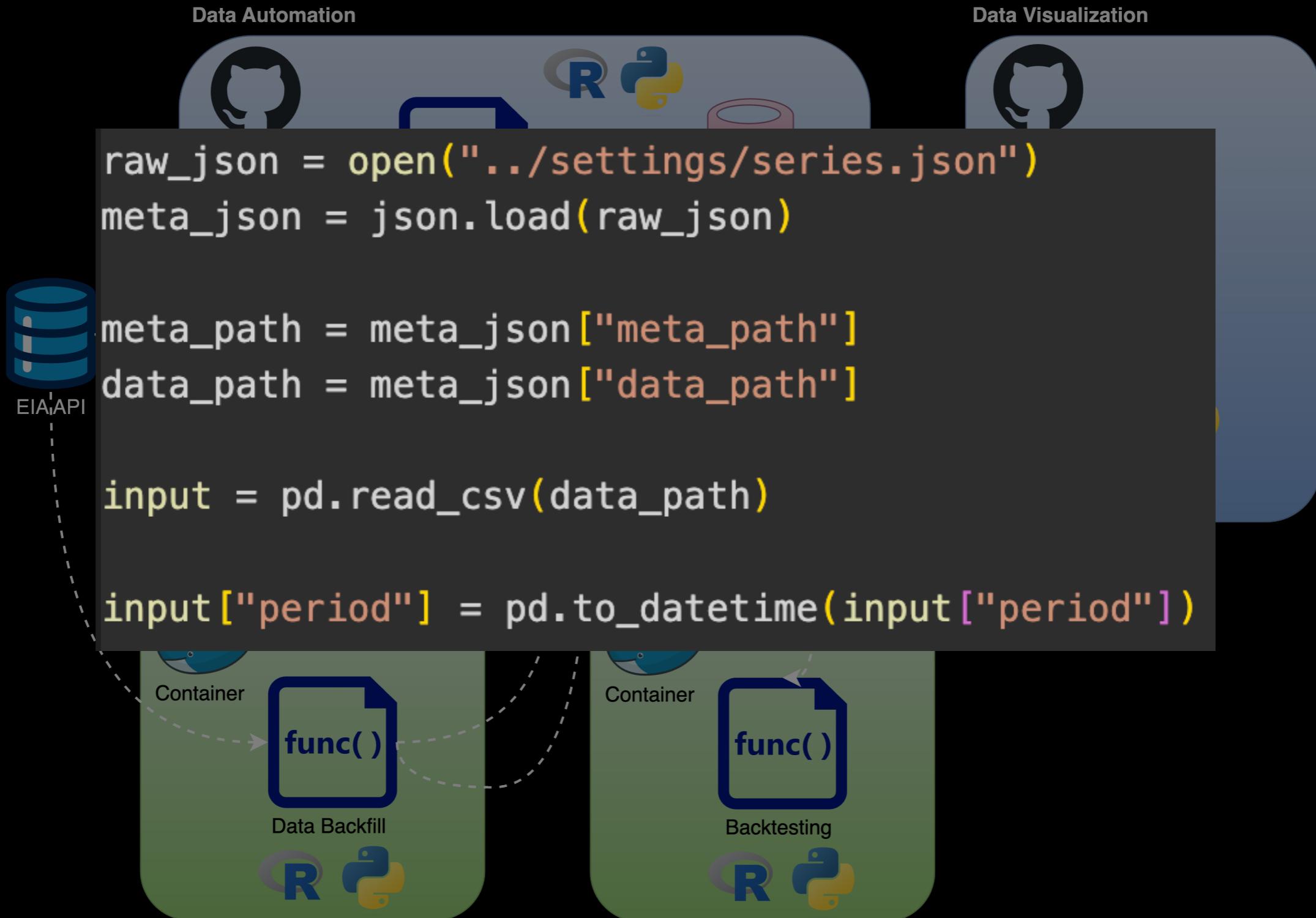
Pipeline Design



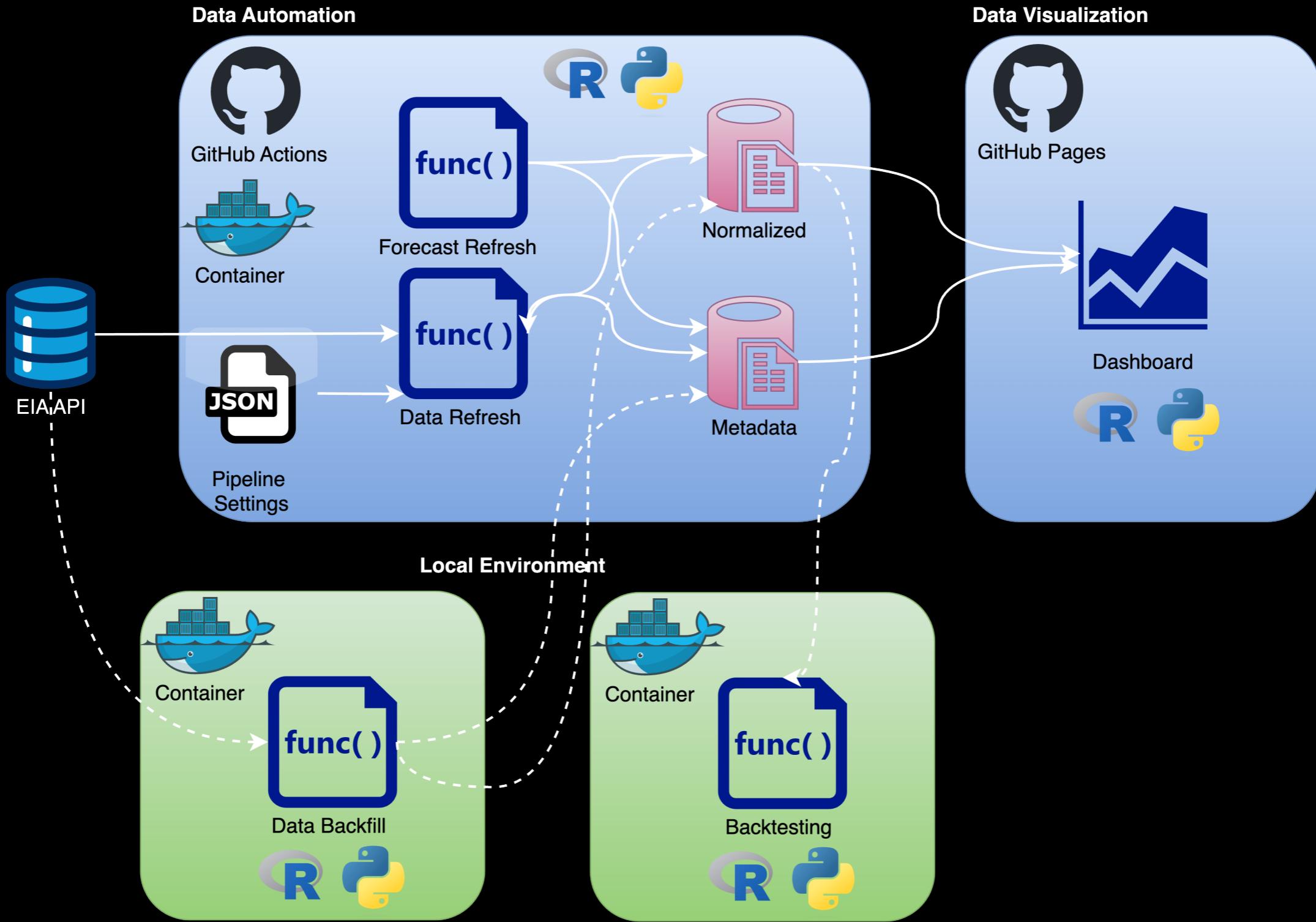
Pipeline Design



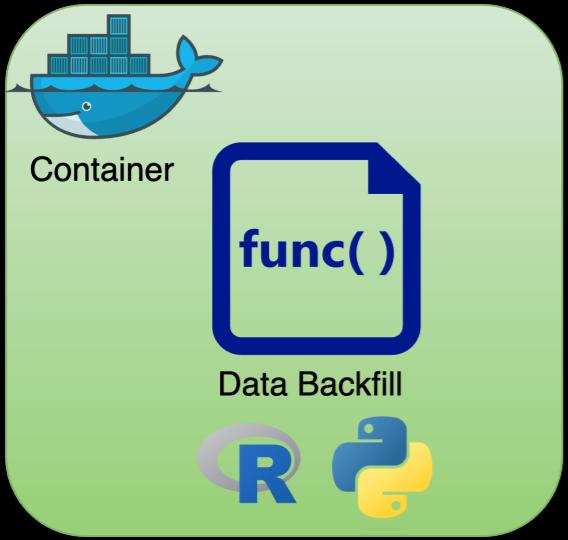
Pipeline Design



Pipeline Design



Pipeline Design



- Initiate the pipeline
- Large dataset
- Use Quarto
- Data profiling

Data Backfill

The goal of the backfill process is to pull the historical data for the required series using the settings.json file. This includes the following steps:

- Setting parameters and pulling the data
- Data quality checks
- Saving the data and creating a log file

Load Libraries

▼ Code

```
import eia_api as api
import eia_data
import pandas as pd
import numpy as np
import requests
import json
import os
import datetime
import plotly.express as px
```

▼ Code

```
raw_json = open("../settings/series.json")
meta_json = json.load(raw_json)
series = pd.DataFrame(meta_json["series"])
api_path = meta_json["api_path"]
```

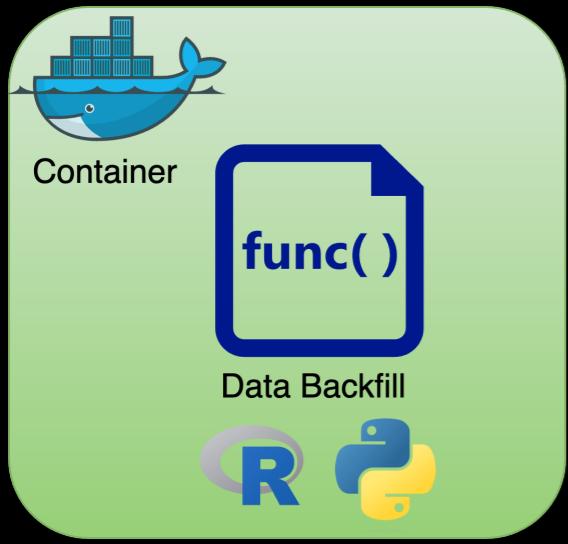
▼ Code

```
facets_template = {
    "parent" : None,
    "subba" : None
}

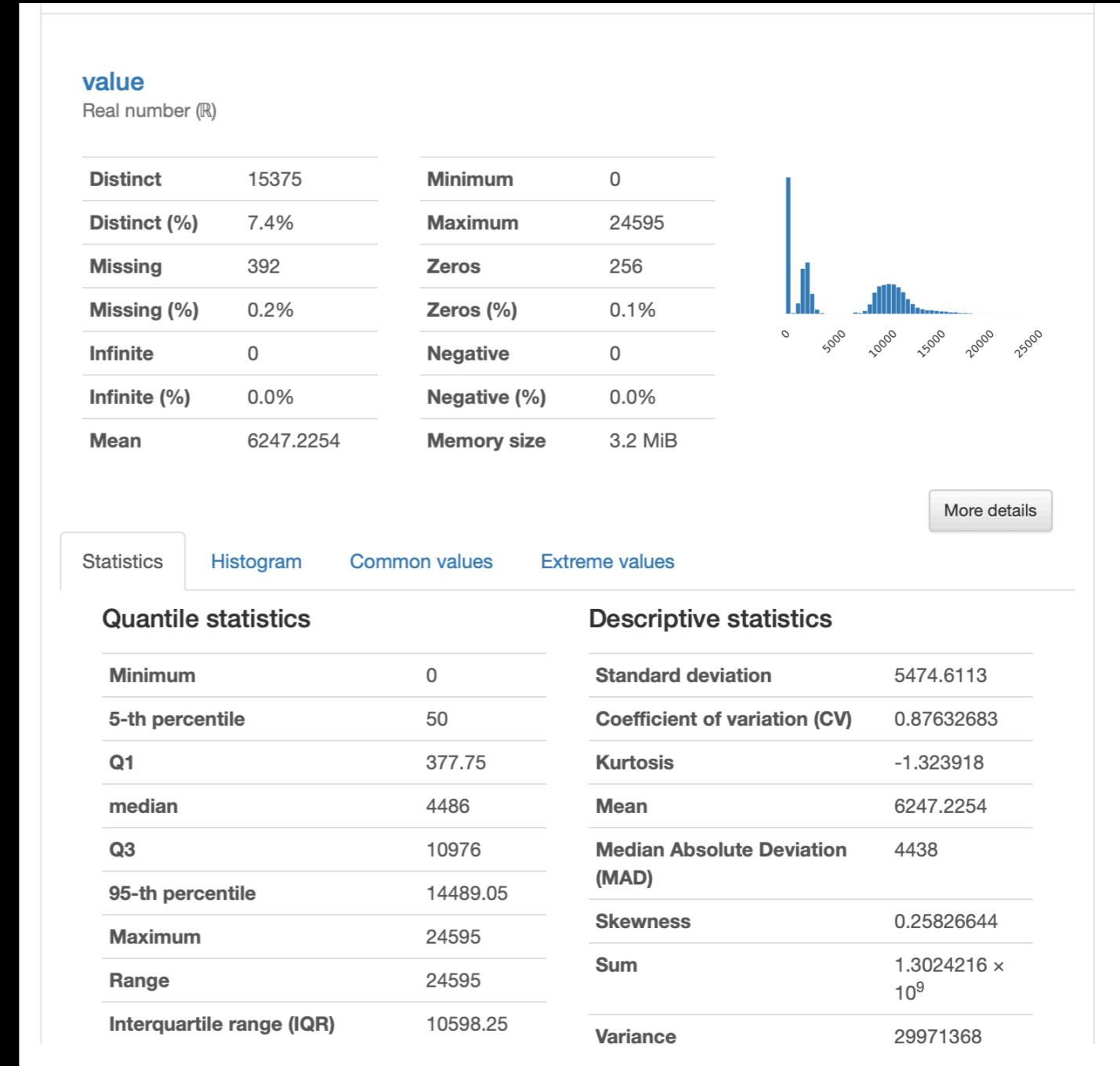
start = datetime.datetime(meta_json["start"]["year"],
    meta_json["start"]["month"],
    meta_json["start"]["day"],
    meta_json["start"]["hour"])

end = datetime.datetime(meta_json["end"]["year"],
```

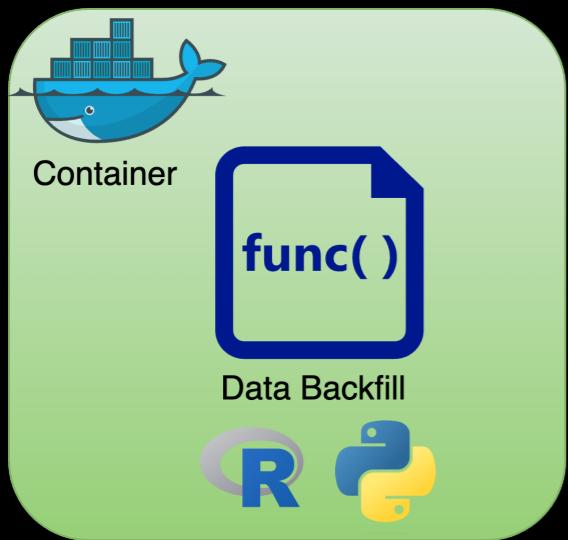
Pipeline Design



- Initiate the pipeline
- Large dataset
- Use Quarto
- Data profiling



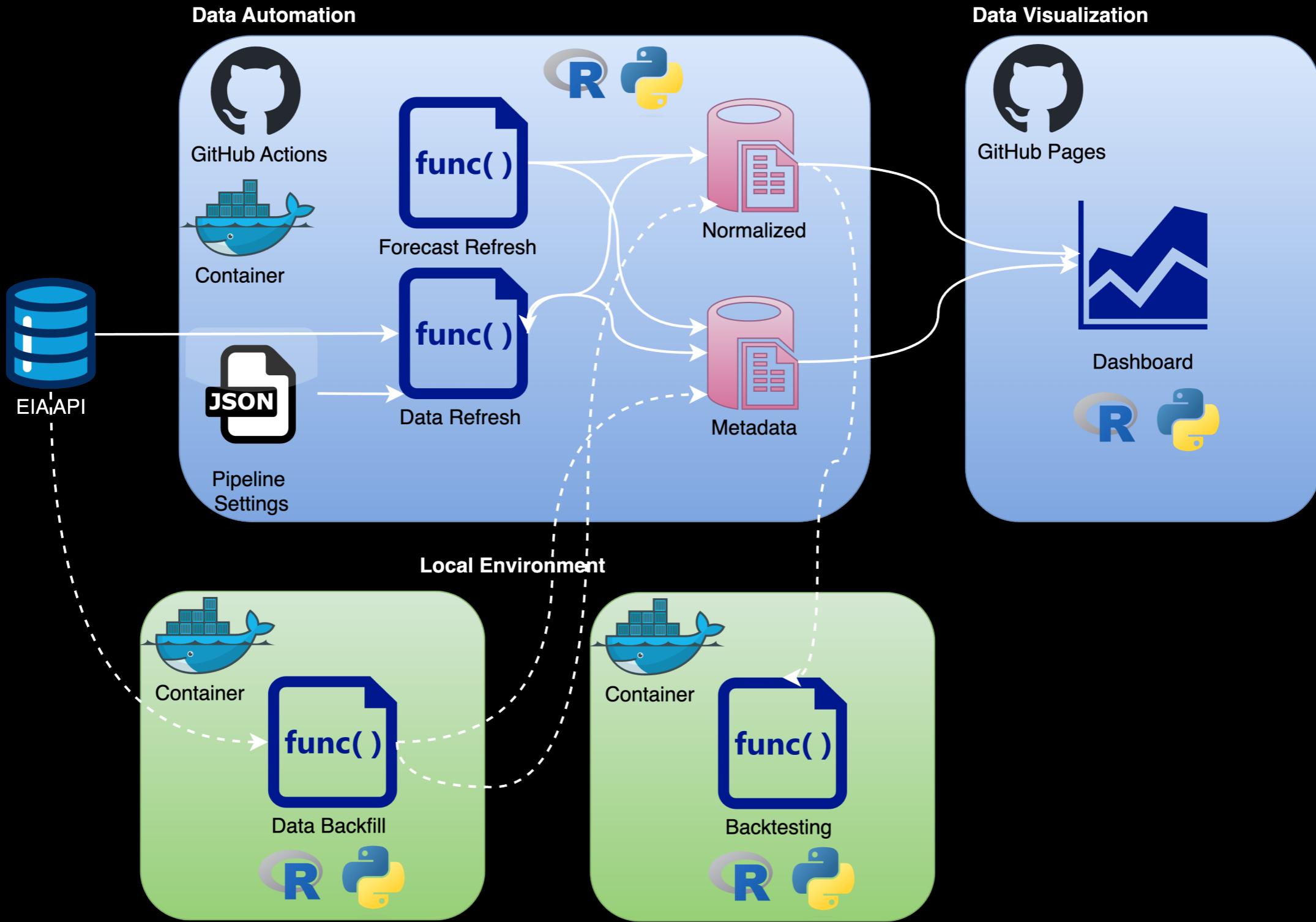
Pipeline Design



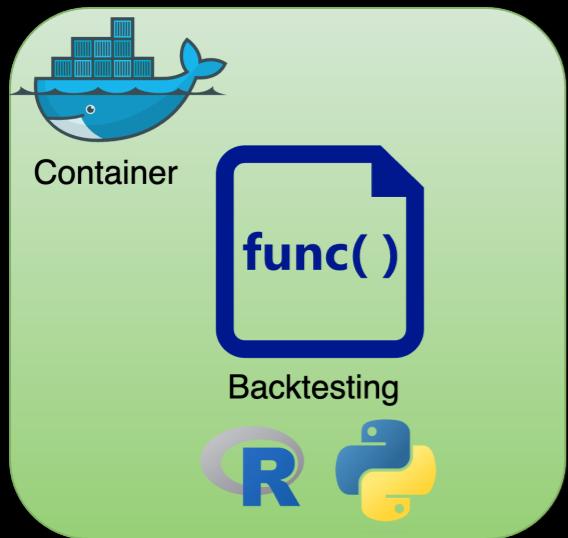
- Initiate the pipeline
- Large dataset
- Use Quarto
- Data profiling

```
functions > eia_api.py > ...
1  import pandas as pd
2  import datetime
3  import requests
4
5  def day_offset(start, end, offset):
6      current = [start]
7      while max(current) < end:
8          if(max(current) + datetime.timedelta(days= offset) < end):
9              current.append(max(current) + datetime.timedelta(days= offset))
10         else:
11             current.append(end)
12
13     return current
14
15  def hour_offset(start, end, offset):
16      current = [start]
17      while max(current) < end:
18          if(max(current) + datetime.timedelta(hours = offset) < end):
19              current.append(max(current) + datetime.timedelta(hours = offset))
20          else:
21              current.append(end)
22
23      return current
24
25  def eia_get(api_key,
26             api_path,
27             data = "value",
28             facets = None,
29             start = None,
30             end = None,
31             length = None,
32             offset = None,
33             frequency = None):
```

Pipeline Design



Pipeline Design



Model Backtesting

▶ Code

Load data

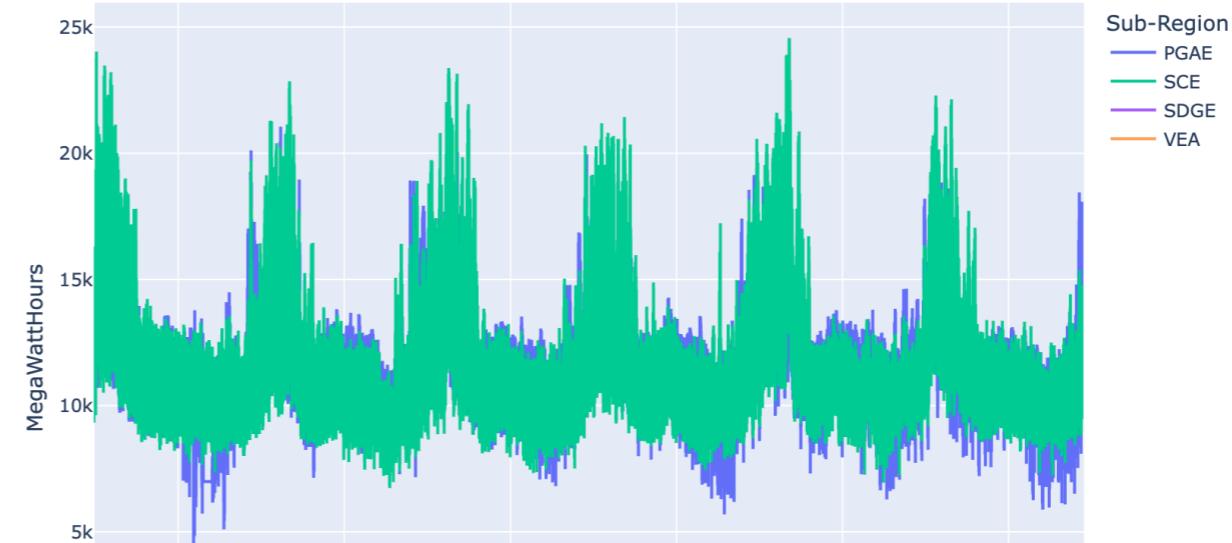
▶ Code

2024-06-14 23:00:00

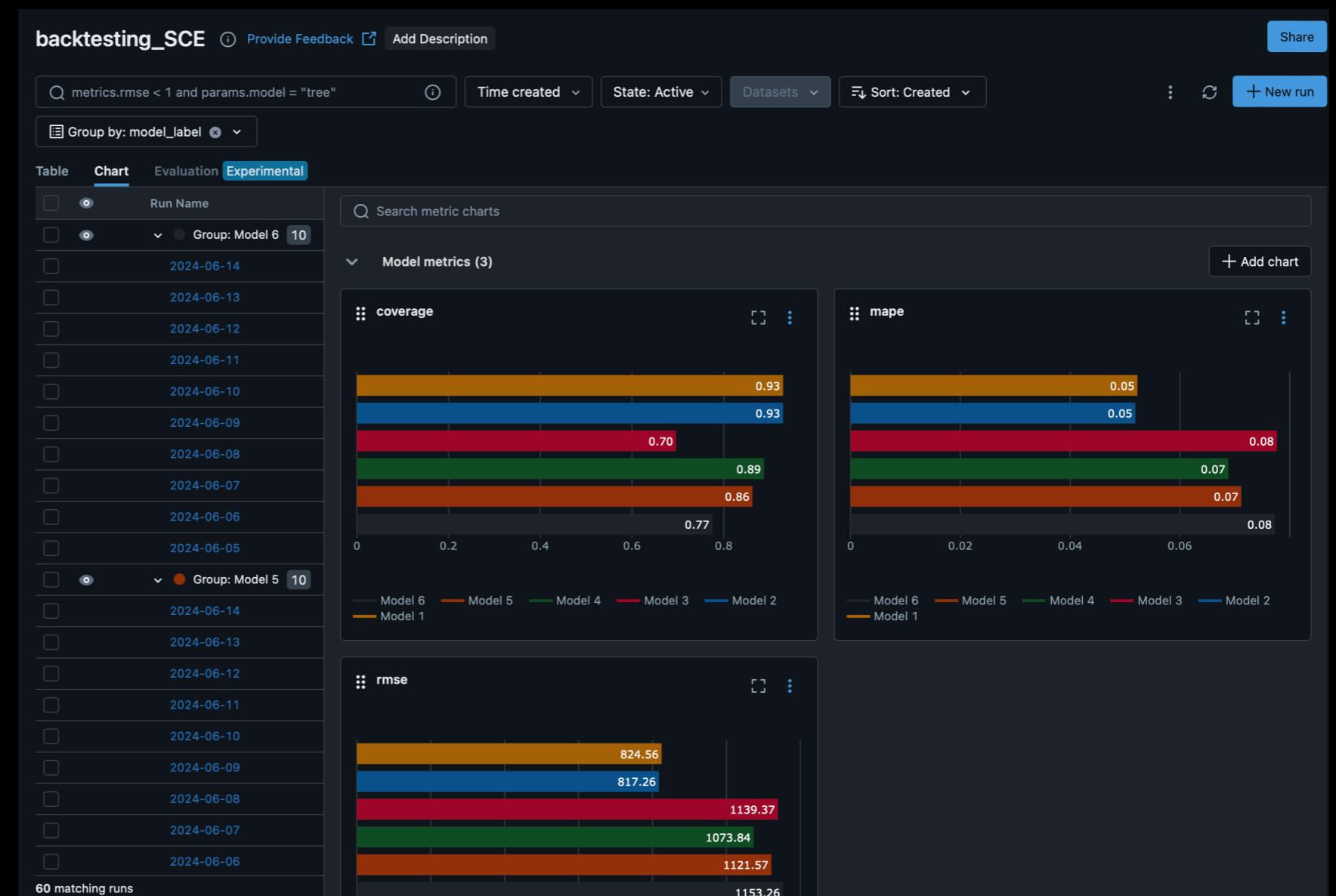
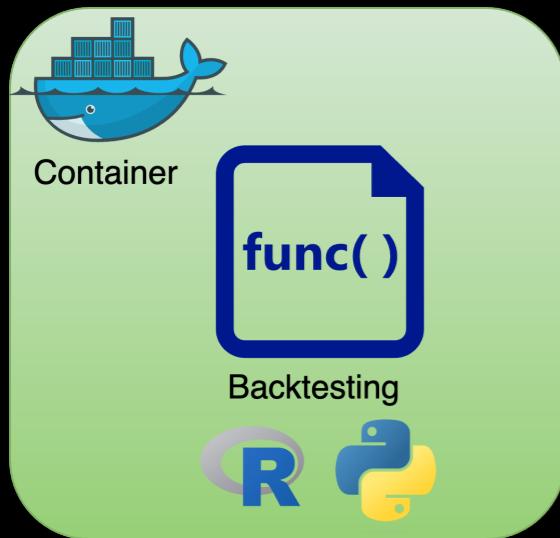
period	subba	subba-name	parent	parent-name	value	value-units
0	2018-07-01 08:00:00	PGAE	Pacific Gas and Electric	CISO	California Independent System Operator	12522.0 megawatthours
1	2018-07-01 09:00:00	PGAE	Pacific Gas and Electric	CISO	California Independent System Operator	11745.0 megawatthours
2	2018-07-01 10:00:00	PGAE	Pacific Gas and Electric	CISO	California Independent System Operator	11200.0 megawatthours
3	2018-07-01 11:00:00	PGAE	Pacific Gas and Electric	CISO	California Independent System Operator	10822.0 megawatthours
4	2018-07-01 12:00:00	PGAE	Pacific Gas and Electric	CISO	California Independent System Operator	10644.0 megawatthours

▶ Code

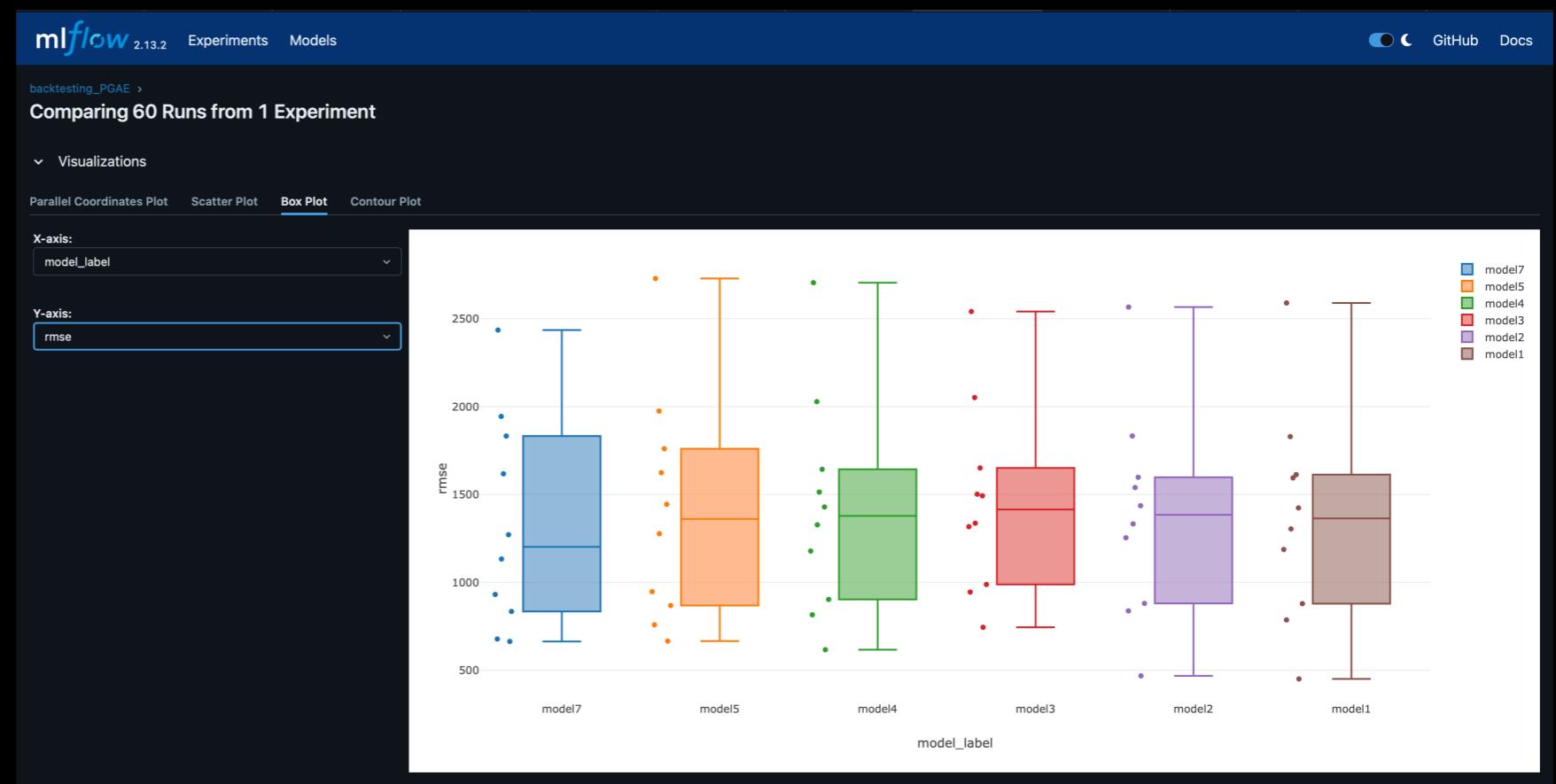
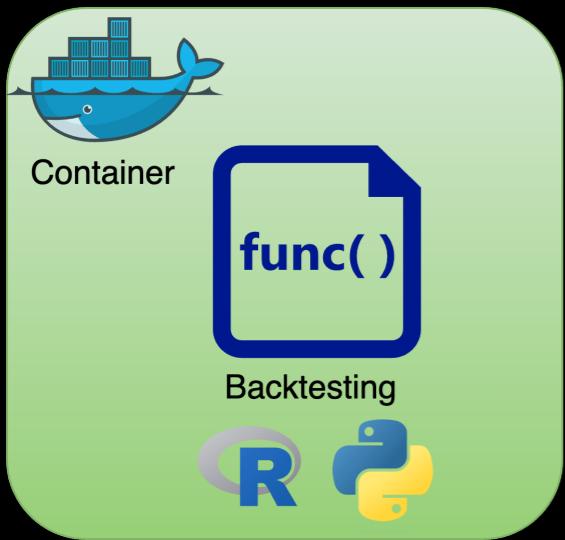
California Hourly Demand By Operating Provider



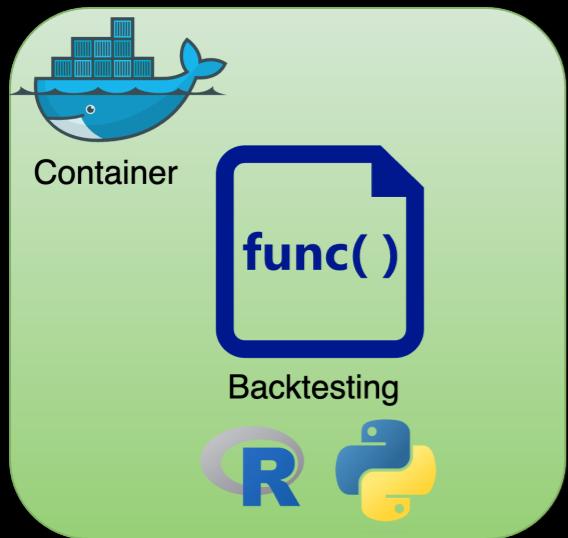
Pipeline Design



Pipeline Design

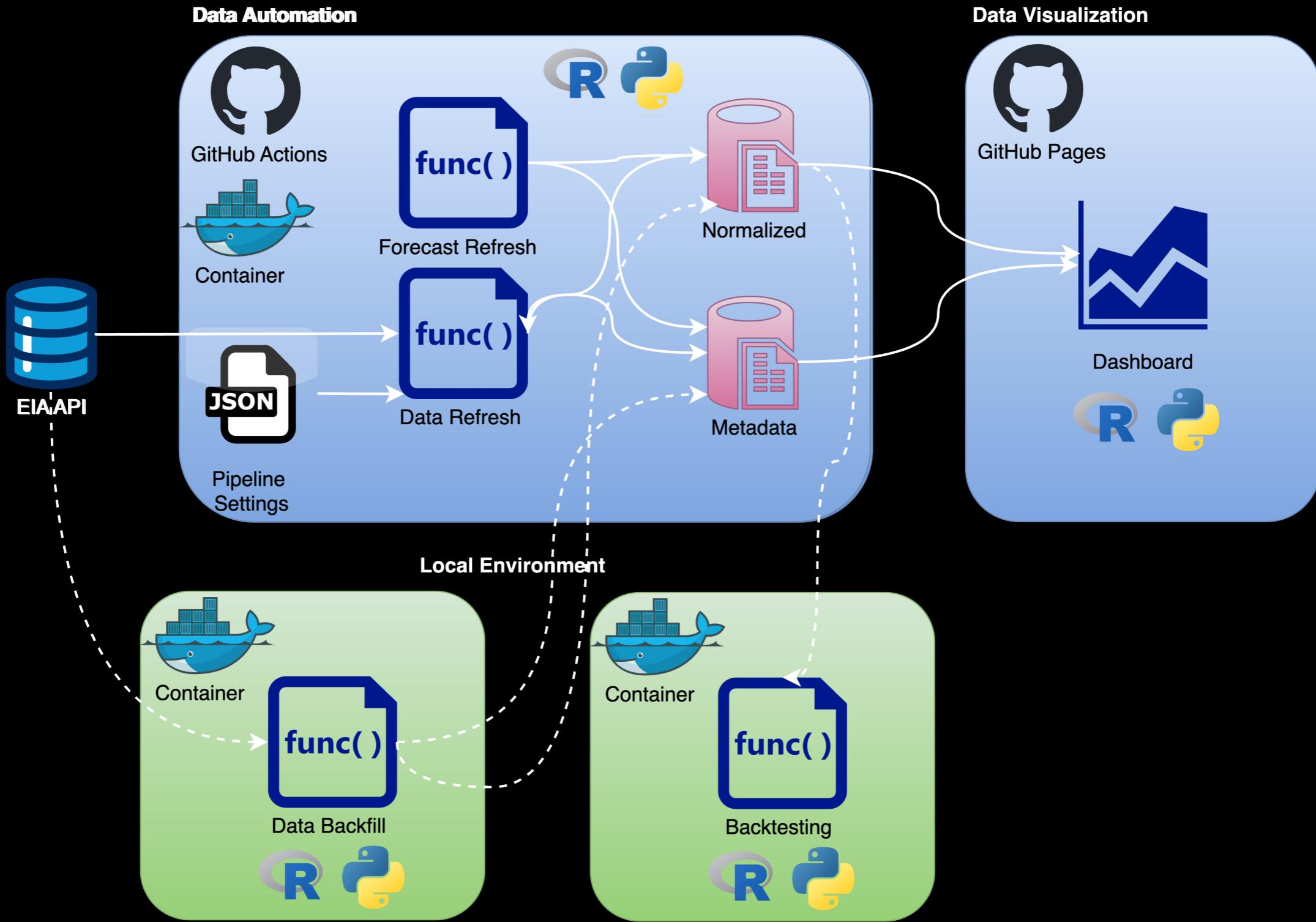


Pipeline Design



	forecast_label	model_label	model	mape	rmse	coverage	comments	subba
0	2024-06-14	model1	LinearRegressionModel	0.089334	1365.539586	0.725000	LM model with lags, training with 2 years of h...	PGAE
1	2024-06-14	model2	LinearRegressionModel	0.051878	817.262735	0.929167	LM model with lags, training with 3 years of h...	SCE
1	2024-06-14	model2	LinearRegressionModel	0.083141	188.657150	0.933333	LM model with lags, training with 3 years of h...	SDGE
0	2024-06-14	model1	LinearRegressionModel	0.100615	18.513913	0.887500	LM model with lags, training with 2 years of h...	VEA

Pipeline Design



Pipeline Design

RamiKrispin Auto update of the data ✓ 14f259e · 22 minutes ago ⏱ History

Preview Code Blame 361 lines (361 loc) · 63.3 KB ⚡ Code 55% faster with GitHub Copilot Raw ⌂ ⌄ ⌅ ⌆ ⌇

Q Search this file

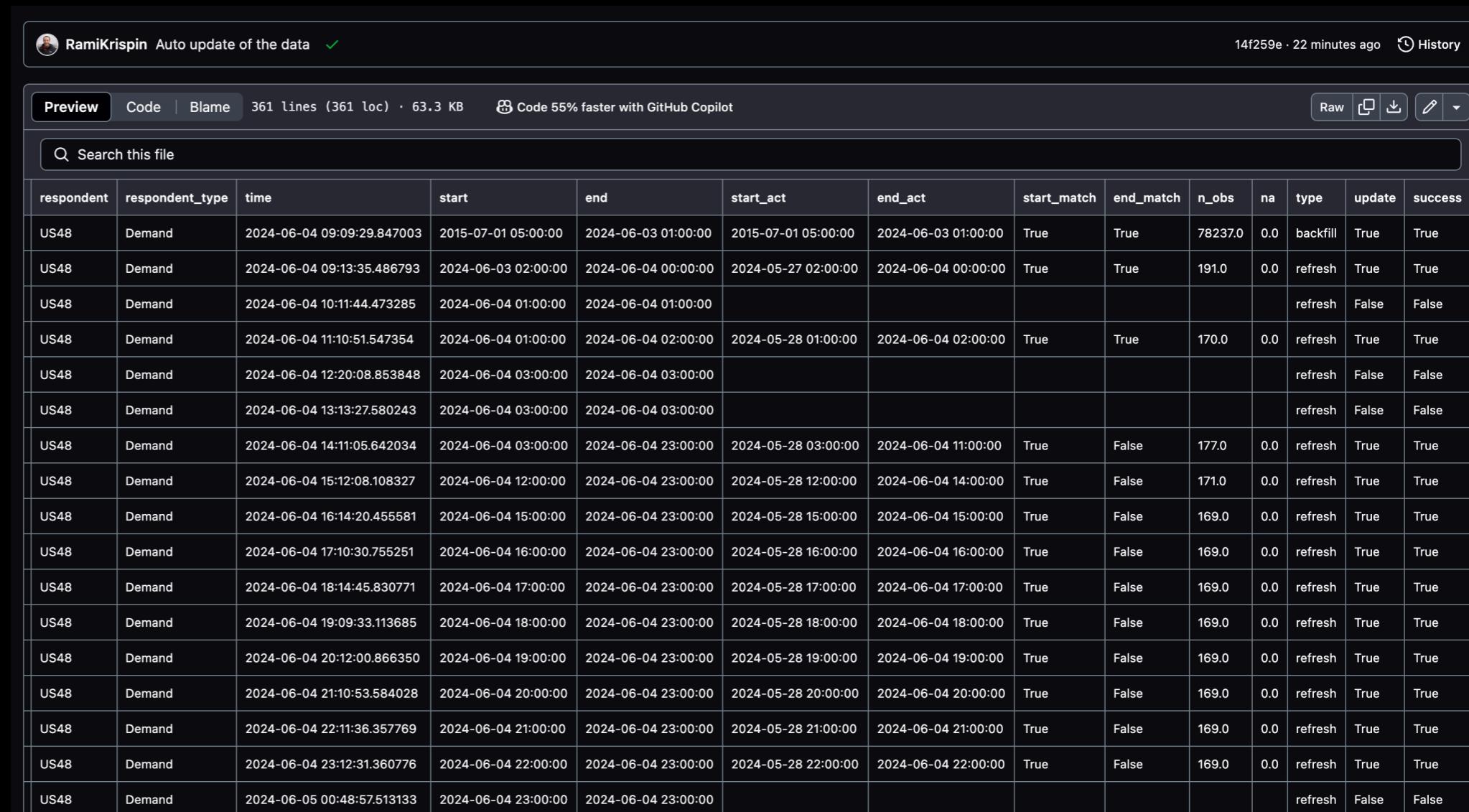
respondent	respondent_type	time	start	end	start_act	end_act	start_match	end_match	n_obs	na	type	update	success
US48	Demand	2024-06-04 09:09:29.847003	2015-07-01 05:00:00	2024-06-03 01:00:00	2015-07-01 05:00:00	2024-06-03 01:00:00	True	True	78237.0	0.0	backfill	True	True
US48	Demand	2024-06-04 09:13:35.486793	2024-06-03 02:00:00	2024-06-04 00:00:00	2024-05-27 02:00:00	2024-06-04 00:00:00	True	True	191.0	0.0	refresh	True	True
US48	Demand	2024-06-04 10:11:44.473285	2024-06-04 01:00:00	2024-06-04 01:00:00							refresh	False	False
US48	Demand	2024-06-04 11:10:51.547354	2024-06-04 01:00:00	2024-06-04 02:00:00	2024-05-28 01:00:00	2024-06-04 02:00:00	True	True	170.0	0.0	refresh	True	True
US48	Demand	2024-06-04 12:20:08.853848	2024-06-04 03:00:00	2024-06-04 03:00:00							refresh	False	False
US48	Demand	2024-06-04 13:13:27.580243	2024-06-04 03:00:00	2024-06-04 03:00:00							refresh	False	False
US48	Demand	2024-06-04 14:11:05.642034	2024-06-04 03:00:00	2024-06-04 23:00:00	2024-05-28 03:00:00	2024-06-04 11:00:00	True	False	177.0	0.0	refresh	True	True
US48	Demand	2024-06-04 15:12:08.108327	2024-06-04 12:00:00	2024-06-04 23:00:00	2024-05-28 12:00:00	2024-06-04 14:00:00	True	False	171.0	0.0	refresh	True	True
US48	Demand	2024-06-04 16:14:20.455581	2024-06-04 15:00:00	2024-06-04 23:00:00	2024-05-28 15:00:00	2024-06-04 15:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 17:10:30.755251	2024-06-04 16:00:00	2024-06-04 23:00:00	2024-05-28 16:00:00	2024-06-04 16:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 18:14:45.830771	2024-06-04 17:00:00	2024-06-04 23:00:00	2024-05-28 17:00:00	2024-06-04 17:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 19:09:33.113685	2024-06-04 18:00:00	2024-06-04 23:00:00	2024-05-28 18:00:00	2024-06-04 18:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 20:12:00.866350	2024-06-04 19:00:00	2024-06-04 23:00:00	2024-05-28 19:00:00	2024-06-04 19:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 21:10:53.584028	2024-06-04 20:00:00	2024-06-04 23:00:00	2024-05-28 20:00:00	2024-06-04 20:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 22:11:36.357769	2024-06-04 21:00:00	2024-06-04 23:00:00	2024-05-28 21:00:00	2024-06-04 21:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 23:12:31.360776	2024-06-04 22:00:00	2024-06-04 23:00:00	2024-05-28 22:00:00	2024-06-04 22:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-05 00:48:57.513133	2024-06-04 23:00:00	2024-06-04 23:00:00							refresh	False	False

Pipeline Design

What?

- Logs

- Unit tests

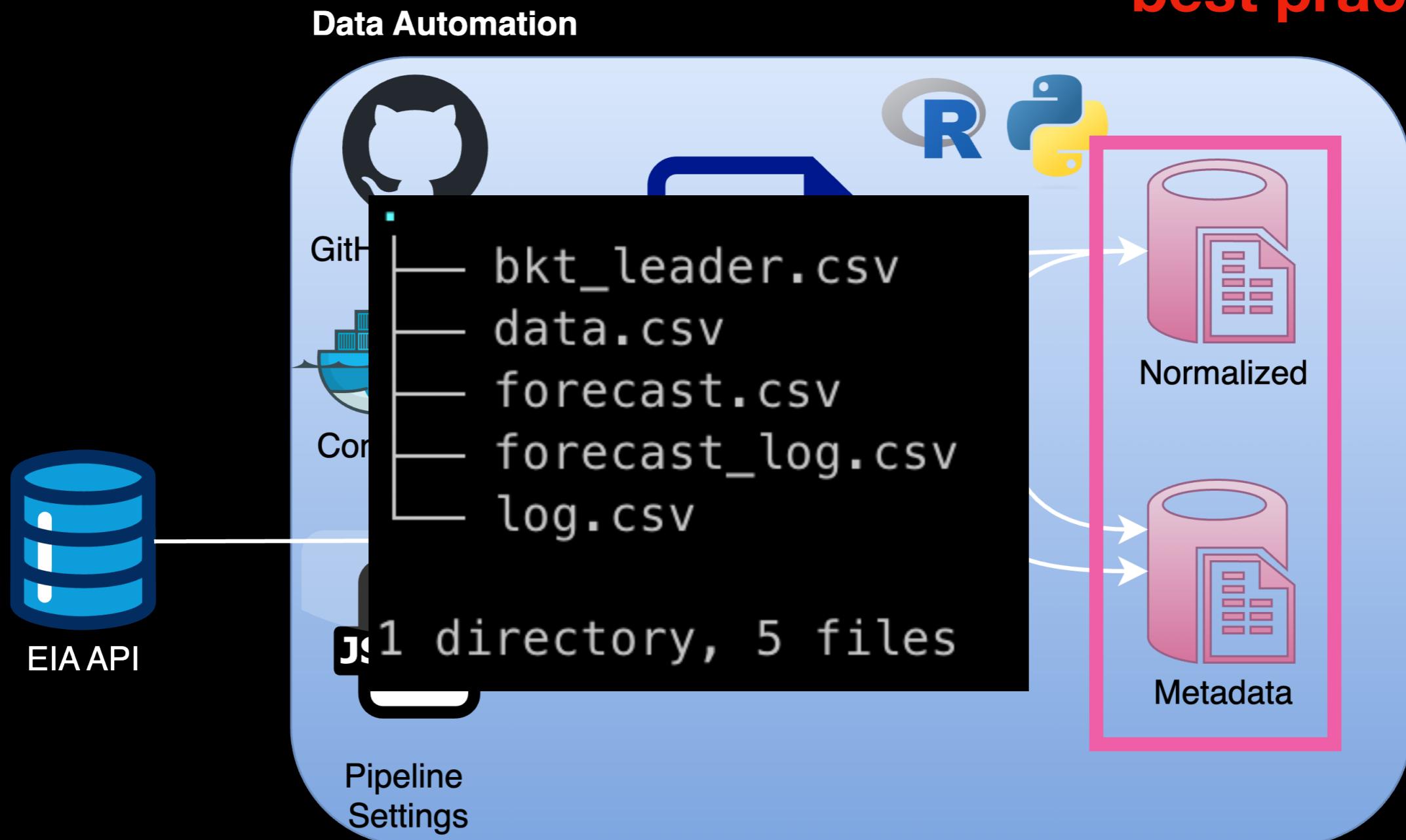


A screenshot of a GitHub code review interface. At the top, it shows a commit by "RamiKrispin" with the message "Auto update of the data" and a green checkmark. To the right, it says "14f259e · 22 minutes ago" and has a "History" button. Below the commit, there are tabs for "Preview", "Code", "Blame", and "361 lines (361 loc) · 63.3 KB". A note says "Code 55% faster with GitHub Copilot". On the far right, there are buttons for "Raw", "Copy", "Delete", "Edit", and a dropdown. A search bar at the top of the table area contains the placeholder "Search this file". The main content is a table with 16 columns and 17 rows of data. The columns are labeled: respondent, respondent_type, time, start, end, start_act, end_act, start_match, end_match, n_obs, na, type, update, and success. The data consists of 17 rows of log entries, each starting with "US48" and followed by a timestamp and various status indicators.

respondent	respondent_type	time	start	end	start_act	end_act	start_match	end_match	n_obs	na	type	update	success
US48	Demand	2024-06-04 09:09:29.847003	2015-07-01 05:00:00	2024-06-03 01:00:00	2015-07-01 05:00:00	2024-06-03 01:00:00	True	True	78237.0	0.0	backfill	True	True
US48	Demand	2024-06-04 09:13:35.486793	2024-06-03 02:00:00	2024-06-04 00:00:00	2024-05-27 02:00:00	2024-06-04 00:00:00	True	True	191.0	0.0	refresh	True	True
US48	Demand	2024-06-04 10:11:44.473285	2024-06-04 01:00:00	2024-06-04 01:00:00							refresh	False	False
US48	Demand	2024-06-04 11:10:51.547354	2024-06-04 01:00:00	2024-06-04 02:00:00	2024-05-28 01:00:00	2024-06-04 02:00:00	True	True	170.0	0.0	refresh	True	True
US48	Demand	2024-06-04 12:20:08.853848	2024-06-04 03:00:00	2024-06-04 03:00:00							refresh	False	False
US48	Demand	2024-06-04 13:13:27.580243	2024-06-04 03:00:00	2024-06-04 03:00:00							refresh	False	False
US48	Demand	2024-06-04 14:11:05.642034	2024-06-04 03:00:00	2024-06-04 23:00:00	2024-05-28 03:00:00	2024-06-04 11:00:00	True	False	177.0	0.0	refresh	True	True
US48	Demand	2024-06-04 15:12:08.108327	2024-06-04 12:00:00	2024-06-04 23:00:00	2024-05-28 12:00:00	2024-06-04 14:00:00	True	False	171.0	0.0	refresh	True	True
US48	Demand	2024-06-04 16:14:20.455581	2024-06-04 15:00:00	2024-06-04 23:00:00	2024-05-28 15:00:00	2024-06-04 15:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 17:10:30.755251	2024-06-04 16:00:00	2024-06-04 23:00:00	2024-05-28 16:00:00	2024-06-04 16:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 18:14:45.830771	2024-06-04 17:00:00	2024-06-04 23:00:00	2024-05-28 17:00:00	2024-06-04 17:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 19:09:33.113685	2024-06-04 18:00:00	2024-06-04 23:00:00	2024-05-28 18:00:00	2024-06-04 18:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 20:12:00.866350	2024-06-04 19:00:00	2024-06-04 23:00:00	2024-05-28 19:00:00	2024-06-04 19:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 21:10:53.584028	2024-06-04 20:00:00	2024-06-04 23:00:00	2024-05-28 20:00:00	2024-06-04 20:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 22:11:36.357769	2024-06-04 21:00:00	2024-06-04 23:00:00	2024-05-28 21:00:00	2024-06-04 21:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-04 23:12:31.360776	2024-06-04 22:00:00	2024-06-04 23:00:00	2024-05-28 22:00:00	2024-06-04 22:00:00	True	False	169.0	0.0	refresh	True	True
US48	Demand	2024-06-05 00:48:57.513133	2024-06-04 23:00:00	2024-06-04 23:00:00							refresh	False	False

Pipeline Design

Not the
best practice!



Pipeline Design



EIA API

EIA API - Data Refresh (Python Version)

Load libraries

```
import eia_api as api
import eia_data
import pandas as pd
import numpy as np
import requests
import json
import os
import datetime
import plotly.express as px
import great_expectations as gt
```

API Settings:

```
raw_json = open("../settings/series.json")
meta_json = json.load(raw_json)
series = pd.DataFrame(meta_json["series"])
api_path = meta_json["api_path"]

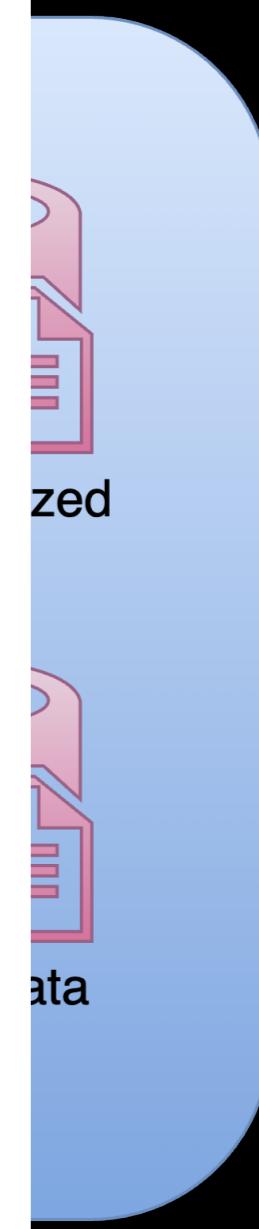
facets_template = {
    "parent" : None,
    "subba" : None
}

offset = 2250

eia_api_key = os.getenv('EIA_API_KEY')

meta_path = meta_json["meta_path"]
data_path = meta_json["data_path"]

meta_obj = eia_data.get_metadata(api_key = eia_api_key, api_path = api_path, meta_path =
gt.GT(meta_obj.request_meta, )
```



Pipeline Design



EIA API

data_refresh.yml 1 X

```
.github > workflows > data_refresh.yml
1   name: Data Refresh
2
3   on:
4     schedule:
5       - cron: "0 */1 * * *"
6
7   jobs:
8     refresh-the-dashboard:
9       runs-on: ubuntu-22.04
10      container:
11        image: docker.io/rkrispin/ai-dev:amd64.0.0.2
12      steps:
13        - name: checkout_repo
14          uses: actions/checkout@v3
15          with:
16            ref: "main"
17        - name: Data Refresh
18          run: bash ./functions/data_refresh_py.sh
19          env:
20            EIA_API_KEY: ${{ secrets.EIA_API_KEY }}
21            USER_EMAIL: ${{ secrets.USER_EMAIL }}
22            USER_NAME: ${{ secrets.USER_NAME }}
```

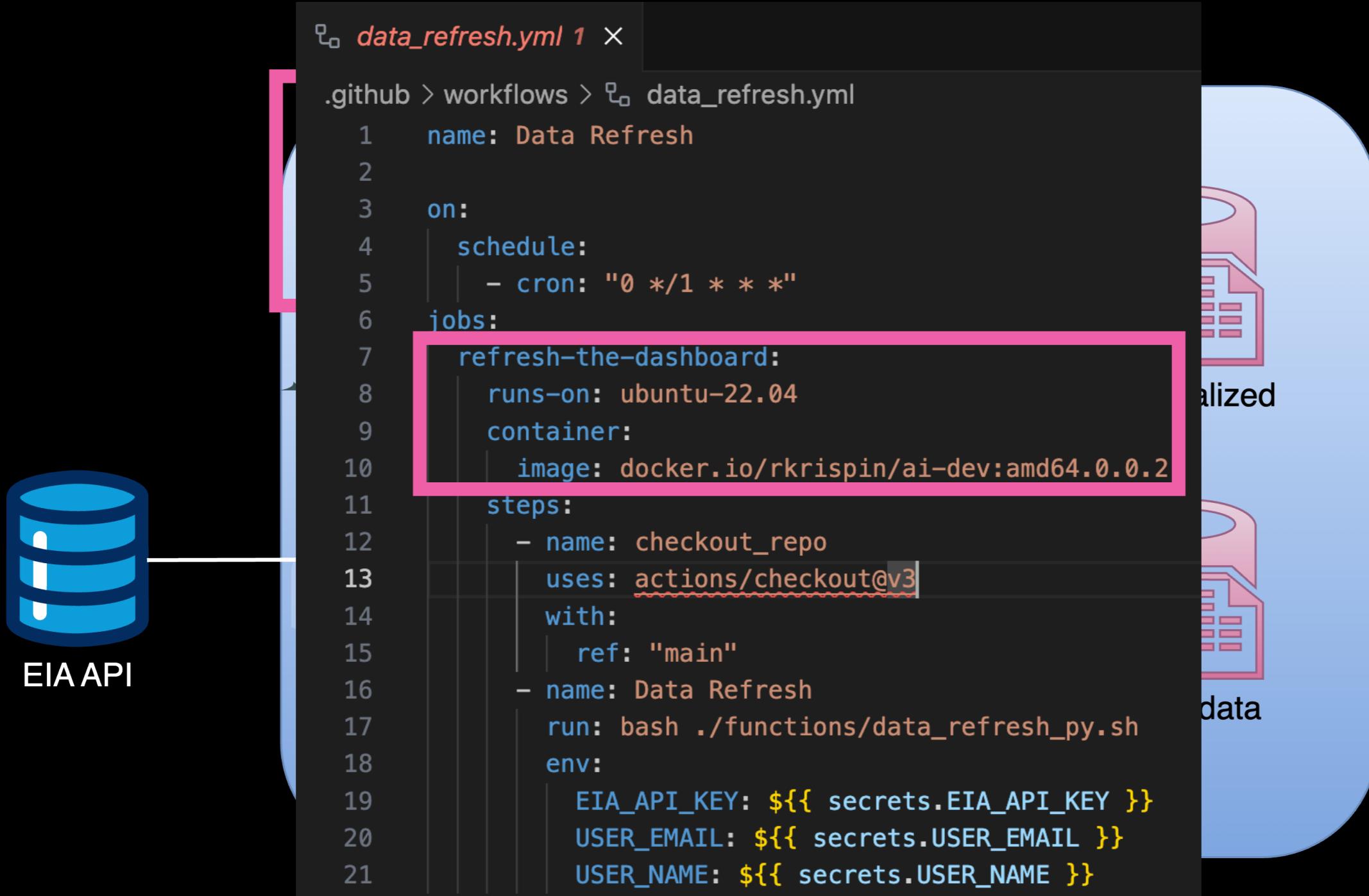


alized



data

Pipeline Design

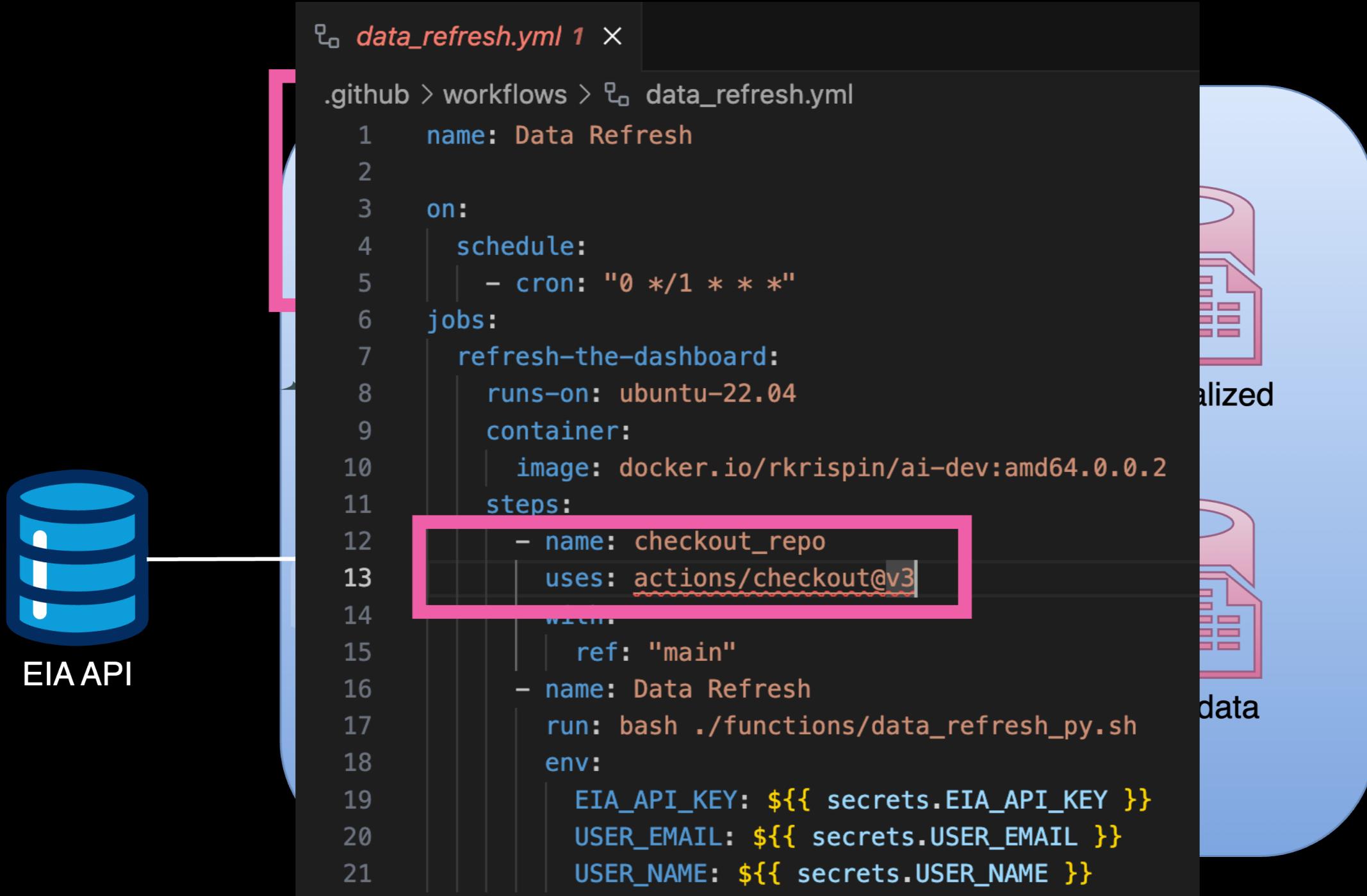


EIA API

```
data_refresh.yml 1 ×  
.github > workflows > data_refresh.yml  
1   name: Data Refresh  
2  
3   on:  
4     schedule:  
5       - cron: "0 */1 * * *"  
6   jobs:  
7     refresh-the-dashboard:  
8       runs-on: ubuntu-22.04  
9       container:  
10         image: docker.io/rkrispin/ai-dev:amd64.0.0.2  
11       steps:  
12         - name: checkout_repo  
13           uses: actions/checkout@v3  
14           with:  
15             ref: "main"  
16         - name: Data Refresh  
17           run: bash ./functions/data_refresh_py.sh  
18           env:  
19             EIA_API_KEY: ${{ secrets.EIA_API_KEY }}  
20             USER_EMAIL: ${{ secrets.USER_EMAIL }}  
21             USER_NAME: ${{ secrets.USER_NAME }}
```

The GitHub code editor shows a workflow named 'Data Refresh'. It includes a scheduled job that runs every hour. The job contains a step to checkout the main branch and another step to run a script named 'data_refresh_py.sh'. This script is expected to interact with the EIA API, as indicated by the 'EIA API' icon and the connection line from the icon to the 'data_refresh_py.sh' command in the workflow.

Pipeline Design



EIA API

```
data_refresh.yml 1 ×  
.github > workflows > data_refresh.yml  
1   name: Data Refresh  
2  
3   on:  
4     schedule:  
5       - cron: "0 */1 * * *"  
6   jobs:  
7     refresh-the-dashboard:  
8       runs-on: ubuntu-22.04  
9       container:  
10      image: docker.io/rkrispin/ai-dev:amd64.0.0.2  
11      steps:  
12        - name: checkout_repo  
13          uses: actions/checkout@v3  
14          with:  
15            ref: "main"  
16        - name: Data Refresh  
17          run: bash ./functions/data_refresh_py.sh  
18          env:  
19            EIA_API_KEY: ${{ secrets.EIA_API_KEY }}  
20            USER_EMAIL: ${{ secrets.USER_EMAIL }}  
21            USER_NAME: ${{ secrets.USER_NAME }}
```

Pipeline Design



EIA API

data_refresh.yml 1 X

```
.github > workflows > data_refresh.yml
1   name: Data Refresh
2
3   on:
4     schedule:
5       - cron: "0 */1 * * *"
6   jobs:
7     refresh-the-dashboard:
8       runs-on: ubuntu-22.04
9       container:
10      image: docker.io/rkrispin/ai-dev:amd64.0.0.2
11      steps:
12        - name: checkout_repo
13          uses: actions/checkout@v3
14          with:
15            ref: "main"
16        - name: Data Refresh
17          run: bash ./functions/data_refresh_py.sh
18          env:
19            EIA_API_KEY: ${{ secrets.EIA_API_KEY }}
20            USER_EMAIL: ${{ secrets.USER_EMAIL }}
21            USER_NAME: ${{ secrets.USER_NAME }}
```

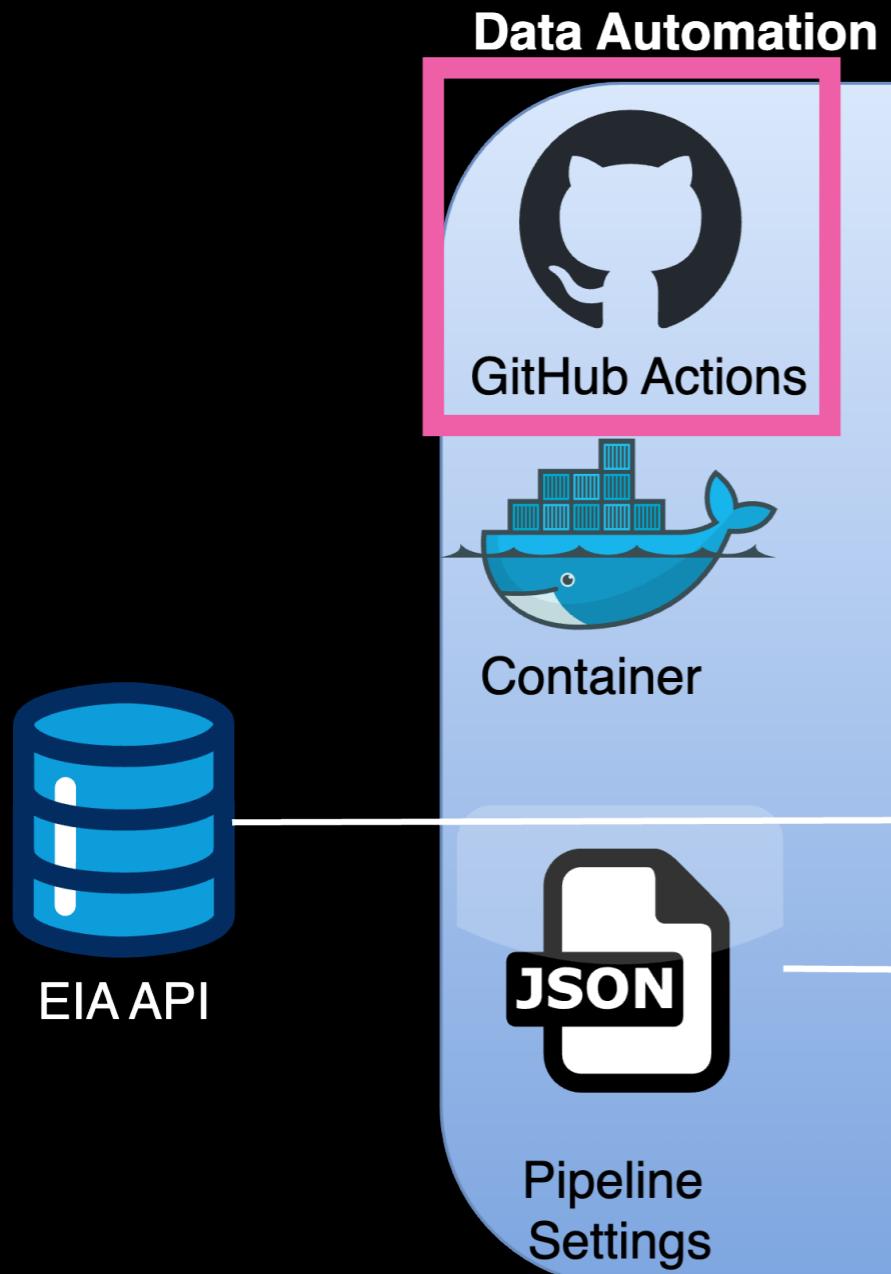


alized



data

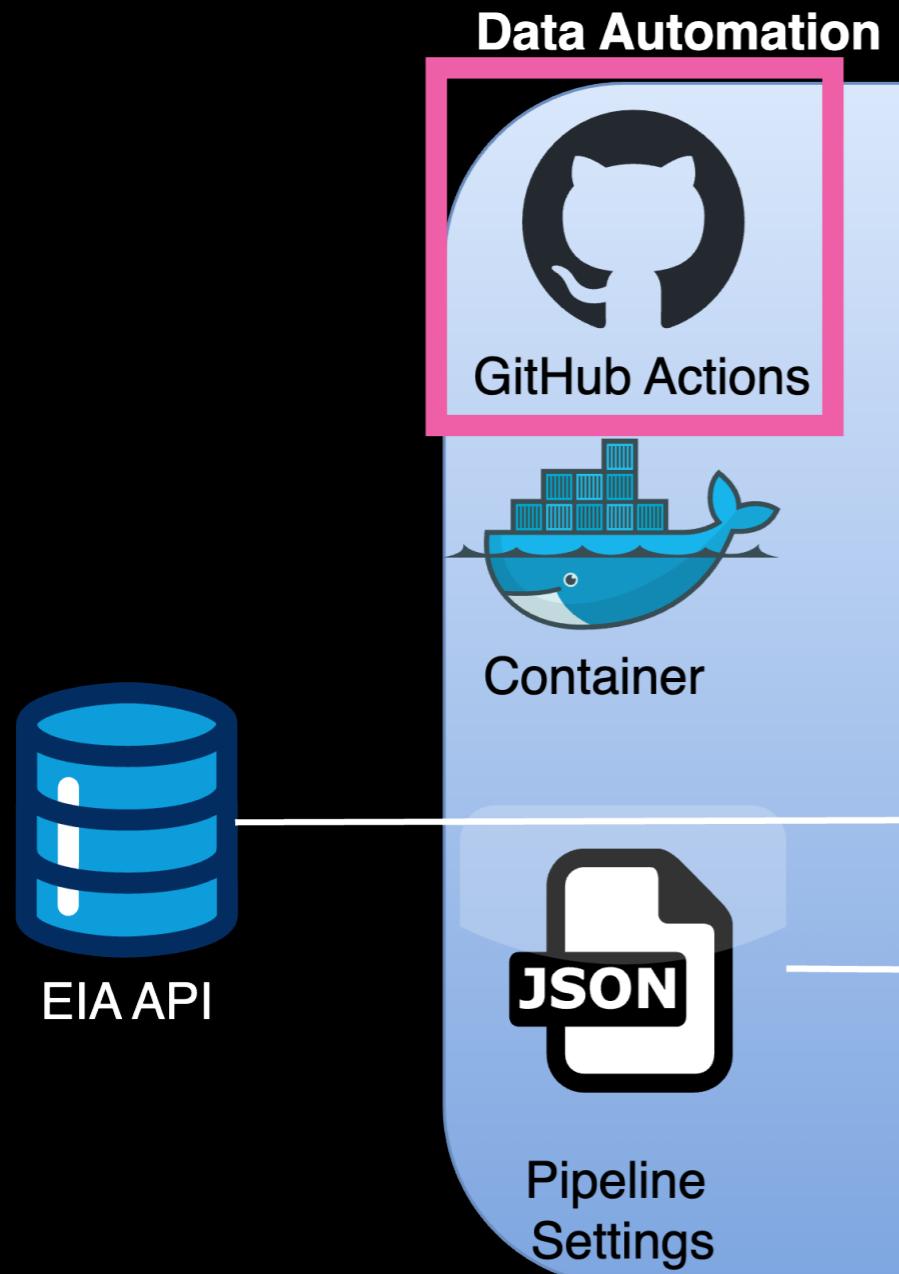
Pipeline Design



```
$ data_refresh_py.sh >

functions > $ data_refresh_py.sh
1 #!/usr/bin/env bash
2 source /opt/$VENV_NAME/bin/activate
3
4 rm -rf ./functions/data_refresh_py_files
5 rm ./functions/data_refresh_py.html
6 quarto render ./functions/data_refresh_py.qmd --to html
7
8 rm -rf docs/data_refresh/
9 mkdir docs/data_refresh
10 cp ./functions/data_refresh_py.html ./docs/data_refresh/
11 cp -R ./functions/data_refresh_py_files ./docs/data_refresh/
12
13 echo "Finish"
14 p=$(pwd)
15 git config --global --add safe.directory $p
16
17
18 # Render the Quarto dashboard
19 if [[ "$(git status --porcelain)" != "" ]]; then
20     quarto render functions/index.qmd
21     cp functions/index.html docs/index.html
22     rm -rf docs/index_files
23     cp -R functions/index_files/ docs/
24     rm functions/index.html
25     rm -rf functions/index_files
26     git config --global user.name $USER_NAME
27     git config --global user.email $USER_EMAIL
28     git add data/*
29     git add docs/*
30     git commit -m "Auto update of the data"
31     git push origin main
32 else
33     echo "Nothing to commit..."
34 fi
```

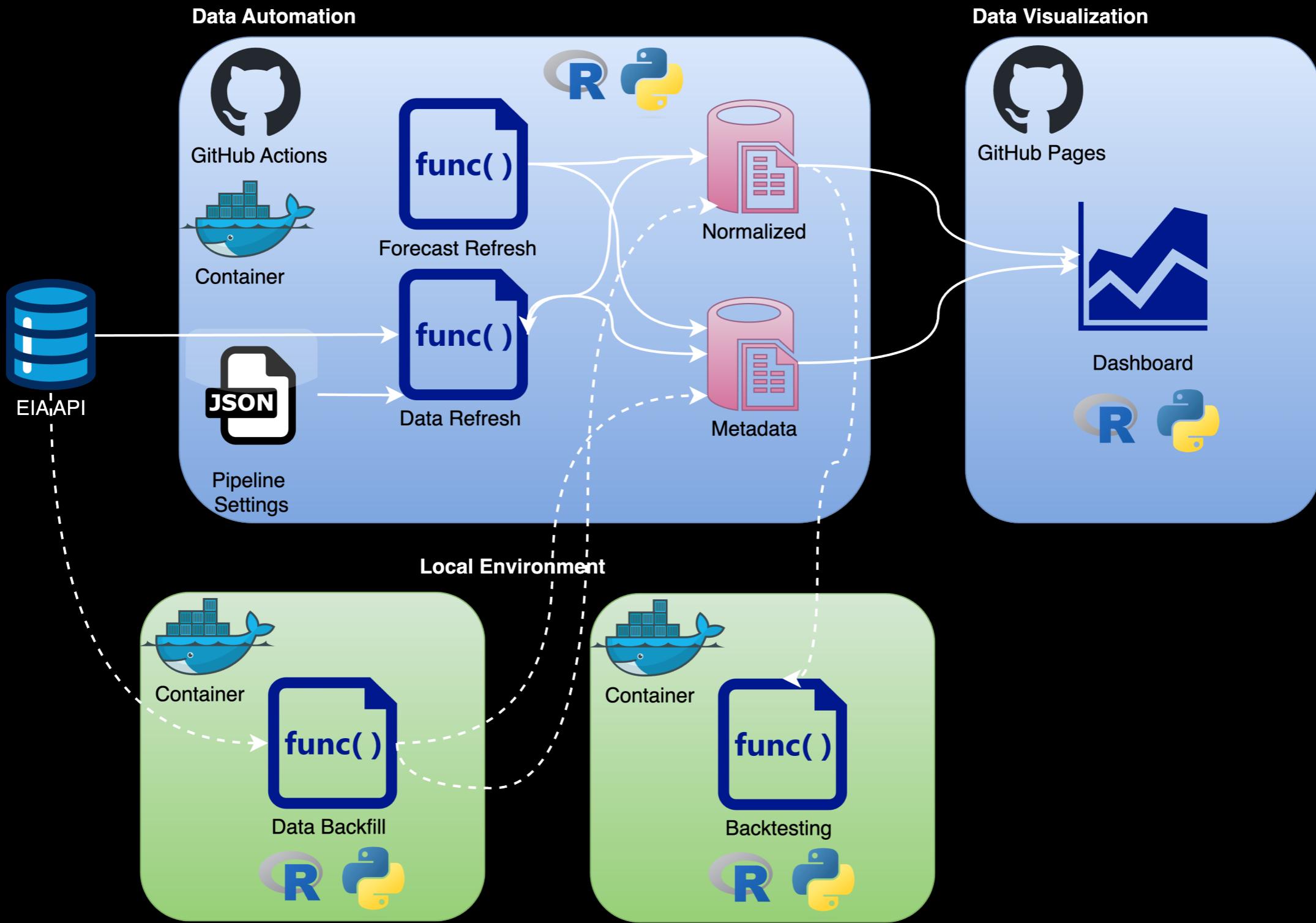
Pipeline Design



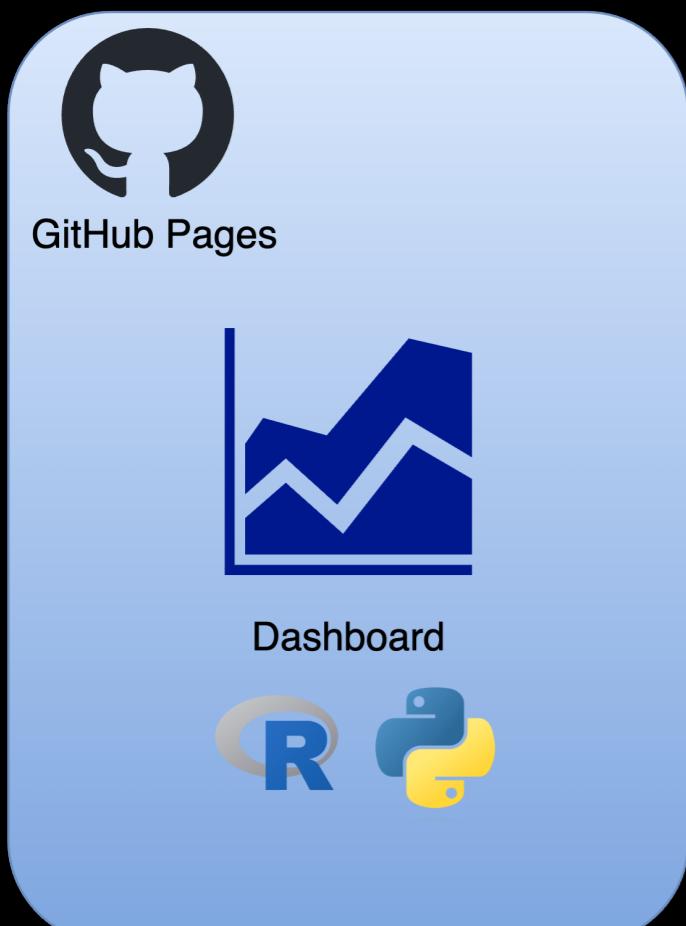
```
$ data_refresh_py.sh >

functions > $ data_refresh_py.sh
1 #!/usr/bin/env bash
2 source /opt/$VENV_NAME/bin/activate
3
4 rm -rf ./functions/data_refresh_py_files
5 rm ./functions/data_refresh_py.html
6 quarto render ./functions/data_refresh_py.qmd --to html
7
8 rm -rf docs/data_refresh/
9 mkdir docs/data_refresh
10 cp ./functions/data_refresh_py.html ./docs/data_refresh/
11 cp -R ./functions/data_refresh_py_files ./docs/data_refresh/
12
13 echo "Finish"
14 p=$(pwd)
15 git config --global --add safe.directory $p
16
17
18 # Render the Quarto dashboard
19 if [[ "$(git status --porcelain)" != "" ]]; then
20     quarto render functions/index.qmd
21     cp functions/index.html docs/index.html
22     rm -rf docs/index_files
23     cp -R functions/index_files/ docs/
24     rm functions/index.html
25     rm -rf functions/index_files
26     git config --global user.name $USER_NAME
27     git config --global user.email $USER_EMAIL
28     git add data/*
29     git add docs/*
30     git commit -m "Auto update of the data"
31     git push origin main
32 else
33     echo "Nothing to commit..."
34 fi
```

Pipeline Design



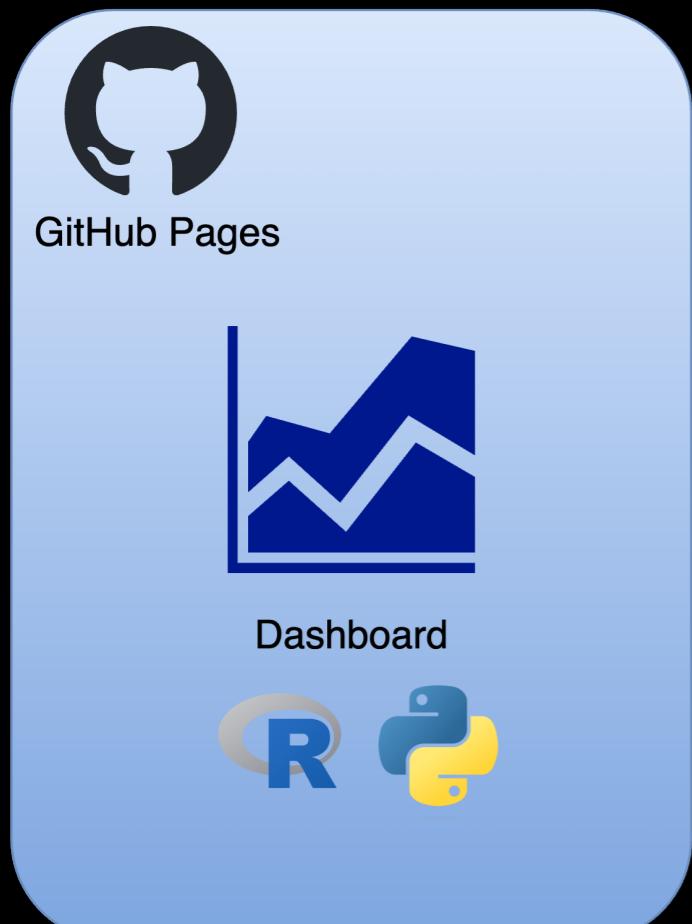
Pipeline Design



Dashboard



Pipeline Design



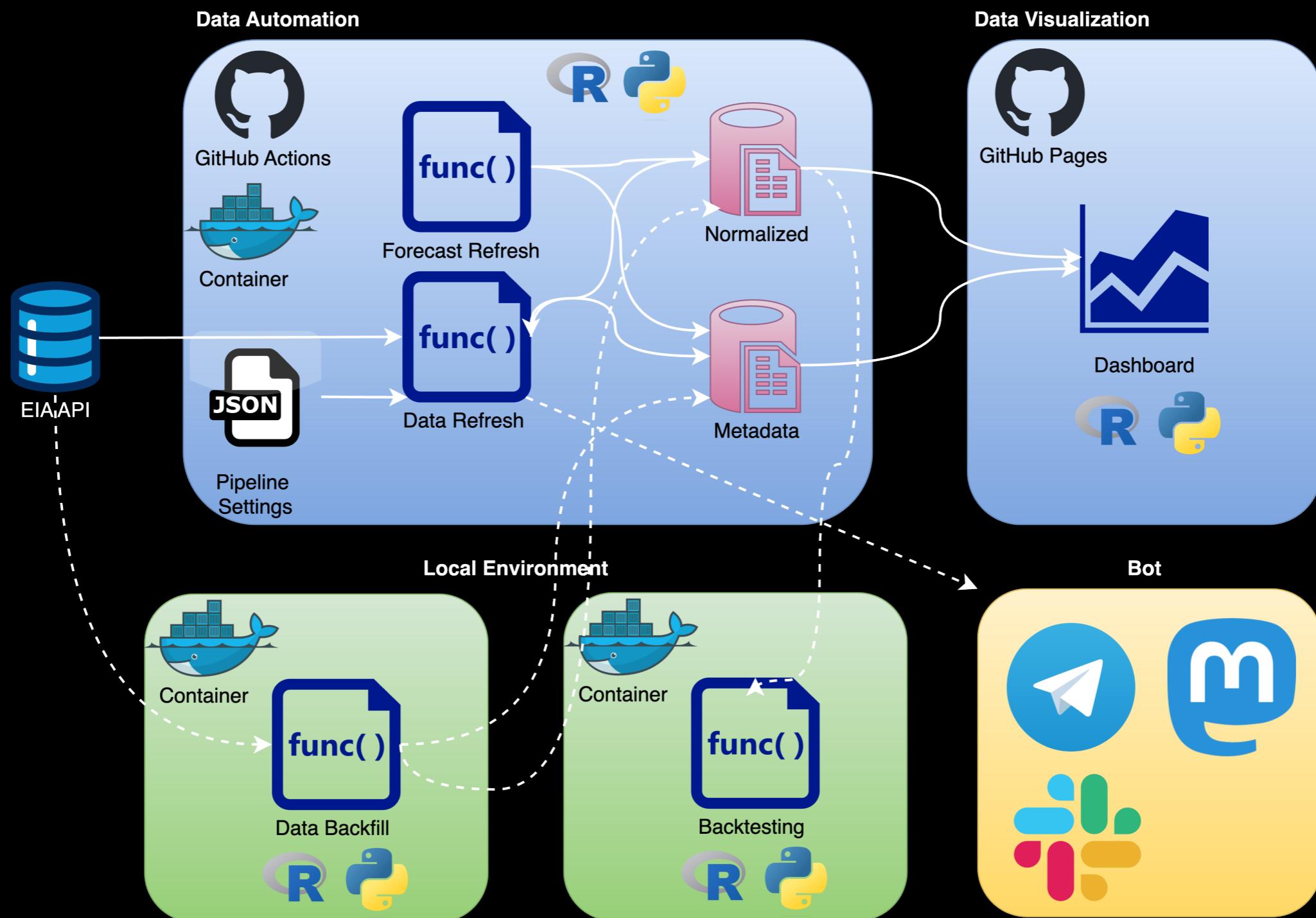
California Independent System Operator Hourly Demand AI Dev Workshop															
Hourly Demand By Provider			Metadata												
index	parent	subba	time	start	end	start_act	end_act	start_match	end_match	n_obs	na	type	update	success	comments
41	CISO	PGAE	2024-06-19 04:19:47.237844+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
41	CISO	SCE	2024-06-19 04:19:47.238733+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
41	CISO	SDGE	2024-06-19 04:19:47.240187+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
41	CISO	VEA	2024-06-19 04:19:47.241557+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
42	CISO	PGAE	2024-06-19 05:16:48.928662+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
42	CISO	SCE	2024-06-19 05:16:48.929611+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
42	CISO	SDGE	2024-06-19 05:16:48.931054+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
42	CISO	VEA	2024-06-19 05:16:48.932411+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
43	CISO	PGAE	2024-06-19 06:22:25.443620+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
43	CISO	SCE	2024-06-19 06:22:25.444909+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
43	CISO	SDGE	2024-06-19 06:22:25.446762+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
43	CISO	VEA	2024-06-19 06:22:25.448279+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
44	CISO	PGAE	2024-06-19 07:15:18.244598+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	
44	CISO	SCE	2024-06-19 07:15:18.245507+00:00	2024-06-18 08:00:00	2024-06-18 07:00:00						refresh	False	False	No new data is available; The data refresh failed, please check the log;	

Demo

What else can we do?

Many other cool things!

Adding Bot



Best Practices

- Know the tools limitations
- Using settings files and environment variables
- Unit tests
- Interactive documents
- GitHub Templates!

Summary

- GitHub Actions enables to schedule workflows
- Docker provides a reproducible environment
- Great use cases - open source projects
- Quarto documents to host the pipeline
- Leverage great open source libraries (MLflow, Y-Data Profile, etc.)
- Not enterprise ready

Questions?

Get In Touch

