NREL's REopt Lite Tool – Open Source

Webinar: March 17, 2020

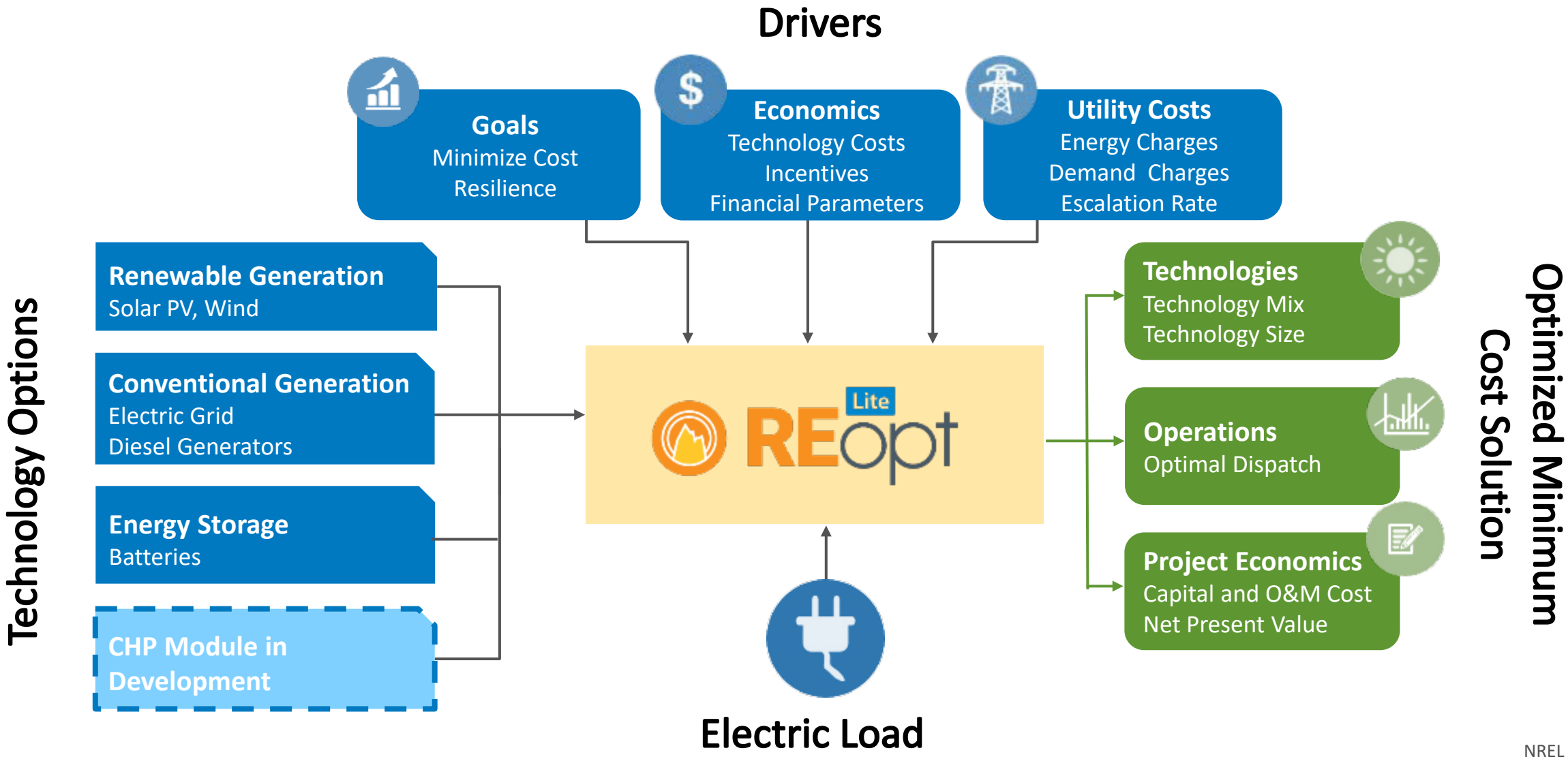
https://reopt.nrel.gov/tool

Agenda

1. REopt Lite – An Introduction
2. Setup and Installation
3. Structure: Full code Stack
4. Modelling Concepts
5. Formulation: Mixed Integer Linear Programming
6. Expansion: Adding New Functionalities
7. Contributions Guidelines + Q&A

REopt Lite: Free Tool to Optimize Economic and Resilience Benefits of DERs

Formulated as a mixed integer linear program, REopt provides an integrated cost-optimal energy solution.



REopt Lite Tool: Transforms Complex Decisions Into Actionable Results

- The free, publicly available tool guides investment in economic, resilient energy technologies;
- Based on decades of NREL decision-support expertise, REopt Lite transforms complex decisions into actionable results for building owners, utilities, and industry; and
- Open Source and API access to the tool enables analysis at scale.

REopt Lite provides solutions for a range of users:
Researchers, developers, building owners, utilities, and industry

Researchers

How will RE deployment change in the future?

Developers

Where are the best market opportunities?

Home and Building Owners

What technologies are best for my site?

Utilities

What value do these systems provide?

Industry

How do I optimize system control for maximum value?

Currently: 1,100+ web tool users & 325,000 API hits

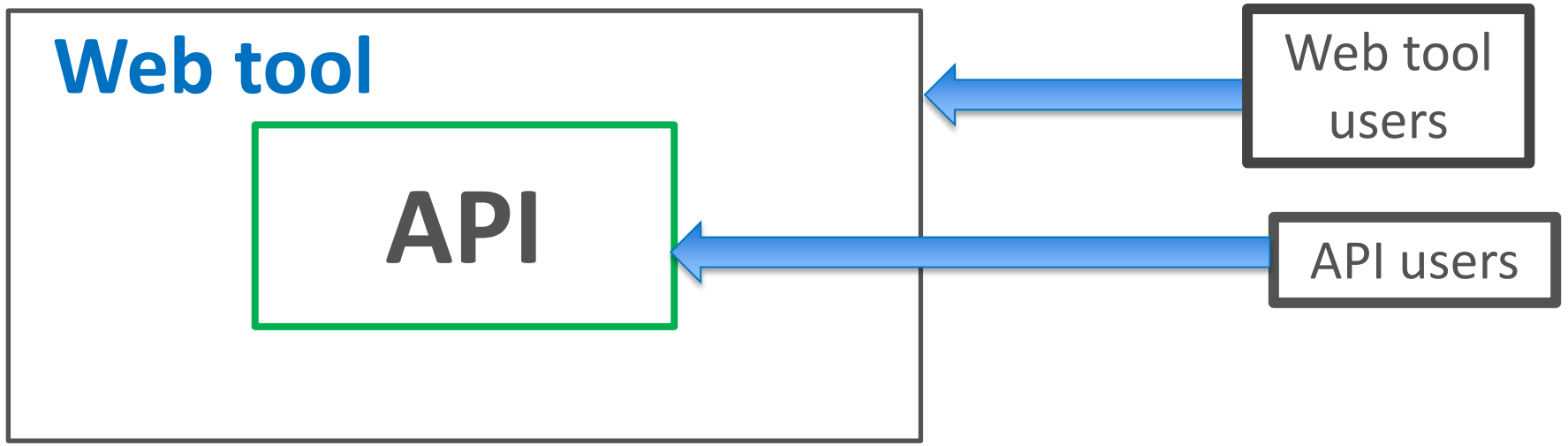
REopt Lite's Accessibility

Multiple formats

Accessing REopt Lite

- REopt Lite can be accessed via the web tool, application programming interface (API), or open source code.
  - Web tool: Use the simple, user-friendly, web-based interface for single-site analysis.
  - API: Get programmatic access to run multisite analysis or embed in custom applications.
  - Open Source: Access the source code to add custom features and capabilities.

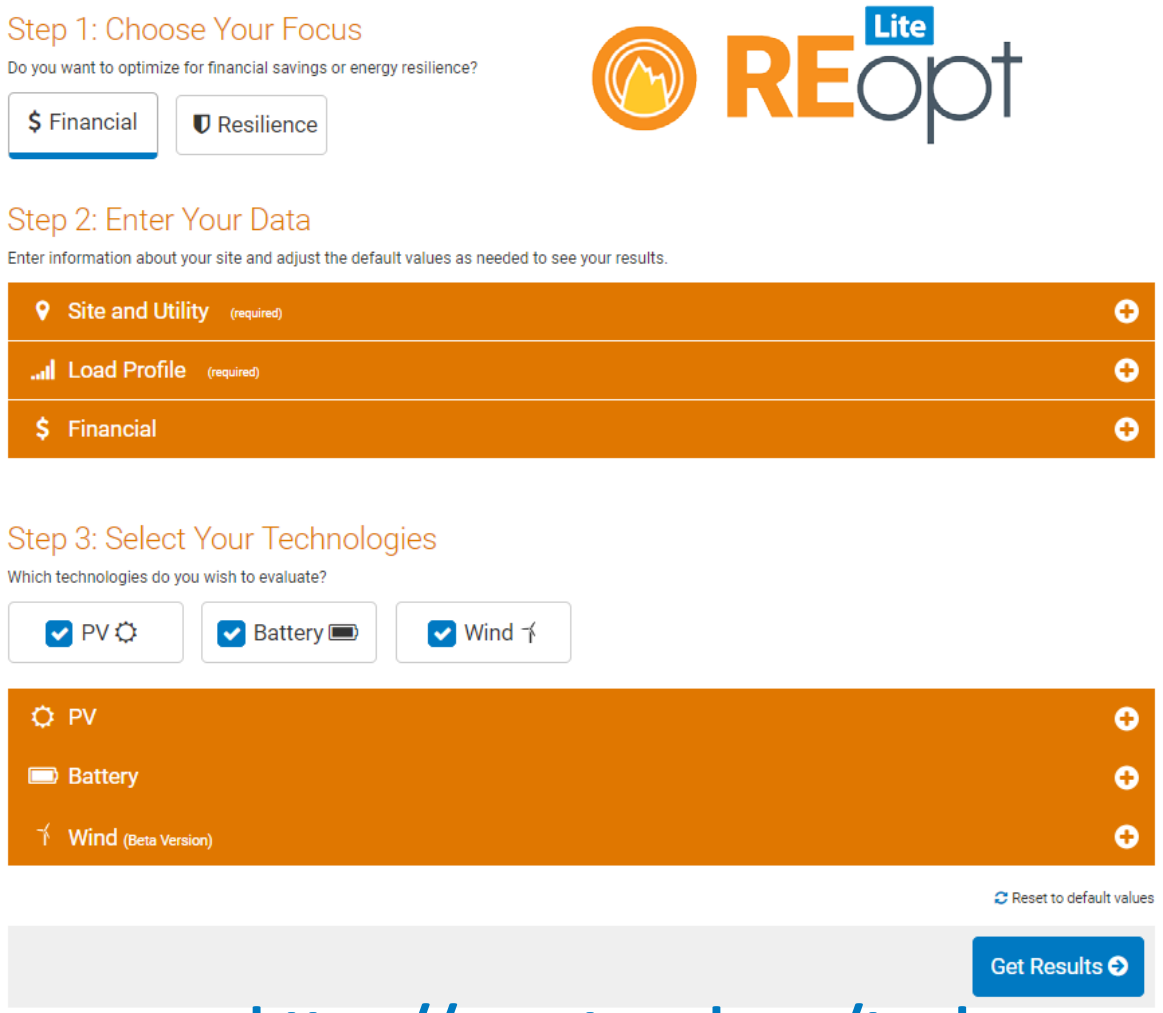
The difference: Web Tool, API, and Open Source



The code-base in the green box (API) is open sourced

REopt Lite Web Tool

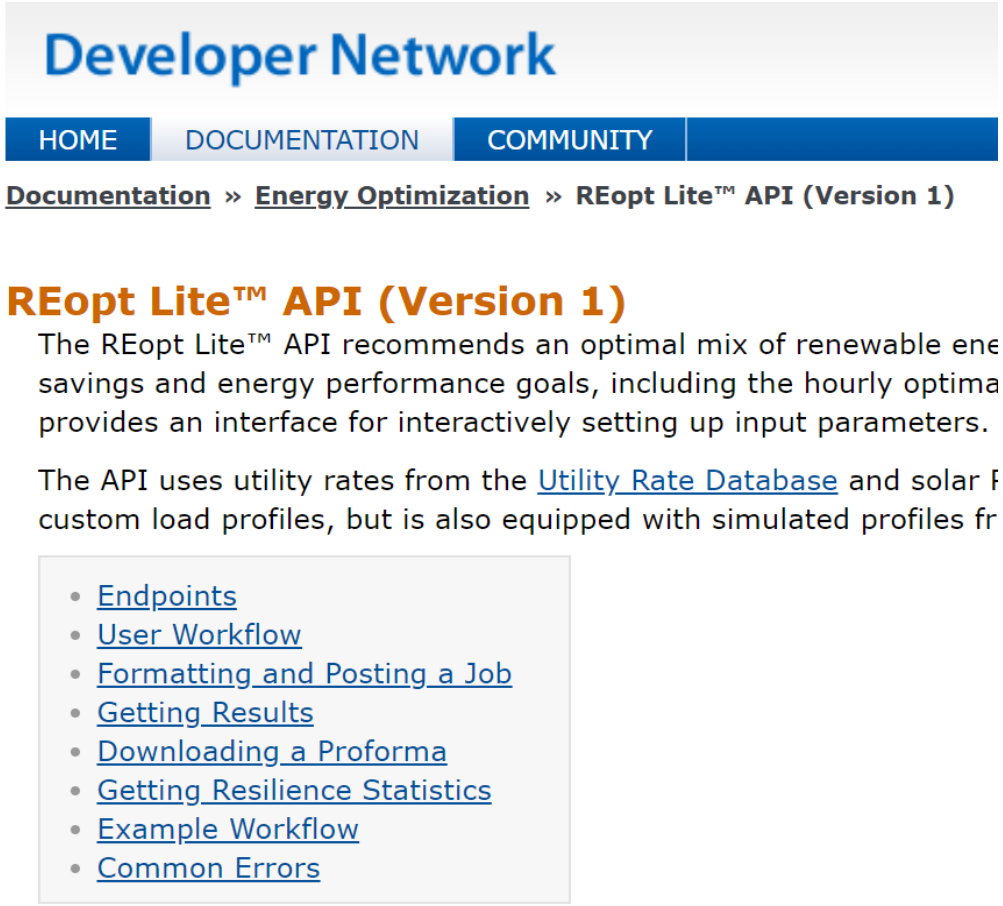
- The REopt Lite Web Tool offers a no-cost subset of NREL's more comprehensive REopt model
- **Financial mode** optimizes PV, wind and battery system sizes and battery dispatch strategy to minimize life cycle cost of energy
- **Resilience mode** optimizes PV, wind, and storage systems along with existing back-up generators to sustain critical load during grid outages



https://reopt.nrel.gov/tool

REopt Lite Application Programming Interface

- What is an API?
  - Application Programming Interface
  - Programmatic way of accessing REopt Lite (sending and receiving data from a server)
  - File format used for sending and receiving the data: JSON
- Advantages:
  - Multiple simulations for different sites can be run programmatically
  - Scenario analysis can be automated
  - Integration with other programs



https://developer.nrel.gov/docs/energy-optimization/reopt-v1/

REopt Lite Open Source

- REopt Lite Open Source provides all of the web tool's robust capabilities, plus:
  - **Transparency**—Explore the source code to find equations and algorithms so you can see exactly how the models work.
  - **Flexibility**—Change the code and build your own versions of REopt Lite to add your customized features and capabilities.
  - **Collaboration**—Contribute new models, fix bugs, or work with NREL to implement new features that can be added to NREL's REopt Lite offerings.

- The source code and tools required to build REopt Lite are available to the public on the REopt Lite GitHub repository.

- NREL will continue to maintain and update the code, and to release NREL versions of REopt Lite Web Tool and the REopt Lite API with new features and updates.

BSD-3 License

- Permissive license with minimal restrictions on use and distribution
- You are not required to share contributions (but the community would appreciate it!)

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

https://github.com/NREL/REopt_Lite_API/blob/develop/LICENSE

Installation

Working with the Open Source model

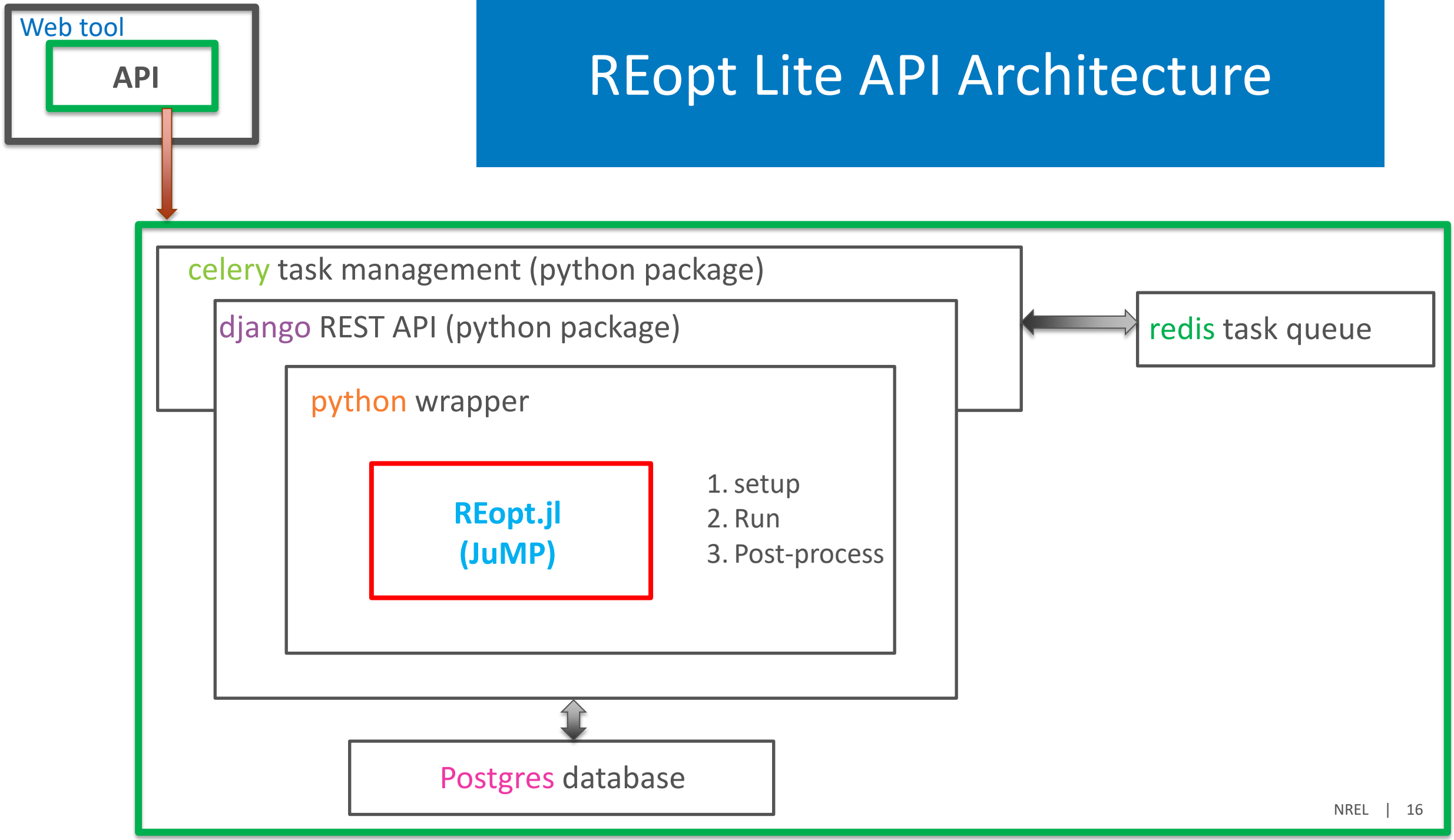
Development Environment Setup

- Platform specific installation instructions provided on the wiki: https://github.com/NREL/REopt_Lite_API/wiki
- Two ways to setup the development environment
  - Docker
  - Piece by piece installation
- New to GitHub? Check out: https://guides.github.com/

Structure

The code stack that handles preprocessing, optimization and post-processing

REopt Lite API Architecture



REopt Lite Modelling Concepts: “Techs”

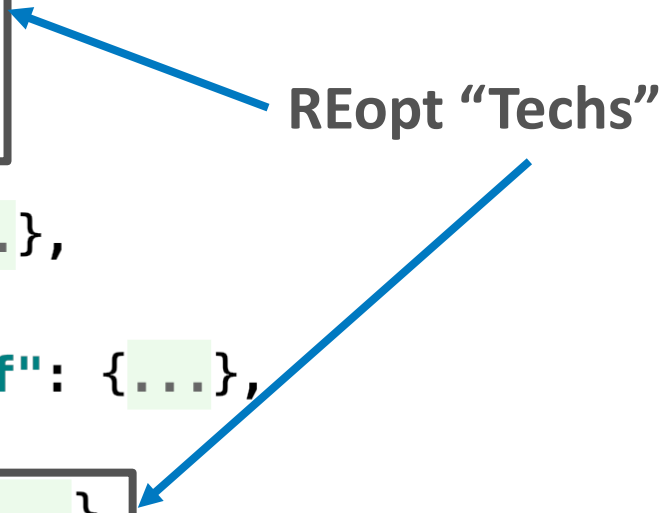
Generic technologies that can meet loads

REopt Lite ”job” POST

```
"Site": {
  "LoadProfile": {...},
  "Financial": {...},
  "PV": {...},
  "Wind": {...},
  "Storage": {...},
  "ElectricTariff": {...},
  "Generator": {...}
}
```

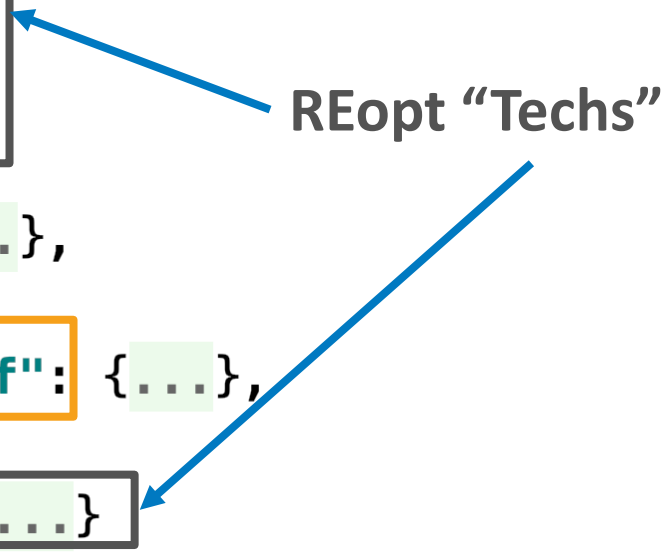
REopt Lite "job" POST

```
"Site": {
  "LoadProfile": {...},
  "Financial": {...},
  "PV": {...},
  "Wind": {...},
  "Storage": {...},
  "ElectricTariff": {...},
  "Generator": {...}
}
```



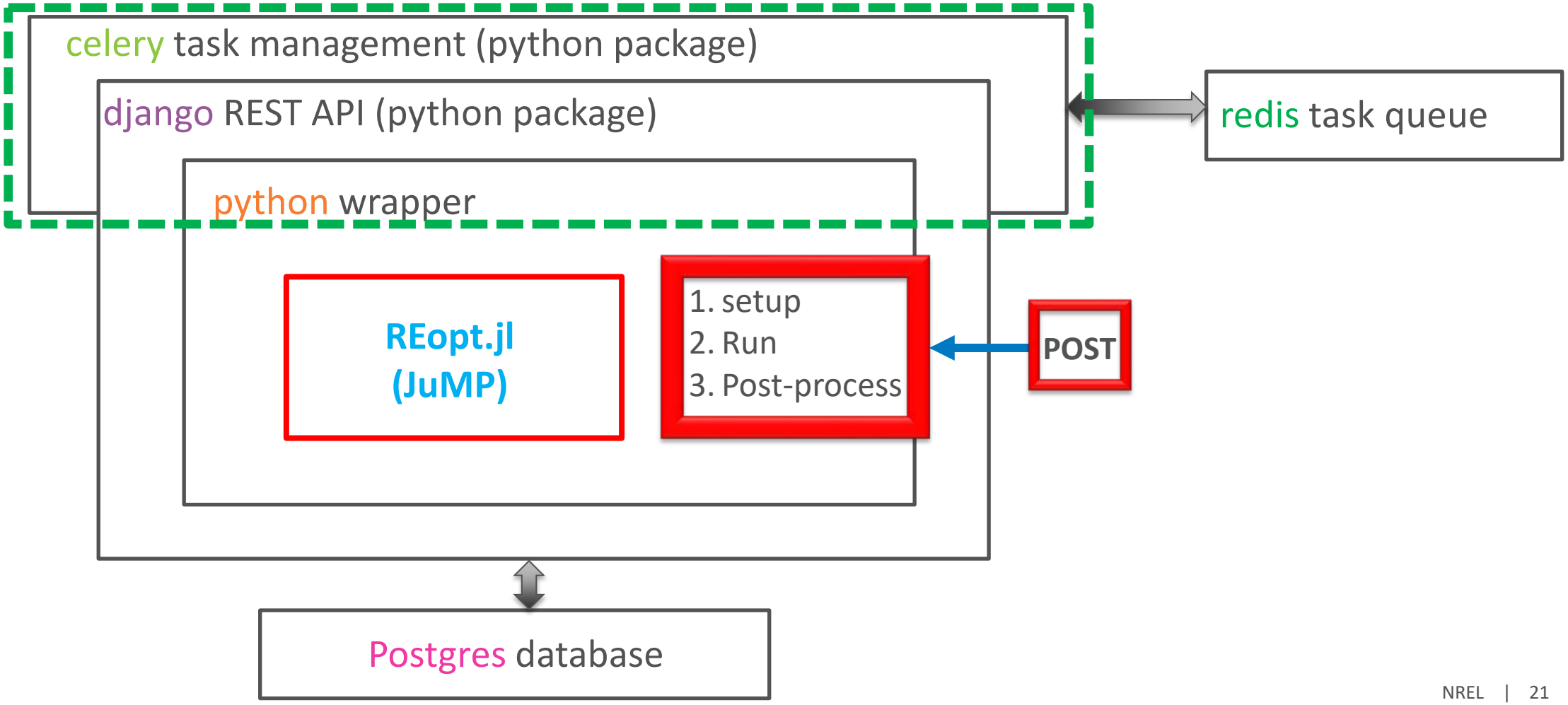
REopt Lite "job" POST

```
"Site": {
  "LoadProfile": {...},
  "Financial": {...},
  "PV": {...},
  "Wind": {...},
  "Storage": {...},
  "ElectricTariff": {...},
  "Generator": {...}
}
```

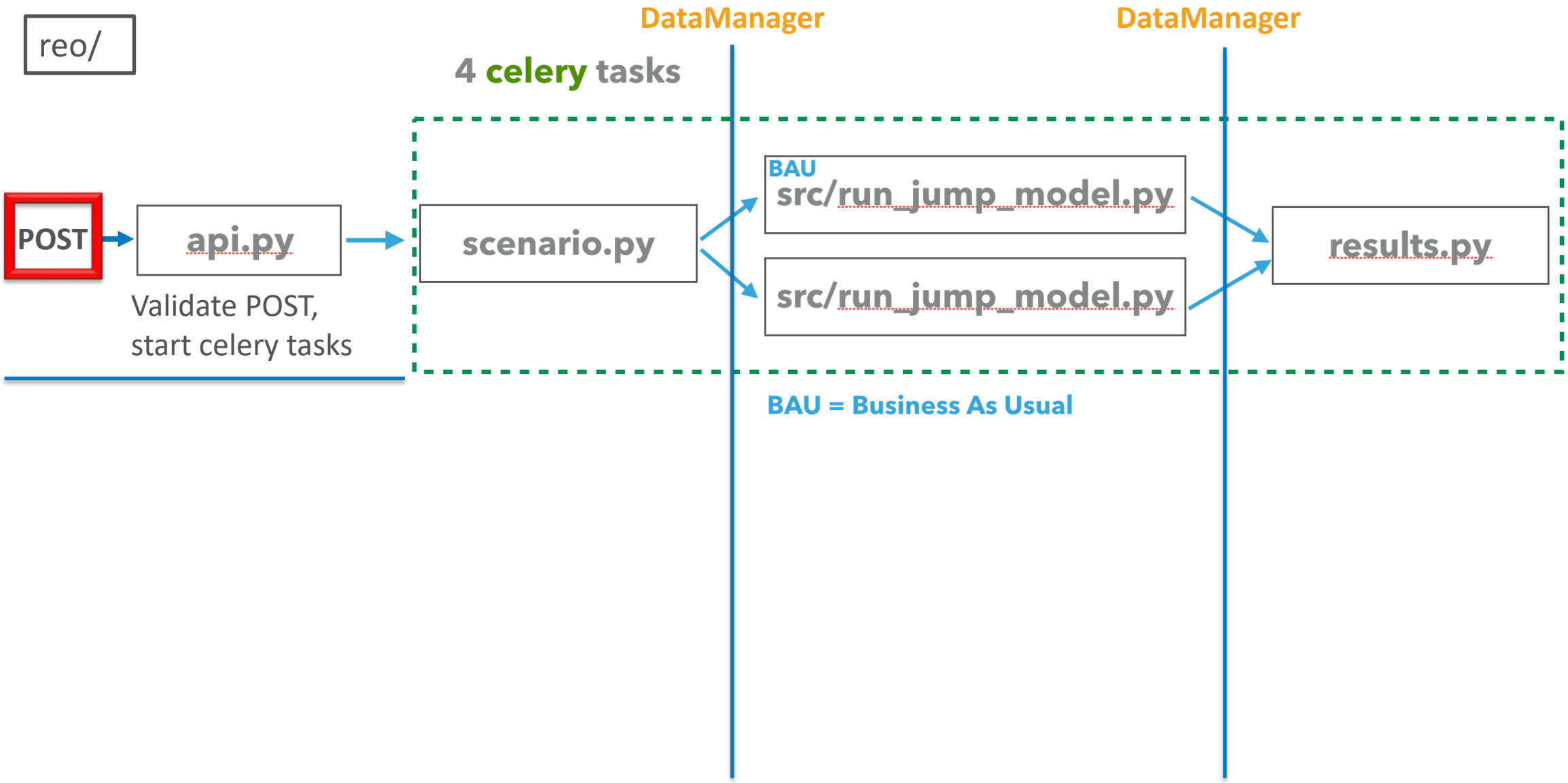


ElectricTariff and **LoadProfile** parameters combine to make ***Util Tech***

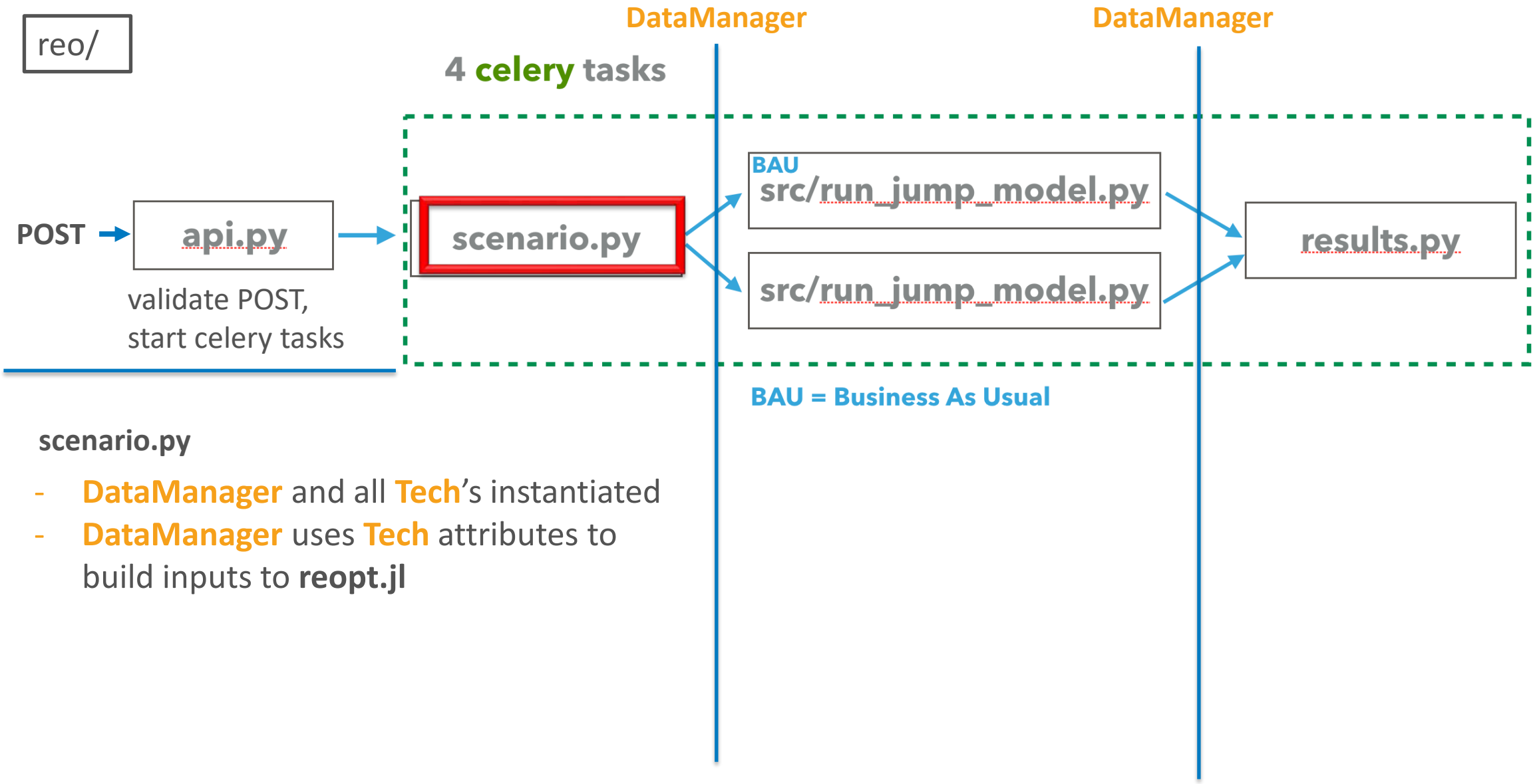
REopt Lite API Architecture



REopt Lite Data-Flow



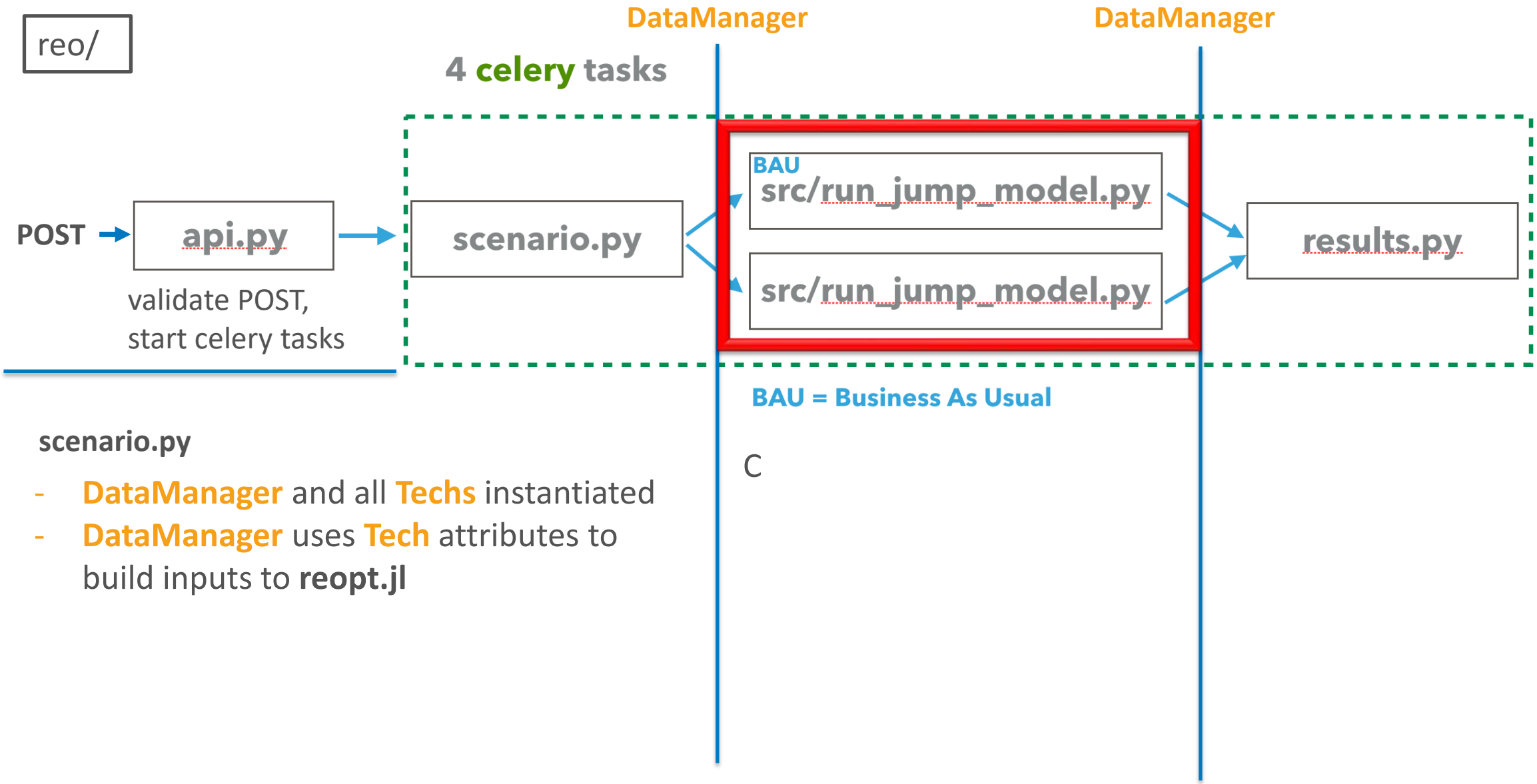
REopt Lite Data-Flow



scenario.py

- **DataManager** and all **Tech**'s instantiated
- **DataManager** uses **Tech** attributes to build inputs to **reopt.jl**

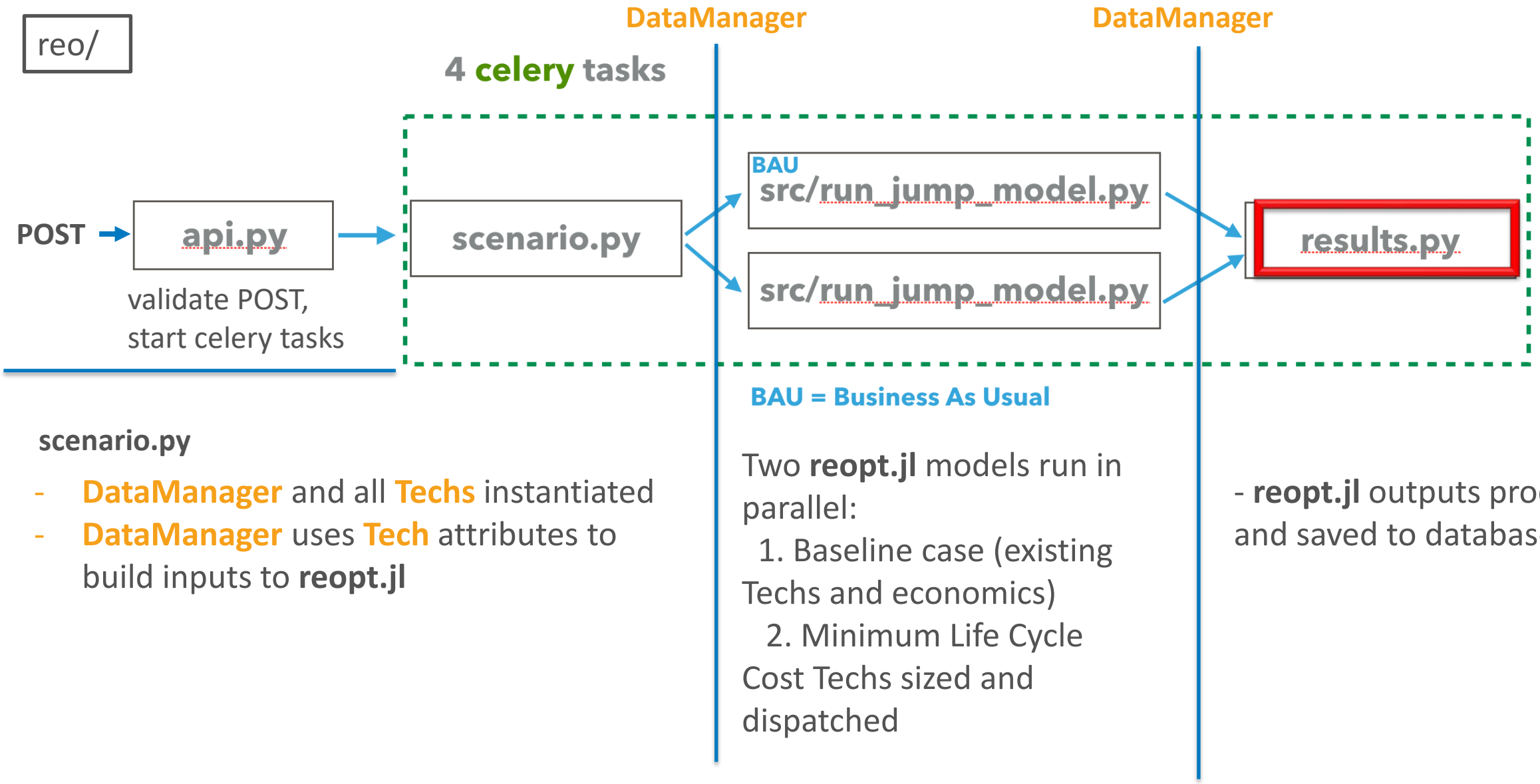
REopt Lite Data-Flow



scenario.py

- **DataManager** and all **Techs** instantiated
- **DataManager** uses **Tech** attributes to build inputs to **reopt.jl**

REopt Lite Data-Flow



scenario.py

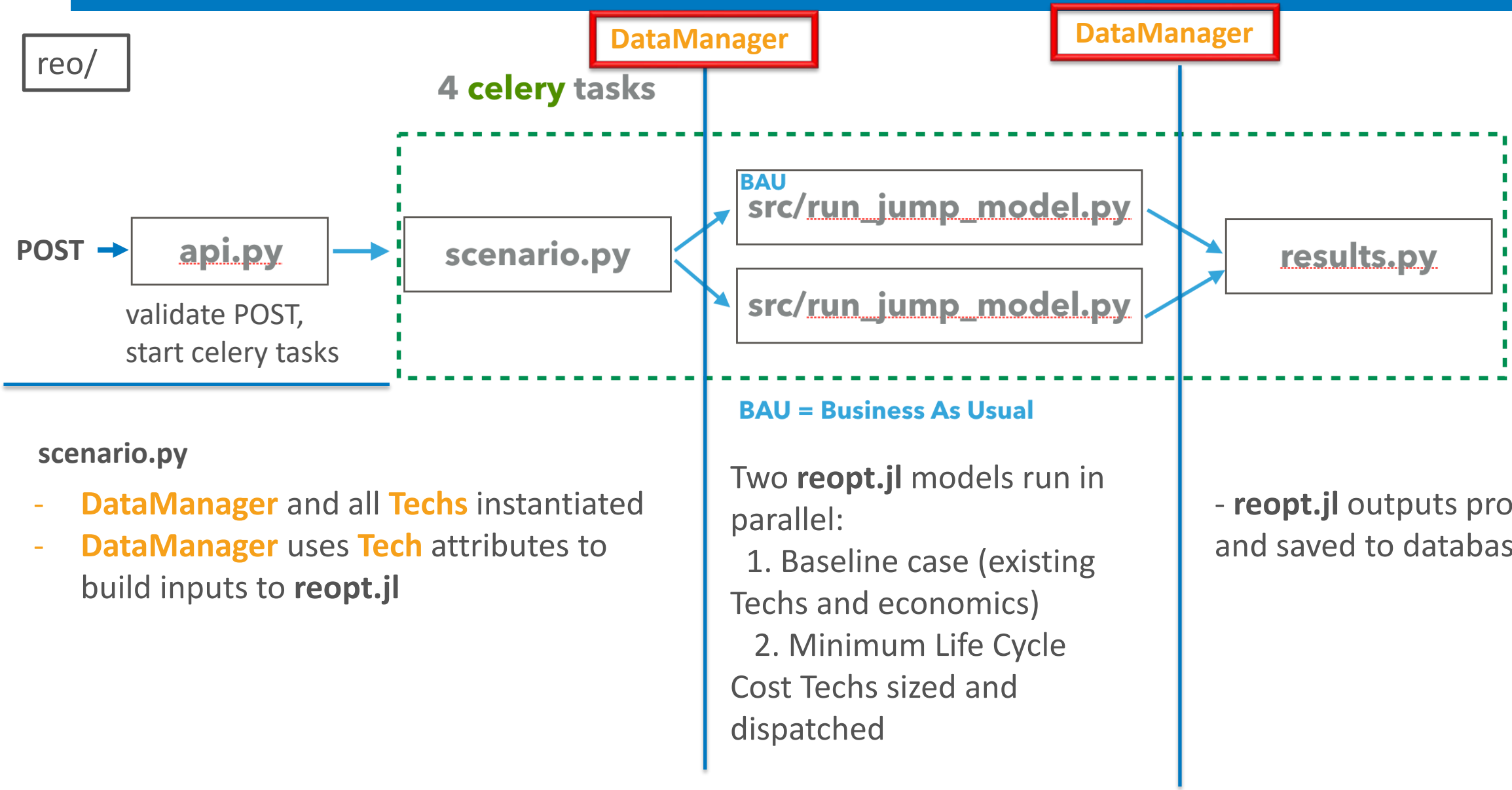
- **DataManager** and all **Techs** instantiated
- **DataManager** uses **Tech** attributes to build inputs to **reopt.jl**

Two **reopt.jl** models run in parallel:

1. Baseline case (existing Techs and economics)
2. Minimum Life Cycle Cost Techs sized and dispatched

- **reopt.jl** outputs processed and saved to database

REopt Lite Data-Flow



scenario.py

- **DataManager** and all **Techs** instantiated
- **DataManager** uses **Tech** attributes to build inputs to **reopt.jl**

Two **reopt.jl** models run in parallel:

1. Baseline case (existing Techs and economics)
2. Minimum Life Cycle Cost Techs sized and dispatched

- **reopt.jl** outputs processed and saved to database

REopt Lite **DataManager**

reo/src/**data_manager.py**

```
class DataManager:
    """
    Creates input dicts for reopt.jl and manages data transfer between Celery tasks
    """
    self.available_techs = ['pv', 'pvnm', 'wind', 'windnm', 'generator', 'util']
    self.available_tech_classes = ['PV', 'WIND', 'GENERATOR', 'UTIL']
    self.available_loads = ['retail', 'wholesale', 'export', 'storage']
```

DataManager uses **Tech** attributes to build inputs to **reopt.jl**

(More on ‘pvnm’ and ‘windnm’ later…)

REopt Lite **Tech** objects

reo/src/**techs.py**

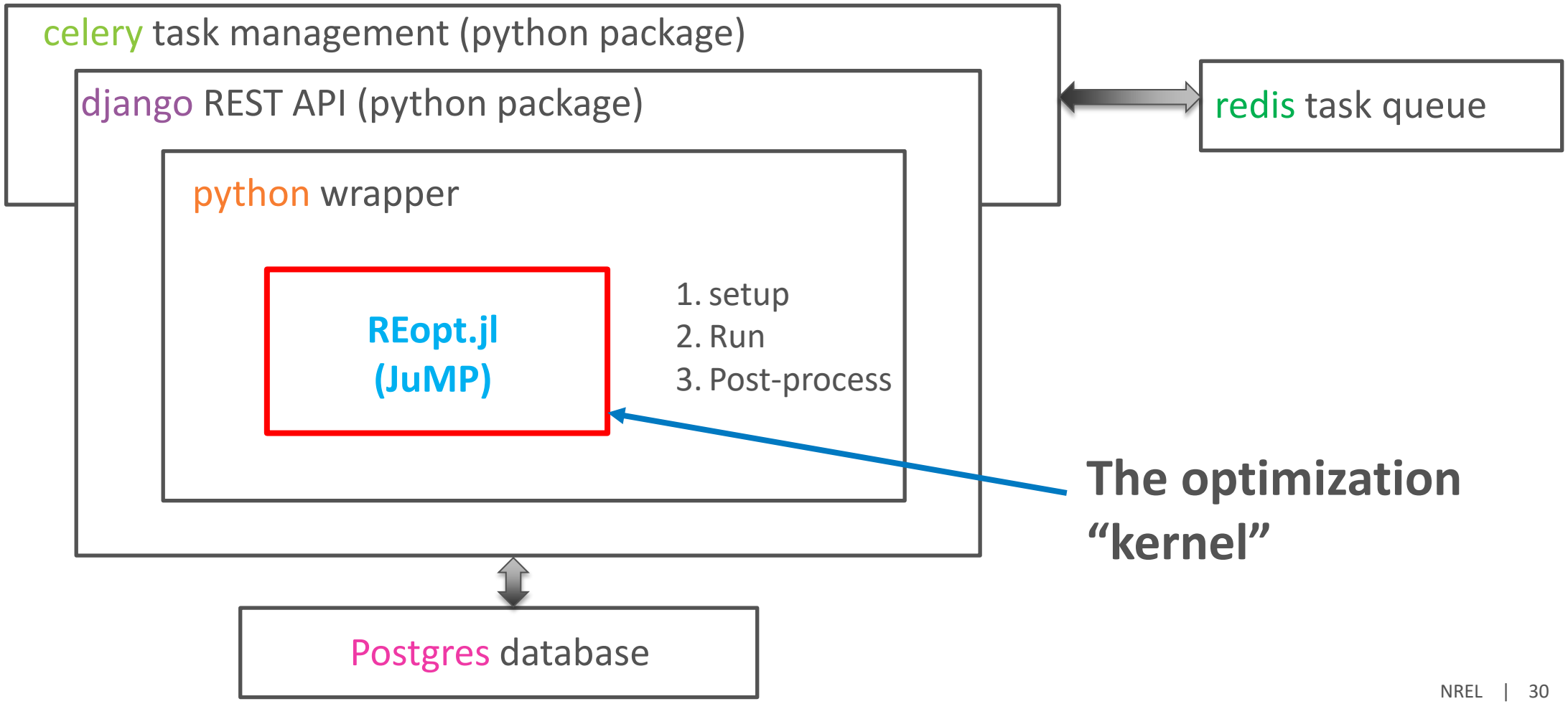
```
class Tech(object):
    """
    base class for REopt energy generation technology
    """
    self.loads_served = ['retail', 'wholesale', 'export', 'storage']
  ...
class Wind(Tech)
  ...
class PV(Tech)
  ...
class Generator(Tech)
  ...
class Util(Tech)
    self.loads_served = ['retail', 'storage']
```

- Overrides the base class attribute
- Utility cannot serve wholesale or export “Loads”

REopt Lite Formulation

Mixed Integer Linear Program (MILP)

REopt Lite API Architecture



REopt Lite MILP Overview

Objective: Minimize life cycle cost of energy

- Uses one year of resource and cost data with present worth factors to account for life-time costs
- Assumes that one year repeats, with degradation and escalation factors

REopt Lite MILP Overview

Objective: Minimize life cycle cost of energy

- Uses one year of resource and cost data with present worth factors to account for life-time costs
- Assumes that one year repeats, with degradation and escalation factors

Energy balance at every time step for entire year

- Load must be met from some combination of grid purchases, on-site generation, or discharge from storage
- Does not consider power flow or transient effects

REopt Lite MILP Overview

Objective: Minimize life cycle cost of energy

- Uses one year of resource and cost data with present worth factors to account for life-time costs
- Assumes that one year repeats, with degradation and escalation factors

Energy balance at every time step for entire year

- Load must be met from some combination of grid purchases, on-site generation, or discharge from storage
- Does not consider power flow or transient effects

Technology modules based on empirical operating data

- Finds optimal technology sizes (can be zero) and optimal dispatch strategy subject to resource, operating, and market constraints

Example from REopt's MILP

reo/src/reopt.jl

"1R" = retail load

```
@constraint(REopt, [LD in ["1R"], ts in p.TimeStep],
  sum(dvRatedProd[t,LD,ts,s,fb] * p.ProdFactor[t,LD,ts] * p.LevelizationFactor[t]
  for t in p.Tech, s in p.Seg, fb in p.FuelBin) + dvElecFromStor[ts] >= p.LoadProfile[LD,ts])
```

"p" = parameter struct (input values)

"dv" = decision variable

Example from REopt's MILP

reo/src/reopt.jl

"1R" = retail load

```
@constraint(REopt, [LD in ["1R"], ts in p.TimeStep],
  sum(dvRatedProd[t,LD,ts,s,fb] * p.ProdFactor[t,LD,ts] * p.LevelizationFactor[t]
  for t in p.Tech, s in p.Seg, fb in p.FuelBin) + dvElecFromStor[ts] >= p.LoadProfile[LD,ts])
```

"p" = parameter struct (input values)

"dv" = decision variable

This constraint says that:

- **the sum of all *Techs'* rated production * resource available * life-cycle-factor**
- **plus discharge from storage**
- **must be >= the "retail" load.**

Example from REopt's MILP

reo/src/reopt.jl

Binary decision variable, dimensioned on Tech and TechClass

```
@variables REopt begin
    binSingleBasicTech[p.Tech,p.TechClass], Bin
```

Example from REopt's MILP

reo/src/reopt.jl

Binary decision variable, dimensioned on Tech and TechClass

```
@variables REopt begin
  binSingleBasicTech[p.Tech,p.TechClass], Bin
...
@constraint(REopt, [b in p.TechClass],
            sum(binSingleBasicTech[t,b] for t in p.Tech) <= 1)

@constraint(REopt, [t in p.Tech, b in p.TechClass],
            sum(dvSystemSize[t, s] * p.TechToTechClassMatrix[t, b] for s in p.Seg)
                                    <= p.MaxSize[t] * binSingleBasicTech[t, b])
```

Example from REopt's MILP

reo/src/reopt.jl

Binary decision variable, dimensioned on Tech and TechClass

```
@variables REopt begin
  binSingleBasicTech[p.Tech,p.TechClass], Bin
...
@constraint(REopt, [b in p.TechClass],
            sum(binSingleBasicTech[t,b] for t in p.Tech) <= 1)

@constraint(REopt, [t in p.Tech, b in p.TechClass],
            sum(dvSystemSize[t, s] * p.TechToTechClassMatrix[t, b] for s in p.Seg)
                                    <= p.MaxSize[t] * binSingleBasicTech[t, b])
```

"dv" = decision variable

"p" = parameter struct (input values)

Together, these constraints say: Can only choose one Tech from each TechClass

Example from REopt's MILP

reo/src/reopt.jl

```
@variables REopt begin
    binSingleBasicTech[p.Tech,p.TechClass], Bin
...
@constraint(REopt, [b in p.TechClass],
            sum(binSingleBasicTech[t,b] for t in p.Tech) <= 1)

@constraint(REopt, [t in p.Tech, b in p.TechClass],
            sum(dvSystemSize[t, s] * p.TechToTechClassMatrix[t, b] for s in p.Seg)
                                    <= p.MaxSize[t] * binSingleBasicTech[t, b])
```

Example from REopt's MILP

reo/src/reopt.jl

"**Tech**": generic technology that can meet load(s)

All **Tech**s (can) have:

-capital, fuel, and O&M costs

- production- and capacity-based incentives

- tax benefits

REopt Lite includes **Tech**'s for:

Wind, Generator, PV, Utility

Example from REopt's MILP

reo/src/reopt.jl

"**Tech**": generic technology that can meet load(s)

All **Tech**'s (can) have:

-capital, fuel, and O&M costs

- production- and capacity-based incentives

- tax benefits

REopt Lite includes **Tech**s for:

Wind, Generator, PV, Utility

"**TechClass**": abstract group of Techs used for constraints

Example constraint: two choices for PV system

1. PV eligible for net metering benefits, that can not generate more than annual site consumption
2. PV *not* eligible for net metering, that can generate an "unlimited" amount of energy

Example from REopt's MILP

reo/src/reopt.jl

"**Tech**": generic technology that can meet load(s)

All **Tech**'s (can) have:

-capital, fuel, and O&M costs

- production- and capacity-based incentives
- tax benefits

REopt Lite includes **Tech**s for:

Wind, Generator, PV, Utility

"**TechClass**": abstract group of Techs used for constraints

Example constraint: two choices for PV system

1. PV eligible for net metering benefits, that can not generate more than annual site consumptions
2. PV *not* eligible for net metering, that can generate an "unlimited" amount of energy

`p.TechToTechClassMatrix`

Tech:	TechClass: PV	WIND
pv	1	0
pvnm	1	0
wind	0	1
windnm	0	1

Contribution guidelines

Adding a Tech to REopt Lite

https://github.com/NREL/REopt_Lite_API/**wiki**

1. How to define a new model?
2. Where will the new model reside?
3. How does the version-control thing work?
4. How to test if the intended functionality is working within the model?
5. How to *actually* code a new feature?!
6. How do I know if the feature is working as intended?

Welcome to the REopt Lite API Wiki!

1. Structure of the API
2. Installing the API
   - Installing in a Docker container
   - Installing on a Mac computer
   - Installing in Ubuntu 18.04 and Windows via Virtual Box
3. Developing the API
4. Contribution guidelines
5. REopt mathematical model documentation

Contributing to REopt Lite

github.com/NREL/REopt_Lite_API/blob/develop/CONTRIBUTING.md

How Can I Contribute?

Reporting Bugs

Please create an issue for a bug that you have encountered. (T

Suggesting Enhancements

First, please check the REopt Lite Development Plan in case y

Please refer to the instructions in the issue template provided i

Note that not all suggestions will be integrated into the public
community of REopt Lite API users as well as preserve backwa
limited so we can not create every suggested enhancement.

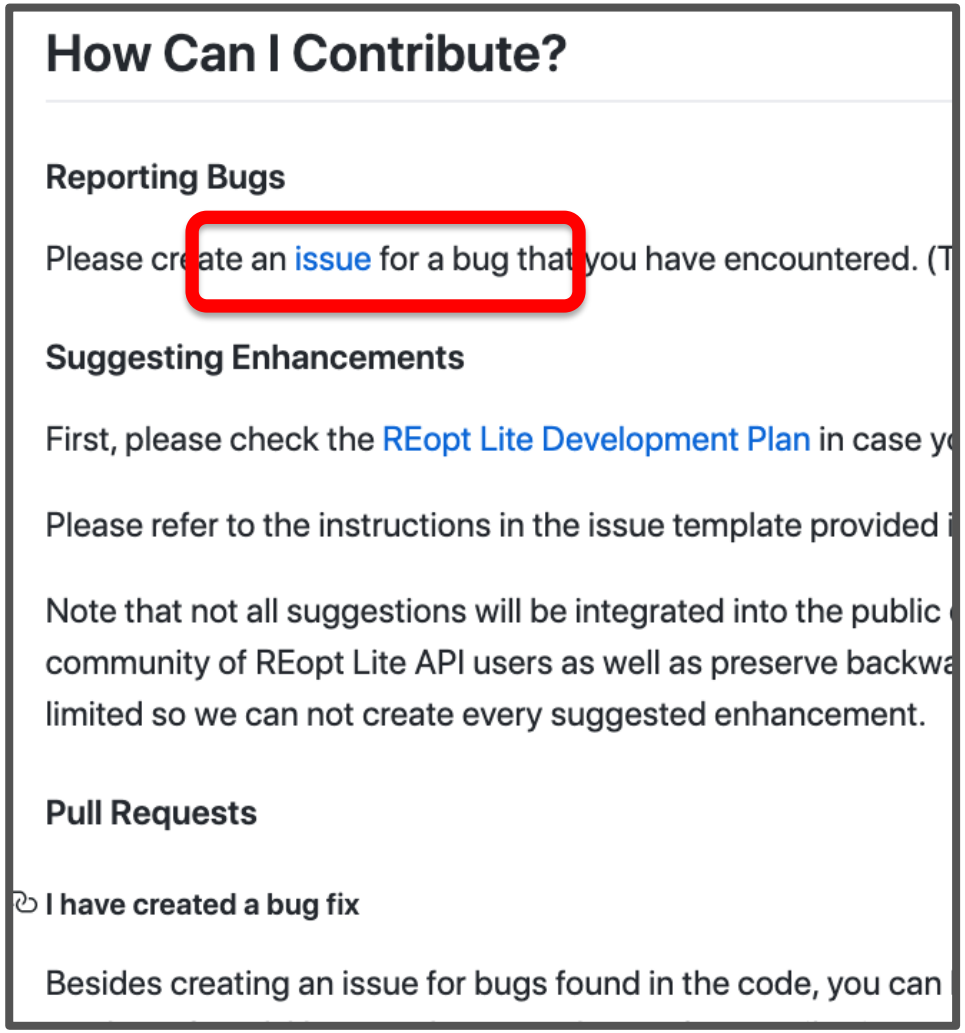
Pull Requests

I have created a bug fix

Besides creating an issue for bugs found in the code, you can

Contributing to REopt Lite

github.com/NREL/REopt_Lite_API/blob/develop/CONTRIBUTING.md



How Can I Contribute?

Reporting Bugs

Please create an issue for a bug that you have encountered. (T

Suggesting Enhancements

First, please check the REopt Lite Development Plan in case y

Please refer to the instructions in the issue template provided i

Note that not all suggestions will be integrated into the public
community of REopt Lite API users as well as preserve backwa
limited so we can not create every suggested enhancement.

Pull Requests

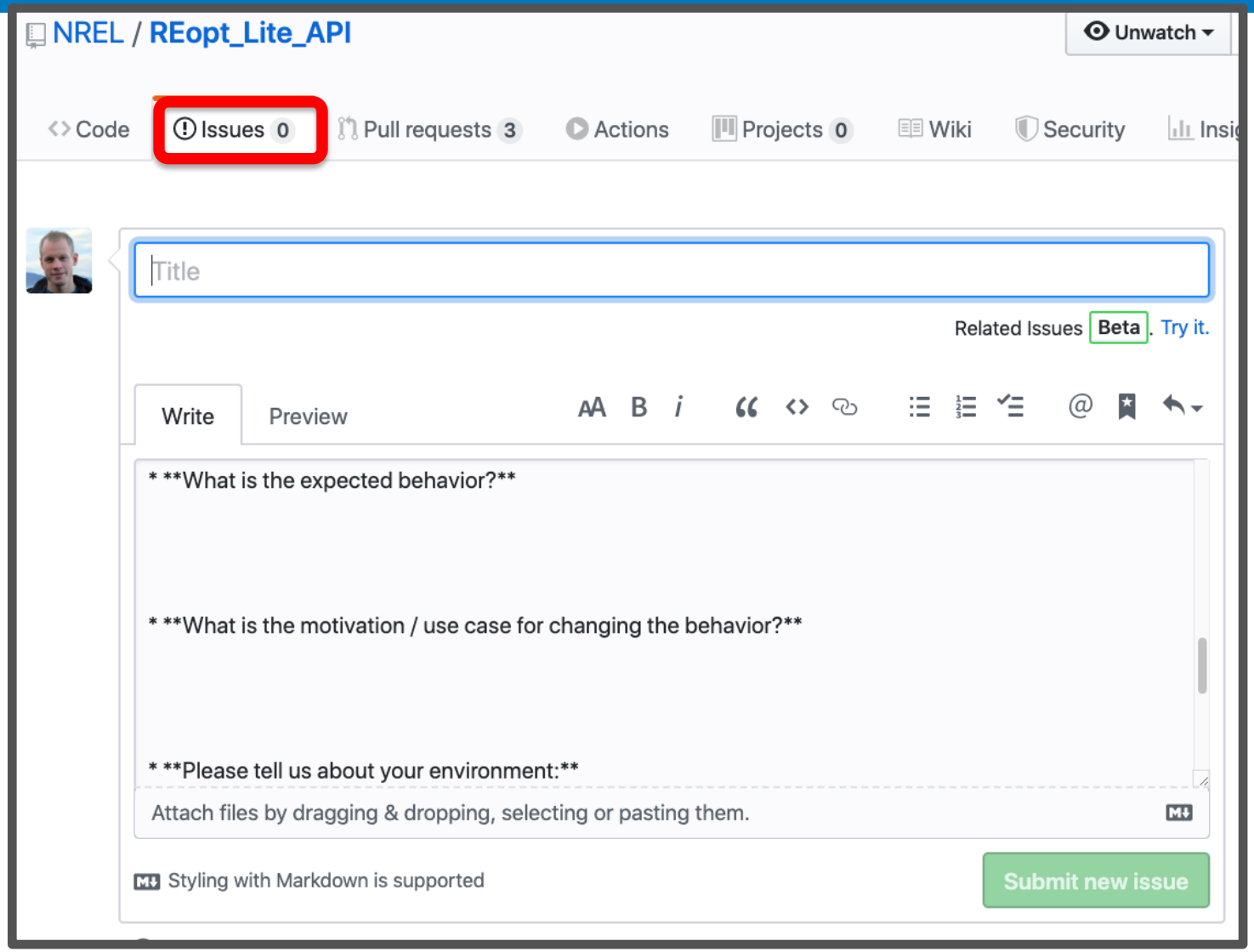
I have created a bug fix

Besides creating an issue for bugs found in the code, you can

Contributing to REopt Lite

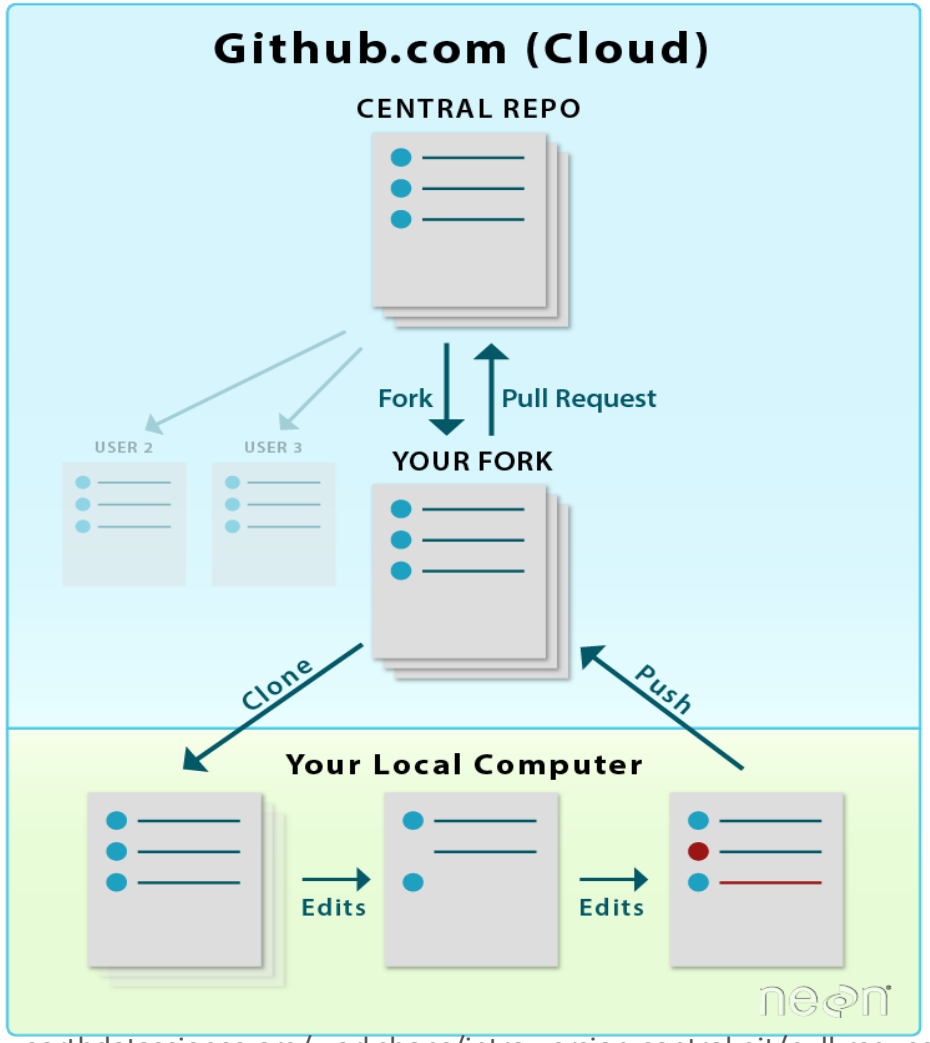
Issues:

- Bug reports
- Feature Requests



Contributing to REopt Lite

Add your own feature or bug fix via **Pull Request**



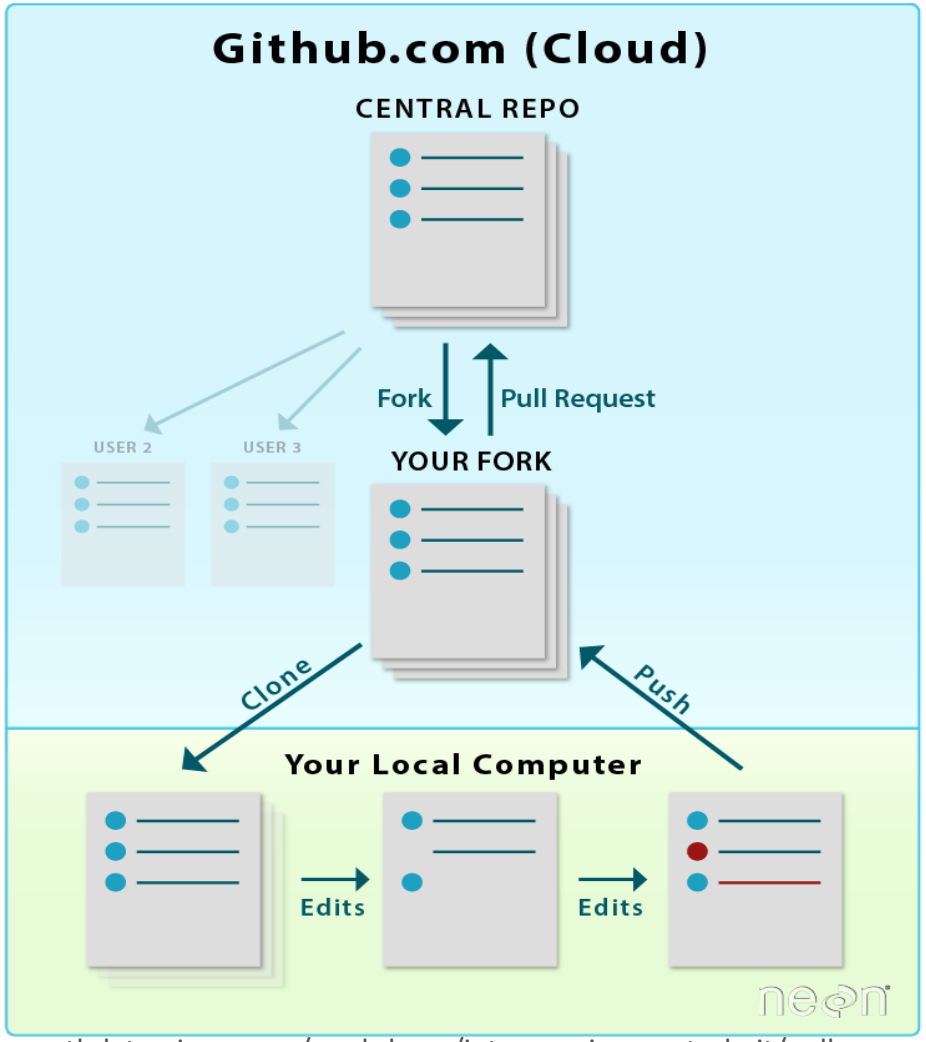
earthdatascience.org/workshops/intro-version-control-git/pull-request/

Contributing to REopt Lite

Add your own feature or bug fix via **Pull Request**

Test-based development

- Leveraging Github Actions



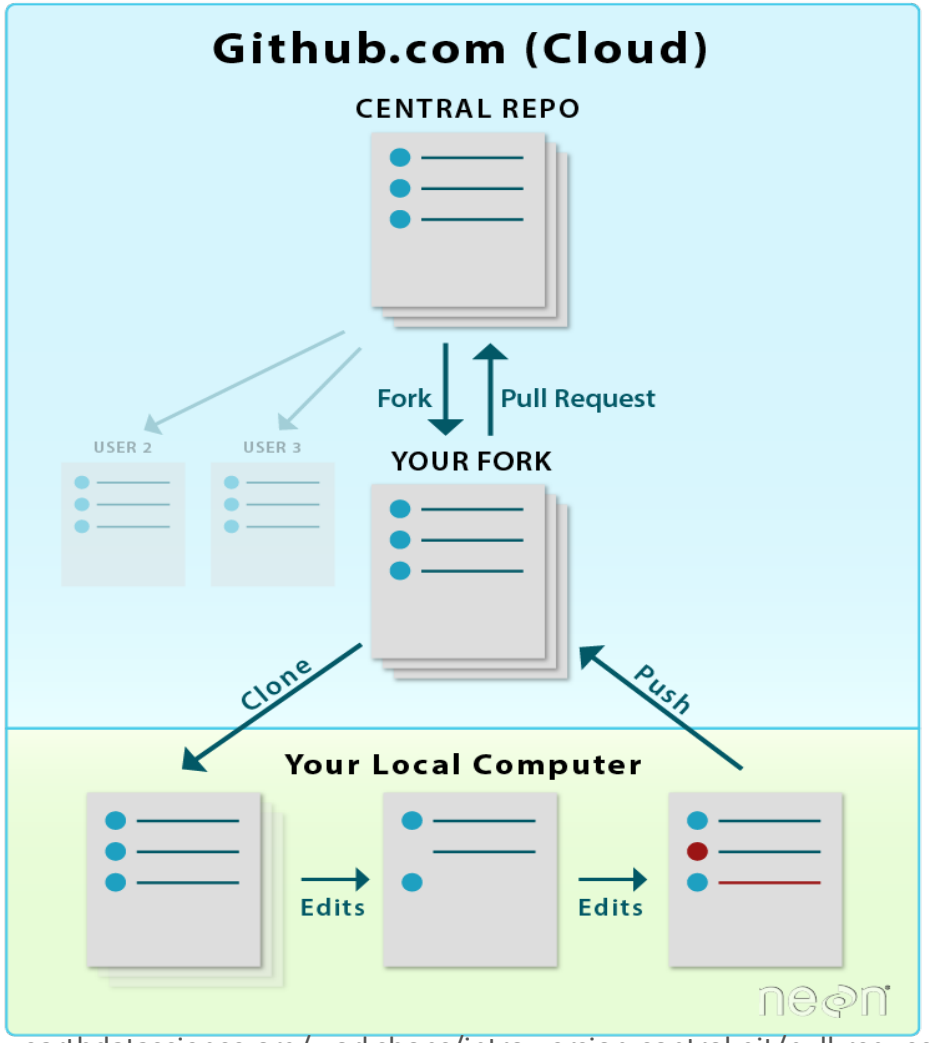
earthdatascience.org/workshops/intro-version-control-git/pull-request/

Contributing to REopt Lite

Add your own feature or bug fix via
Pull Request

Test-based development
- Leveraging Github Actions

Not every Pull Request will be merged
- Backwards compatibility
- Wider user-community benefit
- NREL Internal review



earthdatascience.org/workshops/intro-version-control-git/pull-request/

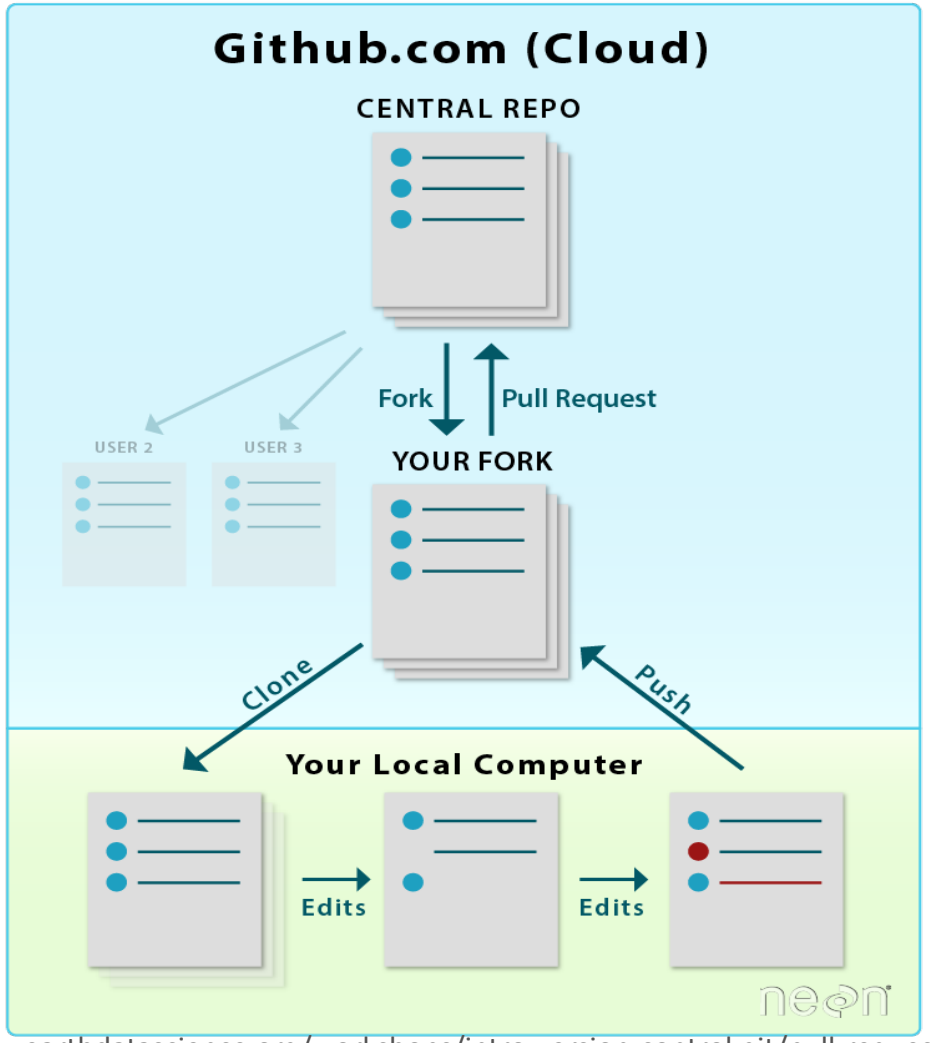
Contributing to REopt Lite

Add your own feature or bug fix via **Pull Request**

Test-based development
- Leveraging Github Actions

Not every Pull Request will be merged
- Backwards compatibility
- Wider user-community benefit
- NREL Internal review

Forum for community support coming



earthdatascience.org/workshops/intro-version-control-git/pull-request/

Other Resources

- **REopt Lite:** https://reopt.nrel.gov/tool
  - Tool: https://reopt.nrel.gov/tool/
  - Help manual: https://reopt.nrel.gov/user-guides.html
  - Accessing REopt API with python: https://github.com/NREL/REopt-API-Analysis
- **REopt Open Source:** https://github.com/NREL/REopt_Lite_API
  - Wiki with more information: https://github.com/nrel/reopt_api/wiki
- **REopt Website:** https://reopt.nrel.gov
  - Analysis services: https://reopt.nrel.gov/analysis/index.html
  - Case studies: https://reopt.nrel.gov/projects/index.html
- **Send tool feedback & ask a question: reopt@nrel.gov**

Share Your Stories!

Do you love REopt Lite? Are you excited to dig into the source code? Share with us how you are using REopt Lite, and your plans for REopt Lite Open Source (reopt@nrel.gov). Testimonials like yours help us secure funding to continue to develop this great tool.

Thank You

Sakshi Mishra Sakshi.Mishra@nrel.gov

Nick Laws Nick.Laws@nrel.gov

www.nrel.gov

NREL/PR-7A40-76408

- REopt Lite (tool and help manual): https://reopt.nrel.gov/tool
- REopt Website (analysis services and case studies): https://reopt.nrel.gov/
- Send tool feedback & ask a question: reopt@nrel.gov

This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Federal Energy Management Program. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.