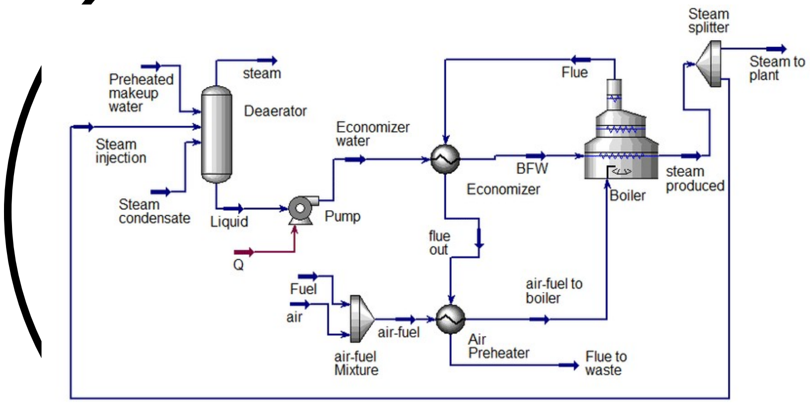
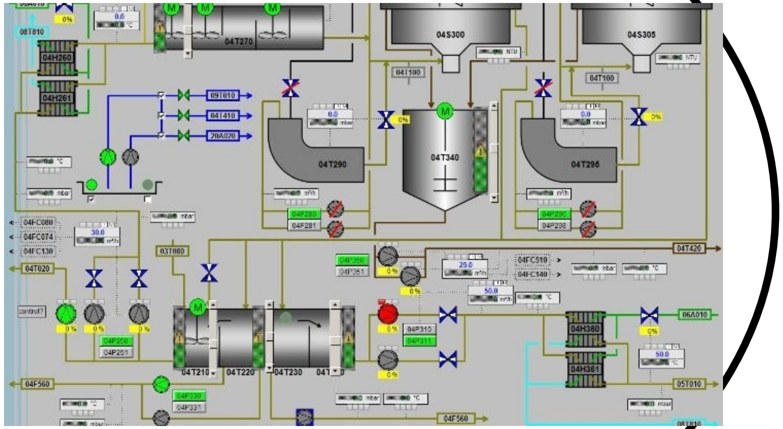
# Live Data in the Ahuora Digital Twin Platform

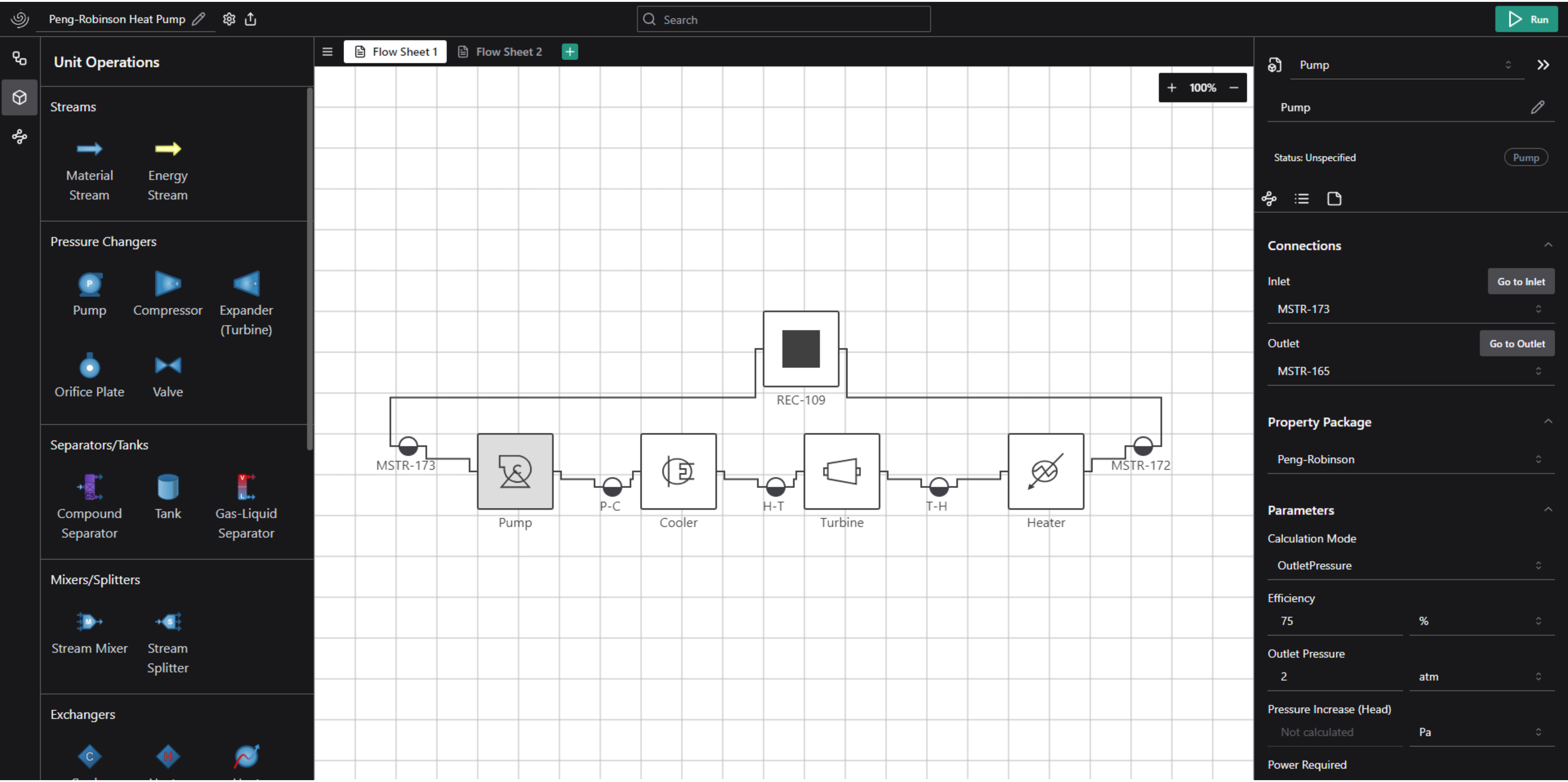
Design



Digital Twins?

Use





Pump

Pump

Status: Unspecified

Pump

Connections

Inlet: MSTR-173

Outlet: MSTR-165

Property Package

Peng-Robinson

Parameters

Calculation Mode: OutletPressure

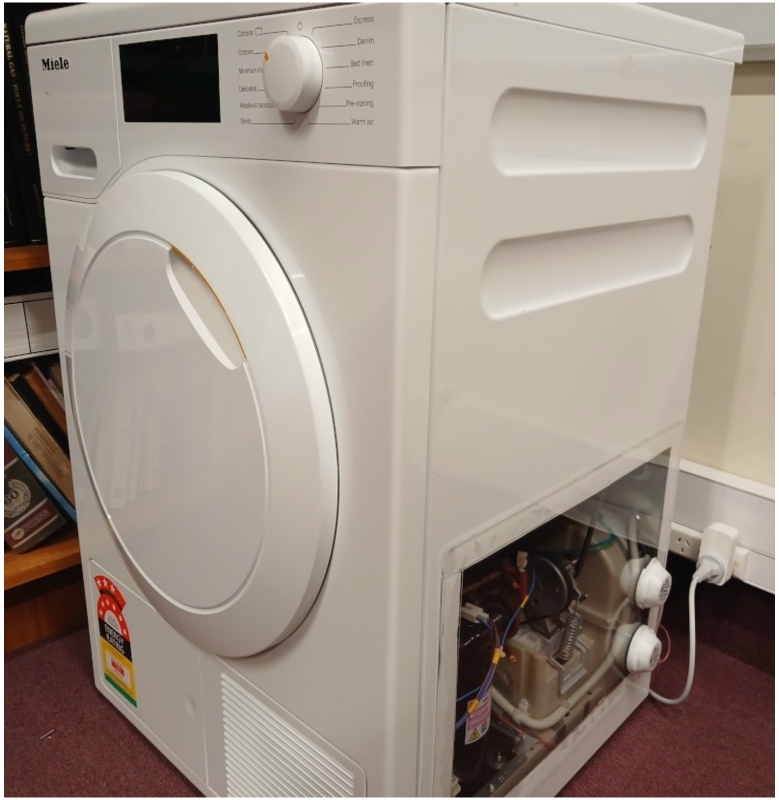
Efficiency: 75 %

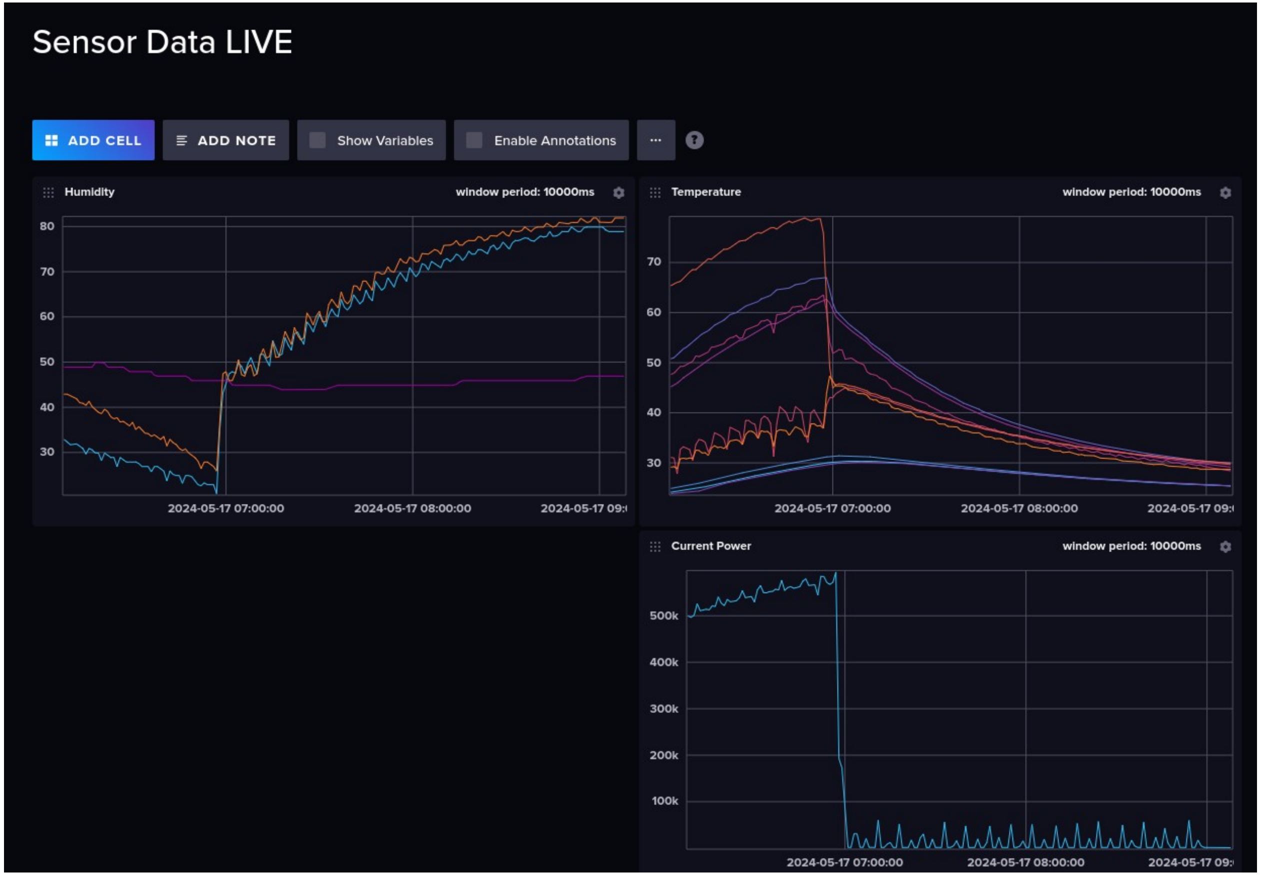
Outlet Pressure: 2 atm

Pressure Increase (Head): Not calculated Pa

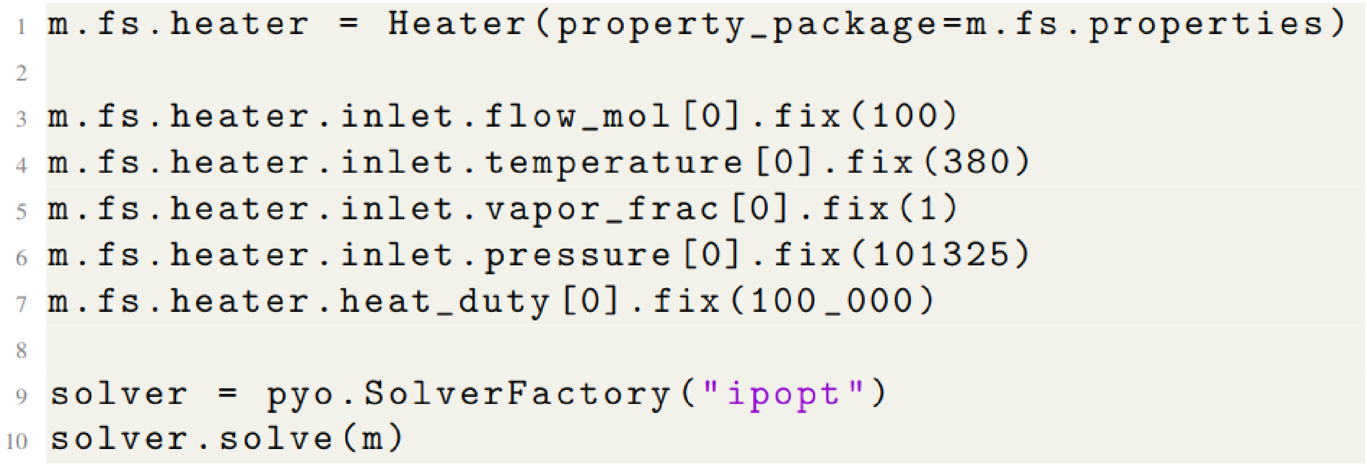
Power Required

# Data Collection





# Modelling

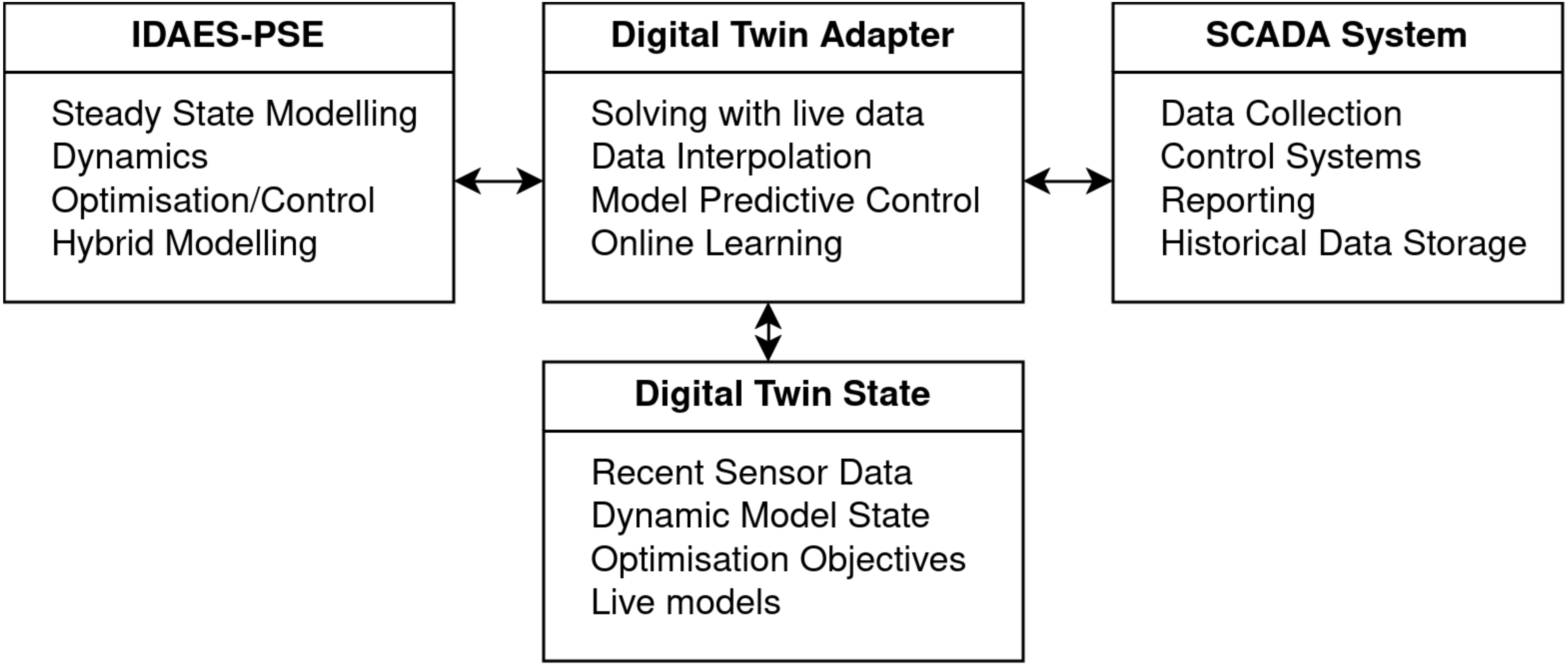


```
m.fs.heater = Heater(property_package=m.fs.properties)

m.fs.heater.inlet.flow_mol[0].fix(100)
m.fs.heater.inlet.temperature[0].fix(380)
m.fs.heater.inlet.vapor_frac[0].fix(1)
m.fs.heater.inlet.pressure[0].fix(101325)
m.fs.heater.heat_duty[0].fix(100_000)

solver = pyo.SolverFactory("ipopt")
solver.solve(m)
```

# Architecture



**IDAES-PSE**

Steady State Modelling
Dynamics
Optimisation/Control
Hybrid Modelling

**Digital Twin Adapter**

Solving with live data
Data Interpolation
Model Predictive Control
Online Learning

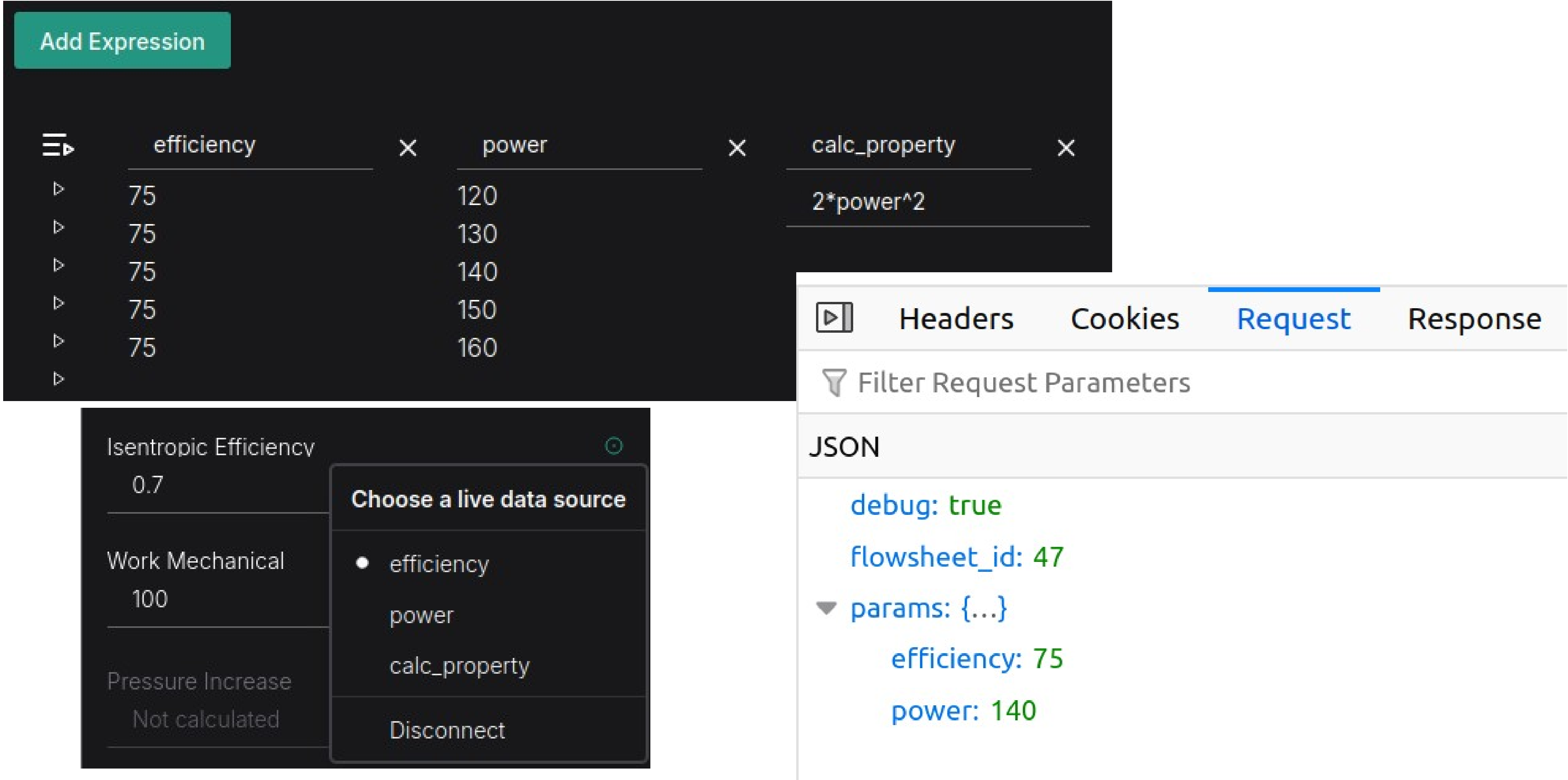
**SCADA System**

Data Collection
Control Systems
Reporting
Historical Data Storage

**Digital Twin State**

Recent Sensor Data
Dynamic Model State
Optimisation Objectives
Live models

# Development



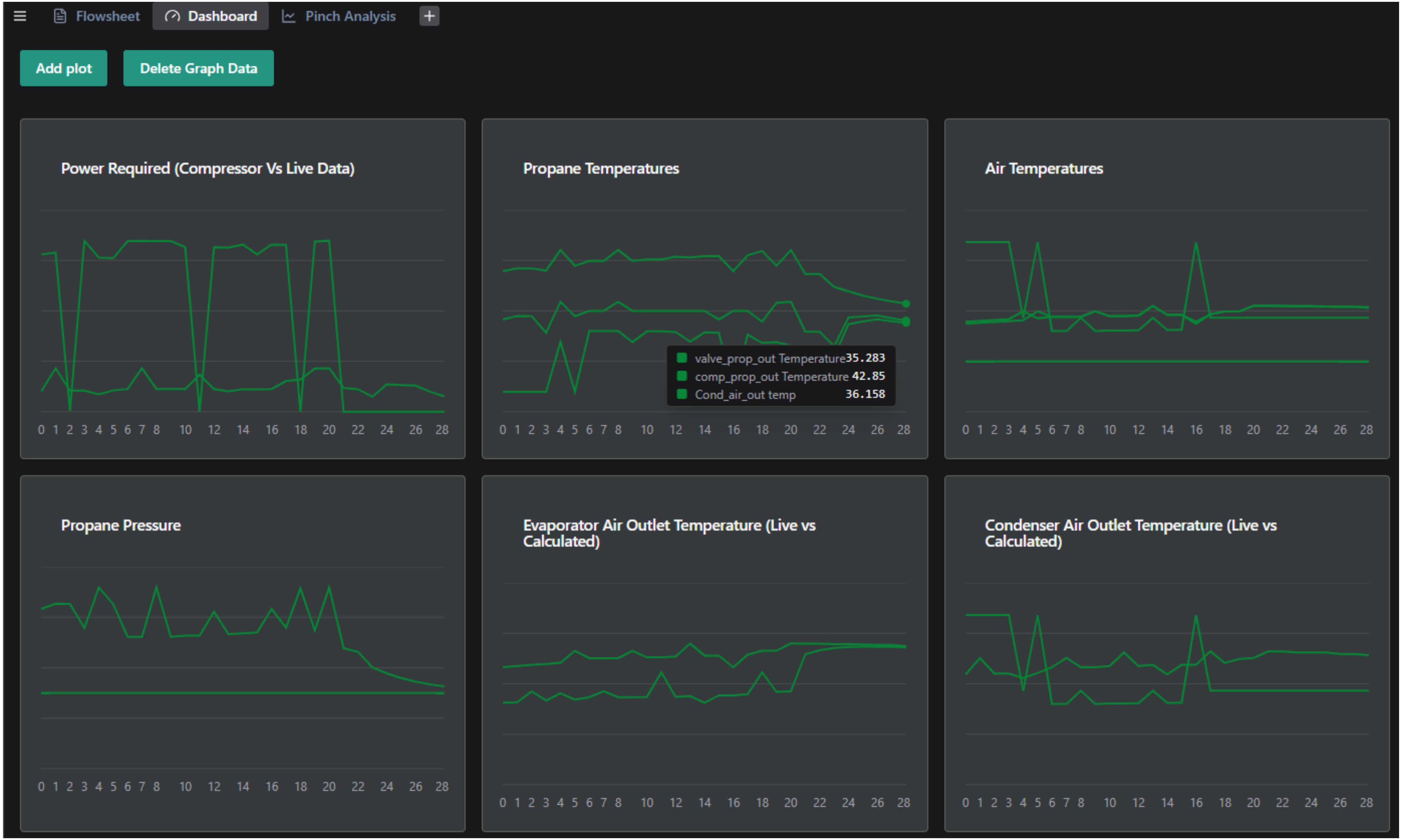
| efficiency | power | calc_property |
|---|---|---|
| 75 | 120 | 2*power^2 |
| 75 | 130 | |
| 75 | 140 | |
| 75 | 150 | |
| 75 | 160 | |

Isentropic Efficiency
0.7

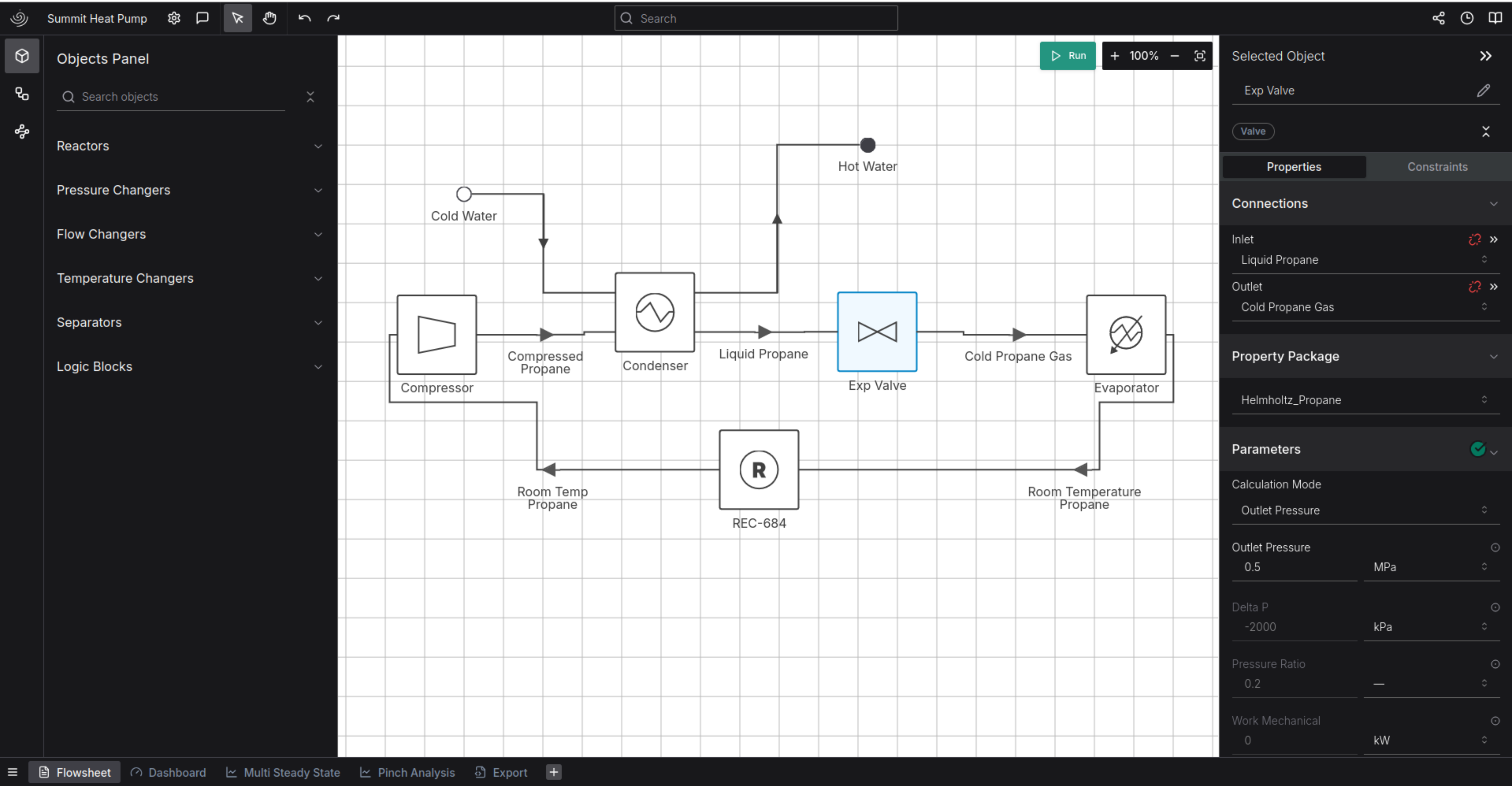
Work Mechanical
100

Pressure Increase
Not calculated

```
debug: true
flowsheet_id: 47
params: {…}
    efficiency: 75
    power: 140
```



```
🟩 valve_prop_out Temperature35.283
🟩 comp_prop_out Temperature 42.85
🟩 Cond_air_out temp         36.158
```



Selected Object

Exp Valve

Valve

Properties | Constraints

Connections

Inlet: Liquid Propane

Outlet: Cold Propane Gas

Property Package: Helmholtz_Propane

Parameters

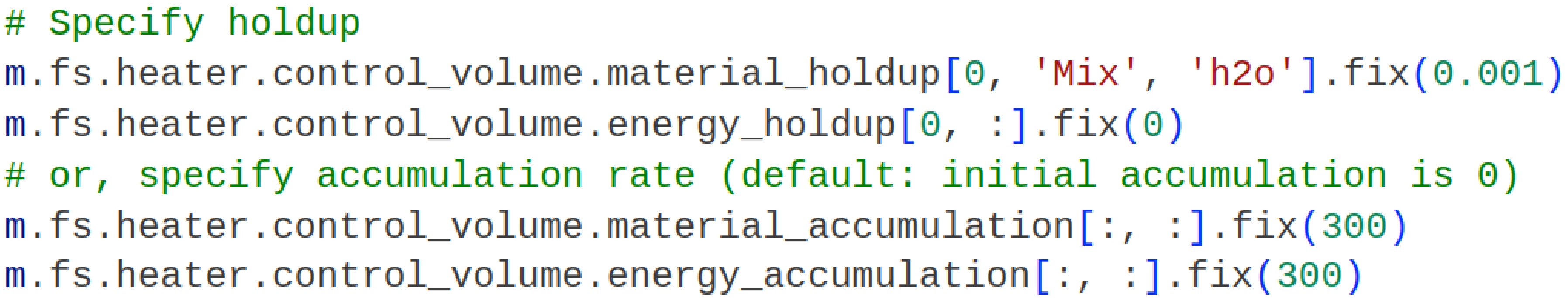
Calculation Mode: Outlet Pressure

Outlet Pressure: 0.5 MPa

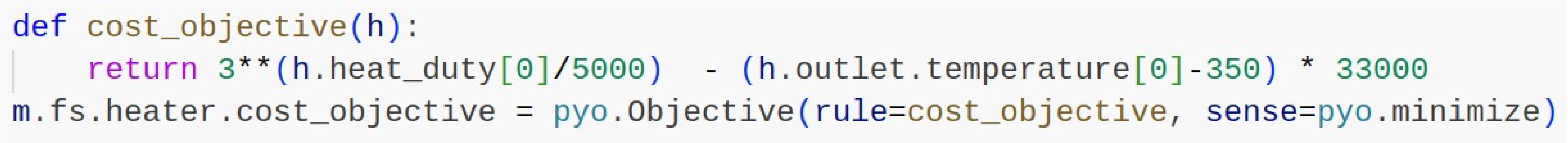
Delta P: -2000 kPa

Pressure Ratio: 0.2 —

Work Mechanical: 0 kW



```
# Specify holdup
m.fs.heater.control_volume.material_holdup[0, 'Mix', 'h2o'].fix(0.001)
m.fs.heater.control_volume.energy_holdup[0, :].fix(0)
# or, specify accumulation rate (default: initial accumulation is 0)
m.fs.heater.control_volume.material_accumulation[:, :].fix(300)
m.fs.heater.control_volume.energy_accumulation[:, :].fix(300)
```



```python
def cost_objective(h):
    return 3**(h.heat_duty[0]/5000)  - (h.outlet.temperature[0]-350) * 33000
m.fs.heater.cost_objective = pyo.Objective(rule=cost_objective, sense=pyo.minimize)
```

```
model = PysmoSurrogate.load_from_file('pysmo_heater_surrogate.json')
inputs = [self.inlet.pressure, self.inlet.temperature, self.heat_duty, self.inlet.flow_mol ]
outputs = [self.outlet.pressure,self.outlet.temperature,self.outlet_vapor]
self.surrogate = SurrogateBlock(concrete=True)
self.surrogate.build_model(model,input_vars=inputs, output_vars=outputs)
```