Experiments 2 - Market 1

In this problem, four energy producers participate in a day-ahead electricity market, including three conventional ones and a renewable one. These producers submit hourly bids that specify a delivery quantity of electricity and an offer price for each hour of the next day. The market operator uses these bids to determine the market-clearing price and dispatch schedule. If a bid is accepted, the producer is obligated to deliver the specified quantity of electricity at the offer price. Otherwise, the producer earns zero profit for that hour.

However, renewable generation is deeply uncertain, as it depends on weather conditions that cannot be predicted. If the producer under-delivers relative to its bid (due to overestimation of production), or over-delivers (due to underestimation), penalties or imbalance costs would apply. These costs are imposed by the system operator to maintain grid stability and to ensure supply-demand balance in real time. Meanwhile, the clearing price is unknown to the producer in advance, so they need to compete in this market.

The renewable energy producer's goal is to maximize its expected profit by strategically selecting the quantity of electricity to bid for each hour of the next day, accounting for the uncertainty in both renewable generation and clearing prices.

Market structure and timing:
1. The day-ahead market is cleared once per day (e.g., at noon) for 24 hourly intervals of the following day.
2. Each producer submits a bid (b_t, p_t) for each hour $t=1, ..., 24$, where:
2.1. b_t: the quantity of energy (in MWh) the producer is willing to supply
2.2. p_t: the minimum acceptable price per MWh
3. The market operator computes the market-clearing price c_t for each hour.
4. If $p_t \leq c_t$, the bid is accepted, and the producer is committed to delivering b_t MWh in hour t.
5. The total energy demand in the market is D_t, which follows a normal distribution with constant mean and variance.

Penalty structure:
1. If actual generation $P_t < b_t$, the shortfall $b_t - P_t$ incurs a penalty of q_u per MWh.
2. If actual generation $P_t \geq b_t$, the surplus $P_t - b_t$ incurs a curtailment penalty of q_o per MWh.

Conventional producers:
1. For each conventional producer, the delivery quantity of electricity and the offer price in bids follow normal distributions with constant means and variances.

Please formalize this problem description as a model. Please also specify key components, such as the state variables, initial state, uncertainties and transition functions.

Excellent — this is a **renewable bidding problem** under **market and production uncertainty**, interacting with other (probabilistic) producers in a **day-ahead market**.

Let's now formalize it rigorously as a **stochastic optimization model** that fully specifies the environment, state variables, controls, and uncertainty dynamics.

🧩 1. Environment

The market operates over discrete time steps

$$t = 1, 2, \ldots, 24$$

representing the hourly intervals of the next day.

All producers share the same **market environment** characterized by:

- Total demand D_t (random)
- Market-clearing price c_t (random, determined by bids)
- Renewable generation P_t (random, weather-driven)
- Penalty parameters q_u, q_o
- Hourly bidding decision (b_t, p_t)

🧭 2. Decision Maker (Renewable Producer)

At each hour t, the **renewable producer** decides:

$$\text{Controls: } u_t = (b_t, p_t)$$

where

- $b_t \in \mathbb{R}_+$: bid quantity (MWh)
- $p_t \in \mathbb{R}_+$: offer price (USD/MWh)

The decisions are made **before** realization of uncertainties (c_t, P_t).

🌩️ 3. Uncertainties

- **Market clearing price**:

  $$c_t \sim f_c(\cdot \mid D_t, \text{other producers' bids})$$

  random variable representing the market outcome.
- **Renewable generation**:

  $$P_t \sim f_P(\cdot)$$

  stochastic output driven by weather (wind, solar, etc.).
- **Demand**:

$$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$$

All random variables are assumed **independent across hours**, unless specified otherwise (we could later add temporal correlation).

⚙️ 4. State Variables

We define the system state at time t as:

$$s_t = (D_t, c_t, P_t)$$

For the renewable producer, **no intertemporal coupling** exists (each hour independent), but for generality we include:

$$x_t = (b_{t-1}, P_{t-1}, \Pi_{t-1})$$

where Π_{t-1} is cumulative profit up to hour $t-1$.

🚀 5. Transition Functions

Each hour evolves as:

1. **Bidding** (decision step):

$$u_t = (b_t, p_t)$$

2. **Market clearing** (stochastic realization):

$$c_t = \phi(D_t, b_t^{(1)}, p_t^{(1)}, \ldots, b_t^{(N)}, p_t^{(N)})$$

   (where $N = 4$ producers, including the renewable one).

3. **Generation realization**:

$$P_t \sim f_P(\cdot)$$

4. **Profit update**:

$$\Pi_t = \Pi_{t-1} + \pi_t$$

   where instantaneous profit π_t is defined below.

Thus, the state transition function is:

$$x_{t+1} = f(x_t, u_t, \omega_t)$$

with $\omega_t = (c_t, P_t)$ representing the uncertainty realization.

💰 6. Profit and Penalty Structure

For each hour t:

Market acceptance condition

$$\delta_t = \begin{cases} 1, & \text{if } p_t \leq c_t \\ 0, & \text{otherwise} \end{cases}$$

Profit components

If accepted ($\delta_t = 1$):

$$\pi_t = c_t b_t - q_u \max(0, b_t - P_t) - q_o \max(0, P_t - b_t)$$

If not accepted ($\delta_t = 0$):

$$\pi_t = 0$$

Expected hourly profit:

$$\mathbb{E}[\pi_t] = \mathbb{E}\Big[\delta_t \Big(c_t b_t - q_u \max(0, b_t - P_t) - q_o \max(0, P_t - b_t)\Big)\Big]$$

🎯 7. Objective Function

The renewable producer aims to maximize expected total profit:

$$\max_{\{b_t, p_t\}_{t=1}^{24}} \mathbb{E}\left[\sum_{t=1}^{24} \pi_t(b_t, p_t; c_t, P_t)\right]$$

subject to:

$$b_t, p_t \geq 0, \quad \forall t$$

🌍 8. Initial State and Parameters

- $\Pi_0 = 0$
- P_0 known (or irrelevant)
- $D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$
- $q_u, q_o > 0$
- f_P, f_c given distributions (estimated from data)

🔗 9. Conventional Producers (for completeness)

For each conventional producer $i = 1, 2, 3$:

$$(b_t^{(i)}, p_t^{(i)}) \sim \mathcal{N}(\mu_b^{(i)}, \sigma_b^{(i)}), \quad \mathcal{N}(\mu_p^{(i)}, \sigma_p^{(i)})$$

These producers affect the **market-clearing price function**:

$$c_t = \Phi\big(D_t, \{b_t^{(i)}, p_t^{(i)}\}_{i=1}^4\big)$$

⚖️ 10. Model Summary

Component	Symbol	Description
Time	$t = 1, \ldots, 24$	Hour of next day
State	$x_t = (b_{t-1}, P_{t-1}, \Pi_{t-1})$	Internal/cumulative state
Controls	$u_t = (b_t, p_t)$	Renewable bid decision
Random vars	$\omega_t = (c_t, P_t)$	Market price and generation
Transition	$x_{t+1} = f(x_t, u_t, \omega_t)$	Profit update
Objective	$\max \mathbb{E}[\sum_t \pi_t]$	Expected profit maximization
Profit	$\pi_t = \delta_t[c_t b_t - q_u \max(0, b_t - P_t) - q_o \max(0, P_t - b_t)]$	Hourly profit
Market acceptance	$\delta_t = 1[p_t \leq c_t]$	Binary acceptance indicator
Demand	$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$	Market demand
Price formation	$c_t = \Phi(D_t, \text{bids})$	Market-clearing rule

---

Would you like me to extend this into a **multi-agent unified model** (including the three conventional producers and the renewable one), similar to your previous “Lake” tasks — with shared environment, transition functions, and perspective-specific objectives?

Various stakeholders approach this problem from different perspectives. Please identify four different perspectives related to this problem. These perspectives should share a common environment, along with global parameters, state variables, and an initial state. For each perspective, also provide the corresponding model with the key components, such as control variables, transition functions, and objective functions. If there are multiple objectives, provide multiple objective functions instead of a weighted objective function.

Excellent — let’s formalize this electricity market as a **shared-environment, multi-perspective system**. We’ll identify four stakeholders (each with distinct objectives and controls) but defined over **a single, consistent environment** that describes the **day-ahead market** and its uncertainties.

🌍 1. Common Environment

All perspectives interact through the same underlying market and physical system.

(a) Time horizon

$$t = 1, 2, \ldots, 24$$

(b) Global parameters

Symbol	Description
q_u	under-delivery penalty rate ($/MWh)
q_o	over-delivery (curtailment) penalty rate ($/MWh)
μ_D, σ_D^2	mean and variance of total demand distribution
$N = 4$	total number of producers (3 conventional, 1 renewable)
$\Phi(\cdot)$	market-clearing price function
$f_P(\cdot)$	renewable generation distribution
$\mu_b^{(i)}, \sigma_b^{(i)}$	conventional bid quantity parameters
$\mu_p^{(i)}, \sigma_p^{(i)}$	conventional offer price parameters

(c) State variables

$$s_t = (D_t, c_t, P_t)$$

where

- D_t: total demand (random)
- c_t: market-clearing price
- P_t: renewable generation output

(d) Initial state

$$s_0 = (\mathbb{E}[D_t],\ c_0,\ P_0)$$

where c_0 and P_0 are nominal baseline values.

(e) Transition dynamics

The state evolves as:

$$s_{t+1} = f(s_t, u_t, \omega_t)$$

where $u_t = \{u_t^{(i)}\}_{i=1}^4$ are all producers' controls,
and $\omega_t = (D_t, P_t)$ are stochastic shocks.

⚙️ 2. Shared Market Mechanism

At each hour t:

1. Each producer i submits a bid $u_t^{(i)} = (b_t^{(i)}, p_t^{(i)})$.
2. Market operator computes the **market-clearing price**:

$$c_t = \Phi(D_t, \{b_t^{(i)}, p_t^{(i)}\}_{i=1}^N)$$

3. Accepted bids ($p_t^{(i)} \leq c_t$) are dispatched, and profits or penalties accrue.

👥 3. Perspectives

We identify **four perspectives**:

1. Renewable Producer (stochastic production & strategic bidding)
2. Conventional Producer (cost-minimizing, capacity-constrained)
3. Market Operator (system welfare & balance)
4. Regulator / System Planner (policy and risk oversight)

Each shares the same environment but has unique **controls**, **transition functions**, and **objectives**.

🔹 Perspective 1: Renewable Producer

Control variables

$$u_t^{(R)} = (b_t^{(R)}, p_t^{(R)})$$

- $b_t^{(R)}$: bid quantity
- $p_t^{(R)}$: bid price

State variables

Shares $s_t = (D_t, c_t, P_t)$

Transition function

$$\Pi_{t+1}^{(R)} = \Pi_t^{(R)} + \pi_t^{(R)}(b_t^{(R)}, p_t^{(R)}, c_t, P_t)$$

Profit function

$$\pi_t^{(R)} = \begin{cases} c_t b_t^{(R)} - q_u \max(0, b_t^{(R)} - P_t) - q_o \max(0, P_t - b_t^{(R)}), & p_t^{(R)} \leq c_t \\ 0, & p_t^{(R)} > c_t \end{cases}$$

Objective

$$\max_{\{b_t^{(R)}, p_t^{(R)}\}} \mathbb{E}\left[\sum_{t=1}^{24} \pi_t^{(R)}\right]$$

🔹 Perspective 2: Conventional Producer $i \in \{1, 2, 3\}$

Control variables

$$u_t^{(i)} = (b_t^{(i)}, p_t^{(i)})$$

subject to production capacity $0 \le b_t^{(i)} \le \bar{B}^{(i)}$.

State variables

$$s_t = (D_t, c_t)$$

(generation deterministic, no weather dependence).

Cost structure

$$C_t^{(i)} = \alpha^{(i)} b_t^{(i)} + \beta^{(i)} (b_t^{(i)})^2$$

Transition

$$\Pi_{t+1}^{(i)} = \Pi_t^{(i)} + \pi_t^{(i)}, \quad \pi_t^{(i)} = \begin{cases} c_t b_t^{(i)} - C_t^{(i)}, & p_t^{(i)} \le c_t \\ 0, & p_t^{(i)} > c_t \end{cases}$$

Objective

$$\max_{\{b_t^{(i)}, p_t^{(i)}\}} \mathbb{E}\left[\sum_{t=1}^{24} \pi_t^{(i)}\right]$$

🔹 Perspective 3: Market Operator

Control variables

None (acts as system mechanism).
Determines market-clearing price via:

$$c_t = \Phi(D_t, \{b_t^{(i)}, p_t^{(i)}\}_{i=1}^N)$$

Objective functions

(1) Market balance constraint

$$\sum_{i=1}^N b_t^{(i)} \delta_t^{(i)} = D_t$$

(2) System welfare (social surplus)

$$W_t = \text{Consumer surplus} - \sum_{i=1}^N \text{Producer costs}$$

Objective

$$\max_{\{c_t\}} \mathbb{E}\left[\sum_{t=1}^{24} W_t\right]$$

subject to balance and dispatch constraints.

🔹 Perspective 4: Regulator / System Planner

This actor oversees system-wide reliability and efficiency, introducing **policy instruments**.

Control variables

$$u_t^{(G)} = (\tau_t, q_u, q_o)$$

where

- τ_t: carbon tax or renewable subsidy per MWh
- q_u, q_o: imbalance penalty rates

State variables

Aggregate market outcomes:

$$s_t^{(G)} = (c_t, \text{emissions}_t, \text{renewable share}_t)$$

Transition function

$$s_{t+1}^{(G)} = h(s_t^{(G)}, u_t^{(G)}, \{b_t^{(i)}\})$$

Objectives (multi-objective)

1. **Market efficiency**

   $$J_1 = -\operatorname{Var}(c_t)$$

   (stabilize prices)

2. **Reliability**

   $$J_2 = -\mathbb{E}\left[\sum_t |D_t - \sum_i b_t^{(i)} \delta_t^{(i)}|\right]$$

3. **Sustainability**

   $$J_3 = \mathbb{E}\left[\sum_t \text{renewable share}_t\right]$$

No weighted sum — the regulator evaluates each J_k separately.

🧩 4. Shared Structure Summary

Component	Shared Symbol	Description
Time	$t = 1, \ldots, 24$	Hourly intervals
State	$s_t = (D_t, c_t, P_t)$	Demand, price, renewable output
Transition	$s_{t+1} = f(s_t, u_t, \omega_t)$	Market evolution
Controls	$u_t^{(i)} = (b_t^{(i)}, p_t^{(i)})$	Producer bids
Common randomness	$\omega_t = (D_t, P_t)$	Demand and weather uncertainty
Initial state	$s_0 = (\mu_D, c_0, P_0)$	Starting market condition

---

🧮 5. Schematic of Interaction

```scss
Regulator sets (q_u, q_o, τ_t)
         ↓
Producers submit bids (b_t, p_t)
         ↓
Market operator clears market → price c_t
         ↓
Realizations (D_t, P_t) → payoffs, penalties
         ↓
State updates for next hour
```

🧠 6. Conceptual Summary

Perspective	Controls	Objective(s)	Core Uncertainty	Decision Horizon
Renewable Producer	$b_t^{(R)}, p_t^{(R)}$	Maximize expected profit	c_t, P_t	Hourly
Conventional Producer	$b_t^{(i)}, p_t^{(i)}$	Maximize expected profit	c_t, D_t	Hourly
Market Operator	—	Maximize welfare, enforce balance	D_t	Instantaneous
Regulator / Planner	τ_t, q_u, q_o	Multi-objective: efficiency, reliability, sustainability	Aggregate outcomes	Long-term

---

Would you like me to **compose these four perspectives into a unified modular model** next (with shared state and perspective-specific controls/objectives), as in your earlier “Lake” examples — and then provide the **Python class implementation** for that unified framework?

The second perspective is not consistent with the initial description, since I have specified that the bid quantity and bid price of conventional producers follow normal distributions with constant means and variances. These variables cannot be used as control variables. Please refine the model specification.

Excellent — that's a very important correction.
You're absolutely right: the **conventional producers' bids are *exogenous stochastic variables***, not strategic control variables.
They serve as part of the **environment's uncertainty** from the renewable producer's viewpoint (and also from the market operator's perspective).

Let's refine the four-perspective formalization accordingly — ensuring all perspectives are internally consistent, share a unified state and transition structure, and correctly treat the conventional producers' bids as random inputs rather than controls.

🌍 1. Shared Market Environment

All perspectives operate within the same stochastic day-ahead electricity market.

(a) Time horizon

$$t = 1, 2, \ldots, 24$$

(b) Global parameters

Symbol	Meaning
q_u	penalty for under-delivery ($/MWh)
q_o	penalty for over-delivery ($/MWh)
μ_D, σ_D^2	mean and variance of hourly demand
$N_c = 3$	number of conventional producers
$\Phi(\cdot)$	market-clearing price function
$f_P(\cdot)$	renewable generation distribution
$\mu_b^{(i)}, \sigma_b^{(i)}$	mean and variance of bid quantity for conventional producer i
$\mu_p^{(i)}, \sigma_p^{(i)}$	mean and variance of offer price for conventional producer i

(c) State variables

$$s_t = (D_t, c_t, P_t)$$

(d) Initial state

$$s_0 = (\mathbb{E}[D_t], c_0, P_0)$$

where c_0, P_0 are baseline nominal values.

(e) Stochastic processes

- Demand: $D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$

- Renewable generation: $P_t \sim f_P(\cdot)$
- Conventional bids:

$$b_t^{(i)} \sim \mathcal{N}(\mu_b^{(i)}, \sigma_b^{(i)}), \quad p_t^{(i)} \sim \mathcal{N}(\mu_p^{(i)}, \sigma_p^{(i)}), \quad i = 1, 2, 3$$

These are **random draws** and **not decision variables**.

(f) Market clearing

Given all bids (conventional + renewable),

$$c_t = \Phi\left(D_t, \{b_t^{(i)}, p_t^{(i)}\}_{i=1}^{3}, b_t^{(R)}, p_t^{(R)}\right)$$

👥 2. Four Perspectives

🔹 Perspective 1 — Renewable Producer (Strategic decision-maker)

Control variables

$$u_t^{(R)} = (b_t^{(R)}, p_t^{(R)})$$

(bid quantity and price chosen strategically before market clearing).

State variables

$s_t = (D_t, c_t, P_t)$

Transition function

Profit accumulation:

$$\Pi_{t+1}^{(R)} = \Pi_t^{(R)} + \pi_t^{(R)}$$

Profit function

$$\pi_t^{(R)} = \begin{cases} c_t b_t^{(R)} - q_u \max(0, b_t^{(R)} - P_t) - q_o \max(0, P_t - b_t^{(R)}), & p_t^{(R)} \le c_t \\ 0, & p_t^{(R)} > c_t \end{cases}$$

Objective

$$\max_{\{b_t^{(R)}, p_t^{(R)}\}} \mathbb{E}\left[\sum_{t=1}^{24} \pi_t^{(R)}\right]$$

subject to $b_t^{(R)}, p_t^{(R)} \ge 0.$

🔹 Perspective 2 — Conventional Producers (Exogenous random bidders)

These participants **do not choose** $b_t^{(i)}$ or $p_t^{(i)}$; instead, their bids are modeled as **stochastic processes** representing historical bidding behavior.

Random variables

$$b_t^{(i)} \sim \mathcal{N}(\mu_b^{(i)}, \sigma_b^{(i)}), \quad p_t^{(i)} \sim \mathcal{N}(\mu_p^{(i)}, \sigma_p^{(i)})$$

Profit realization (given price outcome)

$$\pi_t^{(i)} = \begin{cases} c_t b_t^{(i)} - C_t^{(i)}(b_t^{(i)}), & p_t^{(i)} \le c_t \\ 0, & p_t^{(i)} > c_t \end{cases}$$

where $C_t^{(i)}(b_t^{(i)})$ is a deterministic cost function, e.g.

$$C_t^{(i)} = \alpha^{(i)} b_t^{(i)} + \beta^{(i)} (b_t^{(i)})^2.$$

Objective

No optimization (behavioral model).
Their expected profit distribution can be **analyzed** but not **controlled**:

$$\mathbb{E}[\pi_t^{(i)}] = \int \pi_t^{(i)} f(b_t^{(i)}, p_t^{(i)})\, db_t^{(i)}\, dp_t^{(i)}.$$

🔹 Perspective 3 — Market Operator (System balancer)

Control variables

None (market-clearing is mechanical).

Market-clearing rule

$$c_t = \Phi(D_t, \{b_t^{(i)}, p_t^{(i)}\}_{i=1}^3, b_t^{(R)}, p_t^{(R)})$$

with balance condition:

$$\sum_{i=1}^{3} b_t^{(i)} 1_{[p_t^{(i)} \le c_t]} + b_t^{(R)} 1_{[p_t^{(R)} \le c_t]} = D_t$$

Objective functions

The operator's implicit goals are **feasibility** and **market efficiency**:

1. **Balance constraint** (system feasibility):

$$J_1^{(O)} = \sum_t \left| D_t - \sum_i b_t^{(i)} \delta_t^{(i)} \right|^2$$

to be minimized.

2. **Welfare (efficiency)**:

$$J_2^{(O)} = -\sum_t W_t, \quad W_t = \text{consumer surplus} - \sum_i \text{producer costs}$$

The operator does not directly optimize, but these represent evaluative metrics.

🔹 Perspective 4 — Regulator / System Planner

Control variables

$$u_t^{(G)} = (q_u, q_o, \tau_t)$$

where

- q_u, q_o: imbalance penalties (policy levers),
- τ_t: renewable incentive (subsidy or tax per MWh).

State variables

Aggregate system outcomes:

$$s_t^{(G)} = (c_t, \text{reliability}_t, \text{renewable share}_t)$$

Transition

$$s_{t+1}^{(G)} = h(s_t^{(G)}, u_t^{(G)}, \{b_t^{(R)}, p_t^{(R)}\})$$

Objective functions (multi-objective)

1. **Reliability:**

$$J_1^{(G)} = -\mathbb{E}\left[\sum_t |D_t - \sum_i b_t^{(i)} \delta_t^{(i)}|\right]$$

2. **Price stability:**

$$J_2^{(G)} = -\operatorname{Var}(c_t)$$

3. **Renewable penetration:**

$$J_3^{(G)} = \mathbb{E}\left[\sum_t \frac{b_t^{(R)} \delta_t^{(R)}}{D_t}\right]$$

No single weighted objective — each represents a policy dimension.

🧩 3. Shared System Summary

Element	Symbol	Description
Time	$t = 1, \dots, 24$	Hourly intervals
State	$s_t = (D_t, c_t, P_t)$	Market environment
Uncertainties	$D_t, P_t, b_t^{(i)}, p_t^{(i)}$	Demand, weather, conventional bids
Renewable controls	$b_t^{(R)}, p_t^{(R)}$	Strategic bid quantity & price
Transition	$s_{t+1} = f(s_t, u_t^{(R)}, \omega_t)$	Next-period environment
Profit (renewable)	$\pi_t^{(R)}$	Profit after penalties
Policy levers	q_u, q_o, τ_t	Regulator's control
Market clearing	$c_t = \Phi(D_t, \text{bids})$	Price formation
Initial state	$s_0 = (\mu_D, c_0, P_0)$	Starting market condition

🔗 4. Interactions Overview

```scss
Regulator sets (q_u, q_o, τ_t)
        ↓
Renewable producer optimizes (b_t^R, p_t^R)
        ↓
Conventional producers generate random bids
        ↓
Market operator clears market → c_t
        ↓
Realizations (D_t, P_t) → profits, penalties
        ↓
State updates for next hour
```

🧠 5. Conceptual Summary

Perspective	Controls	Objective(s)	Nature of behavior	Main Uncertainties
Renewable Producer	$b_t^{(R)}, p_t^{(R)}$	Maximize expected profit	**Strategic optimization**	c_t, P_t
Conventional Producers	— (stochastic bids)	None (exogenous)	**Stochastic process**	—
Market Operator	—	Maintain balance, maximize welfare	**Mechanistic**	D_t, bids
Regulator	q_u, q_o, τ_t	Multi-objective: reliability, stability, renewables	**Policy optimization**	Aggregate outcomes

Would you like me to **compose these refined perspectives** into a **unified, modular model** next — sharing the same state and environment but encapsulating each actor as a class (as in your earlier "Lake" examples)?

To ensure internal consistency across these models, please compose them into a single, unified model that shares a common environment and states, while embedding perspective-specific controls and objectives. The composed model should: 1. embed each perspective's control variables and objectives; 2. enables modular evaluation of each perspective; 3. for each perspective, treat other perspectives' controls as exogenous inputs; and 4. incorporate all other components from every perspective.

Perfect — now that each stakeholder's model is well-defined and consistent, we can **compose them into a unified, modular framework** that satisfies your four conditions.

The unified model will feature:

- a **shared environment** with global parameters, state variables, and stochastic processes;
- **perspective-specific control variables and objectives**,
- and **modular interconnections** allowing each actor's model to be evaluated independently, while treating others' controls as exogenous.

🧩 Unified Multi-Perspective Model for the Day-Ahead Market

Let the set of stakeholders be

$$\mathcal{A} = \{\mathrm{R}, \mathrm{C}_1, \mathrm{C}_2, \mathrm{C}_3, \mathrm{O}, \mathrm{G}\}$$

corresponding respectively to the **Renewable Producer**, three **Conventional Producers**, the **Market Operator**, and the **Regulator**.

All actors operate within a **shared market environment** characterized below.

🌍 1. Shared Environment

(a) Time and Horizon

$$t = 1, 2, \dots, T, \quad T = 24$$

(b) Global Parameters

$$\Theta = \{q_u, q_o, \mu_D, \sigma_D, f_P, \{\mu_b^{(i)}, \sigma_b^{(i)}, \mu_p^{(i)}, \sigma_p^{(i)}\}_{i=1}^3\}$$

- q_u, q_o: penalty rates
- μ_D, σ_D: demand distribution parameters
- f_P: renewable generation distribution
- $(\mu_b^{(i)}, \sigma_b^{(i)}, \mu_p^{(i)}, \sigma_p^{(i)})$: conventional bidding distributions.

(c) State Variables (shared)

$$s_t = (D_t, P_t, c_t)$$

where

- D_t: total demand (stochastic)
- P_t: realized renewable generation (stochastic)
- c_t: market-clearing price (endogenous).

(d) Exogenous Processes

$$\begin{aligned} D_t &\sim \mathcal{N}(\mu_D, \sigma_D^2) \\ P_t &\sim f_P(\cdot) \\ b_t^{(i)} &\sim \mathcal{N}(\mu_b^{(i)}, \sigma_b^{(i)}) \\ p_t^{(i)} &\sim \mathcal{N}(\mu_p^{(i)}, \sigma_p^{(i)}), \quad i = 1, 2, 3 \end{aligned}$$

(e) Market-Clearing Mechanism (common transition component)

$$c_t = \Phi\Big(D_t,\ \{b_t^{(i)}, p_t^{(i)}\}_{i=1}^3,\ b_t^{(R)}, p_t^{(R)}\Big)$$

and the **dispatch condition** for each producer:

$$\delta_t^{(k)} = \begin{cases} 1, & p_t^{(k)} \le c_t \\ 0, & \text{otherwise} \end{cases} \qquad k \in \{\mathrm{R}, 1, 2, 3\}$$

ensuring:

$$\sum_{k \in \{\mathrm{R},1,2,3\}} b_t^{(k)} \delta_t^{(k)} = D_t$$

👥 2. Embedded Perspectives

Each actor $a \in \mathcal{A}$ has:

$$M^{(a)} = \langle s_t, u_t^{(a)}, f^{(a)}, J^{(a)} \rangle$$

where $u_t^{(a)}$ are control variables, $f^{(a)}$ the actor's transition equations, and $J^{(a)}$ the objective(s). Other actors' controls are treated as **exogenous inputs**.

🔹 Perspective 1 — Renewable Producer (R)

- **Controls:**

$$u_t^{(R)} = (b_t^{(R)}, p_t^{(R)})$$

- **Transition:**

$$\Pi_{t+1}^{(R)} = \Pi_t^{(R)} + \pi_t^{(R)}$$

- **Instantaneous profit:**

$$\pi_t^{(R)} = \begin{cases} c_t b_t^{(R)} - q_u \max(0, b_t^{(R)} - P_t) - q_o \max(0, P_t - b_t^{(R)}), & p_t^{(R)} \le c_t \\ 0, & p_t^{(R)} > c_t \end{cases}$$

- **Objective:**

$$J^{(R)} = \max_{\{b_t^{(R)}, p_t^{(R)}\}} \mathbb{E}\left[\sum_{t=1}^{T} \pi_t^{(R)}\right]$$

🔹 Perspective 2 — Conventional Producers (C_1, C_2, C_3)

Each conventional producer i has:

- **Exogenous stochastic bids:**

$$b_t^{(i)} \sim \mathcal{N}(\mu_b^{(i)}, \sigma_b^{(i)}), \quad p_t^{(i)} \sim \mathcal{N}(\mu_p^{(i)}, \sigma_p^{(i)})$$

- **Instantaneous profit:**

$$\pi_t^{(i)} = \begin{cases} c_t b_t^{(i)} - \alpha^{(i)} b_t^{(i)} - \beta^{(i)} (b_t^{(i)})^2, & p_t^{(i)} \le c_t \\ 0, & p_t^{(i)} > c_t \end{cases}$$

- **Transition:**

$$\Pi_{t+1}^{(i)} = \Pi_t^{(i)} + \pi_t^{(i)}$$

- **Objective (evaluative, not control-based):**

$$J^{(i)} = \mathbb{E}\left[\sum_{t=1}^{T} \pi_t^{(i)}\right]$$

These producers influence the environment through their **stochastic bids**.

🔹 Perspective 3 — Market Operator (O)

- **Controls:** None (mechanical rule).
- **Transition rule:**

$$c_t = \Phi(D_t, \{b_t^{(i)}, p_t^{(i)}\}_{i=1}^{3}, b_t^{(R)}, p_t^{(R)})$$

  and enforce balance:

$$D_t = \sum_k b_t^{(k)} \delta_t^{(k)}.$$

- **Objectives (evaluation metrics):**

1. **Market balance:**

$$J_1^{(O)} = -\sum_t \left(D_t - \sum_k b_t^{(k)} \delta_t^{(k)}\right)^2$$

2. **Social welfare:**

$$J_2^{(O)} = \sum_t W_t, \quad W_t = \text{consumer surplus} - \sum_k C_t^{(k)}$$

🔹 Perspective 4 — Regulator / System Planner (G)

- **Controls:**

$$u_t^{(G)} = (q_u, q_o, \tau_t)$$

  where τ_t is a renewable incentive or tax.

- **State variables (derived):**

$$s_t^{(G)} = (c_t, \rho_t, \eta_t)$$

  where

  - $\rho_t = \frac{b_t^{(R)} \delta_t^{(R)}}{D_t}$: renewable penetration
  - $\eta_t = D_t - \sum_k b_t^{(k)} \delta_t^{(k)}$: imbalance.

- **Transition:**

$$s_{t+1}^{(G)} = h(s_t^{(G)}, u_t^{(G)}, \{b_t^{(R)}, p_t^{(R)}\})$$

- **Objectives (multi-objective):**

$$\begin{aligned} J_1^{(G)} &= -\mathbb{E}\left[\sum_t |\eta_t|\right] && \text{(reliability)} \\ J_2^{(G)} &= -\operatorname{Var}(c_t) && \text{(price stability)} \\ J_3^{(G)} &= \mathbb{E}\left[\sum_t \rho_t\right] && \text{(renewable penetration)} \end{aligned}$$

Each is evaluated separately (no weighting).

🔗 3. Unified Transition Structure

The market evolves according to a unified transition function:

$$s_{t+1} = F\big(s_t,\ u_t^{(R)},\ u_t^{(G)},\ \{b_t^{(i)}, p_t^{(i)}\}_{i=1}^3,\ \omega_t\big)$$

where $\omega_t = (D_t, P_t)$ are exogenous stochastic shocks.

Explicitly:

$$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2), \quad P_t \sim f_P(\cdot),$$
$$c_t = \Phi(D_t, \{b_t^{(i)}, p_t^{(i)}\}, b_t^{(R)}, p_t^{(R)}),$$
$$\Pi_{t+1}^{(a)} = \Pi_t^{(a)} + \pi_t^{(a)}(s_t, u_t^{(a)}).$$

🧠 4. Unified Model Summary

Element	Symbol	Shared/Specific	Description
Time	$t = 1, \ldots, 24$	Shared	Hourly periods
State	$s_t = (D_t, P_t, c_t)$	Shared	Market state
Renewable controls	$b_t^{(R)}, p_t^{(R)}$	Specific	Strategic bids
Regulator controls	q_u, q_o, τ_t	Specific	Policy levers
Conventional bids	$b_t^{(i)}, p_t^{(i)}$	Exogenous	Random processes
Market rule	$c_t = \Phi(\cdot)$	Shared	Clearing mechanism
Transition	$s_{t+1} = F(\cdot)$	Shared	System evolution
Objectives	$J^{(a)}$	Specific	Actor-level goals
Initial state	$s_0 = (\mu_D, c_0, P_0)$	Shared	Start conditions

⚙️ 5. Structural Dependencies

Each perspective treats others' controls as exogenous:

Perspective	Endogenous Controls	Exogenous Inputs
Renewable Producer	$(b_t^{(R)}, p_t^{(R)})$	q_u, q_o, c_t, P_t
Conventional Producers	— (random)	c_t
Market Operator	—	All bids, D_t
Regulator	(q_u, q_o, τ_t)	Market outcomes (c_t, D_t, P_t)

This ensures **modular evaluation**:

- each submodel $M^{(a)}$ can be simulated independently,
- yet all share consistent states and stochastic processes.

🧮 6. Unified Model (Compact Form)

$$\begin{aligned}
\text{State: } s_{t+1} &= F(s_t, u_t^{(R)}, u_t^{(G)}, \omega_t) \\
\text{Renewable: } &\max_{u_t^{(R)}} \mathbb{E}\left[\sum_t \pi_t^{(R)}(s_t, u_t^{(R)})\right] \\
\text{Conventional: } b_t^{(i)}, p_t^{(i)} &\sim \mathcal{N}(\mu^{(i)}, \sigma^{(i)}) \\
\text{Market: } c_t &= \Phi(D_t, \text{bids}) \\
\text{Regulator: } &\max_{u_t^{(G)}}(J_1^{(G)}, J_2^{(G)}, J_3^{(G)}) \\
\text{Initial: } s_0 &= (\mu_D, c_0, P_0)
\end{aligned}$$

Would you like me to now **implement this unified model in modular Python form**, with a base **`MarketEnvironment`** class and subclasses for each perspective (renewable, conventional, operator, regulator), plus an interface function to simulate dynamics and return all objectives — matching your "Lake"-style structure?

Now, please provide a modular Python implementation of the unified model, using classes to represent the different perspectives. At the end of the program, define a function that serves as the interface. This function should accept the control variables and the values of uncertain parameters as inputs, simulate the model dynamics, and return the values of each objective function as outputs. There is no need to execute the function.

Here is a **fully modular Python implementation** of the **unified stochastic day-ahead market model**, written to reflect the mathematical specification above.
Each stakeholder (renewable producer, conventional producers, market operator, and regulator) is represented as a class.
All share a common environment and interact through the unified model dynamics.

```python
import numpy as np

# ==============================================================
# 1. Shared Market Environment
# ==============================================================

class MarketEnvironment:
    """
    Shared stochastic environment representing the day-ahead market.
    Contains global parameters, stochastic processes, and the market-clearing rule.
    """

    def __init__(self, T=24, mu_D=100, sigma_D=10,
                 q_u=20, q_o=10, f_P=None,
                 mu_b=None, sigma_b=None, mu_p=None, sigma_p=None):
        self.T = T
        self.mu_D = mu_D
        self.sigma_D = sigma_D
        self.q_u = q_u
        self.q_o = q_o

        # Default renewable generation distribution: truncated normal [0, 1]
```

```
        self.f_P = f_P if f_P is not None else lambda: max(0, np.random.normal(0.5, 0.2))

        # Conventional producer parameters (for 3 producers)
        self.N_c = 3
        self.mu_b = mu_b if mu_b is not None else [30, 40, 50]
        self.sigma_b = sigma_b if sigma_b is not None else [5, 5, 5]
        self.mu_p = mu_p if mu_p is not None else [40, 45, 50]
        self.sigma_p = sigma_p if sigma_p is not None else [5, 5, 5]

    def sample_demand(self):
        return np.random.normal(self.mu_D, self.sigma_D)

    def sample_conventional_bids(self):
        """Sample bids (quantity, price) for each of the 3 conventional producers."""
        bids = []
        for i in range(self.N_c):
            b_i = np.random.normal(self.mu_b[i], self.sigma_b[i])
            p_i = np.random.normal(self.mu_p[i], self.sigma_p[i])
            bids.append((b_i, p_i))
        return bids

    def market_clearing_price(self, D_t, bids_conv, bid_renew):
        """
        Compute market-clearing price c_t.
        Simplified rule: sort offers by price until cumulative quantity ≥ demand.
        """
        offers = []
        for b, p in bids_conv:
            offers.append((p, b))
        offers.append((bid_renew[1], bid_renew[0]))
        offers.sort(key=lambda x: x[0])  # ascending by price

        cum_qty, clearing_price = 0, offers[-1][0]
        for p, b in offers:
            cum_qty += b
            if cum_qty >= D_t:
                clearing_price = p
                break
        return clearing_price


# ============================================================
# 2. Perspective: Renewable Producer
# ============================================================

class RenewableProducer:
    """
    Strategic renewable energy producer.
    Chooses bid quantity and price to maximize expected profit.
    """

    def __init__(self, env: MarketEnvironment):
        self.env = env
        self.profit = 0

    def profit_function(self, b_t, p_t, c_t, P_t):
        q_u, q_o = self.env.q_u, self.env.q_o
        if p_t <= c_t:
            penalty_under = q_u * max(0, b_t - P_t)
            penalty_over = q_o * max(0, P_t - b_t)
            return c_t * b_t - penalty_under - penalty_over
        else:
            return 0

    def step(self, b_t, p_t, c_t, P_t):
        """Compute hourly profit and update total."""
        pi_t = self.profit_function(b_t, p_t, c_t, P_t)
        self.profit += pi_t
        return pi_t


# ============================================================
# 3. Perspective: Conventional Producers
# ============================================================
```

```python
class ConventionalProducer:
    """
    Non-strategic conventional producer with stochastic bids.
    """

    def __init__(self, env: MarketEnvironment, i: int, alpha=5, beta=0.1):
        self.env = env
        self.i = i
        self.alpha = alpha
        self.beta = beta
        self.profit = 0

    def sample_bid(self):
        b_i = np.random.normal(self.env.mu_b[self.i], self.env.sigma_b[self.i])
        p_i = np.random.normal(self.env.mu_p[self.i], self.env.sigma_p[self.i])
        return b_i, p_i

    def cost(self, b_i):
        return self.alpha * b_i + self.beta * b_i ** 2

    def profit_function(self, b_i, p_i, c_t):
        if p_i <= c_t:
            return c_t * b_i - self.cost(b_i)
        else:
            return 0

    def step(self, b_i, p_i, c_t):
        pi_t = self.profit_function(b_i, p_i, c_t)
        self.profit += pi_t
        return pi_t


# ============================================================
# 4. Perspective: Market Operator
# ============================================================

class MarketOperator:
    """
    Market-clearing mechanism and welfare evaluation.
    """

    def __init__(self, env: MarketEnvironment):
        self.env = env
        self.welfare = 0
        self.imbalance = 0

    def compute_welfare(self, D_t, c_t, producers):
        """
        Placeholder welfare measure: consumer value - producer cost.
        """
        total_supply = sum([p[0] for p in producers])
        self.imbalance += abs(D_t - total_supply)
        # Simplified welfare metric
        return D_t * c_t - sum([0.5 * c_t * p[0] for p in producers])

    def step(self, D_t, c_t, producers):
        W_t = self.compute_welfare(D_t, c_t, producers)
        self.welfare += W_t
        return W_t


# ============================================================
# 5. Perspective: Regulator / System Planner
# ============================================================

class Regulator:
    """
    Regulator with policy levers: imbalance penalties (q_u, q_o) and renewable incentives
(tau_t).
    """

    def __init__(self, env: MarketEnvironment, q_u=None, q_o=None, tau_t=0):
        self.env = env
        if q_u is not None:
            self.env.q_u = q_u
```

```
        if q_o is not None:
            self.env.q_o = q_o
        self.tau_t = tau_t
        self.metrics = {"reliability": 0, "price_stability": [], "renewable_share": []}

    def evaluate(self, D_t, c_t, b_R, accepted_R):
        # Renewable share if bid accepted
        rho_t = (b_R / D_t) if accepted_R else 0
        self.metrics["renewable_share"].append(rho_t)
        self.metrics["price_stability"].append(c_t)

    def finalize_metrics(self, total_imbalance):
        self.metrics["reliability"] = -total_imbalance
        self.metrics["price_stability"] = -np.var(self.metrics["price_stability"])
        self.metrics["renewable_share"] = np.mean(self.metrics["renewable_share"])
        return self.metrics


# ==============================================================
# 6. Unified Market Simulation Interface
# ==============================================================

def simulate_market(
    controls_R,
    controls_G,
    uncertain_params,
    T=24
):
    """
    Interface function to simulate the unified day-ahead market model.
    Inputs:
        controls_R: dict with renewable bids {t: (b_t, p_t)}
        controls_G: dict with regulator settings {'q_u': val, 'q_o': val, 'tau_t': val}
        uncertain_params: dict with environment parameters
    Output:
        dict of objective values for each perspective
    """

    # --- Initialize environment and agents ---
    env = MarketEnvironment(
        T=T,
        mu_D=uncertain_params.get("mu_D", 100),
        sigma_D=uncertain_params.get("sigma_D", 10),
        q_u=controls_G.get("q_u", 20),
        q_o=controls_G.get("q_o", 10)
    )

    renewable = RenewableProducer(env)
    conventionals = [ConventionalProducer(env, i) for i in range(3)]
    operator = MarketOperator(env)
    regulator = Regulator(env, q_u=controls_G.get("q_u"), q_o=controls_G.get("q_o"),
                          tau_t=controls_G.get("tau_t", 0))

    # --- Simulation loop ---
    for t in range(T):
        D_t = env.sample_demand()
        P_t = env.f_P()
        bids_conv = env.sample_conventional_bids()
        b_R, p_R = controls_R.get(t, (0, 0))

        # Market clearing
        c_t = env.market_clearing_price(D_t, bids_conv, (b_R, p_R))

        # Compute profits
        pi_R = renewable.step(b_R, p_R, c_t, P_t)
        for i, conv in enumerate(conventionals):
            conv.step(*bids_conv[i], c_t)

        # Operator metrics
        producers = bids_conv + [(b_R, p_R)]
        operator.step(D_t, c_t, producers)

        # Regulator metrics
        accepted_R = (p_R <= c_t)
        regulator.evaluate(D_t, c_t, b_R, accepted_R)
```

```
    # --- Compute final objectives ---
    regulator_metrics = regulator.finalize_metrics(operator.imbalance)
    results = {
        "Renewable_Profit": renewable.profit,
        "Conventional_Profit": sum([conv.profit for conv in conventionals]),
        "Market_Welfare": operator.welfare,
        "Regulator_Objectives": regulator_metrics
    }
    return results
```

🧠 Model Design Notes

- **Common Environment (`MarketEnvironment`)**
  Holds global parameters, stochastic demand and generation, and the market-clearing mechanism.
- **Perspective Classes**
  - **`RenewableProducer`**: controls $b_t^{(R)}, p_t^{(R)}$ and maximizes expected profit.
  - **`ConventionalProducer`**: exogenous bids with random distributions.
  - **`MarketOperator`**: clears the market, tracks welfare and imbalance.
  - **`Regulator`**: defines policy levers q_u, q_o, τ_t and computes system metrics.
- **Interface Function (`simulate_market`)**
  - Inputs: control variables for renewable and regulator, and uncertain parameters.
  - Simulates hourly market dynamics for $T = 24$ hours.
  - Returns objective function values for all perspectives:

    ```
    python

    {
        "Renewable_Profit": ...,
        "Conventional_Profit": ...,
        "Market_Welfare": ...,
        "Regulator_Objectives": {
            "reliability": ...,
            "price_stability": ...,
            "renewable_share": ...
        }
    }
    ```

Would you like me to extend this with **uncertainty sampling (Monte Carlo)** to approximate expected objectives (i.e., $\mathbb{E}[\pi_t]$) across multiple simulated days?

> There is a logic error in your implementation. The demand distribution has constant mean and variance, which are deterministic and you should not ask users to provide values for them in the simulation function. Instead, the renewable generation function is the one that is deeply uncertain. You should ask users to provide a possible instance for that as input.

Excellent — you're absolutely right again.

✅ The **mean and variance of demand** are *fixed global parameters* of the market environment — they are not deeply uncertain.
✅ The **renewable generation function** $f_P(\cdot)$, however, *is deeply uncertain* (as it represents weather- and technology-driven processes).
Hence, the simulation interface should **ask the user to provide a particular realization (instance) of the renewable generation function**, not the demand distribution parameters.

Let's correct this logic error and present the refined, internally consistent unified model implementation.

🧩 Corrected Modular Python Implementation

```python
import numpy as np

# ============================================================
# 1. Shared Market Environment
# ============================================================

class MarketEnvironment:
    """
    Shared stochastic environment representing the day-ahead market.
    Demand is modeled as a normal distribution with fixed mean and variance.
    Renewable generation is provided as an uncertain external function f_P().
    """

    def __init__(self, T=24, q_u=20, q_o=10, f_P=None):
        self.T = T
        self.q_u = q_u
        self.q_o = q_o

        # Fixed demand parameters (deterministic across all simulations)
        self.mu_D = 100
        self.sigma_D = 10

        # Deeply uncertain renewable generation function
        # This must be provided by the user as part of simulation input
        if f_P is None:
            raise ValueError("A renewable generation function f_P must be provided.")
        self.f_P = f_P

        # Conventional producer parameters (fixed, stochastic but with constant mean/variance)
        self.N_c = 3
        self.mu_b = [30, 40, 50]
        self.sigma_b = [5, 5, 5]
        self.mu_p = [40, 45, 50]
        self.sigma_p = [5, 5, 5]

    def sample_demand(self):
        """Hourly total demand (Normal with constant mean and variance)."""
        return np.random.normal(self.mu_D, self.sigma_D)

    def sample_conventional_bids(self):
        """Sample bids (quantity, price) for each of the 3 conventional producers."""
        bids = []
        for i in range(self.N_c):
            b_i = np.random.normal(self.mu_b[i], self.sigma_b[i])
            p_i = np.random.normal(self.mu_p[i], self.sigma_p[i])
            bids.append((b_i, p_i))
        return bids

    def market_clearing_price(self, D_t, bids_conv, bid_renew):
        """
        Compute market-clearing price c_t.
        Simplified rule: sort offers by price until cumulative quantity ≥ demand.
        """
        offers = []
```

```
        for b, p in bids_conv:
            offers.append((p, b))
        offers.append((bid_renew[1], bid_renew[0]))
        offers.sort(key=lambda x: x[0])  # ascending by price

        cum_qty, clearing_price = 0, offers[-1][0]
        for p, b in offers:
            cum_qty += b
            if cum_qty >= D_t:
                clearing_price = p
                break
        return clearing_price


# ================================================================
# 2. Perspective: Renewable Producer
# ================================================================

class RenewableProducer:
    """
    Strategic renewable energy producer.
    Chooses bid quantity and price to maximize expected profit.
    """

    def __init__(self, env: MarketEnvironment):
        self.env = env
        self.profit = 0

    def profit_function(self, b_t, p_t, c_t, P_t):
        q_u, q_o = self.env.q_u, self.env.q_o
        if p_t <= c_t:
            penalty_under = q_u * max(0, b_t - P_t)
            penalty_over = q_o * max(0, P_t - b_t)
            return c_t * b_t - penalty_under - penalty_over
        else:
            return 0

    def step(self, b_t, p_t, c_t, P_t):
        """Compute hourly profit and update total."""
        pi_t = self.profit_function(b_t, p_t, c_t, P_t)
        self.profit += pi_t
        return pi_t


# ================================================================
# 3. Perspective: Conventional Producers
# ================================================================

class ConventionalProducer:
    """
    Non-strategic conventional producer with stochastic bids.
    """

    def __init__(self, env: MarketEnvironment, i: int, alpha=5, beta=0.1):
        self.env = env
        self.i = i
        self.alpha = alpha
        self.beta = beta
        self.profit = 0

    def cost(self, b_i):
        return self.alpha * b_i + self.beta * b_i ** 2

    def profit_function(self, b_i, p_i, c_t):
        if p_i <= c_t:
            return c_t * b_i - self.cost(b_i)
        else:
            return 0

    def step(self, b_i, p_i, c_t):
        pi_t = self.profit_function(b_i, p_i, c_t)
        self.profit += pi_t
        return pi_t
```

```
# ============================================================
# 4. Perspective: Market Operator
# ============================================================

class MarketOperator:
    """
    Market-clearing mechanism and welfare evaluation.
    """

    def __init__(self, env: MarketEnvironment):
        self.env = env
        self.welfare = 0
        self.imbalance = 0

    def compute_welfare(self, D_t, c_t, producers):
        total_supply = sum([p[0] for p in producers])
        self.imbalance += abs(D_t - total_supply)
        # Simplified welfare metric
        return D_t * c_t - sum([0.5 * c_t * p[0] for p in producers])

    def step(self, D_t, c_t, producers):
        W_t = self.compute_welfare(D_t, c_t, producers)
        self.welfare += W_t
        return W_t


# ============================================================
# 5. Perspective: Regulator / System Planner
# ============================================================

class Regulator:
    """
    Regulator with policy levers: imbalance penalties (q_u, q_o) and renewable incentives
(tau_t).
    """

    def __init__(self, env: MarketEnvironment, tau_t=0):
        self.env = env
        self.tau_t = tau_t
        self.metrics = {"reliability": 0, "price_stability": [], "renewable_share": []}

    def evaluate(self, D_t, c_t, b_R, accepted_R):
        rho_t = (b_R / D_t) if accepted_R else 0
        self.metrics["renewable_share"].append(rho_t)
        self.metrics["price_stability"].append(c_t)

    def finalize_metrics(self, total_imbalance):
        self.metrics["reliability"] = -total_imbalance
        self.metrics["price_stability"] = -np.var(self.metrics["price_stability"])
        self.metrics["renewable_share"] = np.mean(self.metrics["renewable_share"])
        return self.metrics


# ============================================================
# 6. Unified Market Simulation Interface
# ============================================================

def simulate_market(
    controls_R,
    controls_G,
    f_P_instance,
    T=24
):
    """
    Unified simulation interface for the day-ahead electricity market.

    Inputs:
        controls_R : dict
            Renewable producer bids {t: (b_t, p_t)}.
        controls_G : dict
            Regulator settings {'q_u': val, 'q_o': val, 'tau_t': val}.
        f_P_instance : callable
            A function representing a possible instance of renewable generation.
            Must return a random sample of P_t (MWh) each time it's called.
        T : int
```

```
            Number of hourly intervals (default 24).

    Output:
        dict containing objective function values for each perspective.
    """

    # --- Initialize environment and agents ---
    env = MarketEnvironment(
        T=T,
        q_u=controls_G.get("q_u", 20),
        q_o=controls_G.get("q_o", 10),
        f_P=f_P_instance
    )

    renewable = RenewableProducer(env)
    conventionals = [ConventionalProducer(env, i) for i in range(3)]
    operator = MarketOperator(env)
    regulator = Regulator(env, tau_t=controls_G.get("tau_t", 0))

    # --- Simulation loop ---
    for t in range(T):
        D_t = env.sample_demand()  # deterministic distribution with fixed parameters
        P_t = env.f_P()            # deeply uncertain renewable generation realization
        bids_conv = env.sample_conventional_bids()
        b_R, p_R = controls_R.get(t, (0, 0))

        # Market clearing
        c_t = env.market_clearing_price(D_t, bids_conv, (b_R, p_R))

        # Compute profits
        renewable.step(b_R, p_R, c_t, P_t)
        for i, conv in enumerate(conventionals):
            conv.step(*bids_conv[i], c_t)

        # Operator metrics
        producers = bids_conv + [(b_R, p_R)]
        operator.step(D_t, c_t, producers)

        # Regulator metrics
        accepted_R = (p_R <= c_t)
        regulator.evaluate(D_t, c_t, b_R, accepted_R)

    # --- Compute final objectives ---
    regulator_metrics = regulator.finalize_metrics(operator.imbalance)
    results = {
        "Renewable_Profit": renewable.profit,
        "Conventional_Profit": sum([conv.profit for conv in conventionals]),
        "Market_Welfare": operator.welfare,
        "Regulator_Objectives": regulator_metrics
    }
    return results
```

✅ Key Corrections and Improvements

Issue	Correction
Demand mean and variance were treated as uncertain	Now fixed internally in the environment (**`mu_D=100`**, **`sigma_D=10`**).
Renewable generation was deterministic in code	Now passed as **`f_P_instance`** by the user, representing deep uncertainty.
User interface asked for demand parameters	Now it only asks for a function describing renewable generation uncertainty.
Unified structure maintained	Classes for all perspectives remain modular and consistent with the shared environment.

🧠 Usage (conceptual example)

```python
# Example of deeply uncertain renewable generation function (e.g., wind)
def uncertain_generation():
    return max(0, np.random.lognormal(mean=3.0, sigma=0.4) / 100)

controls_R = {t: (20, 35) for t in range(24)}  # renewable bid quantity & price
controls_G = {"q_u": 25, "q_o": 8, "tau_t": 2}

# Simulate one scenario of renewable generation
results = simulate_market(controls_R, controls_G, f_P_instance=uncertain_generation)

# results contains all objective values (no need to execute here)
```

Would you like me to extend this version to allow **Monte Carlo averaging over multiple possible renewable generation instances** (to estimate expected objectives for the renewable producer and regulator)?