**Machine learning prep**

**A. Mitigation Plan Structure**

1. **Metadata**
   * **action\_id**: unique, human‑readable (e.g. "ACT\_20250608\_001")
   * **contingency\_id**: links back to the scenario it addresses
   * **created\_by** / **timestamp**: provenance
2. **Pre‑Conditions**
   * **Static**: e.g. "spinning\_reserve\_mw >= 20"
   * **Dynamic**: e.g. "no line overload > 80%"
   * **Check Functions**: code hooks that verify the grid state before executing each step
3. **Action Sequence**  
   An *ordered list* of steps; each step has:

jsonc

CopyEdit

{

"step": 1,

"type": "redispatch", // enum: redispatch, load\_shed, tap\_change, facts\_adjust, island, etc.

"target": "G12", // equipment ID

"parameters": { // step‑specific args

"delta\_mw": +30,

"ramp\_rate\_mw\_per\_min": 10

},

"timeout\_s": 60 // max time allotted for this step

}

* + **Branching**: you can introduce conditional steps, e.g.

json

CopyEdit

"if": { "max\_line\_loading\_pct > 105" }, "then\_step": 3

1. **Expected Outcome**

json

CopyEdit

{

"metrics": {

"max\_loading\_pct": "<95",

"min\_voltage\_pu": ">=0.98",

"frequency\_hz": "[59.9,60.1]"

},

"tolerance": {

"voltage\_deadband\_pu": 0.02,

"loading\_deadband\_pct": 2

}

}

1. **Cost & Impact Breakdown**

json

CopyEdit

{

"fuel\_cost\_usd": 1200.00,

"shed\_cost\_usd": 750.00,

"startup\_costs\_usd": 300.00,

"wear\_and\_tear\_cost\_usd": 50.00,

"co2\_emissions\_kg": 450.0,

"voLL\_usd\_per\_mw": 10000.0

}

1. **Execution Profile**
   * **estimated\_time\_s** (e.g. 180)
   * **actual\_time\_s** (populated post‑execution)
   * **step\_durations**: list of timings per step

**B. JSON Schema Example**

json

CopyEdit

{

"$schema": "http://json-schema.org/draft-07/schema#",

"title": "MitigationPlan",

"type": "object",

"properties": {

"action\_id": { "type": "string" },

"contingency\_id": { "type": "string" },

"pre\_conditions": {

"type": "object",

"patternProperties": {

".\*": { "type": "string" }

}

},

"action\_sequence": {

"type": "array",

"items": {

"type": "object",

"required": ["step","type","target","parameters"],

"properties": {

"step": { "type": "integer" },

"type": { "type": "string" },

"target": { "type": "string" },

"parameters": { "type": "object" },

"timeout\_s": { "type": "number" },

"if": { "type": "object" },

"then\_step": { "type": "integer" }

}

}

},

"expected\_outcome": { "$ref": "#/definitions/Outcome" },

"cost\_breakdown": { "$ref": "#/definitions/Cost" },

"execution\_profile": {

"type": "object",

"properties": {

"estimated\_time\_s": { "type": "number" },

"actual\_time\_s": { "type": "number" },

"step\_durations": {

"type": "array",

"items": { "type": "number" }

}

}

}

},

"definitions": {

"Outcome": {

"type": "object",

"properties": {

"metrics": {

"type": "object",

"patternProperties": {

".\*": { "type": "string" }

}

},

"tolerance": {

"type": "object",

"patternProperties": {

".\*": { "type": "number" }

}

}

}

},

"Cost": {

"type": "object",

"properties": {

"fuel\_cost\_usd": { "type": "number" },

"shed\_cost\_usd": { "type": "number" },

"startup\_costs\_usd": { "type": "number" },

"wear\_and\_tear\_cost\_usd":{ "type": "number" },

"co2\_emissions\_kg": { "type": "number" },

"voLL\_usd\_per\_mw": { "type": "number" }

}

}

}

}

**C. Integrating in Your Simulator**

1. **Rule‑Based Engine**
   * Write a factory that, given a violation, returns one or more **MitigationPlan** instances.
   * Each plan is scored (cost vs. speed vs. reliability) so you can log the top N options.
2. **Execution Harness**
   * A driver that reads the plan’s JSON, applies each step to the pandapower network, and times/measures the outcome.
   * Automatically populates actual\_time\_s and step\_durations.
3. **Result Logging**
   * Store the full plan JSON alongside outcome metrics in your DataFrame or relational DB for later analysis.

**D. Exporting to Excel**

Because your plan JSON can be large, you have two options:

1. **Flatten into Columns**
   * action\_sequence → "step1\_type", "step1\_target", "step1\_param\_delta\_mw", …
   * Repeat for N steps (up to a max sequence length).
2. **Embed JSON Cells**
   * Excel cells can hold JSON strings in a single column (e.g. action\_sequence\_json).
   * Provide a companion sheet that explains the schema.

**E. Downstream Benefits**

* **Traceability**: every mitigation step is explicitly documented.
* **Comparability**: multiple plans per contingency can be ranked by cost, speed, or environmental impact.
* **ML‑Ready**: features can include plan sequences and parameters—allowing models to learn not just *what* action, but *how* to execute it optimally.