

# Evaluation Test Cases for Conflict-Aware Smart Home Automation

This document contains the nine test cases used to evaluate the system's JSON generation accuracy, conflict detection performance, and explanation quality.

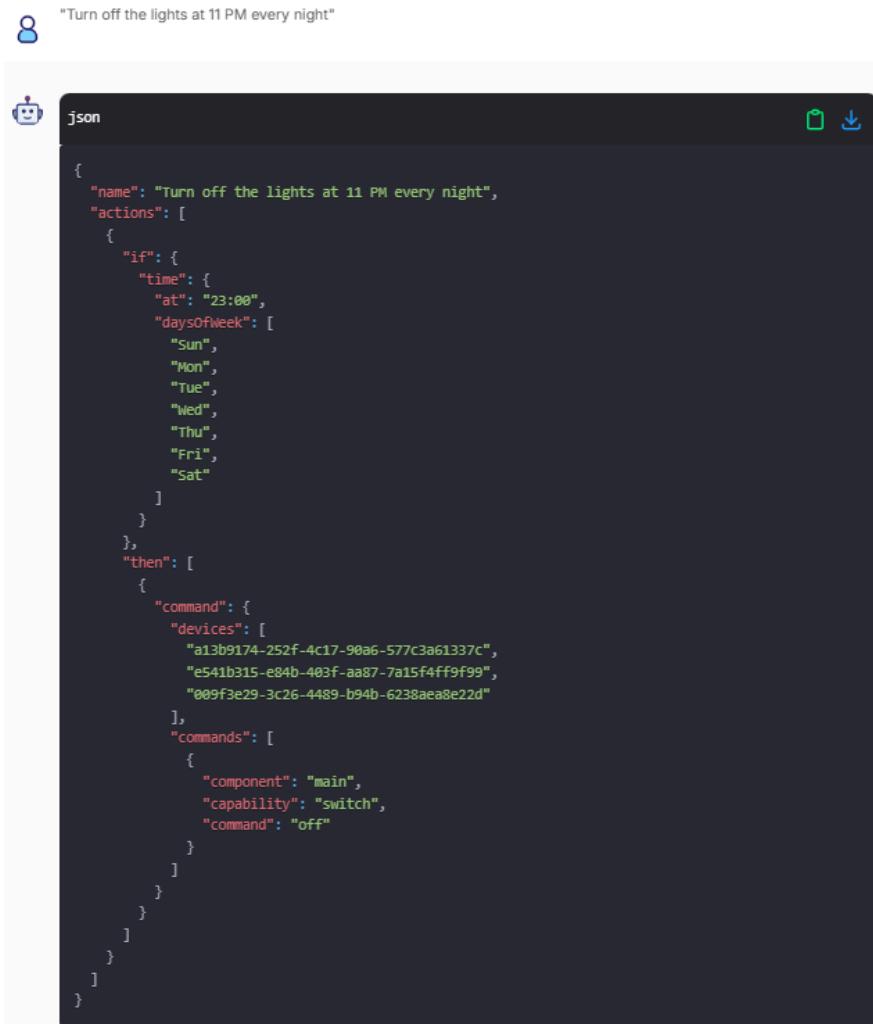
## TYPE 1: DIRECT CONTRADICTIONS (3 cases)

### Test Case 1.1: Time-Based Light Contradiction

#### Step 1 - Baseline Rule:

**Input:** "Turn off all lights at 11 PM every night"

**Output:**



The screenshot shows a JSON editor interface with a dark theme. At the top, there are icons for user, robot, and file operations. The file tab is labeled "json". The main area displays the following JSON code:

```
[{"name": "Turn off the lights at 11 PM every night", "actions": [{"if": {"time": {"at": "23:00", "daysOfWeek": ["Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"]}}, "then": [{"command": {"devices": ["a13b9174-252f-4c17-90a6-577c3a61337c", "e541b315-e84b-403f-aa87-7a15f4ff9f99", "009f3e29-3c26-4489-b94b-6238aea8e22d"], "commands": [{"component": "main", "capability": "switch", "command": "off"}]}]}]}]
```

**Conflict:** None (first rule)

**Success Criteria:** Valid JSON generated, matches SmartThings schema

### Step 2 - Conflicting Rule:

**Input:** "Turn on all lights at 11 PM every night"

**Expected Behavior:** Conflict Detected (No JSON generated).

**Success Criteria:** Correct ID, Explanation, and Resolution strategies provided.

### Output:

#### CONFLICT DETECTED: Direct Contradiction

- **Analysis:** New rule triggers "Turn ON" at 23:00, conflicting with existing rule "Turn OFF" at 23:00.
- **Impact:** Lights will flicker on and off repeatedly, potentially damaging bulbs.
- **Resolutions:**
  1. Delete the existing "Turn OFF at 11 PM" rule.
  2. Change the time to 11:01 PM to avoid overlap.
  3. Add a condition (e.g., "Only turn on if I am home").



"Turn on the lights at 11 PM every night"



⚠ CONFLICT DETECTED

Conflict Type: Direct Contradiction

Existing Rule: Turn off the lights at 11 PM every night

Your New Rule: Turn on the lights at 11 PM every night

The Problem: Both rules trigger at the same time (11 PM) but perform opposite actions (ON vs OFF).

Real-World Impact: The lights would turn ON and immediately turn OFF at 11:00 PM, creating pointless switching.

Resolution Options:

Remove the existing "Turn off the lights at 11:00 PM" rule.

Change the time of the new rule to a different time.

Add a condition that the lights must be off before turning them on.

How would you like to proceed?

### Test Case 1.2: Temperature-Based AC Contradiction

### Step 1 - Baseline Rule:

**Input:** "Turn on the AC when temperature exceeds 80 degrees"

**Output:**

Turn on the AC when temperature exceeds 80 degrees"

The screenshot shows a SmartThings rule editor window. At the top left is a user icon. To its right is the rule name: "Turn on the AC when temperature exceeds 80 degrees". Below the name is a "json" label. The main area contains the JSON configuration for the rule:

```
{  "name": "Turn on the AC when temperature exceeds 80 degrees",  "actions": [    {      "if": {        "greaterThan": {          "right": {            "device": {              "devices": [                "02445572-451a-447b-ac50-efdf2a8c556f"              ],              "component": "main",              "capability": "temperatureMeasurement",              "attribute": "temperature"            }          },          "left": {            "integer": 80          }        }      },      "then": [        {          "command": {            "devices": [              "02445572-451a-447b-ac50-efdf2a8c556f"            ],            "commands": [              {                "component": "main",                "capability": "thermostatMode",                "command": "cool"              }            ]          }        },        {          "else": []        }      ]    }  ]}
```

**Conflict:** None (first rule)

**Success Criteria:** Valid JSON generated, matches SmartThings schema

## Step 2 - Conflicting Rule:

**Input:** "Turn off the AC when temperature exceeds 80 degrees"

**Expected Behavior:** Conflict Detected (No JSON generated).

**Success Criteria:** Correct ID, Explanation, and Resolution strategies provided.

**Output:**

#### CONFLICT DETECTED: Direct Contradiction

- **Analysis:** Same trigger ( $> 80^{\circ}\text{F}$ ) attempts to Turn OFF while existing rule Turns ON.
- **Impact:** AC unit will short-cycle continuously, leading to compressor failure.
- **Resolutions:**
  1. Delete the existing "Turn ON  $> 80^{\circ}\text{F}$ " rule.
  2. Change the threshold (e.g., Turn OFF at  $75^{\circ}\text{F}$  instead of  $80^{\circ}\text{F}$ ).
  3. Add a mode condition (e.g., "Only turn OFF if mode is 'Away'").



"Turn off the AC when temperature exceeds 80 degrees"



#### ⚠ CONFLICT DETECTED

Conflict Type: Condition Overlap

Existing Rule: Turn on the AC when temperature exceeds 80 degrees

Your New Rule: Turn off the AC when temperature exceeds 80 degrees

The Problem: Both rules trigger when the temperature exceeds 80 degrees, but they perform opposite actions (ON vs OFF).

Real-World Impact: When the temperature is above 80, the system won't know whether to turn the AC ON or OFF, creating unpredictable behavior. The AC could toggle rapidly.

Resolution Options:

Add a temperature range: Turn ON when temperature  $> 80$ , turn OFF when temperature  $< 75$  (creates 5 degree deadband)

Use a single thermostat rule: Set target temperature to 78 degrees and let SmartThings manage on/off automatically

Reverse the logic: Turn ON when temp  $< 80$ , OFF when temp  $> 80$  (cooling mode)

How would you like to proceed?

### Test Case 1.3: Door Lock/Unlock Contradiction

#### Step 1 - Baseline Rule:

**Input:** "Lock the doors when presence is not detected."

**Output:**

"Lock the front door when presence is not detected"

The screenshot shows a SmartThings rule editor window. At the top left is a blue user icon. To its right is the rule name: "Lock the front door when presence is not detected". Below the name is a "json" label. The main area contains the JSON configuration for the rule:

```
{  "name": "Lock the front door when presence is not detected",  "actions": [    {      "if": {        "equals": {          "right": {            "device": {              "devices": [                "69c7c196-4482-4b89-8e99-694984a44681"              ],              "component": "main",              "capability": "presenceSensor",              "attribute": "presence"            }          }        },        "left": {          "string": "not present"        }      }    },    "then": [      {        "command": {          "devices": [            "69c7c196-4482-4b89-8e99-694984a44681"          ],          "commands": [            {              "component": "main",              "capability": "lock",              "command": "lock"            }          ]        }      }    ],    "else": []  }]
```

**Conflict:** None (first rule)

**Success Criteria:** Valid JSON generated, matches SmartThings schema

## Step 2 - Conflicting Rule:

**Input:** "Unlock the front door when presence is not detected"

**Expected Behavior:** Conflict Detected (No JSON generated).

**Success Criteria:** Correct ID, Explanation, and Resolution strategies provided.

### Output:

#### CONFLICT DETECTED: Direct Contradiction

- **Analysis:** Trigger "Not Present" attempts to UNLOCK while existing rule LOCKS.
- **Impact:** Door state becomes indeterminate, leaving home potentially unlocked while vacant.
- **Resolutions:**
  1. Delete the existing "Auto-Lock when absent" rule.
  2. Change the action to a different device (e.g., "Unlock Back Door").
  3. Add a time condition (e.g., "Only unlock if it is between 5-6 PM").



Unlock the front door when presence is not detected"



⚠ CONFLICT DETECTED

Conflict Type: Direct Contradiction

Existing Rule: Lock the front door when presence is not detected

Your New Rule: Unlock the front door when presence is not detected

The Problem: Both rules trigger on the same condition (presence not detected) but perform opposite actions (lock vs unlock).

Real-World Impact: When presence is not detected, the system won't know whether to lock or unlock the front door, creating unpredictable behavior.

Resolution Options:

Add time restriction: Unlock front door when presence not detected only between certain hours.

Add a mode restriction: Unlock front door when presence not detected only when in 'Home' mode.

Replace existing rule: Remove the lock rule and keep only the unlock rule.

How would you like to proceed?

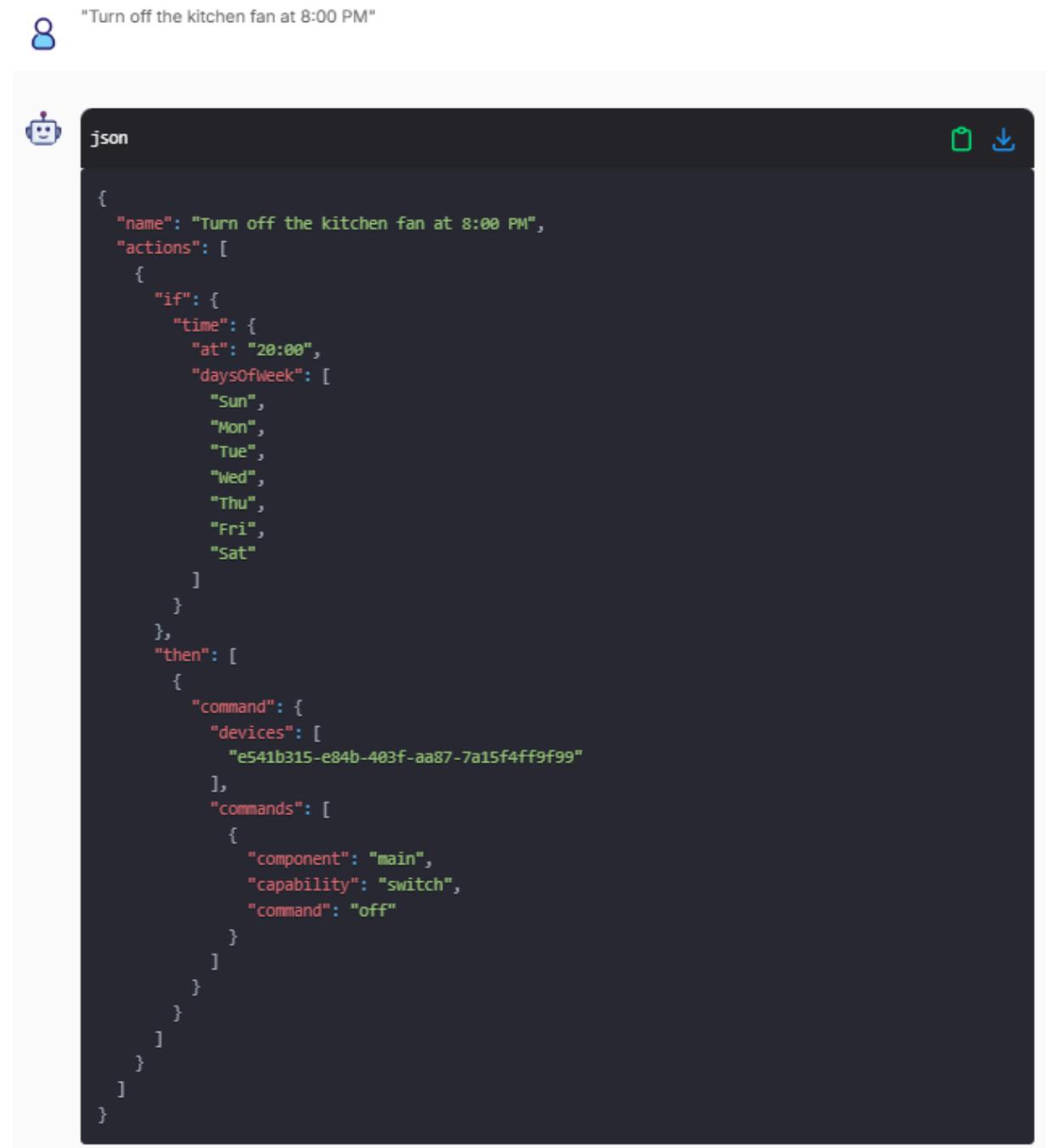
## TYPE 2: TEMPORAL ILLOGICALITY (3 cases)

### Test Case 2.1: Fan Quick Reversal

#### Step 1 - Baseline Rule:

Input: "Turn off the kitchen fan at 8:00 PM"

Output:



The screenshot shows a JSON editor interface with a dark theme. At the top, there are icons for user, robot, and file operations (New, Open, Save). The file name is 'json'. The JSON code defines a rule with the following structure:

```
{
  "name": "Turn off the kitchen fan at 8:00 PM",
  "actions": [
    {
      "if": {
        "time": {
          "at": "20:00",
          "daysOfWeek": [
            "Sun",
            "Mon",
            "Tue",
            "Wed",
            "Thu",
            "Fri",
            "Sat"
          ]
        }
      },
      "then": [
        {
          "command": {
            "devices": [
              "e541b315-e84b-403f-aa87-7a15f4ff9f99"
            ],
            "commands": [
              {
                "component": "main",
                "capability": "switch",
                "command": "off"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

**Conflict:** None

**Success Criteria:** Valid JSON generated.

### Step 2 - Conflicting Rule:

**Input:** "Turn on the kitchen fan at 8:03 PM"

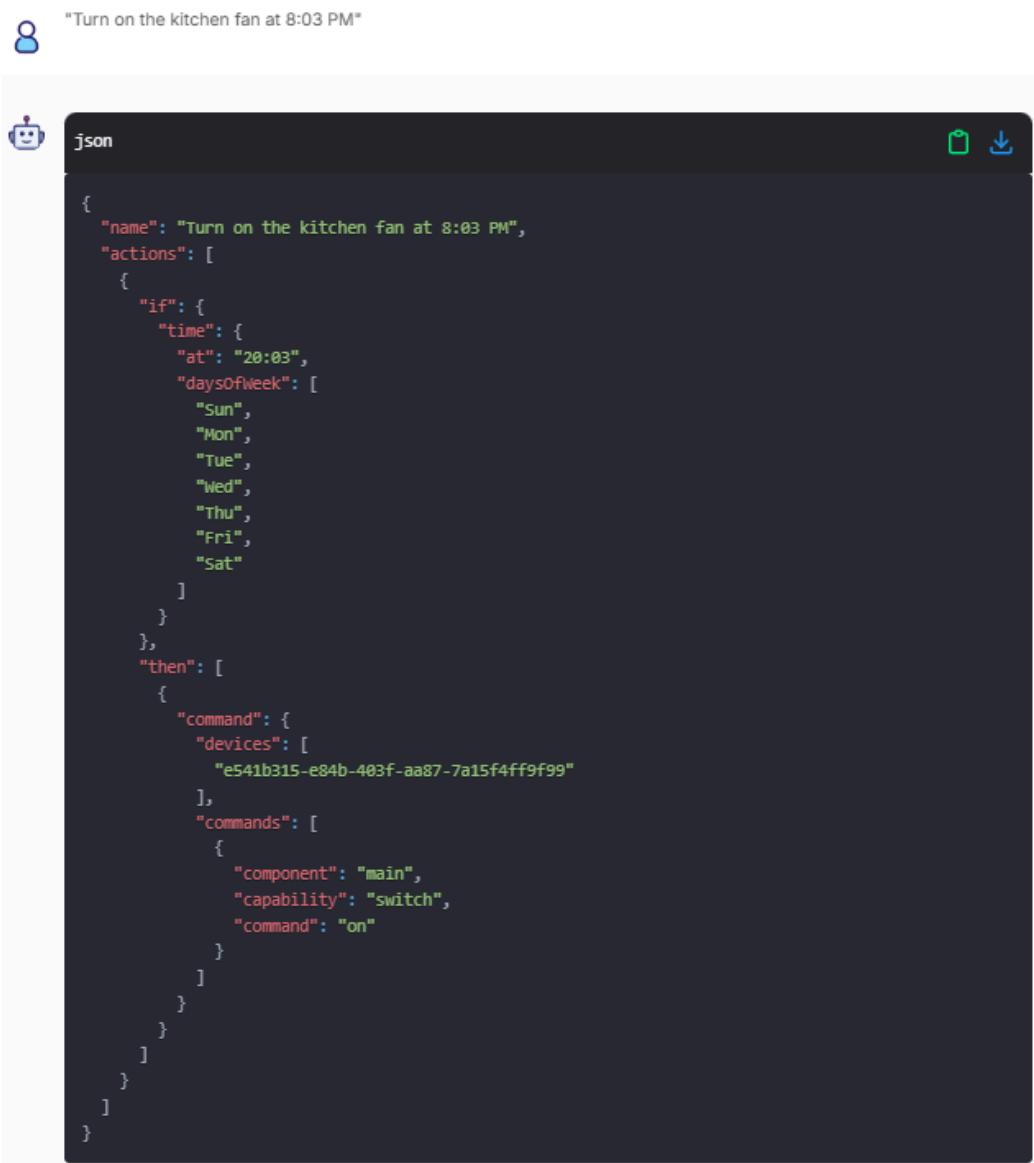
**Expected Behavior:** **CONFLICT DETECTED.** System should flag the 3-minute gap as an illogical rapid cycle and suggest a longer duration.

### Actual Result:

**CONFLICT NOT DETECTED**

The system generated a valid JSON, treating 8:03 PM as a distinct, valid event.

**Failure Analysis:** The vector search identified "Kitchen Fan" but the LLM failed to calculate the 3-minute delta as a "Rapid Cycle" violation.



The screenshot shows a JSON configuration for a scheduled action. The JSON is as follows:

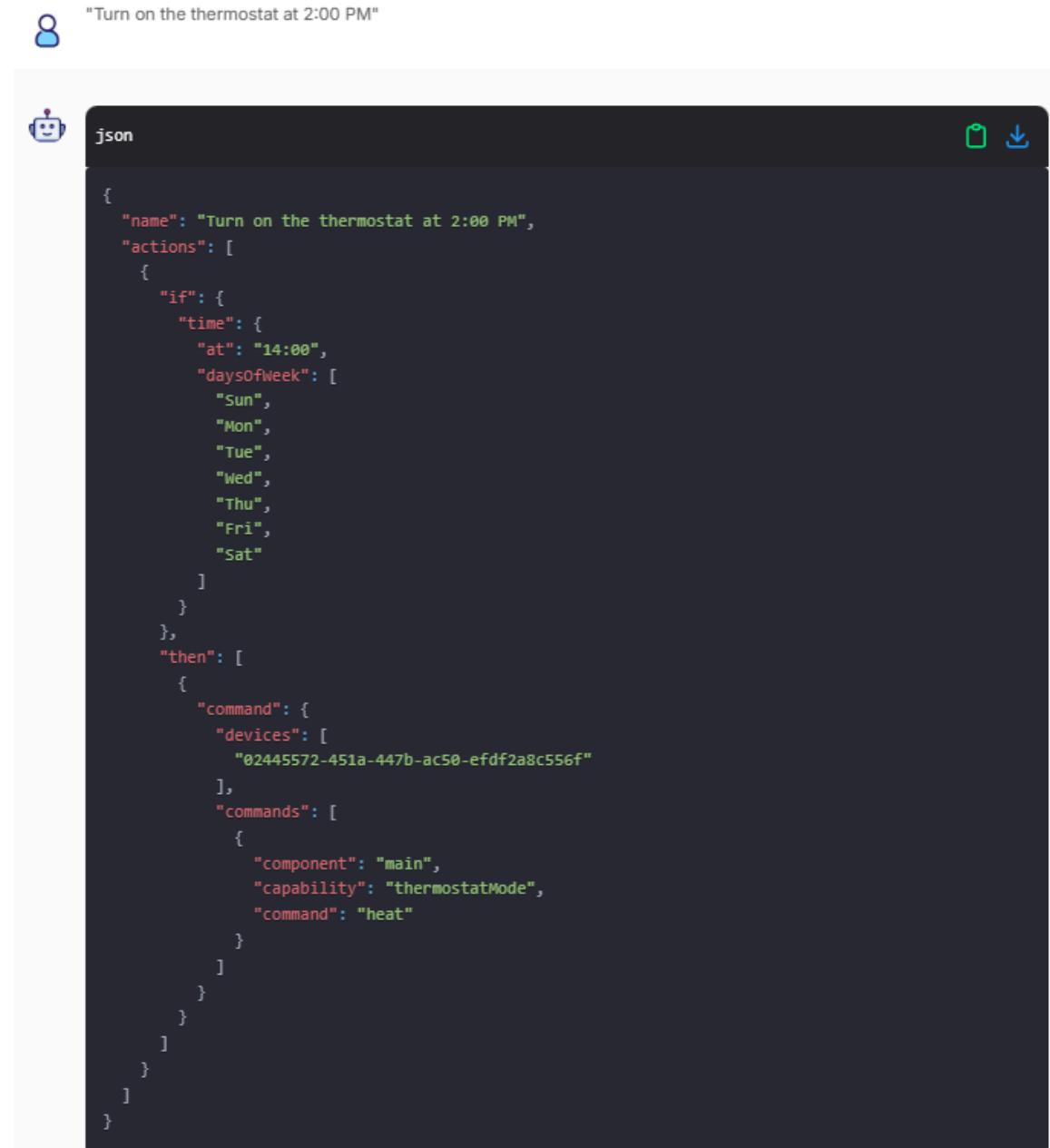
```
{
  "name": "Turn on the kitchen fan at 8:03 PM",
  "actions": [
    {
      "if": {
        "time": {
          "at": "20:03",
          "daysofweek": [
            "Sun",
            "Mon",
            "Tue",
            "Wed",
            "Thu",
            "Fri",
            "Sat"
          ]
        }
      },
      "then": [
        {
          "command": {
            "devices": [
              "e541b315-e84b-403f-aa87-7a15f4ff9f99"
            ],
            "commands": [
              {
                "component": "main",
                "capability": "switch",
                "command": "on"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

## Test Case 2.2: Thermostat Short Cycling

### Step 1 - Baseline Rule:

Input: "Turn on the thermostat at 2:00 PM"

Output:



The screenshot shows a JSON editor interface with a dark theme. On the left, there's a sidebar with icons for a person, a robot, and a file. The main area is titled 'json' and contains the following JSON code:

```
{
  "name": "Turn on the thermostat at 2:00 PM",
  "actions": [
    {
      "if": {
        "time": {
          "at": "14:00",
          "daysOfWeek": [
            "Sun",
            "Mon",
            "Tue",
            "Wed",
            "Thu",
            "Fri",
            "Sat"
          ]
        }
      },
      "then": [
        {
          "command": {
            "devices": [
              "02445572-451a-447b-ac50-efdf2a8c556f"
            ],
            "commands": [
              {
                "component": "main",
                "capability": "thermostatMode",
                "command": "heat"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

**Conflict:** None

**Success Criteria:** Valid JSON generated.

## Step 2 - Conflicting Rule:

**Input:** "Turn off the thermostat at 2:10 PM"

**Expected Behavior:** Conflict Detected

**Success Criteria:** Correct ID, Explanation, and Resolution strategies provided.

### Output:

**CONFLICT DETECTED:** Temporal Illogicality

- **Analysis:** Rule attempts to Turn OFF thermostat 10 minutes after start
- **Impact:** "Short cycling" causes excessive compressor wear and significantly increases energy costs.
- **Resolutions:**
  1. Change the "OFF" time to a later time to ensure the minimum 15-minute operating threshold is met.
  2. Instead of a timer, set the thermostat to turn off when a specific temperature is reached
  3. Create a single rule that handles both the start and stop times with an appropriate duration between them.



"Turn off the thermostat at 2:10 PM"



⚠ CONFLICT DETECTED

Conflict Type: Temporal Illogicality

Existing Rule: Turn on the thermostat at 2:00 PM

Your New Rule: Turn off the thermostat at 2:10 PM

The Problem: Rules are only 10 minutes apart with opposite actions, creating a pointless rapid cycle.

Real-World Impact: The thermostat would turn ON at 2:00 PM and immediately turn OFF at 2:10 PM, wasting energy and potentially damaging the equipment.

Resolution Options:

Extend timing: Change to turn OFF thermostat at a later time (e.g., 10:00 PM)

Remove existing rule: Delete the 2:00 PM ON rule if not needed

Use temperature trigger: Replace with 'turn off when temperature is reached' for automatic control

How would you like to proceed?

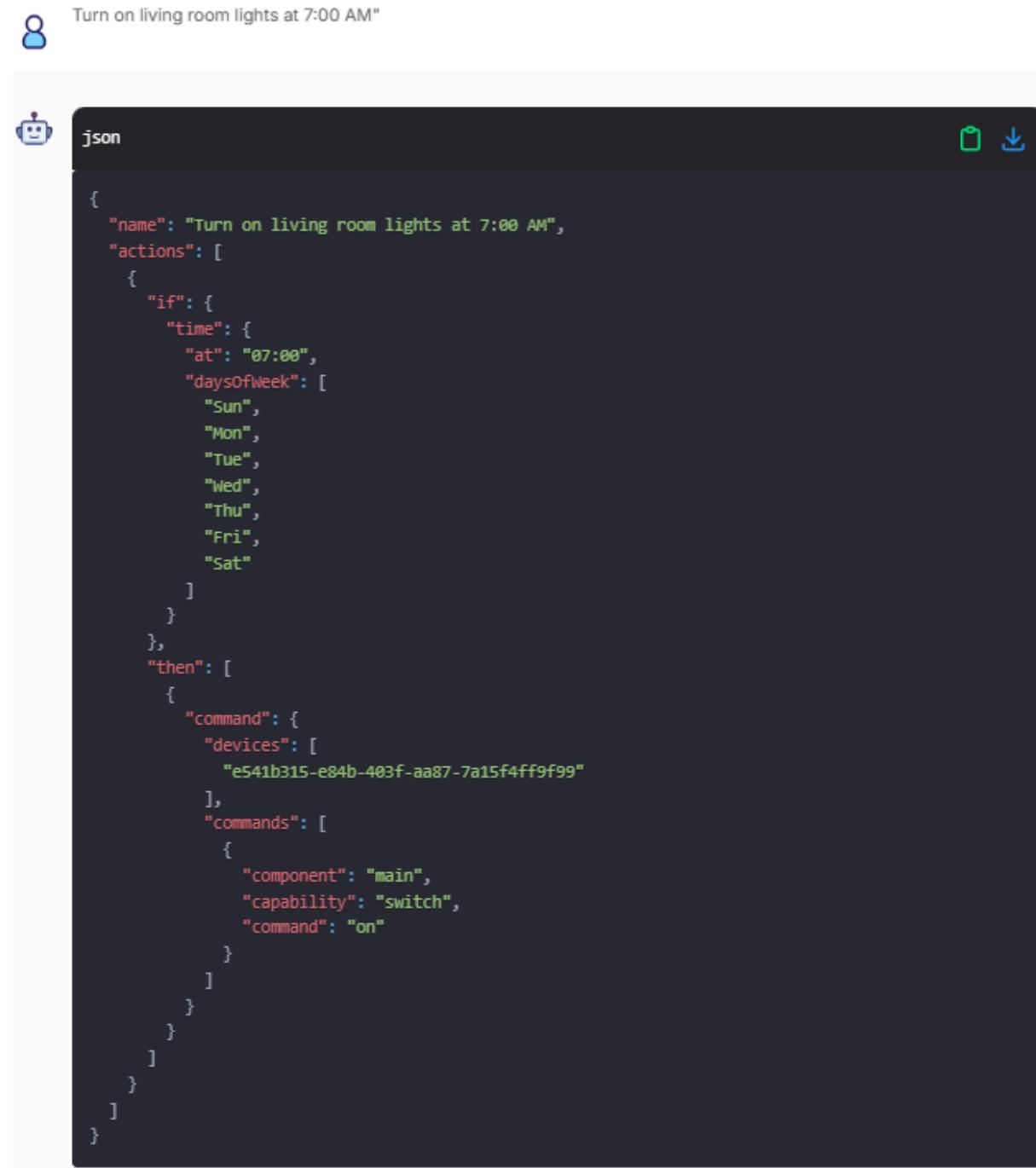
### **Test Case 2.3: Valid Schedule**

#### **Step 1 - Baseline Rule:**

**Input:** "Turn on living room lights at 7:00 AM"

**Output:**

Turn on living room lights at 7:00 AM



The screenshot shows a JSON editor interface with a dark theme. The file is named 'json'. The code is a valid JSON object representing a scheduled action. It defines a rule named 'Turn on living room lights at 7:00 AM' that triggers on specific days at 7:00 AM and turns on a device with a specific component and capability.

```
{
  "name": "Turn on living room lights at 7:00 AM",
  "actions": [
    {
      "if": {
        "time": {
          "at": "07:00",
          "daysOfWeek": [
            "Sun",
            "Mon",
            "Tue",
            "Wed",
            "Thu",
            "Fri",
            "Sat"
          ]
        }
      },
      "then": [
        {
          "command": {
            "devices": [
              "e541b315-e84b-403f-aa87-7a15f4ff9f99"
            ],
            "commands": [
              {
                "component": "main",
                "capability": "switch",
                "command": "on"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

**Conflict:** None

**Success Criteria:** Valid JSON generated.

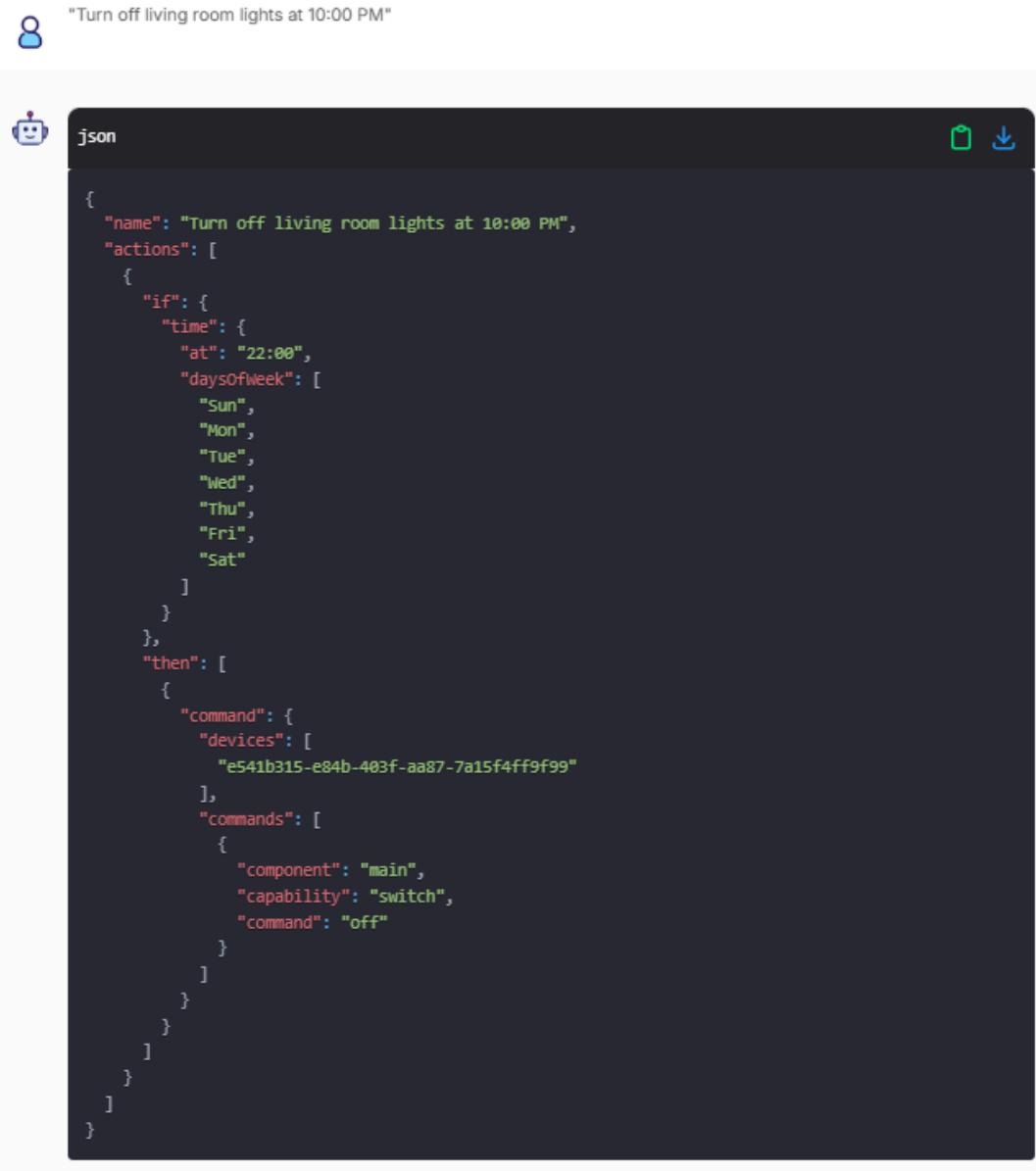
## Step 2 - Non - Conflicting Rule:

**Input:** "Turn off living room lights at 10:00 PM"

**Expected Behavior:** NO CONFLICT DETECTED.

**Success Criteria:** Valid JSON generated

## Output:



The screenshot shows a JSON configuration for a smart home rule. The rule is named "Turn off living room lights at 10:00 PM". It triggers at 22:00 on all days of the week. The action involves turning off a specific device (with ID e541b315-e84b-403f-aa87-7a15f4ff9f99) by sending a "switch" command with component "main" and capability "switch" to turn it off.

```
{
  "name": "Turn off living room lights at 10:00 PM",
  "actions": [
    {
      "if": {
        "time": {
          "at": "22:00",
          "daysofWeek": [
            "Sun",
            "Mon",
            "Tue",
            "Wed",
            "Thu",
            "Fri",
            "Sat"
          ]
        }
      },
      "then": [
        {
          "command": {
            "devices": [
              "e541b315-e84b-403f-aa87-7a15f4ff9f99"
            ],
            "commands": [
              {
                "component": "main",
                "capability": "switch",
                "command": "off"
              }
            ]
          }
        }
      ]
    }
  ]
}
```

## TYPE 3: CONDITION OVERLAPS (3 cases)

### Test Case 3.1: Humidity Threshold Overlap

#### Step 1 - Baseline Rule

**Input:** "Turn on dehumidifier when humidity is above 60%"

**Output:**

👤 "Turn on security lights between 6 PM and 10 PM on weekdays"



The screenshot shows a JSON editor interface with a dark theme. The JSON code is as follows:

```
{
  "name": "Turn on security lights between 6 PM and 10 PM on weekdays",
  "actions": [
    {
      "if": {
        "and": [
          {
            "between": {
              "startTime": "18:00",
              "endTime": "22:00"
            }
          },
          {
            "dayOfWeek": [
              "Mon",
              "Tue",
              "Wed",
              "Thu",
              "Fri"
            ]
          }
        ]
      },
      "then": [
        {
          "command": {
            "devices": [
              "009f3e29-3c26-4489-b94b-6238aea8e22d"
            ],
            "commands": [
              {
                "component": "main",
                "capability": "switch",
                "command": "on"
              }
            ]
          }
        }
      ],
      "else": []
    }
  ]
}
```

**Conflict:** None

**Success Criteria:** Valid JSON generated

## Step 2 - Conflicting Rule

**Input:** "Turn off dehumidifier when humidity is above 65%"

**Expected Behavior:** Conflict Detected

**Success Criteria:** Correct ID, Explanation, and Resolution strategies provided.

**Output:**

### CONFLICT DETECTED: Condition Overlap

- **Analysis:** Overlapping humidity thresholds create ambiguity. When humidity is 66%, both conditions are true ( $> 60$  and  $> 65$ ) but with opposite actions.
- **Impact:** Dehumidifier would turn ON at 61%, then immediately turn OFF at 66%, potentially causing rapid cycling and equipment damage.
- **Resolutions:**
  1. Turn ON when  $> 60\%$ , turn OFF when  $< 55\%$  (creates 5% deadband).
  2. Set target humidity to 60% and let SmartThings manage on/off automatically.
  3. Turn ON when humidity  $> 65\%$ , OFF when humidity  $< 60\%$ .



Turn off security lights between 8 PM and 11 PM every day"



#### ⚠ CONFLICT DETECTED

Conflict Type: Condition Overlap

Existing Rule: Turn on security lights between 6 PM and 10 PM on weekdays

Your New Rule: Turn off security lights between 8 PM and 11 PM every day

The Problem: Overlapping time ranges create ambiguity. On weekdays, between 8 PM and 10 PM, one rule is turning the lights ON and another rule is turning them OFF.

Real-World Impact: The security lights would toggle rapidly between 8 PM and 10 PM on weekdays, wasting energy and potentially damaging the equipment.

Resolution Options:

Adjust time ranges: Change the new rule to turn OFF security lights between 10 PM and 11 PM every day to avoid overlap.

Restrict days: Limit the new rule to weekends only (Sat and Sun) to avoid overlap with the weekday rule.

Replace existing rule: Remove the existing ON rule and keep only the new OFF rule.

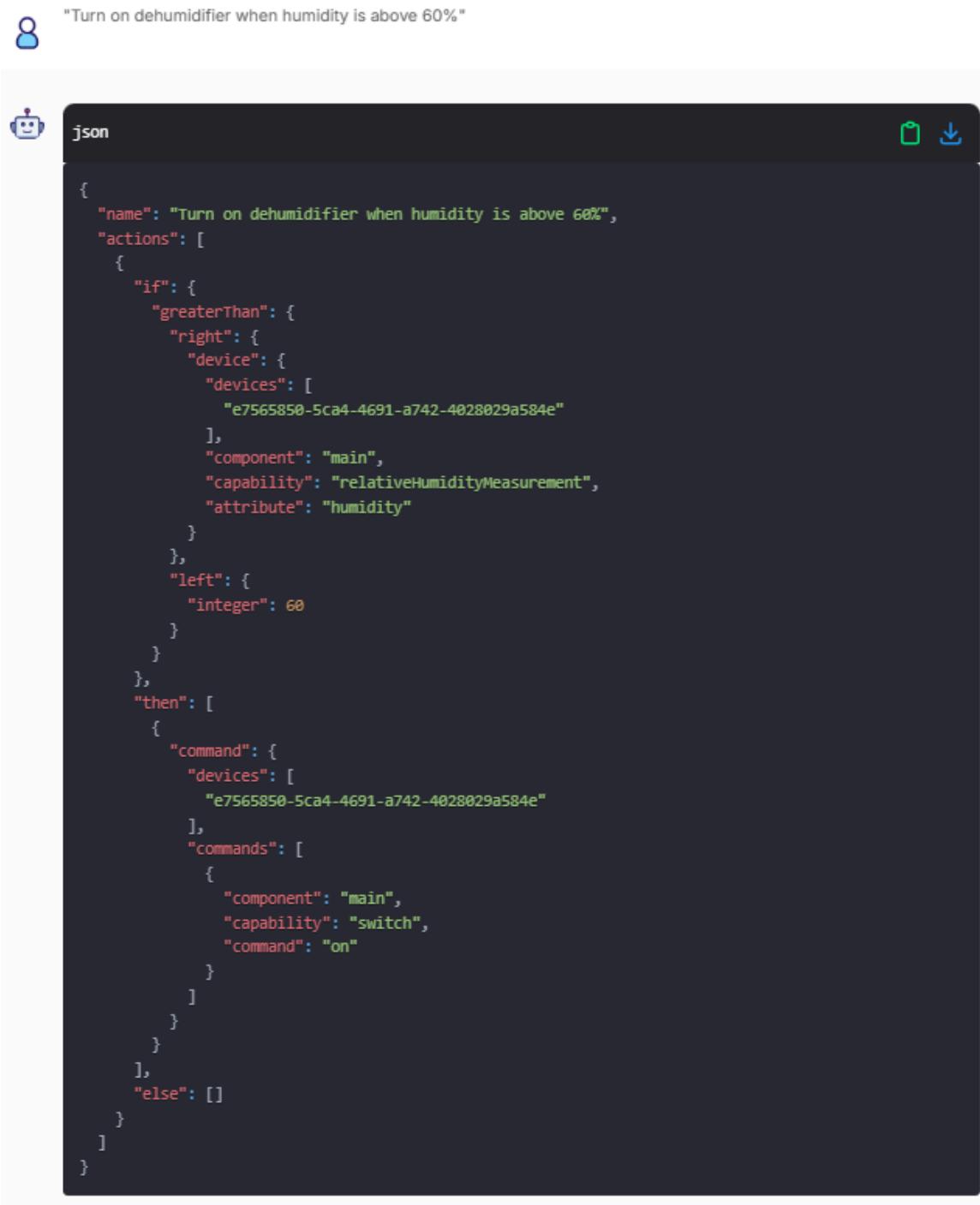
How would you like to proceed?

## Test Case 3.2: Time Range Overlap

### Step 1 - Baseline Rule

**Input:** "Turn on security lights between 6 PM and 10 PM on weekdays"

**Output:**



The screenshot shows a JSON editor interface with a dark theme. At the top, there's a blue user icon, a 'json' file type indicator, and two green circular icons. The main area contains a JSON object representing a rule. The code is as follows:

```
{
  "name": "Turn on dehumidifier when humidity is above 60%",
  "actions": [
    {
      "if": {
        "greaterThan": {
          "right": {
            "device": {
              "devices": [
                "e7565850-5ca4-4691-a742-4028029a584e"
              ],
              "component": "main",
              "capability": "relativeHumidityMeasurement",
              "attribute": "humidity"
            }
          },
          "left": {
            "integer": 60
          }
        }
      },
      "then": [
        {
          "command": {
            "devices": [
              "e7565850-5ca4-4691-a742-4028029a584e"
            ],
            "commands": [
              {
                "component": "main",
                "capability": "switch",
                "command": "on"
              }
            ]
          }
        ],
        "else": []
      }
    ]
}
```

**Conflict:** None

**Success Criteria:** Valid JSON generated

## Step 2 - Conflicting Rule

**Input:** "Turn off security lights between 8 PM and 11 PM every day"

**Expected Behavior:** Conflict Detected

**Success Criteria:** Correct ID, Explanation, and Resolution strategies provided.

**Output:**

**CONFLICT DETECTED: Condition Overlap**

- **Analysis:** Overlapping time ranges create ambiguity. On weekdays, between 8 PM and 10 PM, one rule is turning the lights ON and another rule is turning them OFF.
- **Impact:** The security lights would toggle rapidly between 8 PM and 10 PM on weekdays, wasting energy and potentially damaging the equipment.
- **Resolutions:**
  1. **Adjust time ranges:** Change the new rule to turn OFF security lights between 10 PM and 11 PM every day to avoid overlap.
  2. **Restrict days:** Limit the new rule to weekends only (Sat and Sun) to avoid overlap with the weekday rule.
  3. **Replace existing rule:** Remove the existing ON rule and keep only the new OFF rule.



"Turn off dehumidifier when humidity is above 65%"



⚠ CONFLICT DETECTED

Conflict Type: Condition Overlap

Existing Rule: Turn on dehumidifier when humidity is above 60%

Your New Rule: Turn off dehumidifier when humidity is above 65%

The Problem: Overlapping humidity thresholds create ambiguity. When humidity is 66%, both conditions are true (> 60 and > 65) but with opposite actions.

Real-World Impact: Dehumidifier would turn ON at 61%, then immediately turn OFF at 66%, then potentially toggle repeatedly as humidity fluctuates, causing rapid cycling and equipment damage.

Resolution Options:

Use humidity ranges: Turn ON when > 60%, turn OFF when < 55% (creates 5% deadband)

Single dehumidifier rule: Set target humidity to 60% and let SmartThings manage on/off automatically

Reverse the logic: Turn ON when humidity > 65%, OFF when humidity < 60% (dehumidifying mode)

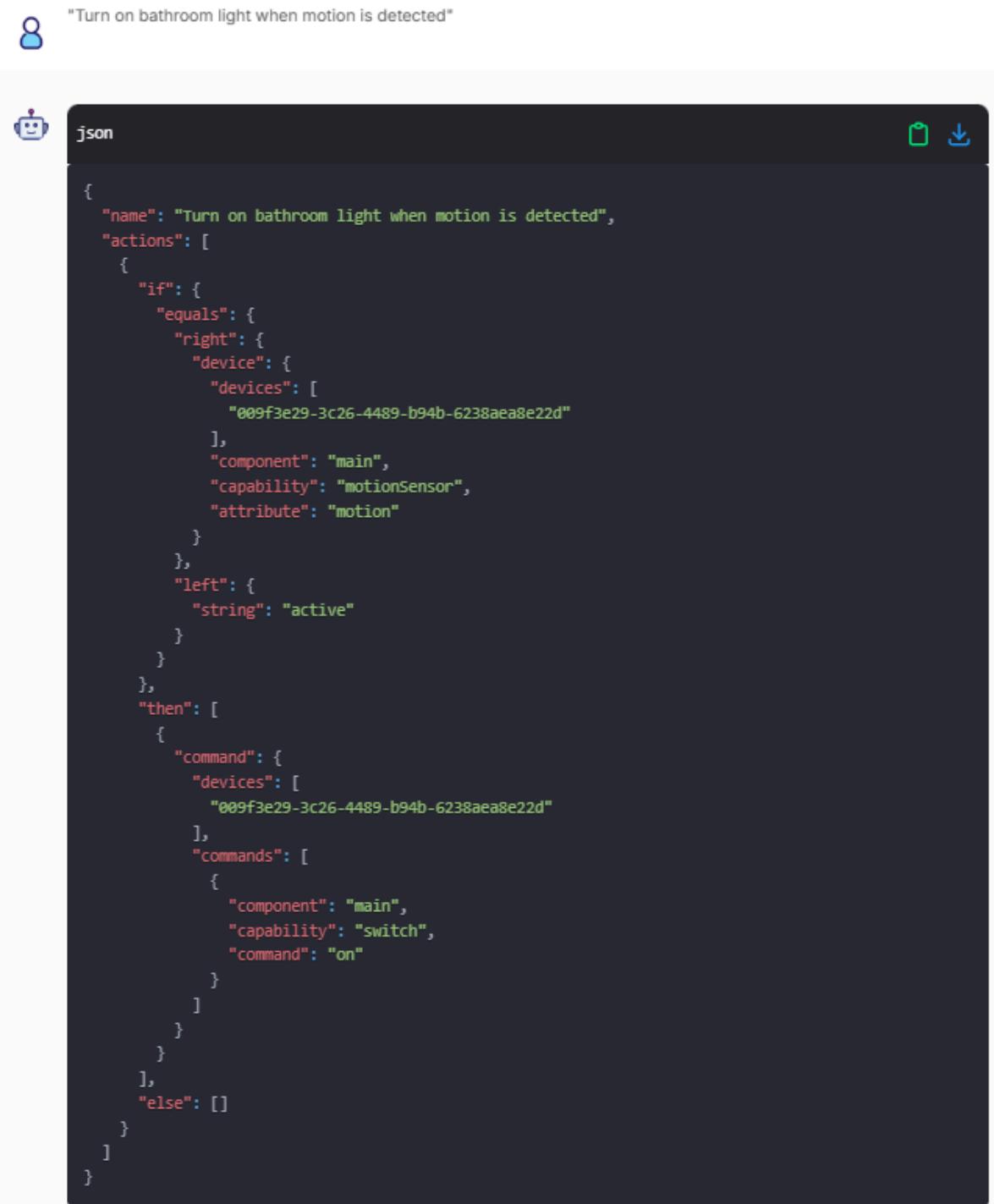
How would you like to proceed?

### Test Case 3.3: Semantic Sensor Overlap (Motion vs. Presence)

#### Step 1 - Baseline Rule

**Input:** "Turn on bathroom light when motion is detected"

**Output:**



The screenshot shows a JSON editor interface with a dark theme. At the top left is a user icon, followed by the text "Turn on bathroom light when motion is detected". To the right are icons for saving and download. The main area contains the following JSON code:

```
{
  "name": "Turn on bathroom light when motion is detected",
  "actions": [
    {
      "if": {
        "equals": {
          "right": {
            "device": {
              "devices": [
                "009f3e29-3c26-4489-b94b-6238aea8e22d"
              ],
              "component": "main",
              "capability": "motionSensor",
              "attribute": "motion"
            }
          },
          "left": {
            "string": "active"
          }
        }
      },
      "then": [
        {
          "command": {
            "devices": [
              "009f3e29-3c26-4489-b94b-6238aea8e22d"
            ],
            "commands": [
              {
                "component": "main",
                "capability": "switch",
                "command": "on"
              }
            ]
          }
        }
      ],
      "else": []
    }
  ]
}
```

**Conflict:** None

**Success Criteria:** Valid JSON generated

## Step 2 - Conflicting Rule

**Input:** "Turn off bathroom light when motion is detected"

**Expected Behavior:** Conflict Detected

**Success Criteria:** Correct ID, Explanation, and Resolution strategies provided.

**Output:**

**CONFLICT DETECTED: Condition Overlap**

- **Analysis:** Both rules trigger on the same condition (motion detected) but perform opposite actions (ON vs OFF).
- **Impact:** When there is motion detected, the system won't know whether to turn the bathroom light ON or OFF, creating unpredictable behavior.
- **Resolutions:**
  1. **Add time restriction:** Turn OFF bathroom light when motion is detected only between certain hours.
  2. **Add a mode restriction:** Turn OFF bathroom light when motion is detected only when in 'Night' mode.
  3. **Replace existing rule:** Remove the ON rule and keep only the OFF rule.



"Turn off bathroom light when motion is detected"



⚠ CONFLICT DETECTED

Conflict Type: Condition Overlap

Existing Rule: Turn on bathroom light when motion is detected

Your New Rule: Turn off bathroom light when motion is detected

The Problem: Both rules trigger on the same condition (motion detected) but perform opposite actions (ON vs OFF).

Real-World Impact: When there is motion detected, the system won't know whether to turn the bathroom light ON or OFF, creating unpredictable behavior.

Resolution Options:

Add time restriction: Turn OFF bathroom light when motion is detected only between certain hours.

Add a mode restriction: Turn OFF bathroom light when motion is detected only when in 'Night' mode.

Replace existing rule: Remove the ON rule and keep only the OFF rule.

How would you like to proceed?