I'll help you understand which files need to be considered when adding new fields to `master.json` and `user\_input.json` to ensure the JSON generation works properly. Let me examine the current structure and identify the key files.

Read file: master.json

Read file: user\_input.json

Now let me examine the core generation files to understand how they work:

Read file: src/mockgen/core.py

Read file: src/mockgen/core.py

Read file: src/mockgen/consolidated\_generator.py

Read file: src/mockgen/consolidated\_generator.py

Read file: src/mockgen/consolidated\_generator.py

Now let me check the data validation file to see if there are any validation rules:

Read file: src/mockgen/data\_validation.py

Based on my analysis of the codebase, here are the \*\*key files you need to consider\*\* when adding new fields to `master.json` and `user\_input.json` to ensure JSON generation works properly:

## 🔑 \*\*Critical Files to Update:\*\*

### 1. \*\*`src/mockgen/consolidated\_generator.py`\*\* (Most Important)

- \*\*Lines 70-85\*\*: Contains the `template\_fields` list that defines the exact order and structure of fields

- \*\*Lines 100-130\*\*: Handles field processing and ensures all template fields have values

- \*\*Lines 140-170\*\*: Processes different field types (simple lists vs. nested objects like ClaimDetails)

### 2. \*\*`src/mockgen/data\_validation.py`\*\*

- \*\*Lines 320-340\*\*: Contains validation rules for required base fields and scenario fields

- \*\*Lines 350-370\*\*: Checks JSON format compliance

### 3. \*\*`src/mockgen/core.py`\*\*

- \*\*Lines 60-80\*\*: Handles data generation and field processing

- \*\*Lines 100-120\*\*: Creates output structure and handles nested data

## �� \*\*What You Need to Update:\*\*

### \*\*In `consolidated\_generator.py`:\*\*

```python

# Update this list (around line 70-75):

template\_fields = [

"first\_proc\_cd", "last\_proc\_cd", "email", "phone", "date\_of\_birth",

"street\_address", "proc\_cd", "mail\_id", "address", "city", "state",

"zip\_code", "country", "PRICNG\_ZIP\_STATE", "CLM\_TYPE", "SRVC\_FROM\_DT",

"HCID", "PAT\_BRTH\_DT", "PAT\_FRST\_NME", "PAT\_LAST\_NME", "ClaimDetails"

# ADD YOUR NEW FIELDS HERE

]

```

### \*\*In `data\_validation.py`:\*\*

```python

# Update required base fields (around line 320-330):

required\_base\_fields = [

"first\_proc\_cd", "last\_proc\_cd", "email", "phone", "date\_of\_birth",

"street\_address", "proc\_cd", "mail\_id", "address", "city", "state",

"zip\_code", "country"

# ADD YOUR NEW BASE FIELDS HERE

]

# Update required scenario fields (around line 340-350):

required\_scenario\_fields = [

"PRICNG\_ZIP\_STATE", "CLM\_TYPE", "SRVC\_FROM\_DT", "HCID",

"PAT\_BRTH\_DT", "PAT\_FRST\_NME", "PAT\_LAST\_NME", "ClaimDetails"

# ADD YOUR NEW SCENARIO FIELDS HERE

]

```

## ⚠️ \*\*Important Considerations:\*\*

1. \*\*Field Order\*\*: The `template\_fields` list determines the exact order of fields in generated JSONs

2. \*\*Data Types\*\*: Ensure your new fields in `user\_input.json` follow the same pattern:

- Simple fields: `"field\_name": ["value1", "value2"]`

- Nested objects: `"field\_name": [{"subfield": ["value1", "value2"]}]`

3. \*\*All Models\*\*: Add your new fields to all model variants:

- `Model\_1`

- `Model\_1\_positive`

- `Model\_1\_negative`

- `Model\_1\_exclusion`

4. \*\*Validation\*\*: The system will automatically validate that all required fields are present

## 🚀 \*\*After Adding Fields:\*\*

1. \*\*Test Generation\*\*: Run `python -m src.mockgen.consolidated\_generator --all` to test

2. \*\*Validate Output\*\*: Run `python -m src.mockgen.data\_validation --test` to verify format

3. \*\*Check Compliance\*\*: Run `python -m src.mockgen.data\_validation --verify` to ensure data compliance

This approach ensures your new fields are properly integrated into the generation pipeline and maintain consistency across all generated JSON files.