📄Exam Factory Documentation

Overview

The **Exam Factory** (`exam_factory.py`) is the core utility that **builds an Exam object from a JSON template**.
It ties together **backend models** (`Exam`, `Segment`, `Measurement`) with **template definitions** (`carotid.json`, `renal.json`, etc.).

This function is used whenever a **new ultrasound exam is created** in the Lumen system — whether it comes from: - **EPIC HL7 arrival messages** (patient scheduled and marked "arrived")
- **Manual entry by a technologist**
- **Agent-based test cases or imports**

💣Responsibilities

The factory has **one clear responsibility**:

> **Given an exam type and site, generate a new `Exam` record with all its segments and measurement placeholders initialized according to the template.**

🕐Workflow (Step-by-Step)

1. Load Template

```
template = get_template(exam_type, site)
```

- Uses `template_registry` to load the correct JSON (e.g., `carotid.json`).
- Template defines: - Segments (prox ICA, mid ICA, vertebral, etc.) - Which measurements each segment has (PSV, EDV, ICA/CCA ratio, etc.) - Units for each measurement - Display groups (Right Side, Left Side, Temporal Arteries)

---

2. Create Exam

```
exam = Exam.objects.create(
    patient_name=patient_name,
    gender=gender,
    mrn=patient_data.get("mrn", ""),
    exam_type=exam_type,
    ...
)
```

- Creates the **parent Exam record** in the database.
- Stores patient metadata, scope, extent, CPT, ICD-10 code, etc.
- Sets initial status = `"draft"`.
- Patient demographics can come from **EPIC HL7 feeds** or manual entry.

---

3. Create Segments

```
segment = Segment.objects.create(
    exam=exam,
    name=seg["id"],
    artery=seg["vessel"].lower(),
    side=seg.get("side", "n/a")
)
```

- Each **anatomical segment** (prox ICA, mid ICA, etc.) becomes a `Segment` DB row.
- Linked to the parent `Exam`.
- `side` is stored as `"left"`, `"right"`, or `"temporal"`.

---

4. Create Measurements

```
measurement = Measurement.objects.create(segment=segment)
```

- For each segment, a **Measurement object** is initialized.
- Defines all the numerical values, categorical flags, and dropdowns.
- Examples: - PSV (cm/s)
- EDV (cm/s)
- ICA/CCA ratio (ratio)
- Plaque morphology
- Direction of flow

---

5. Initialize Fields

```
for m in seg.get("measurements", []):
    field_name = m if isinstance(m, str) else m.get("name")

    if hasattr(measurement, field_name):
        # Core field (DB column)
        setattr(measurement, field_name, None)
    else:
        # Non-core field → store in JSON
        measurement.additional_data[field_name] = None
```

- Supports two styles of template definition:
- **Compact:** `["psv", "edv"]`
- **Verbose:** `[{"name": "psv", "unit": "cm/s"}]`
- Core fields (psv, edv, ica_cca_ratio) → saved directly as DB columns.
- Non-core (artery_diameter, ap_tr, longitudinal, etc.) → saved in `additional_data` JSON field.

---

6. Store Units

```
if field_unit:
    measurement.additional_data[f"{field_name}_unit"] = field_unit
```

- Every measurement has a **unit** (cm/s, ratio, cm, etc.).
- Units stored in `additional_data`, so they are always available for rendering in UI or PDF.

📦Data Model Relationships

Here's the **mental model** for what the factory builds:

```
Exam (Carotid Exam for John Doe)
│
├── Segment (Right ICA Proximal)
│    └── Measurement (PSV=None, EDV=None, Ratio=None, additional_data={})
│
├── Segment (Right ICA Mid)
│    └── Measurement (PSV=None, EDV=None, Ratio=None)
│
├── Segment (Left ICA Proximal)
│    └── Measurement (PSV=None, EDV=None, Ratio=None)
│
└── ...
```

🛠️How It’s Used

- **When HL7 says “Exam Arrived”:**
  `create_exam_from_template("carotid", "mount_sinai_hospital", patient_data, tech_user)`
- **When tech manually creates exam:**
  Same call, but `patient_data` comes from frontend form.
- **When testing with sample patients:**
  Call factory directly in scripts or management commands.

🔎Example Call

```
exam = create_exam_from_template(
    exam_type="carotid",
    site="mount_sinai_hospital",
    patient_data={
        "name": "Jane Doe",
        "gender": "female",
        "mrn": "123456",
        "dob": "1970-01-01",
        "accession": "ACC-98765",
        "scope": "bilateral",
        "extent": "complete",
        "cpt_code": "93880",
        "technique": "Duplex carotid ultrasound",
        "operative_history": "Prior left CEA",
        "indication": "I65.23"  # ICD-10 code
    },
    created_by="tech_001"
)
```

Result in DB: - `Exam` record created
- ~20 `Segment` rows (prox/mid/dist ICA, CCA, vertebral, subclavian, etc.)
- Each with 1 `Measurement` row initialized with empty values

🔗Key Takeaways for New Engineers

- The **factory is the single point of truth** for creating a new exam.
- It ensures template-driven **consistency**: same segments, same fields every time.
- **Core fields** = DB columns.

- **Non-core fields** = `additional_data` JSON.
- **Units** always stored alongside values.
- This enables:
- Flexible frontends (auto-build forms from JSON)
- Reliable backends (calculators and reports always have the right fields)
- Easy onboarding for new exam types (just add a template JSON)