FSIPS — Session Generator Walkthrough

Version 0.1

This document is a **procedural companion** to the FSIPS v0.1 Architecture Overview. It describes, step by step, how the Session Generator produces a concrete, executable practice session from inputs and system state.

---

1. Purpose

The Session Generator is responsible for converting: - abstract exercises - overload and fatigue rules - available practice time

into a **specific, playable practice session** with clear intent and constraints.

This document focuses on *how the generator thinks*, not UI or implementation details.

---

2. Generator Inputs

SessionRequest

- InstrumentProfile
- Available time (minutes)
- Session type: normal | light | heavy | deload
- Goal weights (domain or technique emphasis)

All inputs are explicit and bounded. No free-text interpretation is required.

---

3. Generator State (Conceptual)

At v0.1, the generator assumes minimal or empty state.

Future state (defined elsewhere) will include: - Exercise recency - Rolling volume (7-day / 28-day) - Mastery estimates - Recent fatigue exposure

When state is missing, conservative defaults are used.

---

4. Session Skeleton Construction

The generator first builds a **SessionBlock skeleton** based solely on available time.

Examples: - ≤25 min: Warmup → Technique → Application - 30–45 min: Warmup → Technique → Pitch/Harmony → Application - 60–75 min: Warmup → Technique → Pitch/Harmony → Rhythm → Application

Block order is invariant.

---

5. Block-by-Block Resolution

For each SessionBlock, the generator performs the following steps:

1. Filter exercises by:
2. allowed domains
3. instrument compatibility
4. fatigue compatibility
5. Score remaining exercises using a deterministic need model
6. Select the highest-need exercise
7. Generate a concrete exercise variant within block overload bounds

The generator never invents new exercises.

---

6. Overload Assignment Rules

For a selected exercise:

- Exactly **one primary overload dimension** is progressed at a time
- All overload values are clamped to SessionBlock bounds
- Session type may further cap overloads (e.g., deload)

If the previous attempt failed or was sloppy: - overload is held or reduced

---

7. Fatigue Budgeting

Each session implicitly maintains a fatigue budget.

Rules enforced: - No adjacent F2 blocks - Warmup required if any F2 work appears - F2 exposure limited in total duration

If fatigue budget is exceeded, the generator: - downgrades overload - substitutes a lower-fatigue exercise - or shortens the block

8. Executable Output Requirement

Each SessionBlock must emit: - Selected exercise - Concrete playable variant - Human-readable instructions - Focus cues - Stop / regression conditions

If this cannot be produced, the session is invalid.

---

9. Determinism & Explainability

Given the same inputs and state, the generator will always produce the same session.

Every exercise choice is explainable in terms of: - block intent - exercise need - fatigue constraints

---

Status

Session Generator logic frozen at v0.1