Current State

SME

- 3 heads, quiz head, chat head, content head, which are essentially the same fine tuned model underneath with different prompts in application.
- SME populates KG and uses it (unclear what .ts files do as SME has KG related .py files which seem to do what the .ts files are doing)
- SME uses postgres for chat storage rather than using agent memory layers(mem0).

KG

- No concept of graphs and no graph databases exist.
- KG is just a postgres datastore which maps learning objectives to modules. (no learning journey)

```
def init_database(self):
    """Initialize the database with required tables."""
    with self.get_connection() as conn:
        conn.executescript("""
            CREATE TABLE IF NOT EXISTS learning_objective (
                lo_id TEXT PRIMARY KEY,
                title TEXT NOT NULL,
                description TEXT,
                difficulty SMALLINT CHECK (difficulty BETWEEN 1 AND 5),
                keywords TEXT,  -- JSON array as TEXT
                prereq_lo_ids TEXT DEFAULT '[]',  -- JSON array as TEXT
                created_at TEXT DEFAULT CURRENT_TIMESTAMP
            );

            CREATE TABLE IF NOT EXISTS module (
                module_id TEXT PRIMARY KEY,
                title TEXT NOT NULL,
                description TEXT,
                finalized BOOLEAN DEFAULT FALSE,
                created_at TEXT DEFAULT CURRENT_TIMESTAMP
            );

            CREATE TABLE IF NOT EXISTS module_lo (
                module_id TEXT,
                lo_id TEXT,
                sequence_order INTEGER NOT NULL,
                is_core BOOLEAN DEFAULT TRUE,
                PRIMARY KEY (module_id, lo_id),
                FOREIGN KEY (module_id) REFERENCES module(module_id) ON DELETE CASCADE,
                FOREIGN KEY (lo_id) REFERENCES learning_objective(lo_id) ON DELETE CASCADE
            );

            CREATE INDEX IF NOT EXISTS idx_module_lo_sequence
            ON module_lo (module_id, sequence_order);
        """)
```

- All the functions present inside KG are mostly crud applications for querying the database.
- KG is all about retrieving and storing there is no knowledge inferring.
- KLIfication done as a single prompt to the smaller model.

```
    @staticmethod
    def _klify_single_lo(lo: LearningObjectiveInput) -> Dict[str, Any]:
        """Convert a single Learning Objective to KLI format using 1.7b model."""

        prompt = f"""Convert this Learning Objective to KLI (Knowledge, Learning, In

itle: {lo.title}
escription: {lo.description}
ifficulty: {lo.difficulty}
eywords: {', '.join(lo.keywords)}

eturn ONLY JSON in this format:
{
 "lo_id": "{lo.lo_id}",
 "title": "{lo.title}",
 "kli_format": {{
   "knowledge": "What specific knowledge/concepts are covered",
   "learning": "How the learner will acquire this knowledge",
   "instruction": "What instructional methods/activities will be used"
 }}
}"""
```

- There are two KG Ingestion Modules:

- One within SME
- One as separate TypeScript application

Which essentially do the same thing

Learner

- Has 26 attributes which can be seen in decision trees but SME using only 5.
- They themselves could list only 6 of those.
- Most of the learner analytics is via forms.
- Other analysis involves metrics like retries and time taken.
- No grounding proofs regarding the use of such methods if they can be used to capture user characteristics.