# Upgraded Database & Data Matching Plan for SIH PM Internship Project

## Step 1: Suggested Database Schema (Upgraded)

```
Students Table:
CREATE TABLE students (
    student_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(200),
    email VARCHAR(200) UNIQUE,
    availability DATE,
    cgpa FLOAT,
    resume_url VARCHAR(300)
);

Internships Table:
CREATE TABLE internships (
    internship_id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(200),
    description TEXT,
    start_date DATE,
    end_date DATE,
    location VARCHAR(100)
);

Skills Table:
CREATE TABLE skills (
    skill_id INT AUTO_INCREMENT PRIMARY KEY,
    skill_name VARCHAR(100) UNIQUE
);

Student Skills Mapping:
CREATE TABLE student_skills (
    student_id INT,
    skill_id INT,
    FOREIGN KEY (student_id) REFERENCES students(student_id),
    FOREIGN KEY (skill_id) REFERENCES skills(skill_id)
);

Internship Skills Mapping:
CREATE TABLE internship_skills (
    internship_id INT,
    skill_id INT,
    FOREIGN KEY (internship_id) REFERENCES internships(internship_id),
    FOREIGN KEY (skill_id) REFERENCES skills(skill_id)
);

Matches Table:
CREATE TABLE matches (
    match_id INT AUTO_INCREMENT PRIMARY KEY,
    student_id INT,
    internship_id INT,
    match_score FLOAT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (student_id) REFERENCES students(student_id),
    FOREIGN KEY (internship_id) REFERENCES internships(internship_id)
);
Match Log Table (for transparency):
```

```
CREATE TABLE match_log (
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    run_id VARCHAR(100),
    student_id INT,
    internship_id INT,
    score FLOAT,
    algorithm_version VARCHAR(50),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

## Step 2: Matching Pipeline

1. Fetch: Collect student skills (joined from student_skills) and internship skills (from internship_skills). 2. Preprocess: Lowercase, remove stopwords, lemmatize. 3. Vectorize: Convert text and skill sets into embeddings (Sentence Transformers or OpenAI models). 4. Similarity: Compute cosine similarity between embeddings. 5. Rank: Store results in matches table with a score (0–1). 6. Log: Save all runs in match_log for transparency and audits.

## Step 3: Example Matching Code (Python)

```
from sentence_transformers import SentenceTransformer, util
import mysql.connector

model = SentenceTransformer('all-MiniLM-L6-v2')

conn = mysql.connector.connect(
    host="localhost", user="root", password="yourpass", database="pm_internship"
)
cursor = conn.cursor(dictionary=True)

cursor.execute("SELECT student_id, GROUP_CONCAT(skills.skill_name) as skills "
               "FROM students "
               "JOIN student_skills ON students.student_id = student_skills.student_id "
               "JOIN skills ON student_skills.skill_id = skills.skill_id "
               "GROUP BY student_id")
students = cursor.fetchall()

cursor.execute("SELECT internship_id, GROUP_CONCAT(skills.skill_name) as req_skills "
               "FROM internships "
               "JOIN internship_skills ON internships.internship_id = internship_skills.internsh
               "JOIN skills ON internship_skills.skill_id = skills.skill_id "
               "GROUP BY internship_id")
internships = cursor.fetchall()

for s in students:
    student_vec = model.encode(s['skills'], convert_to_tensor=True)
    for i in internships:
        job_vec = model.encode(i['req_skills'], convert_to_tensor=True)
        score = util.cos_sim(student_vec, job_vec).item()
        cursor.execute(
            "INSERT INTO matches (student_id, internship_id, match_score) VALUES (%s, %s, %s)",
            (s['student_id'], i['internship_id'], round(score, 4))
        )
conn.commit()
```

## Step 4: Reporting & Transparency

- Top 5 matches per student: SELECT * FROM matches WHERE student_id = 101 ORDER BY match_score DESC LIMIT 5; - Audit transparency: match_log records each run with algorithm version. - Fairness check: analyze distribution of scores across demographics/skills.

## Final Deliverables:

1. Strongly Normalized Database Schema with Skills Mapping 2. Matching Engine Code with AI Embeddings 3. Match Log for Transparency & Fairness 4. Reporting Queries for Top Matches and Audits 5. Demo-ready dataset with students, internships, skills