📘 Routine Extractor

The **Routine Extractor** is a personal side project designed specifically for college students. It helps students **predict their future attendance** by calculating it based on selected upcoming classes.

For example, suppose today is Monday, and a student wants to know their expected attendance **after attending all classes on Tuesday, Wednesday, and the first three classes on Thursday**. This tool enables them to calculate that easily.

To make this possible, the project is broken down into multiple stages. For ease of use, students are required to **upload their class routine in PDF format** and provide their **current attendance details**. After that, the system takes care of the rest.

🚩 Problem Statement

Most college routines contain inconsistent or incomplete data, such as missing course codes, incorrect course names, or data anomalies in the PDF files. Therefore, the first step is to **clean and refine** the routine data and prepare a **CSV version** that includes only the correct course codes for accurate processing.

🎯 Objectives

The primary goals of this project are:

1. Extract and modify class routine data from a PDF file
2. Clean and normalize values
3. Remove data anomalies
4. Standardize data format
5. Replace text with appropriate course codes
6. Export the cleaned routine as a structured CSV file

🛠️ Methodology

Step 1: Table Extraction

We use the `pdfplumber` Python library to extract tabular data from the routine PDF file. The extracted table often contains extra metadata rows at the top. These rows are **not relevant** to the class schedule and are always fixed in number (e.g., the first three rows).

After removing them:

- The **first row** is treated as the column header, representing class times.
- The **first column** of each row represents the weekday (e.g., MON, TUE) and is **used only as a row label**, not for data processing.
- Each cell (excluding the header and day label) is sent to a processing function.

Step 2: Data Processing

The data processing function:

- Takes each cell value as a string and splits it using the `\n` character.
- Applies a **regular expression** to extract a valid course code if present.
- If no course code is found, the function attempts to **match the course name** with a preprocessed list using partial string matching.
- If a match is found, it replaces the value with the corresponding **course code**.
- If no valid course code or name is detected or if the entry doesn't exist in the preprocessed list, it returns `None`.

Sample Preprocessed Mapping:


```
{
    "course_code": [
        "APTI401", "BUPRP", "SBC", "BCA47111(T)", "BCA47111(P)",
        "BCA49112", "BCA47113(T)", "BCA47113(P)", "BCA40201", "BCA40202"
    ],
    "course_name": [
        "Aptitude-IV", "Preparatory Paper", "Soft Skill Boot Camp",
        "Design and Analysis of Algorithm", "Design and Analysis of
Algorithm",
        "PHP and MySQL Lab", "Full-Stack Development-I", "Full-Stack
Development-I",
        "Sustainability in Indian Knowledge System", "Computer Network"
    ]
}
```

✅ Output Format

The final result is a **Pandas** `DataFrame` structured like this:

Days	8:00 - 9:00	9:00 - 10:00	10:00 - 11:00	11:00 - 12:00	12:00 - 1:00	1:00 - 2:00	2:00 - 3:00	3:00 - 4:00	4:00 - 5:00	5:00 - 6:00	6:00 - 7:00
MON											
TUE		BCA40202	BCA40202		BCA49112						
WED			SBC		BCA40201	BCA40201		BCA47113(P)	BCA47113(P)	BCA47113(P)	
THU			SBC		BCA40202	BCA47113(P)		BCA47111(P)	BCA47111(P)	BCA47111(P)	
FRI			BCA40201		BCA47111(T)			BCA49112	BCA49112	BCA49112	
SAT		BCA40201	BCA40202	BCA47111(P)		BCA47111(T)	BCA47113(T)	BCA47113(T)	APTI401		

This table serves as the basis for calculating attendance forecasts.

🔍 Possible Implementation Approaches

1. Use Generative AI APIs

Use free APIs from large language models (e.g., ChatGPT, Claude, Gemini). Provide them with a well-crafted prompt and the routine PDF. The model will return the refined CSV data, which can then be parsed and saved.

2. Build a Custom Classifier

Build a Python-based classification module that uses regex, fuzzy matching, and manual rules to extract and map course data accurately.

🔗 GitHub Repository

You can find the complete source code of this project on GitHub:

🙏 Thank You

This project is still evolving. Feedback and contributions are welcome! The aim is to make this tool more accurate and student-friendly over time.