Project Report: SmartTrack

Student Name: VEDANT PARASHAR

Course: BTECH CSE AIML

1. Introduction

SmartTrack is a Python-based software solution designed to automate the attendance marking process. By using Python's robust standard libraries, this project provides a distraction-free Command Line Interface (CLI) for teachers to manage their class records efficiently.

2. Functional Requirements

The system is divided into three distinct modules:

1. **Student Registration Module**:
   - Accepts roll number and name.
   - Validates input to ensure no duplicates.
   - Saves new students to the database immediately.
2. **Attendance Processing Module**:
   - Auto-detects the current date (system date).
   - Iterates through the student list, asking the user for status (Present/Absent).
   - Updates the attendance log.
3. **Reporting Module**:
   - Reads the historical data.
   - Calculates percentage: (Days Present / Total Days) * 100.
   - Displays a formatted table with "Good" or "Low" status indicators.

3. Non-Functional Requirements

1. **Usability**: The menu-driven interface (1-5) is intuitive and requires no training to use.
2. **Reliability**: The application uses try-except blocks to handle file errors (e.g., if the data file is missing or corrupted).
3. **Persistence**: Data is stored in JSON format, ensuring records persist across multiple sessions.
4. **Portability**: Being a single Python script, it can run on any machine (Windows/Mac/Linux) with Python installed.

4. System Architecture

- **Input**: Keyboard (User types in numbers and names).
- **Processing**: Python Logic (Functions like add_student, calculate_percentage).
- **Storage**: attendance_data.json (A lightweight text-based database).

- **Output**: Terminal Console (Displays tables and menus).

5. Implementation Details

The project uses the following Python concepts:

- **Lists and Dictionaries**: To store student data in memory.
- **File I/O (JSON)**: To read and write data to the hard drive.
- **Loops & Conditionals**: For the main menu and validating user input.
- **Functions**: To keep the code modular and organized.

6. Challenges Faced

- **Data Structure**: Designing the JSON structure was tricky. I decided to store attendance as {"YYYY-MM-DD": [List of Present Roll Numbers]} because it makes counting present days very efficient.
- **Error Handling**: Handling cases where the JSON file didn't exist yet. I added a check if not os.path.exists() to create a fresh file automatically.

7. Future Enhancements

1. **GUI**: Add a graphical interface using Tkinter.
2. **Export**: Add a feature to save the report as a .csv file for Excel.
3. **Delete**: Add a feature to remove a student or delete an incorrect attendance record.