Project Title	**SecureCheck: A Python-SQL Digital Ledger for Police Post Logs**
Skills take away From This Project	**Python, SQL, Streamlit**
Domain	**Law Enforcement & Public Safety** **Real-time Monitoring Systems**

Problem Statement:

Police check posts require a **centralized system** for logging, tracking, and analyzing vehicle movements. Currently, **manual logging and inefficient databases** slow down security processes. This project aims to build an **SQL-based check post database** with a **Python-powered dashboard** for real-time insights and alerts.

Business Use Cases:

Real-time logging of vehicles and personnel.

Automated suspect vehicle identification using SQL queries.

Check post efficiency monitoring through data analytics.

Crime pattern analysis with Python scripts.

Centralized database for multi-location check posts.

Approach:

Data Collection & Storage

- Define SQL schema for police stop records.
- Store data in PostgreSQL/MySQL.
- Use Python (pandas, sqlalchemy) to insert and query data.

Step 1: Python for Data Processing

- Remove the columns that only contains missing value
- Handle the NAN values

Step 2: Database Design (SQL)

- **Tables: Example(Create one table and insert that values into sql)**

Step 3: Streamlit Dashboard

- Display vehicle logs, violations, and officer reports.
- Implement SQL-based search filters for quick lookups.
- Generate analytics and trends (e.g., high-risk vehicles).
-

Example:

- 🚗 A **27-year-old male driver** was stopped for **Speeding** at **2:30 PM**. No search was conducted, and he received a **citation**. The stop lasted **6-15 minutes** and was **not drug-related**.

Results:

- Faster check post operations using SQL queries.
- Automated alerts for flagged vehicles.
- Real-time reporting of security violations.
- Data-backed decision-making for law enforcement.

Project Evaluation metrics:

- Query Execution Time: Optimize SQL queries for fast lookups.
- Data Accuracy: Ensure correct log entries and flagged reports.
- System Uptime: Real-time updates without lag.
- User Engagement: Officers' interaction with the system.
- Violation Detection Rate: Percentage of flagged vehicles identified.

Technical Tags:

Python

Data preprocessing

PostgreSQL / MySQL / SQLite

Streamlit for dashboard creation

SQL QUERIES :

(Medium level):

🚗 Vehicle-Based

1. What are the top 10 vehicle_Number involved in drug-related stops?

2. Which vehicles were most frequently searched?

🧍 Demographic-Based

4. Which driver age group had the highest arrest rate?

5. What is the gender distribution of drivers stopped in each country?

6. Which race and gender combination has the highest search rate?

🕑 Time & Duration Based

7. What time of day sees the most traffic stops?

8. What is the average stop duration for different violations?

9. Are stops during the night more likely to lead to arrests?

⚖️ Violation-Based

10. Which violations are most associated with searches or arrests?

11. Which violations are most common among younger drivers (<25)?

12. Is there a violation that rarely results in search or arrest?

🌍 Location-Based

13. Which countries report the highest rate of drug-related stops?

14. What is the arrest rate by country and violation?

15. Which country has the most stops with search conducted?

(Complex):

1. Yearly Breakdown of Stops and Arrests by Country (Using Subquery and Window Functions)

2. Driver Violation Trends Based on Age and Race (Join with Subquery)

3. Time Period Analysis of Stops (Joining with Date Functions) , Number of Stops by Year,Month, Hour of the Day

4. Violations with High Search and Arrest Rates (Window Function)

5. Driver Demographics by Country (Age, Gender, and Race)

6. Top 5 Violations with Highest Arrest Rates

Data Set:

Data_set link:

traffic_stops

Dataset Explanation:

1️⃣ **stop_date** – The date when the stop happened.

2️⃣ **stop_time** – The time of the stop.

3️⃣ **country_name** – The country where the stop took place.

4️⃣ **driver_gender** – The gender of the driver (`Male` or `Female`).

5️⃣ **driver_age_raw** – The recorded age of the driver (before cleaning).

6️⃣ **driver_age** – The actual age of the driver (after cleaning).

7️⃣ **driver_race** – The race/ethnicity of the driver.

8️⃣ **violation_raw** – The original reason for the stop (before cleaning).

9️⃣ **violation** – The type of violation (`Speeding`, `DUI`, etc.).

🔟 **search_conducted** – Whether the police searched the driver or vehicle (`True/False`).

1️⃣1️⃣ **search_type** – The type of search (`Frisk`, `Vehicle Search`, etc.).

1️⃣2️⃣ **stop_outcome** – The result of the stop (`Warning`, `Citation`, `Arrest`).

1️⃣3️⃣ **is_arrested** – Whether the driver was arrested (`True/False`).

1️⃣4️⃣ **stop_duration** – How long the stop lasted (`<5 min`, `6-15 min`, etc.).

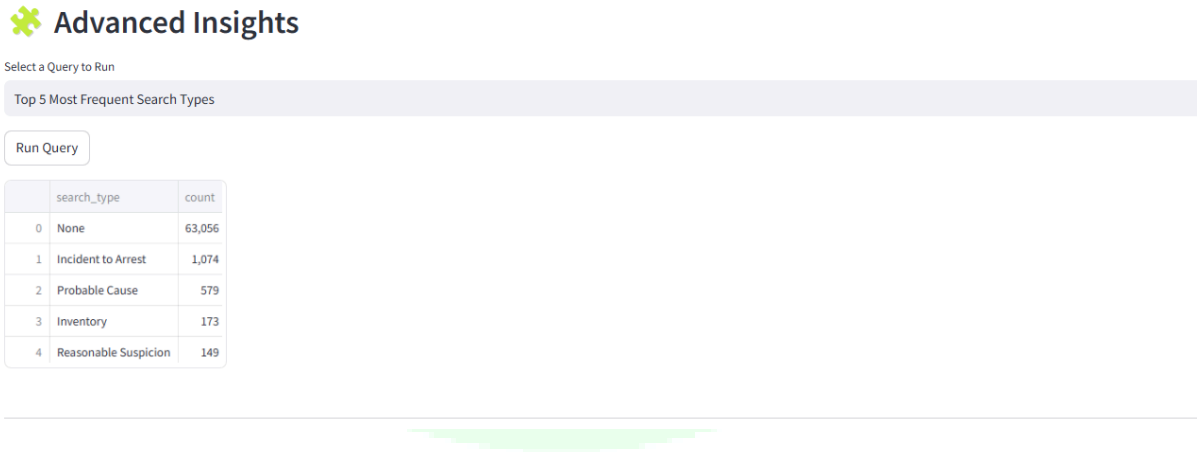
1️⃣5️⃣ **drugs_related_stop** – Whether the stop was drug-related (`True/False`).

Project Deliverables:

- SQL Database Schema (PostgreSQL/MySQL)
- Python Scripts for Data Processing
- Streamlit Dashboard for Check Post Management
- Automated SQL Reports and Logs

- Documentation for System Usage

Sample picture of streamlit :



Project Guidelines:

Use optimized SQL queries with proper indexing.

Implement role-based access for officers.

Ensure data integrity with foreign key constraints.

Regularly update flagged vehicle lists in SQL.

Timeline:

The project must be completed and submitted **within 10 days from the assigned date**.

References:

Streamlit recording (Tamil)	**Recording links**
Reference Document	**https://docs.streamlit.io/develop/api-reference**
Streamlit recording (English)	**Special session for STREAMLIT(11/…**
Project Live Evaluation	**Project Live Evaluation**
Capstone Explanation Guideline	**Capstone Explanation Guideline**
GitHub Reference	**How to Use GitHub.pptx**
Project Orientation (Tamil)	Project Orientation Session : SecureC…
Project Orientation (English)	Project Orientation Session : SecureCheck: A …

Project Excellence series Recordings	Project Excellence Series: Guided Le…
VS_Code installation	How_to_install_vscode_along_with_…

PROJECT DOUBT CLARIFICATION SESSION (PROJECT AND CLASS DOUBTS)

About Session: The Project Doubt Clarification Session is a helpful resource for resolving questions and concerns about projects and class topics. It provides support in understanding project requirements, addressing code issues, and clarifying class concepts. The session aims to enhance comprehension and provide guidance to overcome challenges effectively.
Note: Book the slot at least before 12:00 Pm on the same day

Timing: Monday-Saturday (4:00PM to 5:00PM)

Booking link :https://forms.gle/XC553oSbMJ2Gcfug9

LIVE EVALUATION SESSION (CAPSTONE AND FINAL PROJECT)

About Session: The Live Evaluation Session for Capstone and Final Projects allows participants to showcase their projects and receive real-time feedback for improvement. It assesses project quality and provides an opportunity for discussion and evaluation.
Note: This form will Open only on Saturday (after 2 PM) and Sunday on Every Week

Timing: Monday-Saturday (05:30PM to 07:00PM)

Booking link : https://forms.gle/1m2Gsro41fLtZurRA

Approval Workflow

Created By:	Verified By:	Approved By:
Gomathi A	Shadiya P P	Nehlath Harmain