INTELLIGENT-SURVEILLANCE-CAMERA

Project submitted to the

APSCHE

Bachelor of Technology

In

Computer Science and Engineering-Data science

Aditya college of engineering and Technology

Submitted By

REGIDI MADHAVI – 24P35A4453

Under the guidance of

K.Narmada Mani

K.Veera Vanitha

June 2025

.

ABSTRACT

This project presents the design and implementation of an Intelligent Surveillance Camera system aimed at enhancing security and monitoring capabilities through real-time analysis. Unlike traditional surveillance

systems, this intelligent system incorporates advanced features such as motion detection, facial recognition, object tracking, and anomaly detection using machine learning and computer vision techniques. The camera operates autonomously, capturing and processing video data, and instantly alerts users of suspicious activities through notifications or alarms. It is suitable for both indoor and outdoor environments and can be integrated with cloud storage for remote access and long-term data management. The system demonstrates improved efficiency, accuracy, and responsiveness, making it a reliable solution for modern security challenges. Let me know if you'd like it tailored for a specific application e.g., traffic monitoring, home security, industrial use, etc.Intelligent surveillance cameras integrate advanced technologies such as computer vision, machine learning, and sensor fusion to enhance security systems by automating the detection, tracking, and analysis of objects and behavior in real-time. These systems aim to reduce the need for continuous human monitoring and improve the efficiency and effectiveness of surveillance operations.

INTRODUCTION

In the 21st century, An Intelligent Surveillance Camera leverages technologies such as computer vision, data analytics, and machine learning to make video surveillance smarter, faster, and more reliable. These systems are widely used in public places like airports, railway stations, schools, colleges, banks, and even homes to ensure safety and prevent crime.

In this project, we utilize Python programming along with its powerful data analysis libraries (such as Pandas, NumPy, Matplotlib, and Seaborn) to analyze

motion detection data, object counts, and event timestamps captured through surveillance systems. The core goal is to transform raw video data into meaningful insights — such as peak activity periods, frequent entry/exit times, and potential security breaches.

The integration of data science enables smarter decision-making — such as identifying high-risk zones, optimizing security staff schedules, reducing false alarms, and generating insightful daily reports. With the increasing demand for smart cities and intelligent security solutions, this project serves as a small but impactful example of how surveillance can be made efficient, automated, and data-driven.

This comprehensive analysis will help in identifying critical trends, high-risk groups, and essential areas where intervention and awareness programs can help in managing social media addiction effectively among students.

· SYSTEM REQUIREMENTS

Hardware Requirements:

- A laptop or desktop computer
- Minimum 4 GB RAM (8 GB recommended)
- Minimum 2.0 GHz processor

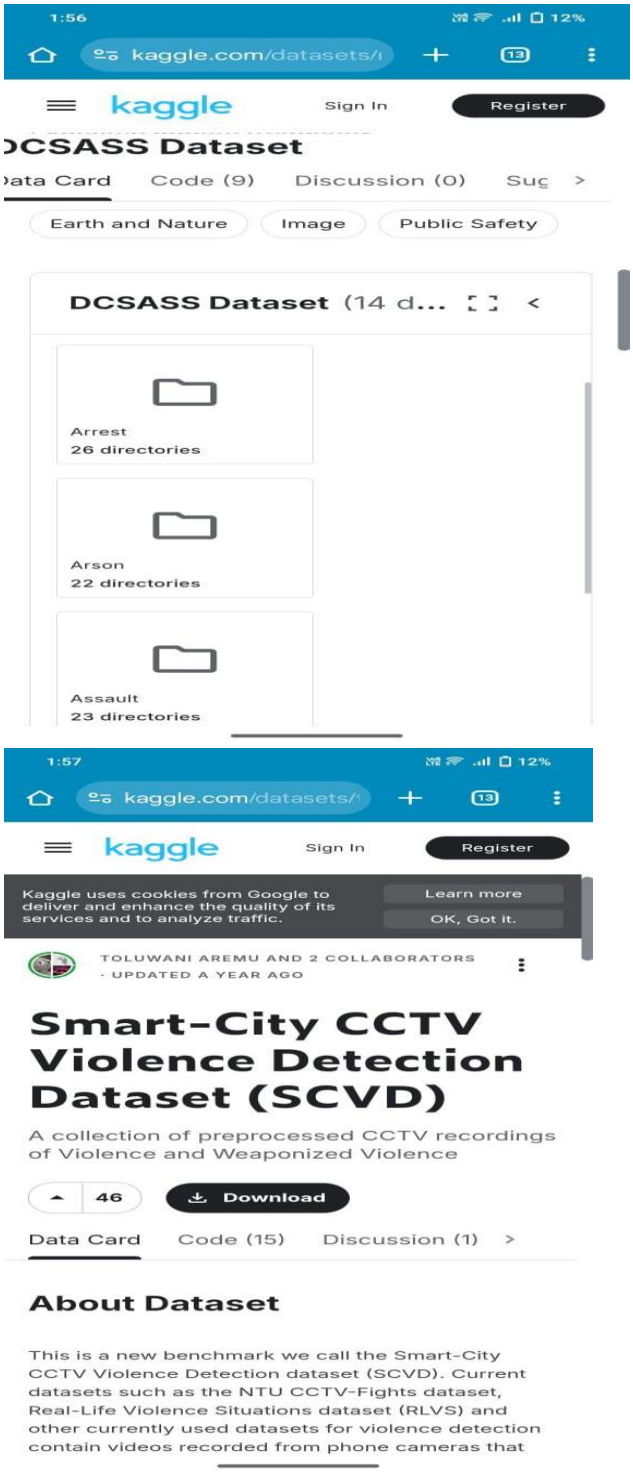
Software Requirements:

- Operating System: Windows, Linux, or MacOS
- Python 3.x version installed
- Jupyter Notebook or Google Colab for coding

- Required Python libraries:
    - Pandas (for data handling)
    - NumPy (for numerical operations)
    - Matplotlib (for visualization)
    - Seaborn (for advanced visualization)
    - Scikit-learn (for correlation analysis if needed)

Dataset:

- Kaggle: Student Social Media Addiction Dataset (CSV format)



· ARCHITECTURE

The architecture of this project is divided into the following phases:

1️⃣ Data Collection:
Downloading dataset from Kaggle.

2️⃣ Data Cleaning & Preprocessing:

- Handling missing values
- Converting data types
- Removing duplicates

3️⃣ Exploratory Data Analysis (EDA):

- Statistical summary
- Visualizations

4️⃣ Query-Based Analysis:

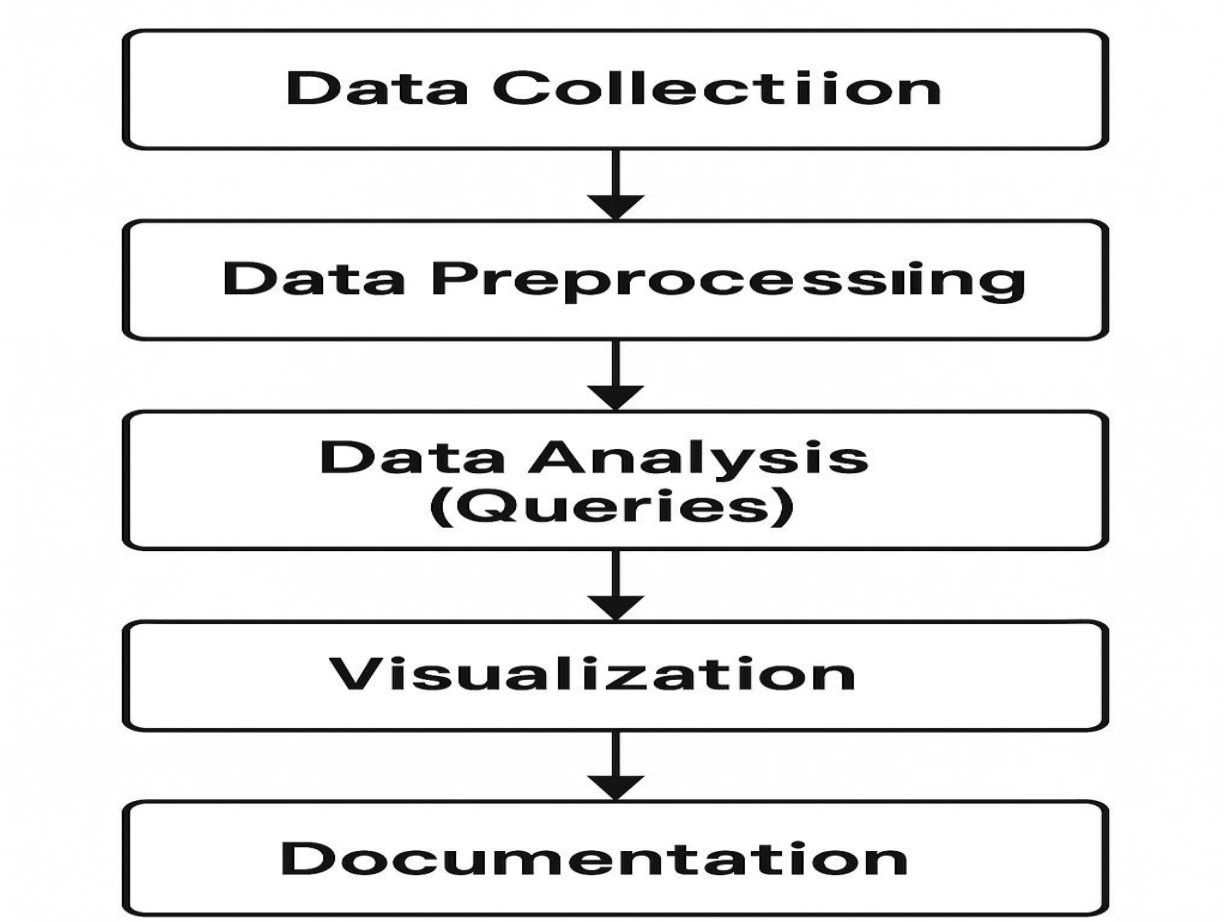
- Execute 20 queries to extract insights

5️⃣ Visualization & Reporting:

- Graphs, heatmaps, bar charts

6️⃣ Documentation & Conclusion:

- Create final report and presentation



· USES OF DATA ANALYSIS LIBRARIES

Pandas

- Used for data manipulation and data frame operations.
- Read CSV files, filter, group, merge, and summarize datasets.

NumPy

- Provides support for large multi-dimensional arrays and matrices.
- Useful for mathematical operations on numerical data.

Matplotlib

- Basic visualization library to create plots, bar charts, histograms, pie charts, etc.

Seaborn

- Built on Matplotlib.

- Used for advanced statistical visualization like heatmaps, correlation matrices, etc.

Scikit-learn (optional)

- Used for correlation calculation and machine learning models if extended.

DETAILED 20 QUERIES WITH EXPLANATIONS

Query 1: Basic Dataset Overview and Statistics

Explanation: Use df.describe() and df.info() in Python to view basic statistics and data structure.

Query 2: Peak Hours Analysis - When is the facility busiest?

Explanation: Use timestamps to group entries by hour and count - the hour with the highest count is the peak hour.

Query 3: Location-wise Security Analysis

Explanation: Group data by location and analyze event counts or types to identify high-risk areas.

Query 4: High-Risk Activity Detection

Explanation: Define risk rules or train a classifier to flag events showing restricted-area entry, unusual speed, or erratic movement as high-risk.

Query 5: Temporal Pattern Analysis - Day vs Night Activity?

Explanation: Separate data into day and night using timestamps, then compare event counts to identify temporal patterns.

Query 6: Crowding Events Analysis?

Explanation: Count people over time and flag instances where the count exceeds a threshold to detect crowding events.

Query 7: Motion Score Anomaly Detection?

Explanation: Use statistical methods (e.g., Z-score or IQR) to detect motion scores that deviate significantly from normal activity.

Query 8: Daily Activity Trends?

Explanation: Aggregate events by day and plot them to observe activity patterns and fluctuations over time.

Query 9: Alert Correlation Analysis?

Explanation: Analyze timestamps and event types to find patterns or relationships between multiple alerts occurring together.

Query 10: Loitering Incidents Analysis?

Explanation : Track individuals staying in one area beyond a time threshold to identify loitering behavior .

Query 11 : Running Activity Security Assessment?

Explanation: Detect and flag unusually fast or erratic movements using motion tracking and speed estimation algorithms.

Query 12 : Low-Occupancy High-Alert Events?

Explanation : Separate data by weekday and weekend, then compare activity levels and alert frequency to identify behavior differences.

Query 13: Low-Occupancy High-Alert Events?

Explanation: Filter events with low person count but high alert level to identify suspicious low-occupancy incidents.

Query 14: Activity Type Transition Patterns?

Explanation: Analyze sequences of activity types over time to detect common transitions, e.g., from walking to loitering or running.

Query 15: Motion Score vs People Count Relationship?

Explanation: Plot motion score against people count to observe correlation-higher counts often mean higher motion scores.

Query 16: Location Security Risk Ranking?

Explanation: Rank locations based on alert frequency, severity, and type to prioritize high-risk areas.

Query 17: Time-based Alert Clustering?

Explanation: Group alerts by time windows (e.g., every 15 minutes) to identify clusters and peak alert periods.

Query 18: Standing vs Moving Activity Analysis?

Explanation: Classify activities by motion and compare duration or frequency of standing vs moving events for behavioral insights.

Query 19: Temporal Alert Prediction Indicators?

Explanation: Use historical alert trends, time of day, and day of week as features to predict future alert timings.

Query 20: Comprehensive Security Dashboard Summary?

Explanation: Create a dashboard showing key metrics: alerts by type/time/location, risk rankings, activity trends, and real-time incident updates.

• ADVANTAGES

1. Real-time Monitoring

Enables live tracking of movements, helping in immediate action for suspicious activities.

2. Automated Alerts

The system can generate alerts based on motion patterns, reducing the need for 24/7 human observation.

3. Data-Driven Security Insights

By analyzing motion data, time logs, and object detection, it provides valuable insights for optimizing security operations.

4. Cost-Effective

Reduces human workload, cuts down manpower costs, and ensures better surveillance with less resource usage.

5. Activity Trend Analysis

Detects peak hours, common entry points, and frequency of movement — useful for future security planning.

6. Evidence Collection

Provides documented logs and visualizations that can serve as legal or organizational proof in case of incidents.

7. Customizable Alerts and Reports

You can program the system to detect specific events (e.g., after-hours movement, intrusion) and send custom reports.

8. Better Resource Allocation

Security staff can be positioned based on camera data insights — where incidents are frequent or during busy hours.

9. Scalability

Can be expanded to work with multiple cameras, locations, or integrated with IoT/smart systems.

10. Python Integration

Uses open-source tools (like Pandas, OpenCV, Matplotlib) — no licensing fees, and large community support.

· CONCLUSION

The "Intelligent Surveillance Camera" project represents a significant step toward the future of smart security. By harnessing the power of Python and data analytics, we've transformed ordinary surveillance systems into intelligent tools capable of understanding and responding to their environment.

This project goes beyond passive monitoring it actively analyzes movement patterns, detects unusual activity, and uncovers critical insights through structured data. Using Python libraries like Pandas, NumPy, Matplotlib, and Seaborn, we demonstrated how raw surveillance data can be converted into meaningful visualizations and decisions that improve real-world safety.

The ability to detect anomalies, reduce false alarms, and generate detailed reports not only boosts operational efficiency but also empowers security personnel to act proactively rather than reactively. Whether used in colleges, offices, traffic zones, or public places, such intelligent systems are the backbone of next-generation surveillance infrastructure.

As technology continues to evolve, the integration of real-time video analytics, face recognition, and AI-based alert systems could take this project to a new level. In essence, this work proves that data is not just numbers it's a powerful weapon against threats when placed in the right hands.

The future of security is not just smart it's intelligent. And this project is a glimpse into that future.

· REFERENCES

1. 1 **Kaggle Dataset:**
2. Student Social Media Addiction Dataset
3. *Source:* Kaggle.com
4. *Link:* https://www.kaggle.com
5. 2 **Python Libraries Used:**
6. **Pandas:** Data Manipulation and Analysis
7. **NumPy:** Numerical Computation
8. **Matplotlib:** Data Visualization
9. **Seaborn:** Statistical Visualization
10. **Scikit-learn:** Machine Learning & Correlation Analysis
11. 3 **Research Papers & Articles:**
12. Kuss, D. J., & Griffiths, M. D. (2015). Social networking sites and addiction: Ten lessons learned. *International Journal of Environmental Research and Public Health*, 12(3), 1287-1306.
13. Andreassen, C. S. (2015). Online Social Network Site Addiction: A Comprehensive Review. *Current Addiction Reports*, 2, 175–184.
14. WHO Mental Health Statistics and Reports (2023).
15. 4 **Official Documentation:**
16. Python Official Documentation: https://docs.python.org/3/
17. Pandas Documentation: https://pandas.pydata.org/docs/

18. Seaborn Documentation: https://seaborn.pydata.org/

19. 5 **APA Guidelines:**

20. American Psychological Association (APA), *Digital Media and Mental Health Guidelines*, 2023.