CHAPTER-1

COMPANY PROFILE

Company Name : EZ Trainings and Technologies Pvt. Ltd. Introduction: EZ Trainings and Technologies Pvt. Ltd. is a dynamic and innovative organization dedicated to providing comprehensive training solutions and expert development services. Established with a vision to bridge the gap between academic learning and industry requirements, we specialize in college trainings for students, focusing on preparing them for successful placements. Additionally, we excel in undertaking development projects, leveraging cutting-edge technologies to bring ideas to life.

Mission: Our mission is to empower the next generation of professionals by imparting relevant skills and knowledge through specialized training programs. We strive to be a catalyst in the career growth of students and contribute to the technological advancement of businesses through our development projects.

- **Services**:

College Trainings:

Tailored training programs designed to enhance the employability of students. • Industry-aligned curriculum covering technical and soft skills. • Placement assistance and career guidance.

Development Projects:

- End-to-end development services, from ideation to execution.

• Expertise in diverse technologies and frameworks.

• Custom solutions to meet specific business needs.

Locations:

Hyderabad | Delhi NCR At EZ Trainings and Technologies Pvt. Ltd., we believe in transforming potential into excellence

CHAPTER-2

DAY-TO-DAY ACTIVITIES.

SL.NO	DATES	TOPICS COVERED
1.	15-04-24	Introduction to Python, Setup & Installation, First Python Program, Variables, Data Types, and Basic I/O Control Structures: If-else
2.	16-04-24	Control Structures: Loops: for, while, pattern problems
3.	17-04-24	Introduction to OOP : Classes and Objects , Lists, Functions . File Handling
4.	18-04-24	Exception Handling, Inheritance, Practice exercises on Python basics
5.	19-04-24	Introduction to DSA, Arrays, and Linked Lists : Singly linked lists & doubly linked lists
6.	20-04-24	Stacks and Queues
7.	22-04-24	Searching and Sorting Algorithms : Binary search , Bubble sort, Selection sort, Insertion sort
8.	23-04-24	Sorting Algorithms: Quick sort & Merge sort
9.	24-04-24	Abstract Classes and Interfaces, Practice exercises on OOP concepts
10.	25-04-24	Trees : Pre-Order, In-order, Post-order, Level-order,Zig-Zag level-order traversals ,BST Tree construction,left-view and right-view of of BST
11.	26-04-24	Top-view, Bottom-view, Boundary traversal of BST, k-th smallest and k-th largest element in BST .
12.	27-04-24	Graphs : BFS and DFS in Graphs
13.	28-04-24	Project Building & Presentations
14.	29-04-24	Project Building & Presentations
15.	30-04-24	Project Building & Presentations
16.	02-05-24	Project Building & Presentations
17.	03-05-24	Project Building & Presentations
18.	04-05-24	Project Building & Presentations

CHAPTER-3

ABSTRACT

The incident report portal is a software system designed to facilitate the reporting, management, and tracking of incidents within an organization. It allows users to create new incident reports, read existing reports, update incident details, and delete reports as necessary. The portal maintains a database of incidents, each with unique identifiers and associated information such as date, person involved, and incident type. Its abstract aims to streamline incident management processes, ensuring timely response, documentation, and resolution of incidents while providing a centralized platform for record-keeping and analysis

Key Features of Incident Report Portal:

1. Incident Submission:

    Users can fill out a form to report an incident. This form typically includes fields such as title, description, severity, category, date, time, location, etc.

2. Incident Tracking:

    Submitted incidents are logged in a database, allowing administrators to track and manage them effectively. Each incident is assigned a unique identifier for reference.

3. Status Updates:

    Administrators can update the status of incidents as they are investigated, resolved, or closed. Users can also receive notifications about the status of their reported incidents.

4. Search and Filter:

    Users and administrators can search for incidents based on various criteria such as date, severity, category, status, etc. Filtering options help streamline the incident management process.

5. Analytics and Reporting:

The portal may include analytics features to generate reports and visualize incident trends over time. This helps organizations identify areas for improvement and take proactive measures to prevent future incidents.

Technology Stack:

Backend Framework: Python's Flask or Django frameworks are commonly used for building the backend of the portal. Flask is lightweight and suitable for smaller projects, while Django offers more built-in features and scalability.

Database: SQLite, PostgreSQL, or MySQL databases can be used to store incident data. SQLite is lightweight and ideal for development and small-scale applications, while PostgreSQL and MySQL are more robust options for larger deployments

Frontend: HTML, CSS, and JavaScript are used to create the user interface (UI) of the portal. Frontend frameworks like Bootstrap or libraries like React can be employed to enhance the UI and user experience.

APIs: RESTful APIs can be implemented to handle communication between the frontend and backend components of the portal. These APIs enable data exchange and interaction with the database.

Basic Structure:

Backend (Python with Flask): The Flask application handles HTTP requests from users, processes form submissions, interacts with the database, and serves data to the frontend. Routes are defined for different functionalities such as incident submission, retrieval, and status updates.

Frontend (HTML/CSS/JavaScript): HTML templates are used to render the UI components of the portal, including forms, tables, and navigation elements. CSS stylesheets are applied to customize the appearance of the portal, while JavaScript adds interactivity and dynamic behavior to the frontend.

Database (SQLite/PostgreSQL/MySQL): The database stores incident records, including details such as title, description, severity, status, etc. Tables are created to organize the data, and SQL queries are executed to perform CRUD (Create, Read, Update, Delete) operations on the database.

Integration and Deployment: Once the portal is developed, it can be integrated with existing systems and deployed to a web server or cloud platform for public access. Continuous monitoring and maintenance ensure the portal remains functional and secure.

CHAPTER-4

INCIDENT REPORT PORTAL

INTRODUCTION OF THE PROJECT

The Incident Report Portal serves as a comprehensive tool for organizations to effectively manage and address various incidents that may occur within their operations. Its introduction is pivotal in enhancing transparency, accountability, and efficiency in incident management processes.

In today's dynamic business landscape, incidents ranging from safety hazards to data breaches can significantly impact an organization's operations, reputation, and stakeholder trust. Hence, the need for a robust incident reporting system becomes imperative.

The Incident Report Portal offers a centralized platform where employees, stakeholders, and authorized personnel can easily report incidents, providing essential details such as date, person involved, and incident type. This centralized approach ensures that incidents are promptly documented and escalated to the appropriate authorities for action.

1. CRUD Operations on Incident Report Portal:

- CRUD (Create, Read, Update, Delete)keeps track from for reporting incident. It should include fields for relevant infromation such as incidents type data/time, location, Description , serverity and any other pertinent details.

2. Mapping Incident Report Portal:

This mapping provides a structured overview of the Incident Report Portal's architecture, functionalities, and integrations, enabling effective development and deployment of the system.

3. Analyzing Incident Report Portal :

Overall, the Incident Report Portal offers significant benefits in terms of centralized incident management, transparency, and data-driven decision-making. However, addressing challenges related to user adoption, data quality, and integration complexity, while continuously improving the portal's functionality and user experience, are essential for maximizing its effectiveness and value to the organization.

CHAPTER-5

MODULE DESCRIPTION

The Incident Report Portal is a software application designed to facilitate the reporting, management, and tracking of incidents within an organization

1. **CRUD Operations for Incident Report Portal Data**: This module handles the basic CRUD (Create, Read, Update, Delete) operations for crime data. It allows users to input, retrieve, update, and delete records of crime incidents stored in a database.
2. **map_Incident Report Portal_locations(crime_id)**: This function is responsible for plotting crime incidents on a simulated map. Given a specific crime ID as input, it retrieves the relevant data from the database and plots the corresponding crime location(s) on a map. This feature helps visualize the spatial distribution of crime incidents.
3. **analyze_Incident Report Portal_patterns(pattern_id)**: This module is designed to analyze crime data in order to identify patterns and trends. By specifying a pattern ID as input, the function retrieves relevant crime data associated with that pattern from the database and performs analysis using Python. This analysis may include identifying recurring patterns, trends over time, correlations between different types of crimes, etc.

Overall, the incident report portal plays a critical role in promoting safety, compliance, and accountability within organizations by providing a centralized platform for incident management and reporting.

CHAPTER-6

ALGORITHM

- Define a class IncidentPortal with methods to create, read, update, and delete incidents.
- The create_incident method adds a new incident to the portal's list of incidents.
- The read_incident_by_report_id method retrieves and displays details of an incident based on the report ID.
- The update_incident method allows updating specific fields of an incident based on the report ID.
- The delete_incident method removes an incident from the portal based on the report ID.

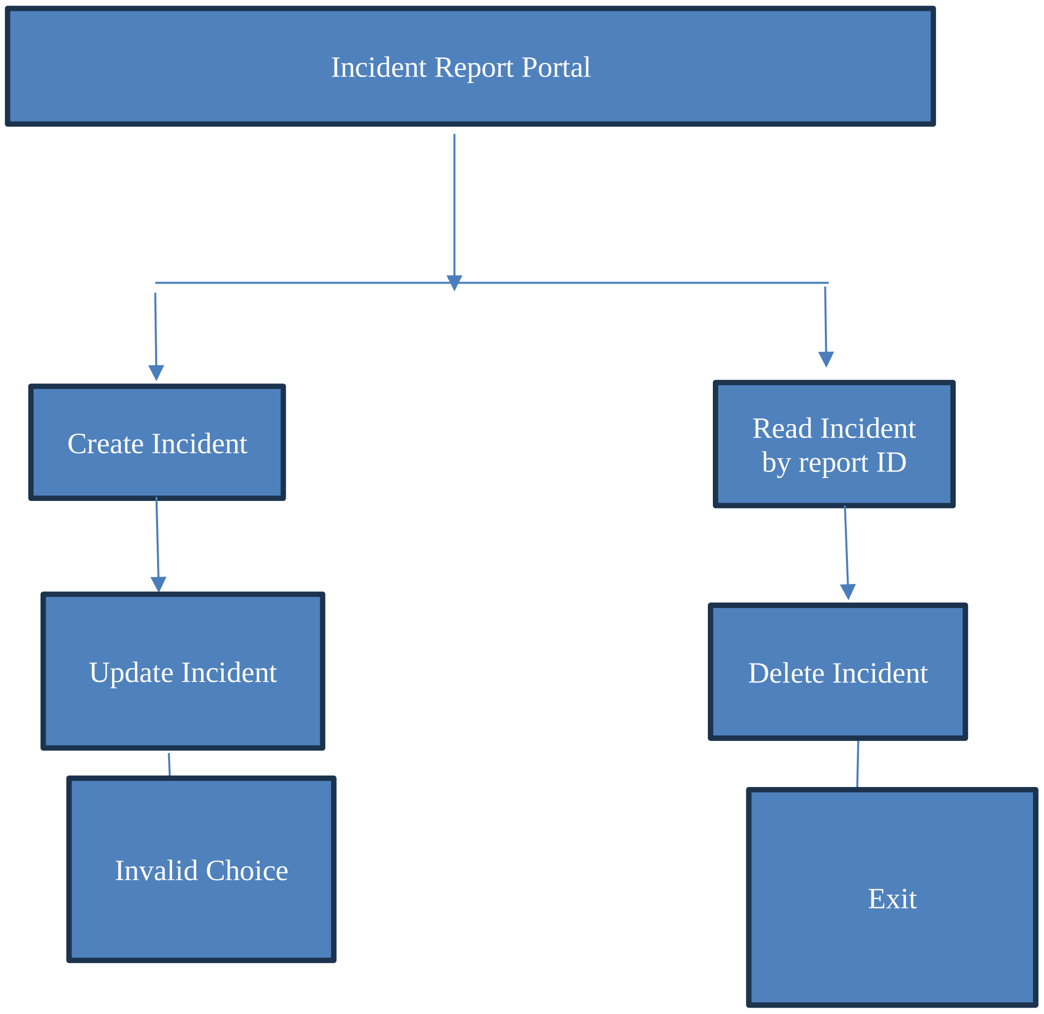
Define main function:

1. Create Incident: Allows the user to input details of a new incident, such as date, person ID, and incident type. This incident is then added to the list of incidents.
2. Read Incident by Report ID: Retrieves and displays the details of an incident based on its report ID.
3. Update Incident: Enables the user to update specific fields of an incident based on its report ID. The user can choose which field to update (date, person ID, or incident type) and provide a new value for that field.
4. Delete Incident: Deletes an incident from the list based on its report ID.
5. Exit:Terminates the program

Source Code:

```
class IncidentPortal:
    def _init_(self):
        self.incidents = []

    def create_incident(self, date, person_id, incident_type):
        report_id = len(self.incidents) + 1  # Assign a unique report ID
        incident = {
            'report_id': report_id,
            'date': date,
            'person_id': person_id,
            'incident_type': incident_type
        }
        self.incidents.append(incident)
        print(f"Incident created with report ID: {report_id}")

    def read_incident_by_report_id(self, report_id):
        for incident in self.incidents:
            if incident['report_id'] == report_id:
                print("Incident details:")
                for key, value in incident.items():
                    print(f"{key}: {value}")
                return
        print("No incident found with the given report ID.")

    def update_incident(self, report_id, **kwargs):
        for incident in self.incidents:
            if incident['report_id'] == report_id:
                for key, value in kwargs.items():
                    if key in incident:
                        incident[key] = value
                print("Incident updated successfully.")
                return
        print("No incident found with the given report ID.")

    def delete_incident(self, report_id):
        for incident in self.incidents:
            if incident['report_id'] == report_id:
                self.incidents.remove(incident)
                print("Incident deleted successfully.")
                return
        print("No incident found with the given report ID.")

def main():
    portal = IncidentPortal()

    while True:
        print("\nIncident Reporting Portal")
        print("1. Create Incident")
        print("2. Read Incident by Report ID")
        print("3. Update Incident")
        print("4. Delete Incident")
        print("5. Exit")

        choice = input("Enter your choice: ")

        if choice == '1':
            date = input("Enter date of the incident: ")
            person_id = input("Enter ID of the person: ")
            incident_type = input("Enter type of the incident: ")
            portal.create_incident(date, person_id, incident_type)

        elif choice == '2':
            report_id = int(input("Enter report ID to read incident: "))
            portal.read_incident_by_report_id(report_id)

        elif choice == '3':
            report_id = int(input("Enter report ID of the incident to update: "))
            updates = {}
            field = input("Enter field to update (date/person_id/incident_type): ")
            if field:
                value = input(f"Enter new value for {field}: ")
                updates[field] = value
            portal.update_incident(report_id, **updates)

        elif choice == '4':
            report_id = int(input("Enter report ID of the incident to delete: "))
            portal.delete_incident(report_id)

        elif choice == '5':
            print("Exiting program.")
            break

        else:
            print("Invalid choice. Please enter a valid option.")

if _name_ == "_main_":
    main()
```

Flow Chart:


Incident Report Portal
Create Incident
Read Incident by report ID
Update Incident
Delete Incident
Invalid Choice
Exit

CHAPTER-7:

Result:

```
Incident Reporting Portal
1. Create Incident
2. Read Incident by Report ID
3. Update Incident
4. Delete Incident
5. Exit
Enter your choice: 1
Enter date of the incident: 1/2/2023
Enter ID of the person: 234
Enter type of the incident: theft
Incident created with report ID: 1

Incident Reporting Portal
1. Create Incident
2. Read Incident by Report ID
3. Update Incident
4. Delete Incident
5. Exit
Enter your choice: 2
Enter report ID to read incident: 1
Incident details:
report_id: 1
date: 1/2/2023
person_id: 234
incident_type: theft

Incident Reporting Portal
1. Create Incident
2. Read Incident by Report ID
3. Update Incident
4. Delete Incident
5. Exit
Enter your choice: 3
Enter report ID of the incident to update: 1
Enter field to update (date/person_id/incident_type): 1
Enter new value for 1: 3/4/2023
Incident updated successfully.

Incident Reporting Portal
1. Create Incident
2. Read Incident by Report ID
3. Update Incident
4. Delete Incident
5. Exit
Enter your choice: 4
Enter report ID of the incident to delete: 1
```

```
Incident deleted successfully.

Incident Reporting Portal
1. Create Incident
2. Read Incident by Report ID
3. Update Incident
4. Delete Incident
5. Exit
Enter your choice: 2
Enter report ID to read incident: 1
No incident found with the given report ID.

Incident Reporting Portal
1. Create Incident
2. Read Incident by Report ID
3. Update Incident
4. Delete Incident
5. Exit
Enter your choice: 5
Exiting program
```

CHAPTER-8

CONCLUSION

An incident reporting portal is a valuable tool for organizations to streamline the process of reporting, tracking, and managing incidents effectively. By leveraging Python and its ecosystem of libraries and frameworks, developers can create robust and scalable portals tailored to the specific needs of their organization. Whether it's addressing IT issues, ensuring workplace safety, or maintaining security standards, an incident reporting portal provides a centralized platform for incident management and resolution.

In conclusion, an incident report portal is a vital tool for organizations to effectively manage and document incidents that occur within their operations. By providing a centralized platform, it facilitates the reporting, tracking, and resolution of incidents, thereby enhancing safety, compliance, and overall operational efficiency.

CHAPTER-9

Reference:

https://chat.openai.com/c/0407d836-6538-46b3-93c3-d4ab1b5d0fc0

https://www.w3schools.in/python/file-handling

https://www.javatpoint.com/python-dictionary

https://www.geeksforgeeks.org/python-classes-and-objects/