PES POLYTECHNIC-563

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

TOPIC :

DATA LEAKAGE DETETCTION PROCESS”

SUBMITTED BY:-

BHUVAN S
HEMANTH HD
MOHAMMED SAAD
SAGAR KG

UNDER THE GUIDANCE OF

Ms. ARCHANA P
Ms. DAKSHAYANI H V

CONTENTS

INTRODUCTION

1. Project Overview:

In today's digital world, keeping our sensitive information safe is more crucial than ever. That's why we're embarking on the "Detecting Data Leaks Using SQL Injections" project. By harnessing the power of cloud technology, we're building a strong defense against cyber threats. Our goal? To create a system that can detect and stop SQL injection attacks in their tracks, making online data safer for everyone.

2. Objective:

Our main aim is simple: to build a smart system that can spot and block SQL injection attacks in cloud-based databases before they cause any harm. By using clever algorithms and the flexibility of cloud computing, we're making it harder for cybercriminals to breach our data security. Ultimately, we want to keep sensitive information out of the wrong hands.

3. Importance:

Why does this project matter? Because data breaches caused by SQL injections can have serious consequences. They can cost companies money, damage their reputation, and even land them in legal trouble. By tackling this vulnerability head-on with cloud-based solutions, we're making the online world safer for businesses, governments, and individuals.

4. Key Components:

So, what's making our project tick? Here are the main ingredients:

Cloud power: We're using the scalability and reliability of cloud platforms to make our detection system fast and efficient.
Smart algorithms: Our system is powered by advanced algorithms that analyze database queries, spotting any signs of suspicious activity.
Always watching: We've set up continuous monitoring to keep an eye on things in real-time, so we can catch any potential threats before they become a problem.
Quick fixes: We're also working on automated tools that can swoop in and fix any vulnerabilities, keeping our databases safe and sound.

5. Outcome:

What do we hope to achieve? Plenty:

Safer data: By staying one step ahead of SQL injection attacks, we're ensuring that our online data stays out of harm's way.
Trustworthy systems: By sticking to the rules and regulations, we show everyone we take data privacy seriously, building trust in our online platforms.
Smarter spending: By using cloud resources wisely and automating where we can, we're making sure that our defenses are strong without breaking the bank.

Problem Statement

In today's interconnected world, data security is paramount. We're on a mission to develop an application that not only detects data leaks but also helps identify the culprit behind them. Our solution leverages Probability Distribution to pinpoint potential breaches and safeguard sensitive information.

Our goal is clear: to create a robust solution for detecting and mitigating SQL injection vulnerabilities. By doing so, we aim to protect sensitive data from unauthorized access and potential leakage. With our application, we're empowering organizations to take proactive measures against cyber threats.

PROBLEM STATEMENT : To build an application that helps in Detecting the data that has been leaked. It also helps in finding the Guilt of an Agent from the given set of agents that has leaked the data using Probability Distribution.

PROBLEM STATEMENT

Our application employs advanced techniques to detect data leaks and identify the responsible agent. Using Probability Distribution, we analyze patterns to determine the likelihood of a breach. By integrating comprehensive monitoring and analysis capabilities, we ensure timely detection and response to potential threats.

Benefits:

Enhanced Data Security: Our solution provides organizations with the tools they need to safeguard their sensitive information effectively.

Proactive Threat Detection: By leveraging Probability Distribution, we enable early detection of potential breaches, minimizing the impact of data leaks.

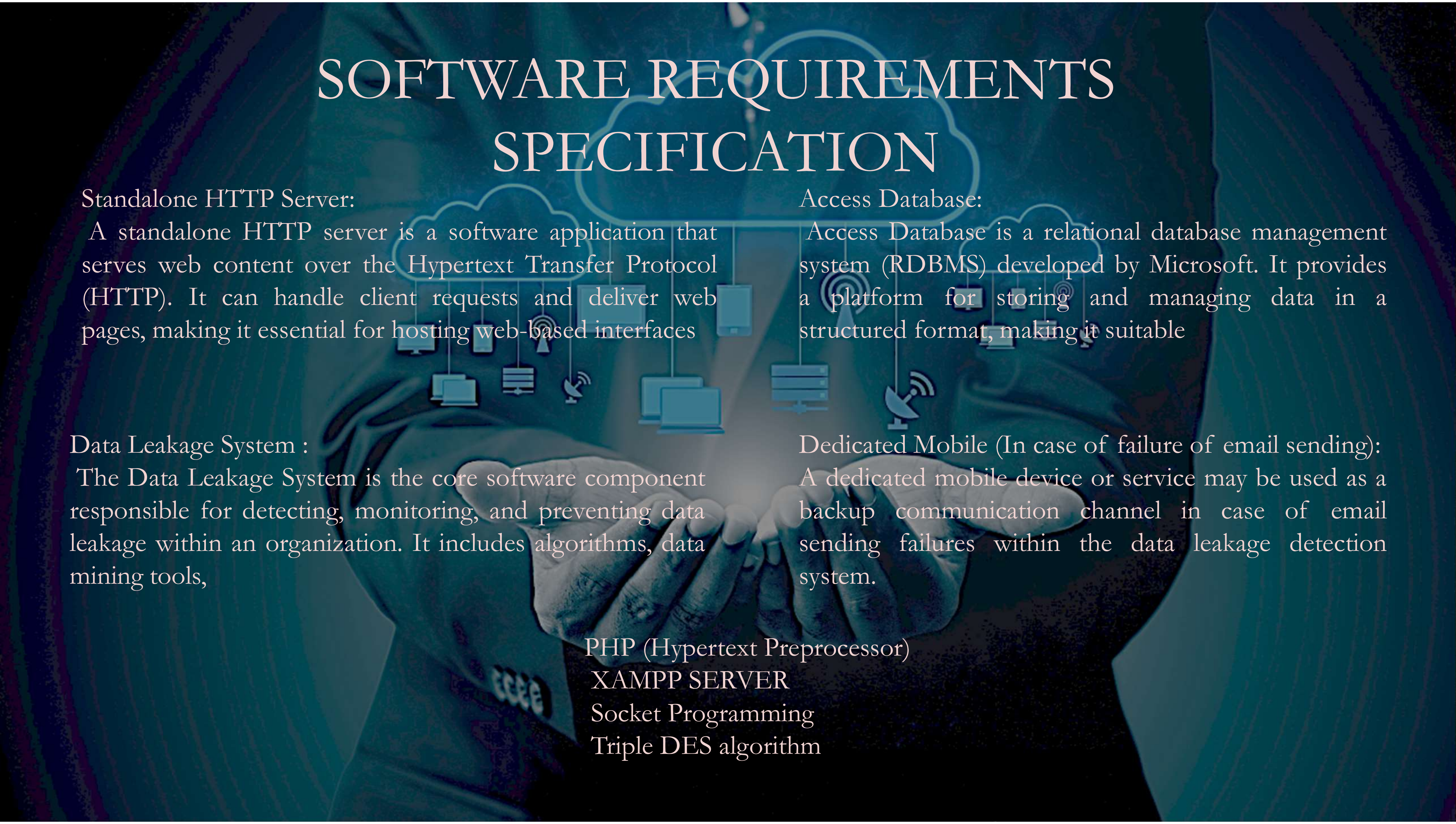
Accountability: With the ability to identify the agent responsible for a leak, our application promotes accountability and strengthens internal security measures.

OBJECTIVES

The objective of this project is to create a system that can effectively detect instances where sensitive data has been leaked by third-party agents. This system aims to analyze the situation when the distributor discovers their data in unauthorized places, such as on a website or obtained through legal processes

By doing so, it seeks to determine the likelihood that the leaked data originated from one or more agents rather than being gathered independently. The ultimate goal is to provide the distributor with the means to assess the evidence and, if necessary, take appropriate action, such as terminating business relationships or initiating legal proceedings against the responsible agent.

THE MAIN OBJECTIVE IS TO
Develop a system capable of detecting instances where sensitive data has been leaked by third-party agents.
Analyze situations where the distributor discovers their data in unauthorized places, such as on a website or through legal processes.
Determine the likelihood that leaked data originated from one or more agents rather than being gathered independently.
Provide the distributor with the means to assess the evidence and take appropriate action, such as terminating business relationships or initiating legal proceedings against the responsible agent.



SOFTWARE REQUIREMENTS SPECIFICATION

Standalone HTTP Server:
A standalone HTTP server is a software application that serves web content over the Hypertext Transfer Protocol (HTTP). It can handle client requests and deliver web pages, making it essential for hosting web-based interfaces

Access Database:
Access Database is a relational database management system (RDBMS) developed by Microsoft. It provides a platform for storing and managing data in a structured format, making it suitable

Data Leakage System :
The Data Leakage System is the core software component responsible for detecting, monitoring, and preventing data leakage within an organization. It includes algorithms, data mining tools,

Dedicated Mobile (In case of failure of email sending):
A dedicated mobile device or service may be used as a backup communication channel in case of email sending failures within the data leakage detection system.

PHP (Hypertext Preprocessor)
XAMPP SERVER
Socket Programming
Triple DES algorithm

Entity relationship diagram

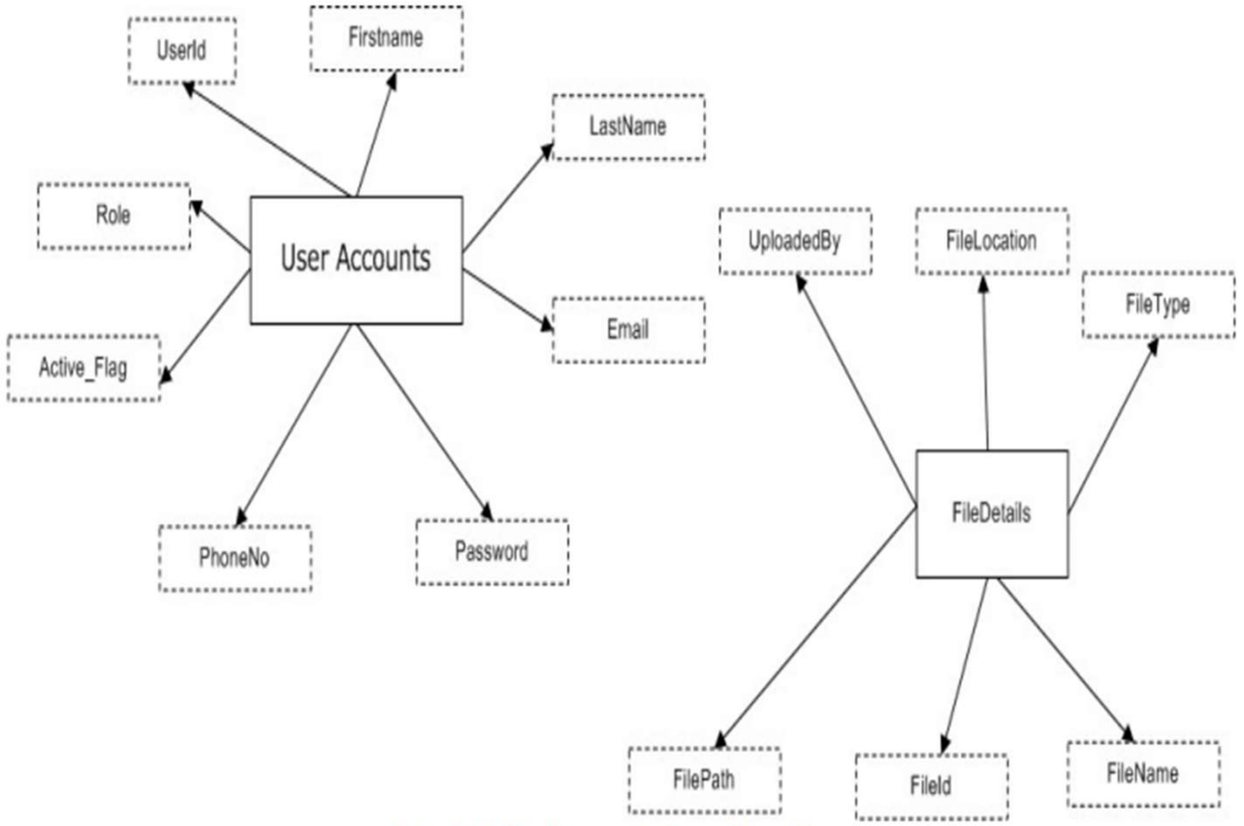
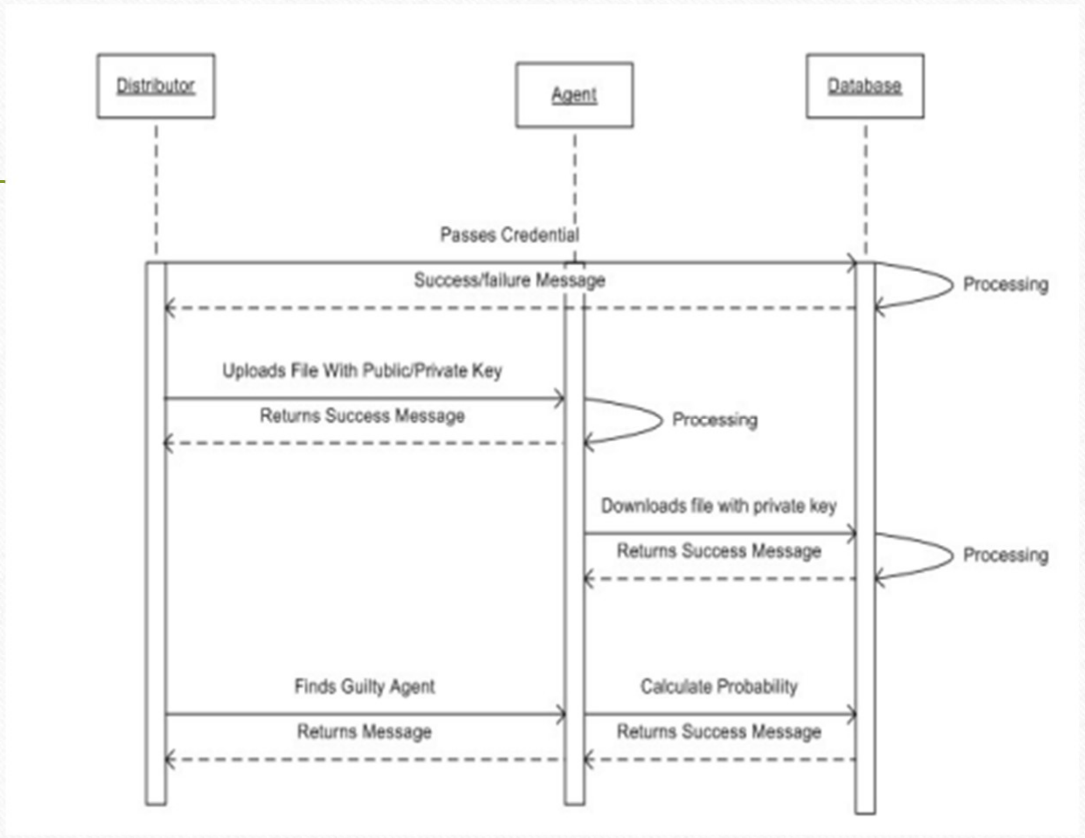
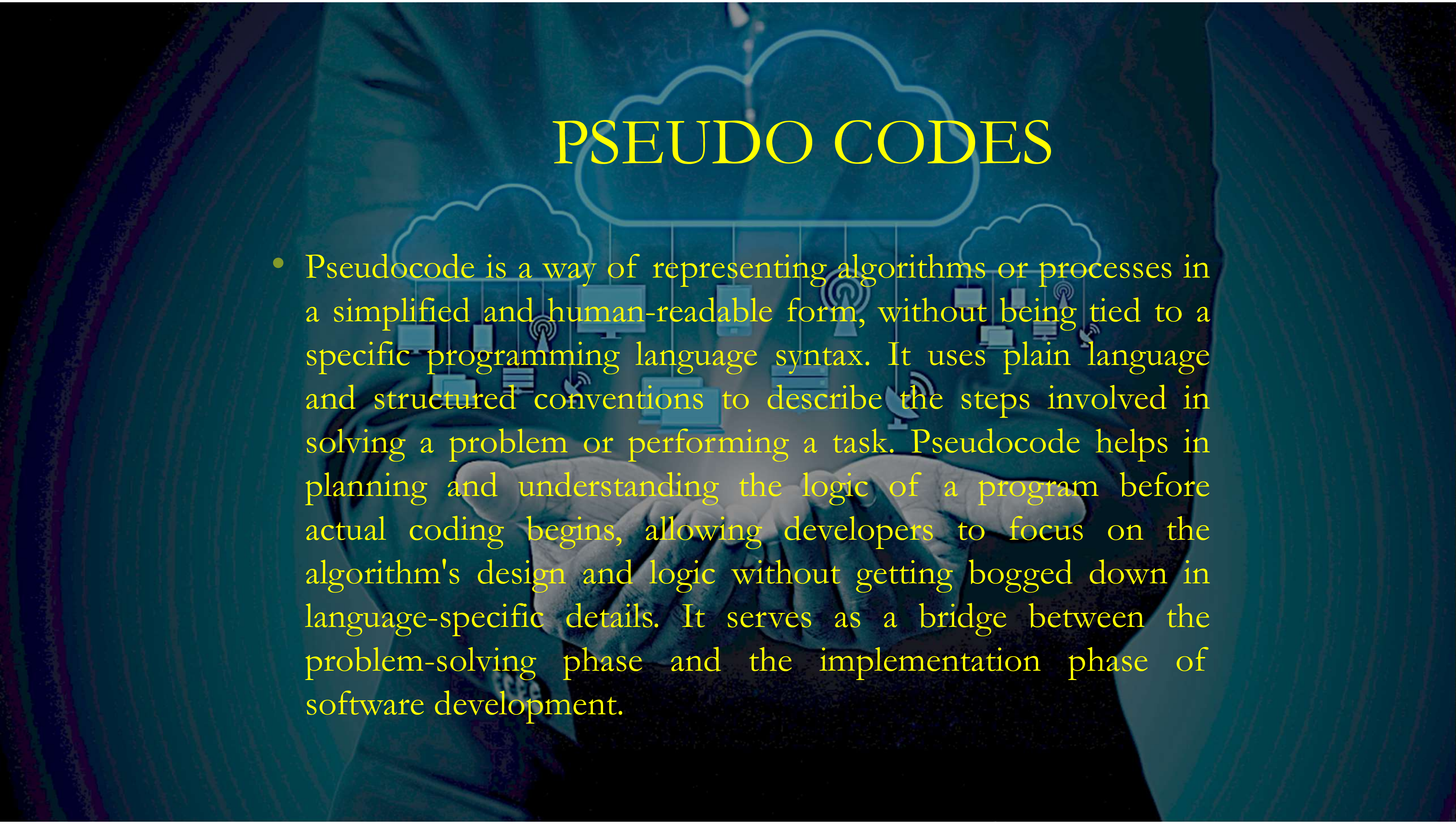


Fig 3.7 Entity relationship diagram

Schematic diagram





PSEUDO CODES

- Pseudocode is a way of representing algorithms or processes in a simplified and human-readable form, without being tied to a specific programming language syntax. It uses plain language and structured conventions to describe the steps involved in solving a problem or performing a task. Pseudocode helps in planning and understanding the logic of a program before actual coding begins, allowing developers to focus on the algorithm's design and logic without getting bogged down in language-specific details. It serves as a bridge between the problem-solving phase and the implementation phase of software development.

```
Function
 detectDataLeak(distributorData, leakedData):
    for each agentData in leakedData:
        similarityScore = calculateSimilarity(distributorData,
agentData)
        if similarityScore > threshold:
            return true, agentData
    return false, null

Function
 calculateSimilarity(distributorData, agentData):
    // Use a similarity calculation method (e.g., cosine
similarity)
    // to determine the similarity between distributorData
and agentData.
    // Return the similarity score.
```

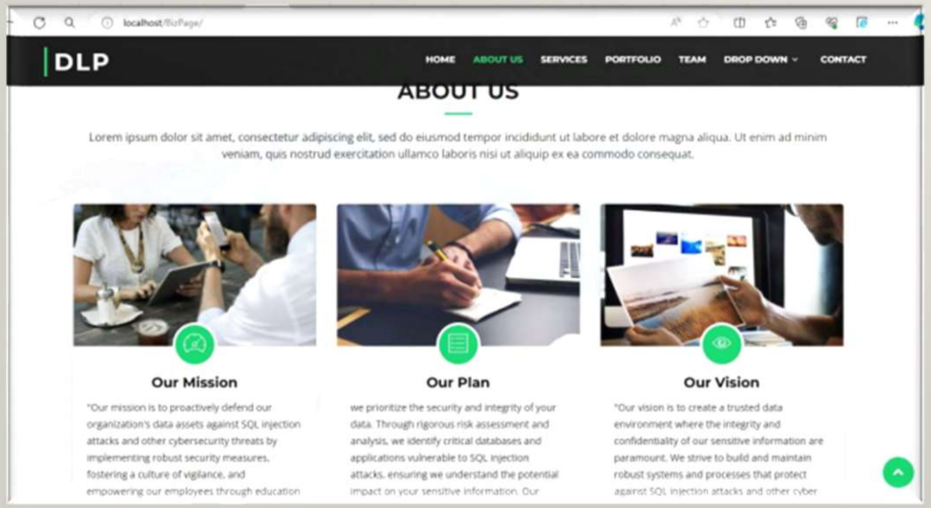
In this pseudocode:
Detect DataLeak is the main function that takes the distributor's data (distributorData) and the leaked data (leakedData) as input.
It iterates through each set of agent data in the leaked data.

For each agent data, it calculates a similarity score between the distributor's data and the agent's data using the calculateSimilarity function.

If the similarity score exceeds a predefined threshold, it indicates a potential data leak, and the function returns true along with the agent's data.

If no potential data leaks are found, the function returns false and null. The calculateSimilarity function calculates the similarity between two sets of data. This could involve using various similarity metrics such as cosine similarity or Jaccard similarity, depending on the nature of the data.

RESULTS(SnapShots)



ABOUT US

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Our Mission

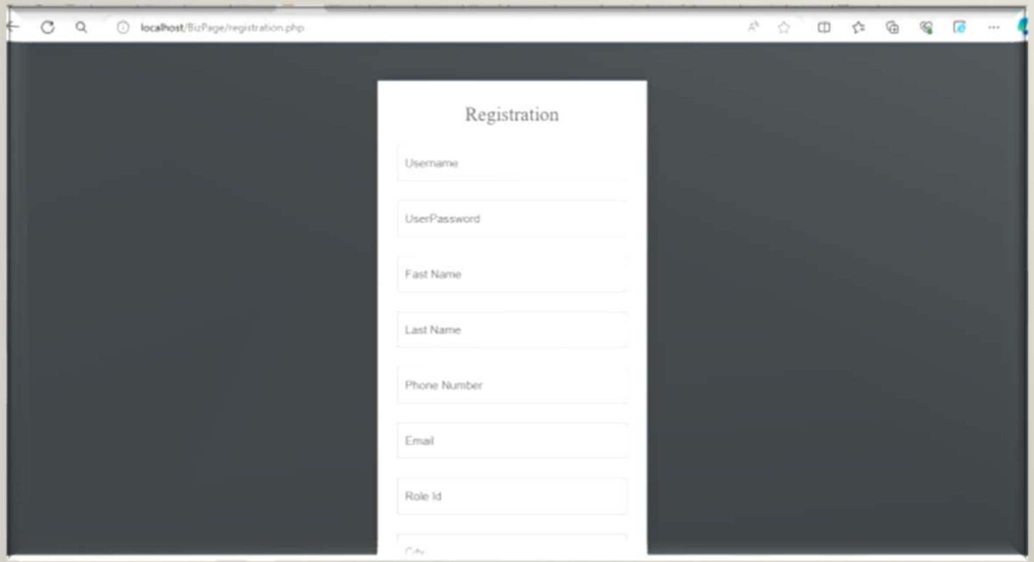
"Our mission is to proactively defend our organization's data assets against SQL injection attacks and other cybersecurity threats by implementing robust security measures, fostering a culture of vigilance, and empowering our employees through education

Our Plan

we prioritize the security and integrity of your data. Through rigorous risk assessment and analysis, we identify critical databases and applications vulnerable to SQL injection attacks, ensuring we understand the potential impact on your sensitive information. Our

Our Vision

"Our vision is to create a trusted data environment where the integrity and confidentiality of our sensitive information are paramount. We strive to build and maintain robust systems and processes that protect against SQL injection attacks and other cyber



Registration

Username

UserPassword

Fast Name

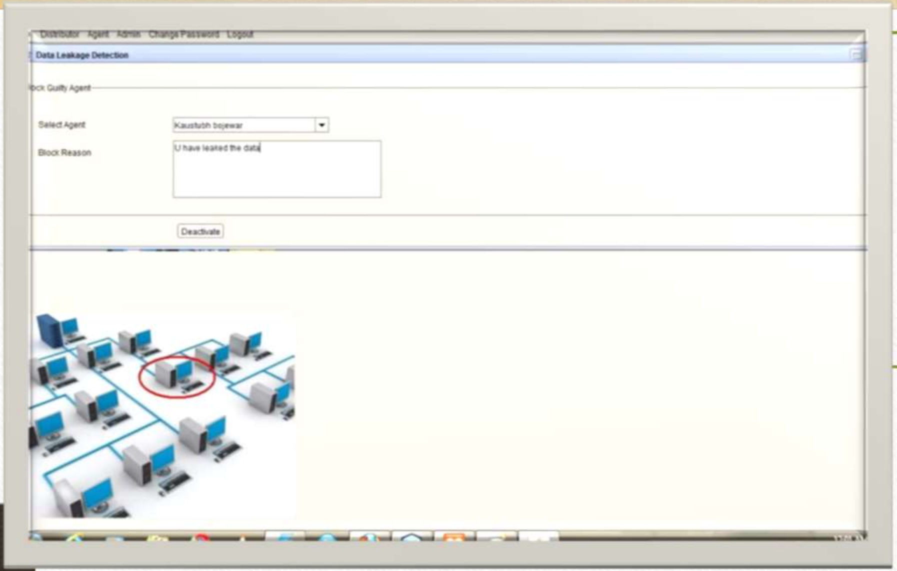
Last Name

Phone Number

Email

Role Id

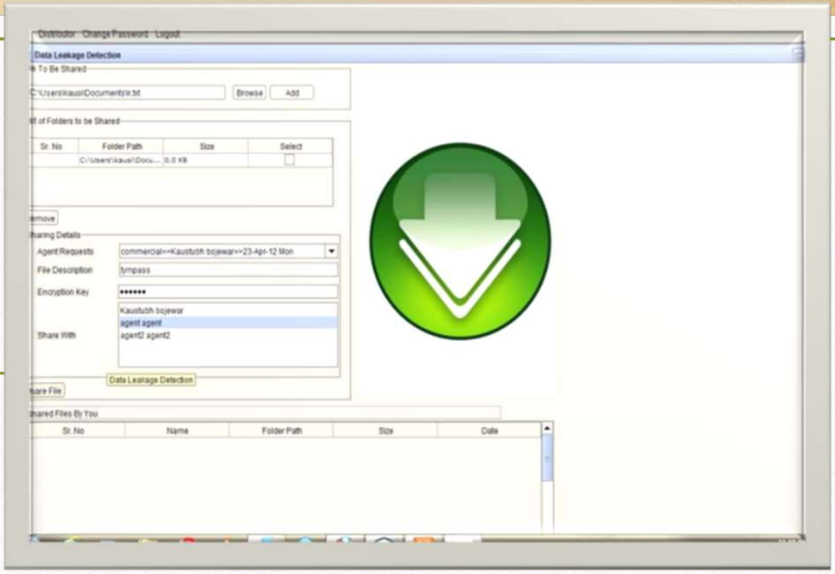
City



ock Guilty Agent

Select Agent: Kaustubh bojewar

Block Reason: U have leaked the data



To Be Shared

C:\Users\kaus\Documents\k.txt

t of Folders to be Shared

Sr. No	Folder Path	Size	Select
	C:\Users\kaus\Docu...	0.0 KB	☐

haring Details

Agent Requests: commercial<->Kaustubh bojewar<->23-Apr-12 Mon

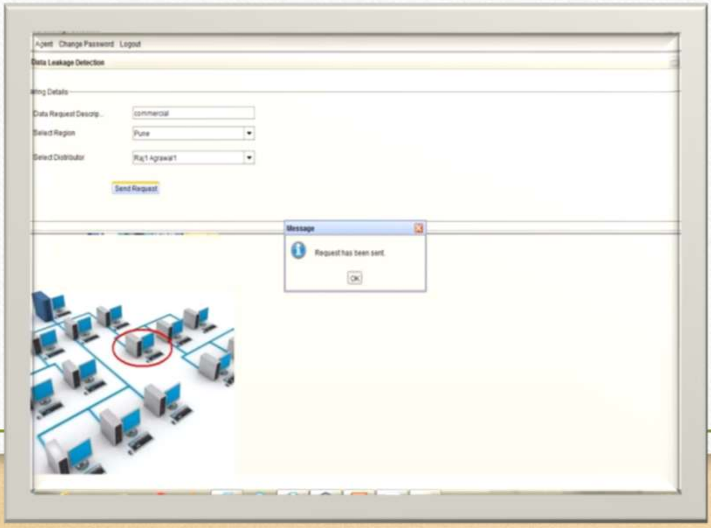
File Description: tympass

Encryption Key: ••••••

Share With: Kaustubh bojewar, agent agent, agent2 agent2

hared Files By You

Sr. No	Name	Folder Path	Size	Date



ting Details

Data Request Descrip...: commercial

Select Region: Pune

Select Distributor: Raj1 Agrawal1

Request has been sent.

CONCLUSION

- In a perfect world, there would be no need to hand over sensitive data to agents that may unknowingly or maliciously leak it. And even if we had to hand over sensitive data, in a perfect world we could watermark each object to trace its origins with absolute certainty.
- However, in many cases, we must indeed work with agents that may not be 100% trusted, and we may not be certain if a leaked object came from an agent or some other source, since certain data cannot admit watermarks.
- Despite these difficulties, we have shown it is possible to assess the likelihood that an agent is responsible for a leak, based on the overlap of his data with the leaked data and the data of other agents, and based on the probability that objects can be "guessed" by other means.
- Our model is relatively simple, but we believe it captures the essential trade-offs. The algorithms we have presented implement a variety of data distribution strategies that can improve the distributor's chances of identifying a leaker.
- We have shown that distributing objects judiciously can make a significant difference in identifying guilty agents, especially in cases where there is a large overlap in the data that agents must receive.

REFERENCES

- "Data Leakage Detection" Panagiotis Papadimitriou, Student Member, IEEE, and Hector Garcia-Molina, Member, IEEE
- R. Agrawal and J. Kiernan, "Watermarking Relational Databases," Proc. 28th Int'l Conf. Very Large Data Bases (VLDB '02), VLDB Endowment, pp. 155-166, 2002.
- [3] P. Bonatti, S.D.C. di Vimercati, and P. Samarati, "An Algebra for Composing Access Control Policies," ACM Trans. Information and System Security, vol. 5, no. 1, pp. 1-35, 2002.
- P. Buneman and W.-C. Tan, "Provenance in Databases," Proc. ACM SIGMOD, pp. 1171-1173, 2007.
- Y. Cui and J. Widom, "Lineage Tracing for General Data Warehouse Transformations," The VLDB J., vol. 12, pp. 41-58, 2003.
- V.N. Murty, "Counting the Integer Solutions of a Linear Equation with Unit Coefficients," Math. Magazine, vol. 54, no. 2, pp. 79-81,1981.
- A programming guide to Java certification
- Cryptography and Network Security by Williams Stallings

thank you