A
Project Based learning
IT Infrastructure Management
On

"Develop Infrastructure Management System(Storage And Security Management)"

Submitted to

INFORMATION TECHNOLOGY, BHARATI VIDYAPEETH
(DEEMED TO BE UNIVERSITY)
COLLEGE OF ENGINEERING , PUNE

In partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN
INFORMATION TECHNOLOGY

Submitted By

Roll No.	PRN No.	Seat No.	Name
36.	**2214110270**	**2422930549**	**RIYA SINGH**
54.	**2214110288**	**2422930563**	**BHUMI**

Under the Guidance Of

Prof. TB Patil

DEPARTMENT OF INFORMATION TECHNOLOGY

BHARATI VIDYAPEETH (DEEMED TO BE UNIVERSITY)

COLLEGE OF ENGINEERING, PUNE-411043

(Academic Year 2023-2024)

Certificate

This is to certify that the work under Synopsis for the topic **"Develop a Infrastructure Management System(Storage and Security Management)"** is carried outby "**Riya Singh, Bhumi"** under my guidance in partial fulfillment of the requirement for the degree of "**Bachelor of Technology in Information Technology Semester-IV**"of **"Bharati Vidyapeeth (Deemed to be University), Pune"** during the academic year **2023-2024**.To the best of my knowledge and belief this work has not been submitted elsewhere forthe award of any other degree.

Date: -

Prof. T.B Patil
PBL GUIDE

Prof. S.B Vanjale
HOD

ACKNOWLEDGEMENT

It was a pleasure doing the project work which helped us discover new things and explore research related areas. This became possible only due to experienced guidance and the inspiration that helpedus to complete this work. The successful completion of this project could not have been possible without the help and guidance of many people.

We express our grateful and sincere appreciation and deep gratitude to our PBL Guide Prof. T.B. Patil for his valuable guidance, constant encouragement, and contribution in completion of this project and for giving us an opportunity to prepare on this interesting topic.

We are grateful to our Head of Department of Information Technology, Prof. Dr.S.B. Vanjale for supporting us and motivating us for the work which has been carried out here.

We would also like to express gratitude to our beloved Principal, Prof. Dr. Vidula Sohoni Madam for providing access to the required resources and the Internet for collecting information for this report.

Last but not the least this acknowledgement would be incomplete without rendering our sincere gratitude for blessings of our parent and friends who have always been a source of inspiration & motivation, without which this project work would not have been possible.

We proudly thank Almighty God for his blessings!

Riya Singh

Bhumi

B.Tech. (IT) Semester - IV

INDEX

INTRODUCTION

IT infrastructure management refers to the coordination and oversight of the components that make up an organization's information technology (IT) infrastructure. This includes hardware, software, networks, data centers, cloud services, and other resources necessary for the operation of IT systems. The primary goal of IT infrastructure management is to ensure that these components are effectively utilized, maintained, and secured to support the organization's business objectives.

It is a method of monitoring and maintaining critical Information Technology (IT) infrastructure to ensure the best use of resources, protect against data loss, and monitor key aspects of local and cloud-based service utilization.

Storage Management:

- This involves managing the physical and virtual storage devices where data is stored.
- This involves allocating storage resources to users or applications based on their needs.
- In our project we have managed to store the data(messages) into a database once user enters the message with its sender and receiver.

Security Management:

- Managing user access to resources to ensure that only authorized individuals or systems can access sensitive data or perform specific actions.
- Protecting data both at rest and in transit by encrypting it using encryption algorithms and secure protocols. This helps prevent unauthorized access and data breaches.
- In our project we are storing the message after encrypting it with a specific algorithm and it will display the message after decrypting it.

MYSQL-CONNECTOR-PYTHON

Code:

```
import mysql.connector as sqltor
import pandas as pd
import random
import sys
from email.utils import parseaddr

global inp
inp = 'y'
global user
user = 'a'
mycon = sqltor.connect(host="localhost", user="root",
password="riya@avanish2007", database="encryption_final")
if mycon.is_connected() == False:
    print("error in connecting to mysql")
else:
    print("successfully connected to mysql")
print()
cur = mycon.cursor()
```

Explanation:

- We have used mysql-connector module of the python to connect our python code to the database in SQL.
- Random() module is also imported as we need to insert the message randomly in the tables , hence it generates random numbers.
- Cursor is object used to retrieve , execute the sql queries and modify the data in the database.

GENERATE_KEY() FUNCTION

Code:

```
def generate_key():
    shift = 3
    alphabet = 'abcdefghijklmnopqrstuvwxyz'
    shifted_alphabet = alphabet[shift:] + alphabet[:shift]
    return shifted_alphabet
```

Explanation:

- This code generate a substring that is passed to the encrypt function and returns the shifted alphabet.

- alphabet is a string used to store string ‘abcdefghijklmnopqrstuvwxyz’

- alphabet[shift:]=This takes a substring of the alphabet starting from the index shift.

- alphabet[:shift]= This takes a substring of the alphabet up to, but not including, the index shift.

- In our project we have taken shift i.e index as 3 so our shifted string becomes 'defghijklmnopqrstuvwxyzabc'.

ENCRYPT_MESSAGE(MESSAGE)

Code:

```
def encrypt_message(message):
    shift = 3
    encrypted_message = ""
    for char in message:
        if char.isalpha():
            shifted_char = chr(((ord(char) - ord('a' if char.islower()
else 'A') + shift) % 26) + ord('a' if char.islower() else 'A'))
            encrypted_message += shifted_char
        else:
            encrypted_message += char
    return encrypted_message
```

Explanation:

- This function takes the message as a parameter and encrypts the message according to the algorithm, further returns the encrypted message.

- *for char in message*: This loop iterates to all the characters in the string message.

- ***if char-isalpha()****:* It checks if the character is alphabet. If so than ***ord(char) - ord('a' if char.islower() else 'A')****:* This calculates the index of the character within its respective case (lowercase or uppercase) in the alphabet. For example, if char is 'c', it calculates (99 - 97) = 2, and if char is 'C', it calculates (67 - 65) = 2.

- Adding the shift value shift and taking modulo 26 ensures that the shift wraps around if it goes beyond the alphabet's length.

- ***chr(... + ord('a' if char.islower() else 'A'))***: This converts the shifted index back to a character, taking into account the character's original case.

- If the character is not alpbabet and it is some special character like : , @, ! , or ! than it is remained untouched.

Example on 'Hello World' encryption:

- Here each letter will be replaced by the letter that appears three positions after in the alphabet series.

- The letter that is in uppercase will be replaced with the uppercase letter appearing after three positions.

H will be replaced by K
e will be replaced by h
l will be replaced by o
l will be replaced by o
' ' will be remained untouched as it is non-alphabetic
W will be replaced by Z
o will be replaced by r
r will be replaced by u
l will be replaced by o
d will be replaced by g
The encrypted message of 'Hello World' will be 'Khoor Zuorg'

ADD_MESSAGE()

Code:

```
def add_message():
    c1 = mycon.cursor()
    table_name = f"messages_{random.randint(1, 3)}"
    c1.execute(f'create table if not exists {table_name}(sender
varchar(50), recipient varchar(50), message varchar(6000),
encryption_key varchar(10000));')
    sender = input("Enter your name:")
    recipient = input("Enter receiver's name:")
    message = input("Message for the user:")
    key = generate_key()
    encrypted_message = encrypt_message(message)
    sql = f"insert into {table_name}(sender, recipient, message,
encryption_key) values(%s, %s, %s, %s)"
    c1.execute(sql, (sender, recipient, encrypted_message, key))
    mycon.commit()
    print("Encrypted message stored successfully!")
```

Explanation:

- This function takes the username ,receiver name and message as input, encryptes the message and stores the data in one of three tables created randomly.

- table_name is initialized with the name messages_{the random number generated from 0 to 3}. For example if the random number generated is ‘2’ than a table named as messages_2 will be created in the database.

- If the table is not present in the database it creates the table.

- After it takes all the input the message is passed as the parameter to the encrypt_message() function.

- Further the data (sender,receiver,key and the encrypted message) is inserted into the table for the particular random number generated.

- *c1.execute()* is used to execute all the sql queries like table creation and table insertion.

- *mycon.commit()* is used to save the changes in the database.

- All this is done dynamically, dynamically the table is chosen and data is inserted in the particular table.

DECRYPT_MESSAGE(ENCRYPTED_MESSAGE)

Code:

```
def decrypt_message(encrypted_message):
    shift = 3
    decrypted_message = ""
    for char in encrypted_message:
        if char.isalpha():
            shifted_char = chr(((ord(char) - ord('a' if char.islower()
else 'A') - shift) % 26) + ord('a' if char.islower() else 'A'))
            decrypted_message += shifted_char
        elif char == ' ':
            decrypted_message += ' '
        else:
            decrypted_message += char
    return decrypted_message
```

Explanation:

- We just reverse the algorithm done in encrypt_message()

- There we replaced the character to the character appearing after 3 positions, here we replace the character to the character appearing before the character.

- If the character is ‘ ’ we add the ‘ ’ (space) in the decrypted message also.

- If non-alphabetic character is encountered , it is added as it is in the decrypted_message.

VIEW_MESSAGESRECIEVER(USER)

Code:

```
def view_messagesreciever(user):
    c1 = mycon.cursor()
    for i in range(1, 4):
        table_name = f"messages_{i}"
        sql = f"SELECT sender, message, encryption_key FROM
    {table_name} WHERE recipient = %s"
        c1.execute(sql, (user,))
        messages = c1.fetchall()
        print(f"Received messages from table {table_name}:")
        for sender, encrypted_message, key in messages:
            decrypted_message =
    decrypt_message(encrypted_message)
                print(f"{sender}: {decrypted_message}")
```

Explanation:

- In this function we can view all the messages received to a particular user.

- As there are total 3 tables , so it iterates through all the tables with the help of the for loop.

- Once the user is found , we get sender, message , encryption_key from the particular table.

- We pass the message (that is in encrypted form) to the decrypt_message() function.

- Further print the sender and messages sent to that user.

VIEW_MESSAGESSENDER(USER)

Code:

```
def view_messagessender(user):
    c1 = mycon.cursor()
    for i in range(1, 4):
        table_name = f"messages_{i}"
        sql = f"SELECT recipient, message, encryption_key
    FROM {table_name} WHERE sender = %s"
        c1.execute(sql, (user,))
        messages = c1.fetchall()
        print(f"Sent messages to table {table_name}:")
        for recipient, encrypted_message, key in messages:
            decrypted_message =
    decrypt_message(encrypted_message)
                print(f"{recipient}: {decrypted_message}")
```

Explanation:

- This function is used to display all the messages sent by a particular user to this user with receiver name.

- It also iterates through the table, once the user is found , its receiver, messages and encryption_key is taken.

- Than message is decrypted with help of the decrypt_message() function.

- Further the messages sent by the user is displayed with the receiver.

MAIN FUNCTION

Code:

```
while True:
    print("\n1. Send Message\n2. View Receiver
    Messages\n3.View Sender Messages\n4. Exit")
    choice = input("Enter your choice: ")
    if choice == '1':
        add_message()
    elif choice == '2':
        user = input("Enter your reciver name whose message
    you want to see:")
        view_messagesreciever(user)
    elif choice == '3':
        user = input("Enter your sender name whose message
    you want to see:")
        view_messagessender(user)
    elif choice == '4':
        break
    else:
            print("Invalid choice. Please try again.")
```

Explanation:

- It uses the while loop for a menu driven program.

- Normal if-elif-else is used to call the functions.

ALL OUTPUTS

```
mysql> use encryption_final;
Database changed
mysql> show tables;
+----------------------------+
| Tables_in_encryption_final |
+----------------------------+
| messages_1                 |
| messages_2                 |
| messages_3                 |
+----------------------------+
3 rows in set (0.06 sec)

mysql> select * from messages_1;
+--------+-----------+------------+----------------------------+
| sender | recipient | message    | encryption_key             |
+--------+-----------+------------+----------------------------+
| sawan  | luv       | kdssb krol | defghijklmnopqrstuvwxyzabc |
| aryan  | shubham   | ndkd       | defghijklmnopqrstuvwxyzabc |
+--------+-----------+------------+----------------------------+
2 rows in set (0.04 sec)

mysql> select * from messages_2;
+--------+-----------+---------+----------------------------+
| sender | recipient | message | encryption_key             |
+--------+-----------+---------+----------------------------+
| riya   | bhumi     | khoor   | defghijklmnopqrstuvwxyzabc |
+--------+-----------+---------+----------------------------+
1 row in set (0.03 sec)

mysql> select * from messages_3;
+--------+-----------+---------+----------------------------+
| sender | recipient | message | encryption_key             |
+--------+-----------+---------+----------------------------+
| sakshi | reena     | ndlvh   | defghijklmnopqrstuvwxyzabc |
+--------+-----------+---------+----------------------------+
1 row in set (0.03 sec)
```

DIFFERENT TABLES AND ITS DATA

OPTION 1

```
>>>
========== RESTART: C:\Users\HP\Desktop\
successfully connected to mysql


1. Send Message
2. View Receiver Messages
3.View Sender Messages
4. Exit
Enter your choice: 1
Enter your name:Rachna Singh
Enter receiver's name:Avanish Singh
Message for the user:How are you?
Encrypted message stored successfully!

1. Send Message
2. View Receiver Messages
3.View Sender Messages
4. Exit
Enter your choice:
```

```
mysql> select * from messages_1;
+--------+-----------+------------+-----------------------------+
| sender | recipient | message    | encryption_key              |
+--------+-----------+------------+-----------------------------+
| sawan  | luv       | kdssb krol | defghijklmnopqrstuvwxyzabc |
| aryan  | shubham   | ndkd       | defghijklmnopqrstuvwxyzabc |
+--------+-----------+------------+-----------------------------+
2 rows in set (0.00 sec)

mysql> select * from messages_2;
+--------------+---------------+--------------+-----------------------------+
| sender       | recipient     | message      | encryption_key              |
+--------------+---------------+--------------+-----------------------------+
| riya         | bhumi         | khoor        | defghijklmnopqrstuvwxyzabc |
| Rachna Singh | Avanish Singh | Krz duh brx? | defghijklmnopqrstuvwxyzabc |
+--------------+---------------+--------------+-----------------------------+
2 rows in set (0.00 sec)

mysql> select * from messages_3;
+--------+-----------+---------+-----------------------------+
| sender | recipient | message | encryption_key              |
+--------+-----------+---------+-----------------------------+
| sakshi | reena     | ndlvh   | defghijklmnopqrstuvwxyzabc |
+--------+-----------+---------+-----------------------------+
1 row in set (0.00 sec)
```

The data is stored in messages_2 table which indicates number 2 is the randomly generated number

OPTION 2

```
1. Send Message
2. View Receiver Messages
3.View Sender Messages
4. Exit
Enter your choice: 2
Enter your reciver name whose message you want to see:luv
Received messages from table messages_1:
sawan : happy holi
Received messages from table messages_2:
Received messages from table messages_3:

1. Send Message
2. View Receiver Messages
3.View Sender Messages
4. Exit
Enter your choice: 2
Enter your reciver name whose message you want to see:sakshi
Received messages from table messages_1:
Received messages from table messages_2:
Received messages from table messages_3:

1. Send Message
2. View Receiver Messages
3.View Sender Messages
4. Exit
Enter your choice:
```

- As a message is sent to a user luv from sawan in table messages_1 the message is displayed.

- But there is no user named sakshi in the database hence no message is displayed in either of the tables.

OPTION 3

```
1. Send Message
2. View Receiver Messages
3.View Sender Messages
4. Exit
Enter your choice: 3
Enter your sender name whose message you want to see:Rachna Singh
Sent messages to table messages_1:
Sent messages to table messages_2:
Avanish Singh: How are you?
Sent messages to table messages_3:

1. Send Message
2. View Receiver Messages
3.View Sender Messages
4. Exit
Enter your choice:
```

As the message was sent from Rachna Singh to Avanish Singh in table 2, the message is successfully displayed.

OPTION 4

```
1. Send Message
2. View Receiver Messages
3.View Sender Messages
4. Exit
Enter your choice: 4
>>>
```

CONCLUSION

- Through this project based learning we learnt about how to connect the python to a database .

- We learnt to store the data in the database and also how to implement the security to the database so that no other users can access it.

- We learnt how storage and security management works.

- Using these concepts from IT Infrastructure Management we made an attempt to make a project where we are taking a message from user and the receivers name , it encrypts the data (message) and stores it in one of the tables in the database randomly.