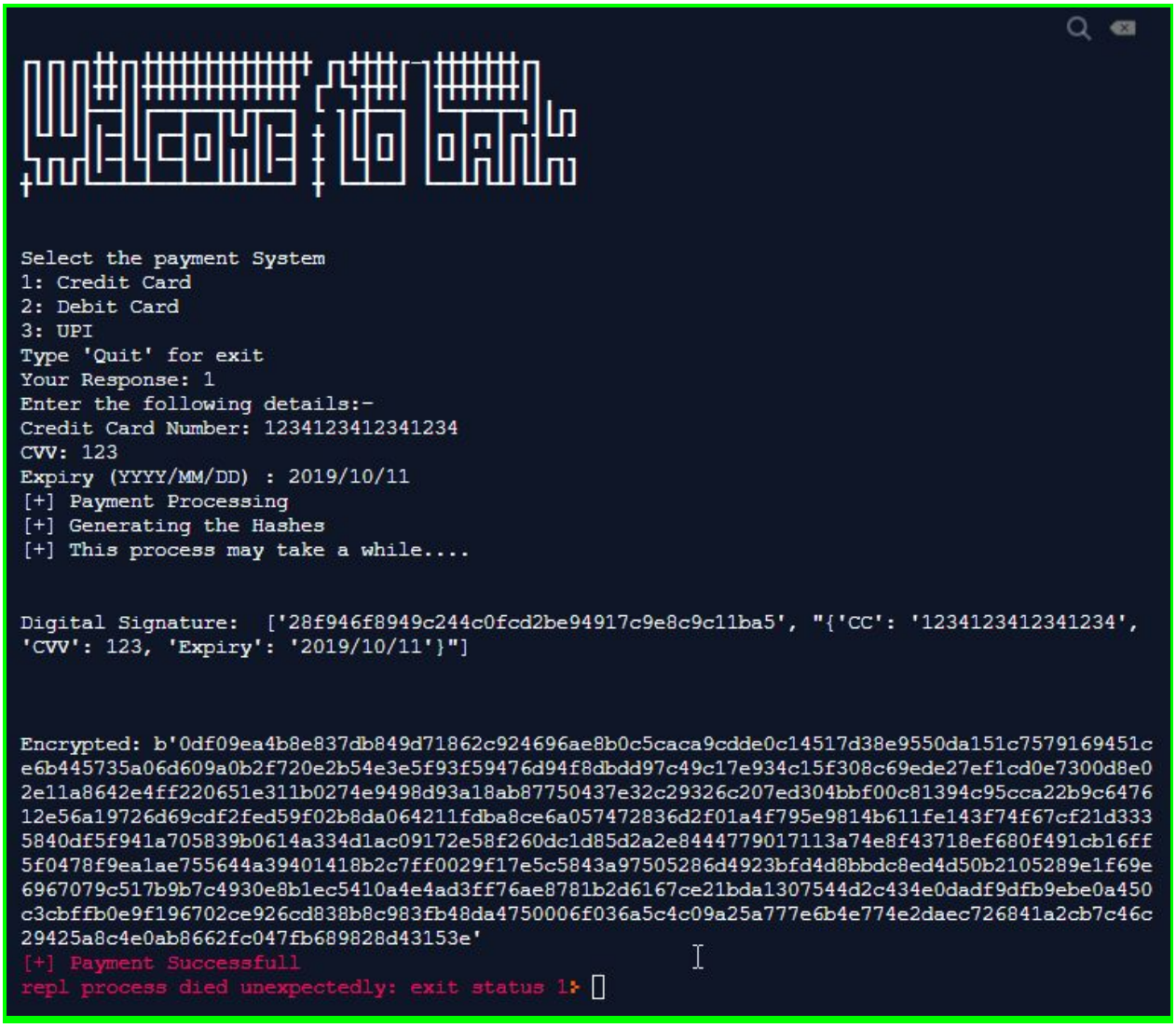
Secure Payment System

I've Created this program by using RSA,SHA-1 and also tried securing the program by Input Validation. Crashing the program is next to impossible and also this program can be implemented in Real World Scenarios.
Doing this assignment was fun.

Hritish Kumar

Output:



OVERVIEW:

1. The program takes basic inputs.
2. Generates hashes of all the inputs.
3. Creates a Digital Signature.
4. Now, uses RSA to encrypt the Digital signature.
5. Uncommenting some lines will give verbose Output.

Input:

Just enter realistic details of cc or debit card.
Or
Check ScreenShot

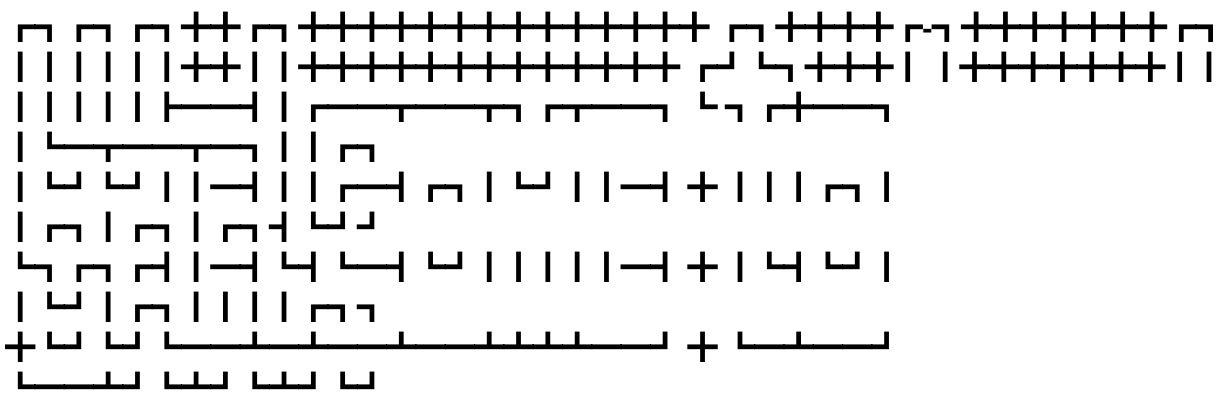
Code:-

```
#Secure Payment System Code

import time
import datetime as dt


from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
import binascii

keyPair = RSA.generate(3072)

pubKey = keyPair.publickey()
#print(f"Public key: (n={hex(pubKey.n)}, e={hex(pubKey.e)})")
pubKeyPEM = pubKey.exportKey()
#print(pubKeyPEM.decode('ascii'))

#print(f"Private key: (n={hex(pubKey.n)}, d={hex(keyPair.d)})")
privKeyPEM = keyPair.exportKey()
#print(privKeyPEM.decode('ascii'))



def validate(date_text):
    try:
        dt.datetime.strptime(date_text, '%Y/%m/%d')
    except Exception:
        exit("[-] Incorrect data format, should be YYYY/MM/DD")

def processing():
    print("[+] Payment Processing ")
    time.sleep(3)
    print("[+] Generating the Hashes")
```

```
    time.sleep(3)
    print("[+] This process may take a while....")
    time.sleep(3)


def ask(value):
    print("Enter the following details:-")
    cc = str(input(f"{value} Number: "))
    if (len(cc) != 16):
        exit(f"[-] Invalid {value} Number")
    cvv = int(input("CVV: "))
    if cvv not in range(100, 1000):
        exit("[-] Invalid CVV")
    expiry = str(input("Expiry (YYYY/MM/DD) : "))
    validate(expiry)
    processing()

    lst = {}
    lst["CC"]=cc
    lst["CVV"]=cvv
    lst["Expiry"]=expiry

    return lst


def sha1(data):
    bytes = ""

    h0 = 0x67452301
    h1 = 0xEFCDAB89
    h2 = 0x98BADCFE
    h3 = 0x10325476
    h4 = 0xC3D2E1F0

    for n in range(len(data)):
        bytes += '{0:08b}'.format(ord(data[n]))
    bits = bytes + "1"
    pBits = bits
    while len(pBits) % 512 != 448:
        pBits += "0"
    pBits += '{0:064b}'.format(len(bits) - 1)

    def chunks(l, n):
```

```
    return [l[i:i + n] for i in range(0, len(l), n)]

def rol(n, b):
    return ((n << b) | (n >> (32 - b))) & 0xffffffff

for c in chunks(pBits, 512):
    words = chunks(c, 32)
    w = [0] * 80
    for n in range(0, 16):
        w[n] = int(words[n], 2)
    for i in range(16, 80):
        w[i] = rol((w[i - 3] ^ w[i - 8] ^ w[i - 14] ^ w[i - 16]), 1)

    a = h0
    b = h1
    c = h2
    d = h3
    e = h4

    for i in range(0, 80):
        if 0 <= i <= 19:
            f = (b & c) | ((~b) & d)
            k = 0x5A827999
        elif 20 <= i <= 39:
            f = b ^ c ^ d
            k = 0x6ED9EBA1
        elif 40 <= i <= 59:
            f = (b & c) | (b & d) | (c & d)
            k = 0x8F1BBCDC
        elif 60 <= i <= 79:
            f = b ^ c ^ d
            k = 0xCA62C1D6

        temp = rol(a, 5) + f + e + k + w[i] & 0xffffffff
        e = d
        d = c
        c = rol(b, 30)
        b = a
        a = temp

    h0 = h0 + a & 0xffffffff
    h1 = h1 + b & 0xffffffff
    h2 = h2 + c & 0xffffffff
```

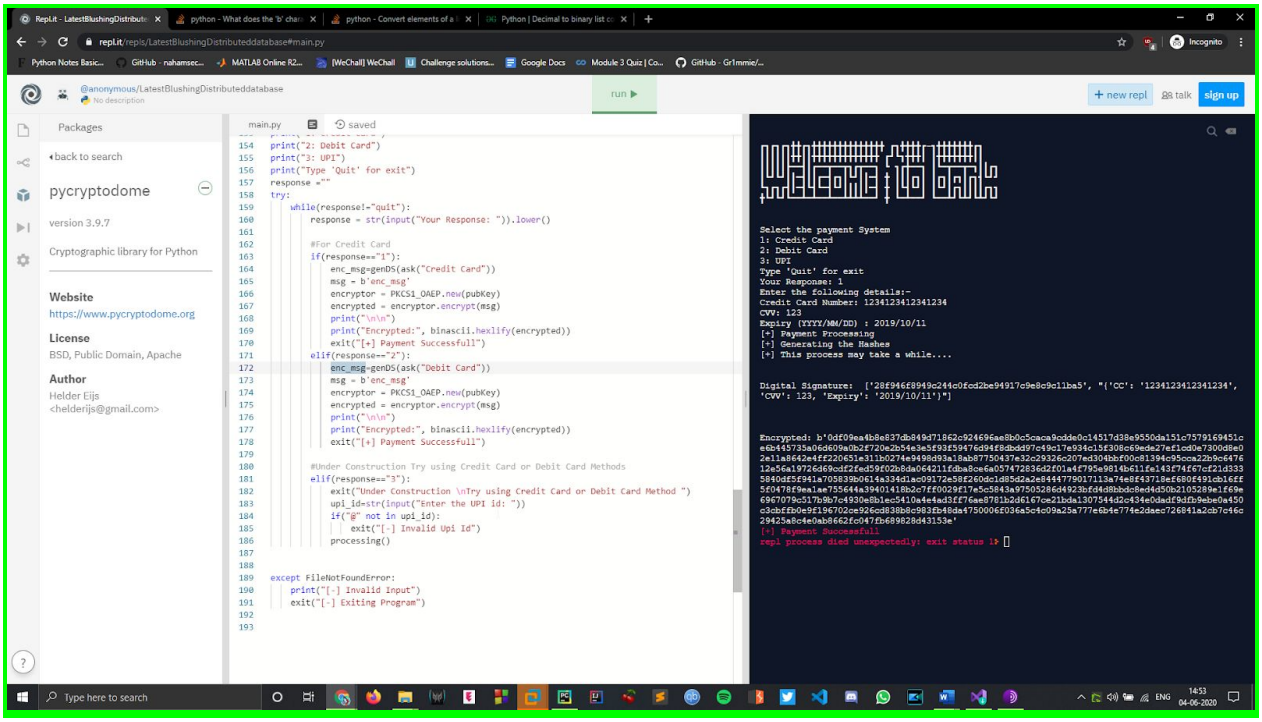
```
        h3 = h3 + d & 0xffffffff
        h4 = h4 + e & 0xffffffff

    return '%08x%08x%08x%08x%08x' % (h0, h1, h2, h3, h4)

def genDS(value):
    digitalSignature = []
    special = ["/", ",", "{", "}"]
    val = value
    val2 = ""
    for i in val:
        if i in special:
            val2 += "A"
            continue
        val2 += i

    val3 = sha1(val2)
    digitalSignature.append(str(val3))
    digitalSignature.append(str(val))
    print("\n\nDigital Signature: ", digitalSignature)

    return str(digitalSignature)

print("""
```

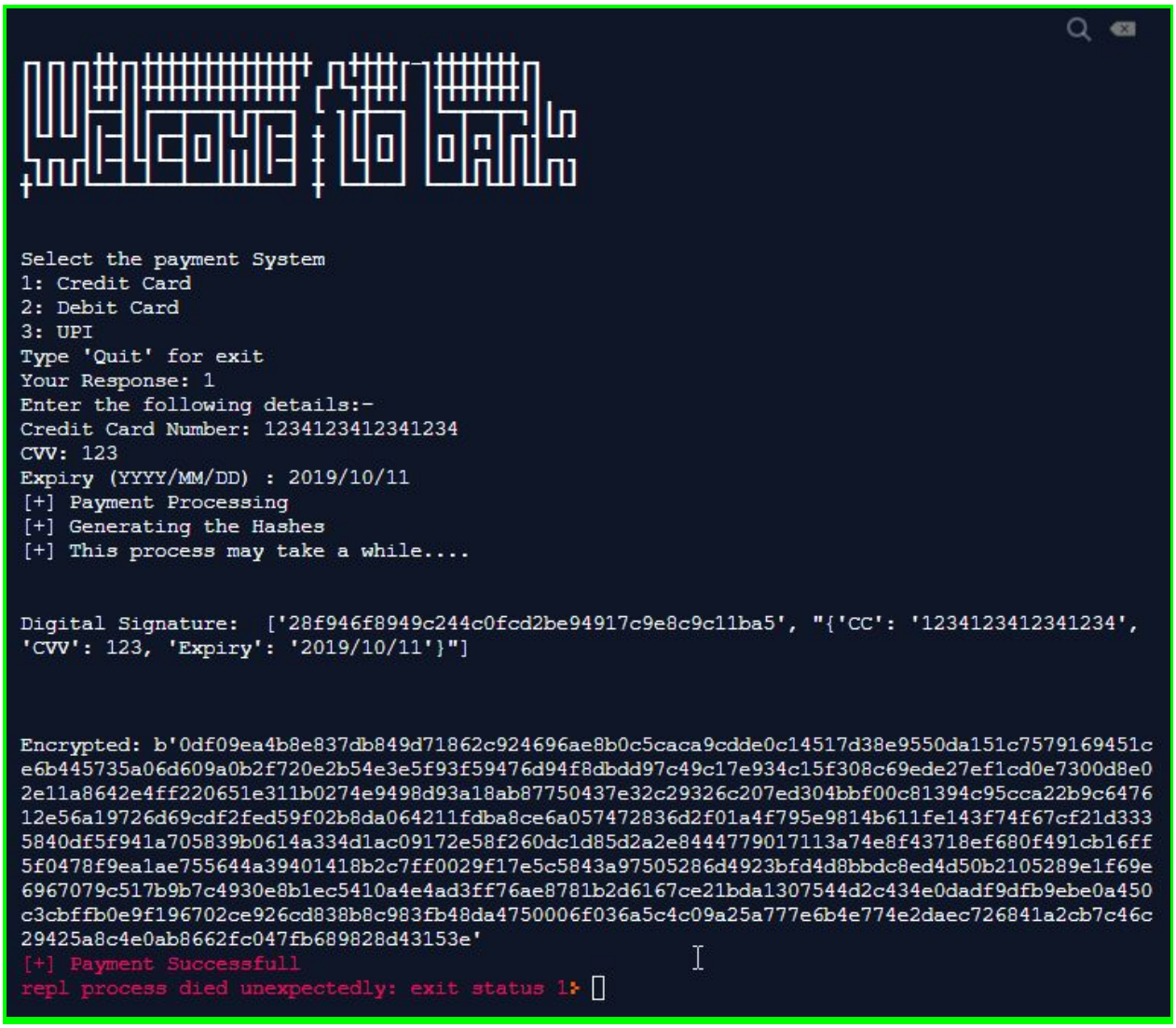


```
 """)

print("Select the payment System")
print("1: Credit Card")
print("2: Debit Card")
print("3: UPI")
print("Type 'Quit' for exit")
```

```
response =""
try:
   while(response!="quit"):
      response = str(input("Your Response: ")).lower()

      #For Credit Card
      if(response=="1"):
         enc_msg=genDS(ask("Credit Card"))
         msg = b'enc_msg'
         encryptor = PKCS1_OAEP.new(pubKey)
         encrypted = encryptor.encrypt(msg)
         print("\n\n")
         print("Encrypted:", binascii.hexlify(encrypted))
         exit("[+] Payment Successfull")
      elif(response=="2"):
         enc_msg=genDS(ask("Debit Card"))
         msg = b'enc_msg'
         encryptor = PKCS1_OAEP.new(pubKey)
         encrypted = encryptor.encrypt(msg)
         print("\n\n")
         print("Encrypted:", binascii.hexlify(encrypted))
         exit("[+] Payment Successfull")

      #Under Construction Try using Credit Card or Debit Card Methods
      elif(response=="3"):
         exit("Under Construction \nTry using Credit Card or Debit Card Method ")
         upi_id=str(input("Enter the UPI id: "))
         if("@" not in upi_id):
            exit("[-] Invalid Upi Id")
         processing()


except FileNotFoundError:
   print("[-] Invalid Input")
   exit("[-] Exiting Program")
```

ScreenShots





Hritish Kumar