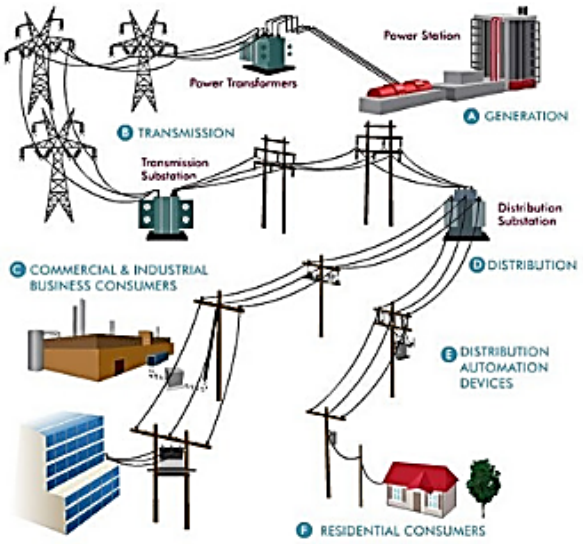
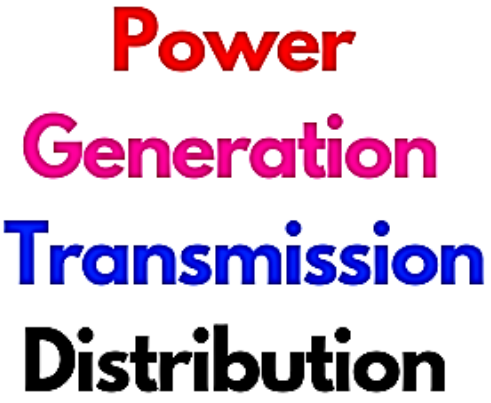
ARMY PUBLIC SCHOOL, KOLKATA

ELECTRICITY BILL ING MANAGEMENT

For AISSCE 2020-21 Examination

ELECTRICITY BILLING MANAGEMENT SYSTEM





NAME	:	**Ayush Saha, Dhruva Shaw, Smyan Kotkar**
CLASS	:	**12-SC-1**
SESSION	:	**2020-2021**
AISSCE ROLL NO	:	

CERTIFICATE

This is to certify that the following students of CLASS 12-SC-1 have prepared the report on the project ELECTRICITY BILLING MANAGEMENT SYSTEM.

The report is the result of their efforts & endeavour as a team. The report is found worthy of acceptance as final project report for the subject **Computer Science** of Class XII for the academic session 2020-21.

They have prepared the report under my guidance.

Ayush Saha	XII / Science-I	
Dhruva Shaw	XII / Science-I	
Smyan Kotkar	XII / Science-I	

(Mrs. Yamini Azhaguvel)

Date :

PGT (Computer Science)

ACKNOWLEDGEMENT

We would like to express a deep sense of thanks & gratitude to our project guide Mrs Yamini Azhaguvel, for guiding me immensely through the course of the project.

She always evinced keen interest in my work. Her constructive advice & constant motivation has been responsible for the successful completion of this project.

We also thank our parents for their motivation & support. I also take this opportunity to thank our classmates and team members for their timely help & support in compilation of this project.

Lastly, I would like to thank all those who had helped directly or indirectly towards the completion of this project.

With Thanks,

- *Dhruva Shaw*
- *Ayush Saha*
- *Smyan Kotkar*

CONTENT

MODULES USED

I. Inbuilt modules :

- **sys** : The system module is used to close the interpreter programmatically using sys.exit()
- **mysql-connector** : This module is used to perform the backend operations with the MySQL database.
- **os** : This module is imported in the program clear the terminal screen programatically, get the current working directory and make the program Operating System independent.
- **json** : This module is used to import data from .json files to the program.
- **math** : From this module the ceil function is imported to roundoff the generated value for the electric bill.
- **smtplib** : This module is imported to send the electric bills to respective customer.
- **email** : This module is imported to work accordance with smtplib module and ease the template making of the emails.
- **datetime** : This module is imported to get the current time.
- **csv** : This module is imported to read and write the csv files.
- **hashlib** : This module is imported to hash the password using the md5 hash algorithm and return the hash in a hexadecimal number
- **time** : From this module sleep function is imported to suspend execution of the calling thread for the given number of seconds
- **cProfile** : This module is to provide a deterministic profiling of the python program
- **re** : From the regular expression module compile function imported and is used to compile a regular expression pattern into a regular expression object
- **pyinstaller** : This is used to convert the python file to exe file.

2. Custom (user made) Modules

- **adminBillGen** : This contains function for the Admin Homepage.
- **clearscreen** : This contains the function for the clearscreen based on the operating system.
- **customerView** : This contains the function for the billing the view bill and this is accessible to customer only.
- **billEmail** : This contains the function for the emailing the bill to respective customer.
- **billGen** : This contains the function for to generate the bill for the corresponding month.
- **login** : This function to logged into the user in correct department.
- **logout** : This contains the function to logout the user.

Objective, Scope & Backbone of the Project

Our project entitled "**Electricity Billing System**" aim is to generate electricity bill with all the charges and penalty. Manual system that is employed is extremely laborious and quite inadequate. It only makes the process more difficult and hard. The aim of our project is to develop a system that is meant to partially computerize the work performed in the Electricity Board like generating monthly electricity bill, record of consuming unit of energy, store record of the customer and previous unpaid record. We used Python 3.8 as front end and MySql-marriaDB engine as back end for developing our project. Our project is independent of any OS and can run on any platform.

The overall project report is divided into further sub-parts which includes developing of the model system with scope for enhancement depending on the functionality of the organisation. The codes written were developed by the team jointly, tested with dummy data and found to be successful worth implementation with suitable modifications for further implementation.

Backbone of the Project :

This Project was completed using the methods which can be used in connecting MySQL and Python together. Python was chosen due to its simple structure, robustness and high capability in creating definitions. MySQL as a backend tool was chosen as a combination to give Python the meaning of flexibility and adaptability due it's simple table management system while primarily used for storing the data related to the billing system and customer details.

We as a team hope that the humble effort taken from our side would be able to create a significant change for the betterment of the lives of the people who would be using the system with adaptations as required.

Table Structure

Table Name	Customer
Field Name	Type
id	integer
meterno	integer
consumerno	biginteger
consumername	varchar()
load_con	varchar()
unit_consumed	integer
month	varchar()
year	integer
email	varchar()
address	text
amountgen	decimal

Table	Login
Field Name	Type
id	int
userid	varchar()
branch	text
session_in	datetime
session_out	datetime
dept_no	int

Table	User
Field Name	Type
id	int
username	varchar()
password	varchar()
branch	varchar()
dept_no	int
useradmin_id	varchar()

Table Name	DEPT
Field Name	Type
id	int
dept_no	int
deptname	text

WORKING DESCRIPTION

- **FILES GENERATED:**

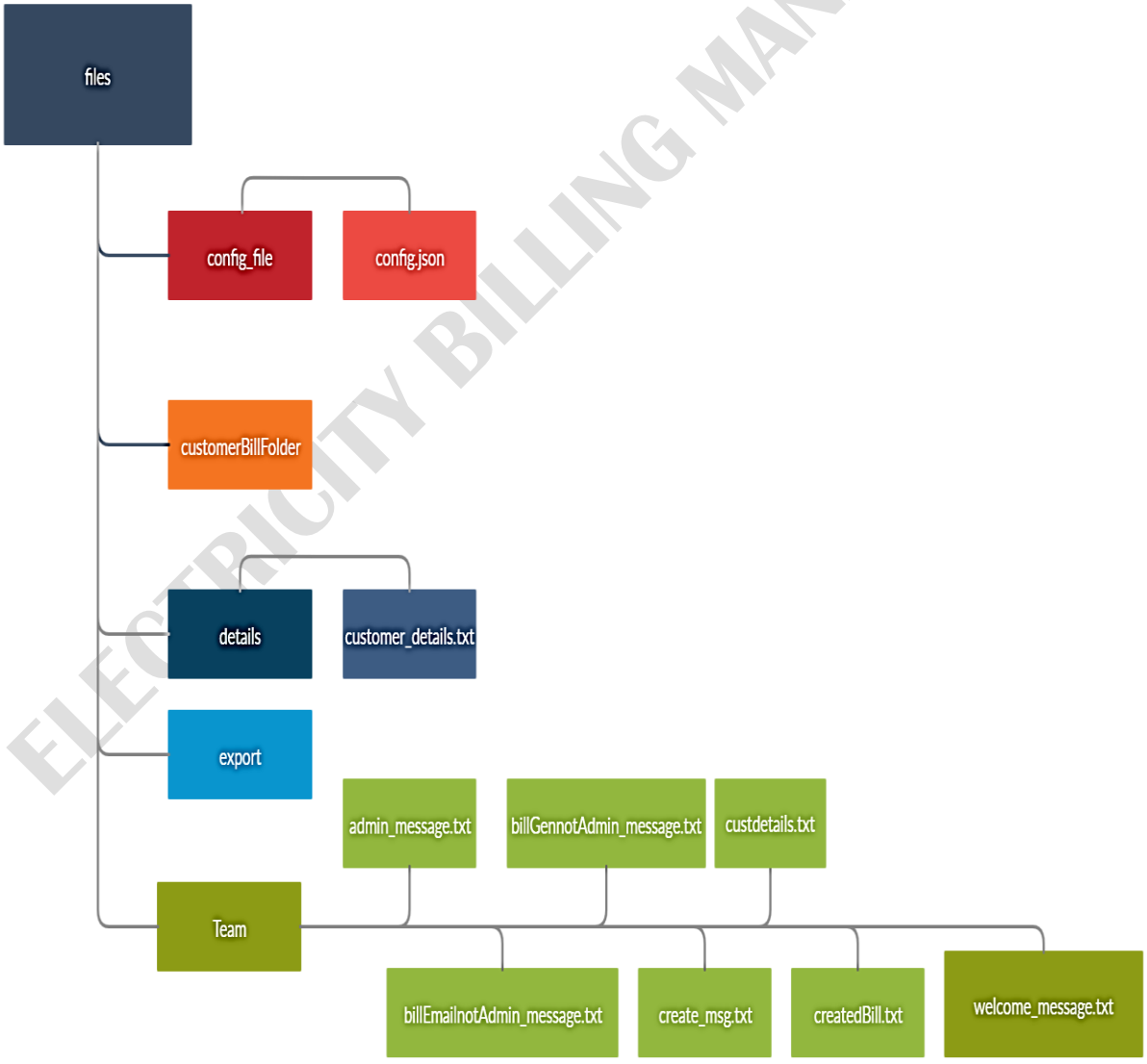
  config.json, customer_details.csv, employee_details.csv, admin_message.txt, billEmailnotAdmin_message.txt, billGennotAdmin_message.txt, create_msg.txt, createdBill.txt, custdetails.txt, welcome_message.txt
  An exe file is generated for distribution.

- **DIRECTORY STRUCTURE :**

  The master folder contains a folder named 'files'.

  Then the files folder contains the following 5 folder.

  config_file', 'customerBillfolder', 'details', 'export', 'messages'



The program has been designed with following modes of operation:

1. **Admin**
2. **Bill Generation**
3. **Bill Delivery**
4. **Customer Bill View**

Admin : It part has the privileges of a super user. It has the power to create, delete and edit, etc.

Bill Generation : This module has been designed to generate electricity bills based on the inputs of meter reading.

Bill Delivery : This module will email the bill to respective customers address and thus bring the concept of a reduce paper and reduce carbon footprint making the environment greener and sustainable.

Customer View Bill : This portal is only for use by the consumer to view the bill for the current month.

This is all in one program where electricity department can enter the data through the MySQL database, where a consumer can view its own bill just by using this program.

Features:

It has an Admin Panel which the super user can access to enter the data of the consumer to database given by the electricity meter department in form of a csv file. It has a login system where the password are hashed using md5 hash algorithm then the hash are converted to the hexadecimal units. The super user can also add the details of a new operator or delete its details.

It also a configurable json file, and configure the contents of a program.

This program is also Operating System independent.

It also has a portal for the Bill Generation and Bill Delivery Department where the respective operator can generate the electricity bill with help of only one command and also deliver the bill to customers using their emails.

It has also portals for the customers where a consumer can enter its consumer no and get the bill details for the current month.

Cons :

A constant Internet Connection is required.

The database of the consumer has to be constantly updated by the admin every month through csv files.

And in the customers or consumer portal in case of any emergency or help requiring situation one cannot contact any authority as would be required to resolve the problem in hand.

PROGRAM CODE

#mainRun.py

```
from datetime import datetime
from os import system
from login import welcome_message
from clearscreen import clear
import mysql.connector as c
connection = c.connect(host='localhost', database='electricity_bill', user='root', password='')
db = connection.cursor()

#__main__
#Checks if the user is already logged in
clear()

db.execute(f'UPDATE login SET session_out="{datetime.now()}" WHERE session_out="0000%"')
connection.commit()
welcome_message()
```

#login.py

```
import hashlib
import json
import sys
import time
from datetime import datetime
from os import path

import mysql.connector as c

from adminBillGen import adminHome
from billGen import bilGenHome
from clearscreen import clear
from billEmail import bilEmailHome
from customerView import consumerDetails

connection = c.connect(host='localhost', database='electricity_bill', user='root', password='')
db = connection.cursor()

#Opening of config.json file
THIS_FOLDER = path.dirname(path.abspath(__file__))
my_file = path.join(THIS_FOLDER,'files','config_file', 'config.json')

with open(my_file, 'r') as c:
    params = json.load(c)["params"]
```

```
#Welcome message
def welcome_message():
   '''The first welcome message'''
   clear()

   #The welcome message
welcome_message =open('files/messages/welcome_message.txt','r').read()
print(welcome_message.format(params['company_name']))

#Calling the login_deptno function
   login_deptno()

#Login system
#####################################################################
# The function 1 and 2 are related to each other
# The first function catches exception and the second function is for validation

# (1) Makes the user to get logged in into the correct deptno
def login_deptno(message=''):
   '''Makes the user to get logged in into the correct deptno'''
   while True:
      print(message)
      try:
         deptno_in = int(input('Please enter the department no.\n'))
         if deptno_in == 15675812:
            consumerDetails()
         else:
            logincheck(deptno_in)
         break
      except ValueError:
         print()
         print('Please Enter a number not alphabets')


# (2) Checks the login (Validation)
def logincheck(deptno):
   '''This is a function to check if the user exists and gets the user logged in'''

   #Department No dictionary
   db.execute('SELECT dept_no FROM dept')
   sqlquery = db.fetchall()
   deptno_dict = (i for i in sqlquery)

   #Check if the department no entered is correct
   newline='\n'
   if (deptno,) not in deptno_dict:
```

```
        login_deptno(f'{deptno} Department No is not valid {newline} Please enter a valid department no !')
    else:
        login_user(deptno)

##########################################################################################
# Similarly here the function 3 and 4 are related to each other
# the 3rd function is used to logged the user answer in and 4th function is used for creating a session and
# then give the user out the appropriate page


#  (3) Make user logged in
def login_user(deptno):
    '''This is the login screen'''
    clear()

    print()
    print('Now please enter your login credentials')
    print('---------------------------------------')
    userid=input('Please enter your USERID\n')
    print('---------------------------------------')
    password=input('Please enter your password\n')
    hashpass = hashlib.md5(password.encode())

    db.execute(f'SELECT * FROM user WHERE password="{hashpass.hexdigest()}" AND dept_no="{deptno}" AND useradmin_id="{userid}";')
    query = db.fetchall()

    if query==None or query==[]:
        print('The given credentials where wrong')
        print('Please wait for 2 sec!')
        time.sleep(2)
        welcome_message()
    else:
        login_user_in(userid,hashpass.hexdigest(),deptno)


#  (4) Checks the logged in user branch and gives out the appropriate page
def login_user_in(userid,hashpass,deptno,work=None):
    '''Checks the logged in user branch and gives out the appropriate page'''
    logintime = datetime.now()   #Creating session

    db.execute(f'SELECT branch FROM user WHERE useradmin_id="{userid}"')
    branch=db.fetchall()
```

```
    if work==None:
        db.execute(f'INSERT INTO login(userid,branch,session_in,dept_no)
VALUES("{userid}","{branch[0][0]}","{logintime}","{deptno}")')
        connection.commit()
    else:
        db.execute(f'UPDATE login set session={datetime.now()} WHERE userid="{userid} AND
session_out="0000%"')

    branchget = userid.split("#")
    print('Please wait you being redirected there! in 3 sec.....')
    time.sleep(3)

    #Validation
    branch = str(branchget[1])
    if branch=='ADMIN':
        adminHome(userid,logintime)
    elif branch=='BILL GENERATION':
        bilGenHome(userid,logintime)
    elif branch=='BILL DELIVERY':
        bilEmailHome(userid,logintime)
##############################################################################
```

#logout.py

```
import sys
import time
from datetime import datetime
import mysql.connector as c
from clearscreen import clear

connection = c.connect(host='localhost', database='electricity_bill', user='root', password='')
db = connection.cursor()

#Logout function
def logout(userid):
    db.execute(f'UPDATE login SET session_out="{datetime.now()}" WHERE userid="{userid}"
AND session_out="0000%"')
    connection.commit()

    clear()
    print(f'You have been logged out!!! {userid}')
    print('The window is closing the 2 sec')
    time.sleep(2)
    clear()
    sys.exit()
```

#clearscreen.py

```
from os import name, system
# define our clear function
def clear():
   # for windows
   if name == 'nt':
      _ = system('cls')
   # for mac and linux(here, os.name is 'posix')
   else:
      _ = system('clear')
```

#customerView.py

```
import mysql.connector as c
import datetime
from os import path
import json
import sys
import time
from clearscreen import clear

connection = c.connect(host='localhost', database='electricity_bill', user='root', password='')
db = connection.cursor()

#Opening of config.json file
THIS_FOLDER = path.dirname(path.abspath(__file__))
my_file = path.join(THIS_FOLDER,'files','config_file', 'config.json')

with open(my_file, 'r') as c:
   params = json.load(c)["params"]

def consumerDetails():
   '''This function is the view page for the customers bill generation'''
 clear()
   db.execute('SELECT consumerno from customer')
   detailsconsumerno = db.fetchall()
   mydate = datetime.datetime.now()
while True:
      try:
         consumerno = int(input('Please enter your consumer no.\n'))
      except ValueError:
         print()
         print("Please enter a valid consumer no")
 if (consumerno,) not in detailsconsumerno:
         print()
         print('The consumer no does not exists!! \nPlease enter a valid consumer no')
      else:
         break
```

```
    db.execute(f'SELECT * from customer where consumerno={consumerno} AND
month="{mydate.strftime("%B")}"')
    custdetails = db.fetchall()[0]

    if custdetails[-1]==0:
        print('No bill is not generated for this month!')
    else:
        my_file1 = path.join(THIS_FOLDER,'files','messages', 'custdetails.txt')
        with open(my_file1, 'r') as c1:
            fileread = c1.read()

print(fileread.format(params['company_name'],custdetails[3],custdetails[1],custdetails[2],c
ustdetails[4],custdetails[5],custdetails[-1],custdetails[8],custdetails[9]))

    print()
    print('Press anything the exit!!!')
    input()
    print(f"Thank you for using the {params['company_name']} ELECTRICITY CUSTOMER
DEPARTMENT SERVICES")
    time.sleep(2)
    sys.exit()
```

#billGen.py

```
import json
import time
from datetime import datetime
from math import ceil
from os import getcwd, path
import mysql.connector as c
from clearscreen import clear
from logout import logout
connection = c.connect(host='localhost', database='electricity_bill', user='root', password='')
db = connection.cursor()

#Opening of config.json file
THIS_FOLDER = path.dirname(path.abspath(__file__))
my_file = path.join(THIS_FOLDER,'files','config_file', 'config.json')

with open(my_file, 'r') as c:
    params = json.load(c)["params"]
def bilGenHome(userid,logintime):
    '''This is the bill generation department homepage function'''

    mydate = datetime.now()
    clear()
```

```
    #The bill generation welcome message
    billGenAdmin_message = open('files/messages/billGennotAdmin_message.txt','r').read()
    funcAdminTuple = ('01#02','00#01')


print(billGenAdmin_message.format(params['company_name'],userid,logintime,datetime.now(),mydate.strftime("%B")))
    userinput = input()

    if userinput not in funcAdminTuple:
        clear()
        bilGenHome(userid,logintime)

    else:
        if userinput=='01#02':
            generateBill(userid,logintime)
        elif userinput=='00#01':
            logout(userid)

def generateBill(userid,logintime):
    mydate= datetime.now()
    month = mydate.strftime("%B")

    db.execute(f'SELECT unit_consumed,consumerno FROM customer WHERE month="{month}" AND amountgen=0.00000')
    consumerno = db.fetchall()

    #Now checking the database if the meter department has given the data
    if consumerno==[] or consumerno==None:
        print()
        print('No data for this month were provided by the Meter Department!')
        print('OR')
        print('The data was was generated already for this month!')
        print('Please contact your Meter Department!')
        time.sleep(2)
        bilGenHome(userid,logintime)

    db.execute(f'SELECT unit_consumed,consumerno FROM customer WHERE month="{month}"')
    consumerno = db.fetchall()

    #Getting the prevous reading and current reading
    counter=0
    for x,y in consumerno:
        db.execute(f'SELECT * From customer WHERE unit_consumed={x} AND consumerno={y}')
        custdetails = db.fetchall()[0]
```

```
        amountgen,rebate,aduj = Bill_Calc1(x)
        db.execute(f'UPDATE customer SET amountgen={amountgen} WHERE
consumerno={y}')
        connection.commit()
        print(f'THE BILL FOR THE CONSUMER NO {y} IS GENERATED Rs.{amountgen}')
        print()

        counter+=1

        with open(path.join(getcwd(),'files','messages','createdBill.txt'),'r') as fileCreated:
            fileReadCreated = fileCreated.read()

            with open(path.join(getcwd(),'files','customerBillFolder',f'{x}{y}.txt'),'w+') as
fileBillCreated:

fileBillCreated.write(fileReadCreated.format(params['company_name'],custdetails[3],custde
tails[1],custdetails[2],custdetails[4],custdetails[5],rebate,aduj,amountgen,custdetails[8],cust
details[9]))
            # UPDATE `customer` SET `amountgen` = '925.60001' WHERE `customer`.`id` = 1

    print(counter, " bills generated.")
    input('Press anything to continue')
    time.sleep(2)
    bilGenHome(userid,logintime)

def Bill_Calc1(unit):
    meter = 10              #Meter Rent
    MVCA = 60               # Metre Load charge
    fixedChrge = 100        #This is the fixed charge
    untstr = str(unit)[-1]  #This is the Adjustment Chrges

    if((unit>=1)and(unit<=50)):#between 1 - 50 units
        return (ceil(unit*4.89)+meter+MVCA+fixedChrge+int(untstr)-1,1,untstr) #At the end the
price deducted is rebate

    elif((unit>50)and(unit<=150)):#between 50 - 150 units

        return                  (ceil((50*4.89)+(unit-50)*5.4)+meter+MVCA+fixedChrge+int(untstr)-
1.4,1.4,untstr) #At the end the price deducted is rebate

    elif((unit>150)and(unit<=250)):#between 150 -  250 units
        return                                                  (ceil((50*4.89)+((150-50)*5.4)+(unit-
150)*6.41)+meter+MVCA+fixedChrge+int(untstr)-1.5,1.5,untstr) #At the end the price
deducted is rebate

    elif(unit>250):         #above 250 units
```

```
        return (ceil((50*4.89)+((150-50)*5.4)+((250-150)*6.41)+(unit-250)*7.16)+meter+MVCA+fixedChrge+int(untstr)-1.6 ,1.6,untstr) #At the end the price deducted is rebate
    else:
        return (0,0,0)
        #amount=0;
```

#billEmail.py

```
import json
import os
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from datetime import datetime
from os import path, getcwd
import time
import mysql.connector as c

from clearscreen import clear
from logout import logout

connection = c.connect(host='localhost', database='electricity_bill', user='root', password='')
db = connection.cursor()

#Opening of config.json file
THIS_FOLDER = path.dirname(path.abspath(__file__))
my_file = path.join(THIS_FOLDER,'files','config_file', 'config.json')

with open(my_file, 'r') as c:
    params = json.load(c)["params"]

def bilEmailHome(userid,logintime):
    '''This is the bill generation department homepage function'''

    mydate = datetime.now()
    clear() #Clear the screen

    billGenAdmin_message = open('files/messages/billEmailnotAdmin_message.txt','r').read()
    funcAdminTuple = ('01#02','00#01')

print(billGenAdmin_message.format(params['company_name'],userid,logintime,datetime.now(),mydate.strftime("%B")))
    userinput = input()
if userinput not in funcAdminTuple:
        clear() #Clear the screen
        bilEmailHome(userid,logintime)
```

```
    else:
        if userinput=='01#02':
            sendmailtocustomers(userid,logintime)
        elif userinput=='00#01':
            logout(userid)

def sendmailtocustomers(userid,logintime):
    port, smtp_server = 465, 'smtp.gmail.com'
    login, password = params['email'], params['password_email']
    mydate = datetime.now()

    db.execute(f'SELECT email,consumername, consumerno FROM customer WHERE
month="{mydate.strftime("%B")}"')
    data = db.fetchall()

    message = MIMEMultipart()
    message["from"] = login

    error,emailno = 0,0
    for x,y,z in data:
        message["subject"] = f"Your electricity bill has been generated for the month
{mydate.strftime('%B')} ({y})"

        db.execute(f'SELECT unit_consumed FROM customer WHERE
month="{mydate.strftime("%B")}" AND consumerno="{z}"')
        unitsConsumed = db.fetchall()[0][0]
        try:
            with  open(path.join(getcwd(),'files','customerBillFolder',f'{unitsConsumed}{z}.txt'),'r')
as bill:
                body = bill.read()

                with smtplib.SMTP(smtp_server, port) as server:
                    server.login(login, password)
                    server.sendmail(message["from"], x, body)
                    print(f"Email (BILL) sent to {y}")
                    print()
                    emailno+=1
        except:
            print('There was some error!')
            print()
            error+=1
    print(emailno, " Email sent!")
    print("With ",error," errors!")
    print()
    print("Now please wait for two seconds!")
    time.sleep(2)
    bilEmailHome(userid,logintime)
```

#adminBillGen.py

```
import csv
import hashlib
import json
import os
import sys
import time
from datetime import datetime
from os import path, system

import mysql.connector as c
from mysql.connector import Error

from billEmail import bilEmailHome
from billGen import bilGenHome
from clearscreen import clear
from logout import logout

connection = c.connect(host='localhost', database='electricity_bill', user='root', password='')
db = connection.cursor()

#Opening of config.json file

THIS_FOLDER = path.dirname(path.abspath(__file__))
my_file = path.join(THIS_FOLDER,'files','config_file', 'config.json')

with open(my_file, 'r') as c:
    params = json.load(c)["params"]


#Admin
################################################################################


def adminHome(userid,logintime):
    #Here userinput is for the functioncode coming from the other function
    '''This the admin homepage'''

    clear() #Clear the screen

    #The admin welcome message
    admin_message = open('files/messages/admin_message.txt','r').read()

print(admin_message.format(params['company_name'],userid,logintime,datetime.now()))

    userinput=input()
    funcAdminTuple = ('01#01','05#02','06#03','04#01','00#01','02#01','07#44','03#01')
```

```
    if userinput not in funcAdminTuple:
        clear() #Clear the screen
        adminHome(userid,logintime)

    else:
        if userinput=='01#01':
            create_user(userid,logintime)
        elif userinput=='05#02':
            delete_user(userid,logintime)

        elif userinput=='06#03':
            dumpdata('customer',userid,logintime)
        elif userinput=='07#44':
            dumpdata('user',userid,logintime)

        elif userinput=='03#01':
            exportdatatoTable(userid,logintime)

        elif userinput=='02#01':
            bilGenHome(userid, logintime)
        elif userinput=='04#01':
            bilEmailHome(userid, logintime)

        #For the Logout
        elif userinput=='00#01':
            logout(userid)


##############################################################################

##############################################################################

def create_user(userid23,logintime23):
    '''This function is used to create a user of the software'''
    clear() #Clear the screen
    db.execute('SELECT dept_no, deptname from dept')
    dept = db.fetchall()

    #Printing the department no
    print('    Department No      |          Department name')
    print('-------------------------------------------------------')
    for i,j in dept:
        print(f'     {i}                    {j}   ')
    print('Following are the department no')
    print()
```

```
    #Department No dictionary
    db.execute('SELECT dept_no FROM dept')
    sqlquery = db.fetchall()

    #Asking to enter the department no
    while True:
        try:
            deptno1 = int(input('Enter the department no\n'))

            if (deptno1,) in sqlquery:
                break
            else:
                print(f'{deptno1} Department No is not valid \n Please enter a valid department no
!')
        except:
            print('Enter no not characters!')

    #Asking to enter the name
    name1 = input('Please enter the name\n')
    name=''
    for i in name1:
        if i.isalpha(): name+=i

    while True:
        #ENTERING THE PASSWORD
        password1 = input('Please enter a password\n')
        password2 = input('Please retype the password\n')
        if password1==password2:
            break
        else:
            clear()
            print('Enter again the two password dosen\'t match!')

    hashpass1 = hashlib.md5(password1.encode())
    db.execute(f'SELECT deptname FROM dept WHERE dept_no={deptno1}')
    #Getting the branch name
    branch = db.fetchall()[0][0]

    # generating the useradminid
    db.execute(f'select username from user where username="{name}"')
    occurence = len(db.fetchall())
    useradminid = f'{deptno1}{occurence+1}{name[:2]}#{branch}'

    #Inserting the data into database
    db.execute(f'INSERT INTO user
VALUES(NULL,"{name}","{hashpass1.hexdigest()}","{branch}",{deptno1},"{useradminid}")')
    connection.commit()
```

```
    #The admin welcome message
    clear()
    created_message = open('files/messages/create_msg.txt','r').read()
    print(created_message.format(name,password1,branch,deptno1,useradminid))
    print()
    input('Press any key to continue')
    adminHome(userid23,logintime23)
################################################################################

################################################################################

def delete_user(userid,logintime):
    clear()
    #Department No dictionary
    db.execute('SELECT useradmin_id FROM user')
    sqlquery = db.fetchall()
    #Asking to enter the department no
    while True:
        UserAdminId = input('Enter the UserAdminId \n')

        if (UserAdminId,) in sqlquery:
            break
        else:
            print(f'{UserAdminId} UserAdminId is not valid \n Please enter a valid UserAdminId !')

    db.execute(f'DELETE FROM user WHERE useradmin_id="{UserAdminId}"')
    connection.commit()
    print('The user succesfully deleted')
    time.sleep(1)
    adminHome(userid,logintime)

################################################################################

################################################################################
def dumpdata(tablename,userid,logintime):
    '''This Function is used to dump all the data from tables to a csv files'''
    QUERY = f'SELECT * FROM {tablename}'
    db.execute(QUERY)
    result=db.fetchall()
    connection.commit()

    if tablename=='user':
        filename = 'employee_details'
    else:
        filename = 'customer_details'
    BASE_DIR = os.getcwd()
```

```
  c1 = csv.writer(open(os.path.join(BASE_DIR,'files','details',f'{filename}.csv'),
'w',newline=''))
  for x in result:
    c1.writerow(x)


  print('The the data has been successfully dumped')
  print('The path of the file is:')
  print(os.path.join(BASE_DIR, 'files', 'details', f'{filename}.csv'))

  time.sleep(2)
  adminHome(userid,logintime)

#############################################################################

#############################################################################

def exportdatatoTable(userid,logintime):
  print()

  BASE_DIR = os.getcwd()

  print('YOU NEED TO WRITE THE DATA IN A CSV FILE')
  print()
  print('AND PLACE IT IN THE FOLWWING PATH:')
  print(os.path.join(BASE_DIR, 'files', 'export'))
  print()

  filename = input('Please you filename that you put in that directory \n(no need of putting
the .csv after the filename)\n')
  n=0
  csv_data = csv.reader(open(os.path.join(BASE_DIR, 'files', 'export',f'{filename}.csv'),'r'))
  for row in csv_data:
    try:
      db.execute(f'INSERT INTO customer
VALUES({row[0]},{row[1]},{row[2]},"{row[3]}","{row[4]}",{row[5]},"{row[6]}","{row[7]}","{ro
w[8]}","{row[9]}",{row[10]})')
      connection.commit()
    except Error: n+=1

  print(n,'number of duplicate values detected!!')
  time.sleep(2)
  adminHome(userid,logintime)
```

#config.json

```
{
   "params": {
      "company_name": "ABC",
      "email": "dhruvashaw@gmail.com",
      "password_email": "cube12345?"
   }
}
```

#admin_message.txt

WELCOME TO ADMIN HOMEPAGE OF {} ELECTRICIY DEPPARTMENT

USERID : {}

LOGIN TIME: {}

CURRENT TIME: {}

WHAT YOU WANT TO DO?

SL NO.	FUNCTIONS AVAILABLE	FUNCTIONS CODE--
(1).	REGISTER OPERATOR	01#01
(2).	DELETE OPERATOR	05#02
(3).	CHECK ALL THE CUSTOMER DETAILS	06#03
(4).	CHECK THE BILL GENERATION DEPARTMENT	02#01
(5).	CHECK the OPERATOR DETAILS	07#44
(6).	INSERT NEW CUSTOMERS USING CSV FILES	03#01
(7).	CHECK THE BILL DELIVERY DEPARTMENT	04#01
(8).	LOGOUT	00#01

NOW PLEASE ENTER THE FOLLOWING FUNCTION NO IN ORDER TO EXECUTE A TASK.

#billEmailnotAdmin_message.txt

WELCOME TO BILL DELIVERY (EMAIL) DEPARTMENT GENERATION HOMEPAGE OF {} ELECTRICIY DEPPARTMENT

USERID : {}

LOGIN TIME: {}

CURRENT TIME: {}

CURRENT MONTH NAME : {}

WHAT YOU WANT TO DO?

SL NO.	FUNCTIONS AVAILABLE	FUNCTIONS CODE
(1).	SEND THE BILLS TO CUSTOMERS FOR THIS MONTH	01#02
(2).	LOGOUT	00#01

NOW PLEASE ENTER THE FOLLOWING FUNCTION NO IN ORDER TO EXECUTE A TASK.

billGennotAdmin_message.txt

WELCOME TO BILL GENERATION DEPARTMENT HOMEPAGE OF {} ELECTRICIY DEPPARTMENT

USERID : {}

LOGIN TIME: {}

CURRENT TIME: {}

CURRENT MONTH NAME : {}

WHAT YOU WANT TO DO?

SL NO.	FUNCTIONS AVAILABLE	FUNCTIONS CODE
(1).	GENERATE THE BILL FOR THIS MONTH	01#02
(2).	LOGOUT	00#01

NOW PLEASE ENTER THE FOLLOWING FUNCTION NO IN ORDER TO EXECUTE A TASK.

#create_msg.txt

The user is created with the follwoing credentials:

USERNAME : {}
PASSWORD : {}
BRANCH : {}
DEPARTMENT NO : {}
USERADMIN ID : {}

createdBill.txt

{} ELECTRICITY CUSTOMER BILL

Consumer Name : {}
Meter No : {}
Consumer No : {}
Meter Load : {}
Units Consumed : {}
Meter Rent : ₹10
MVCA : ₹60
Fixed Charge : ₹100
Rebate : ₹{}
Adujustment Charges : {}
Net Amount Payable : ₹ {}
Email : {}
Address : {}

custdetails.txt

{} ELECTRICITY CUSTOMER DEPARTMENT

Consumer Name : {}

Meter No : {}
Consumer No : {}
Meter Load : {}
Units Consumed : {}
Net Amount Payable : {}
Email : {}
Address : {}

welcome_message.txt

Welcome to the {} ELECTRICITY BILL MANAGEMENT

The following departments are available for the login.

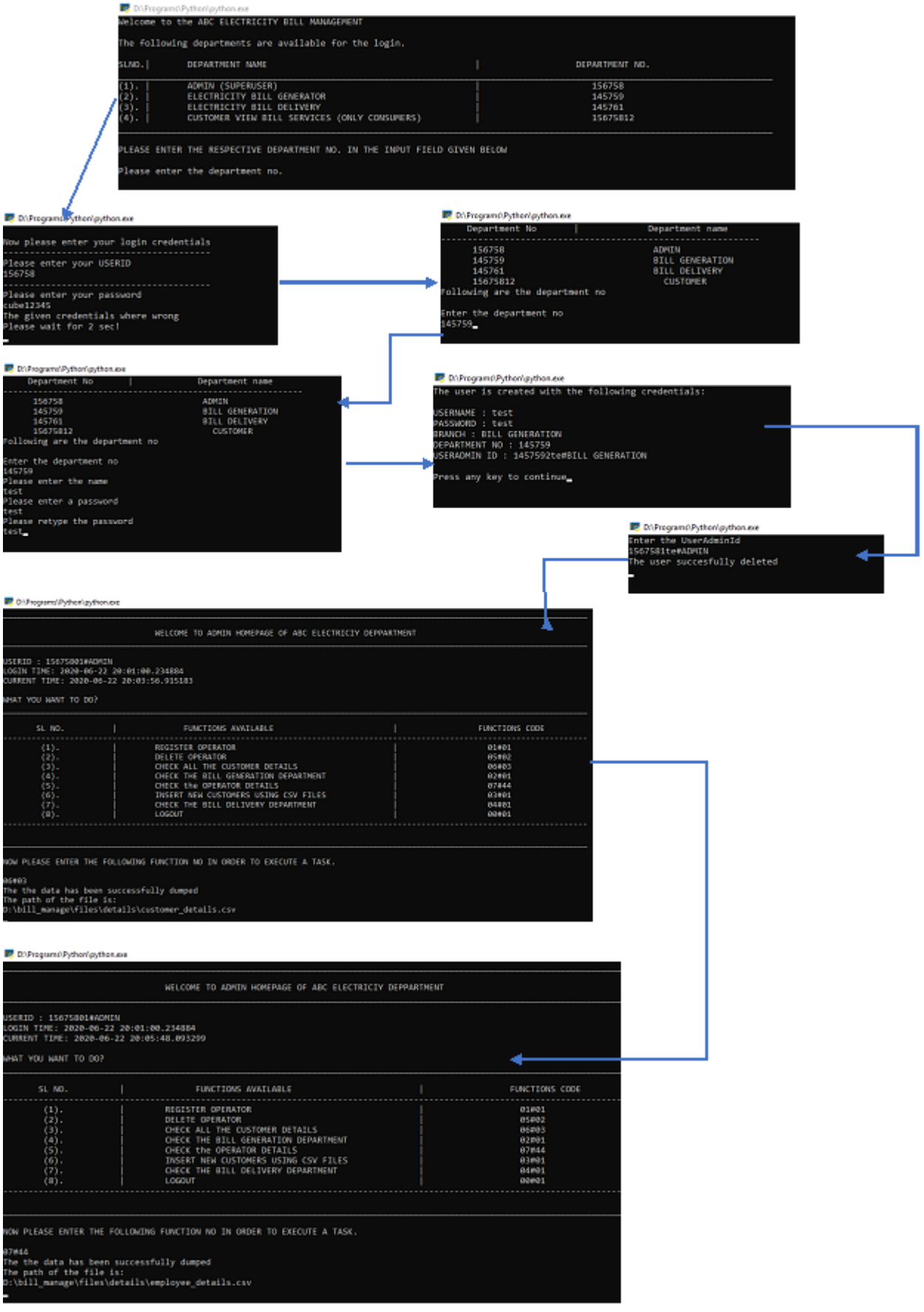
SLNO.	DEPARTMENT NAME	DEPARTMENT NO.
(1).	ADMIN (SUPERUSER)	156758
(2).	ELECTRICITY BILL GENERATOR	145759
(3).	ELECTRICITY BILL DELIVERY	145761
(4).	CUSTOMER VIEW BILL SERVICES (ONLY CONSUMERS)	15675812

PLEASE ENTER THE RESPECTIVE DEPARTMENT NO. IN THE INPUT FIELD GIVEN BELOW

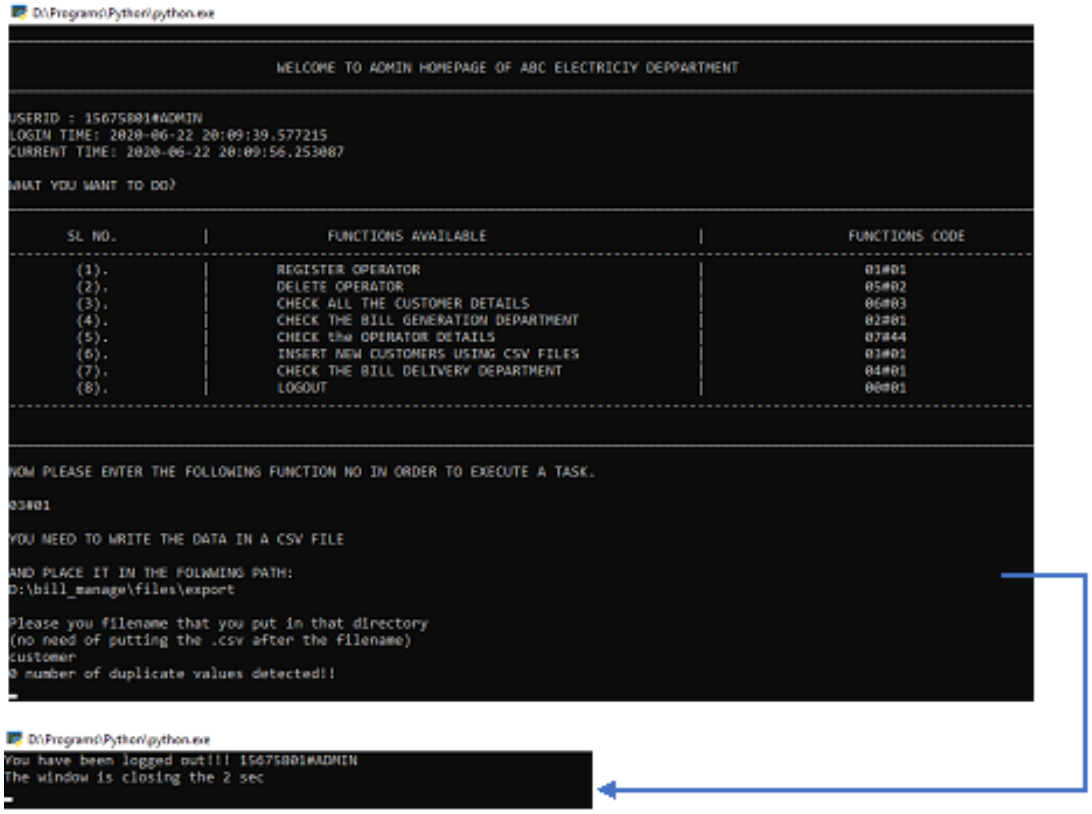
BIBLIO GRAPHY

- https://www.codewithharry.com/
- https://www.geeksforgeeks.org/
- https://www.python.org/doc/
- https://stackoverflow.com/

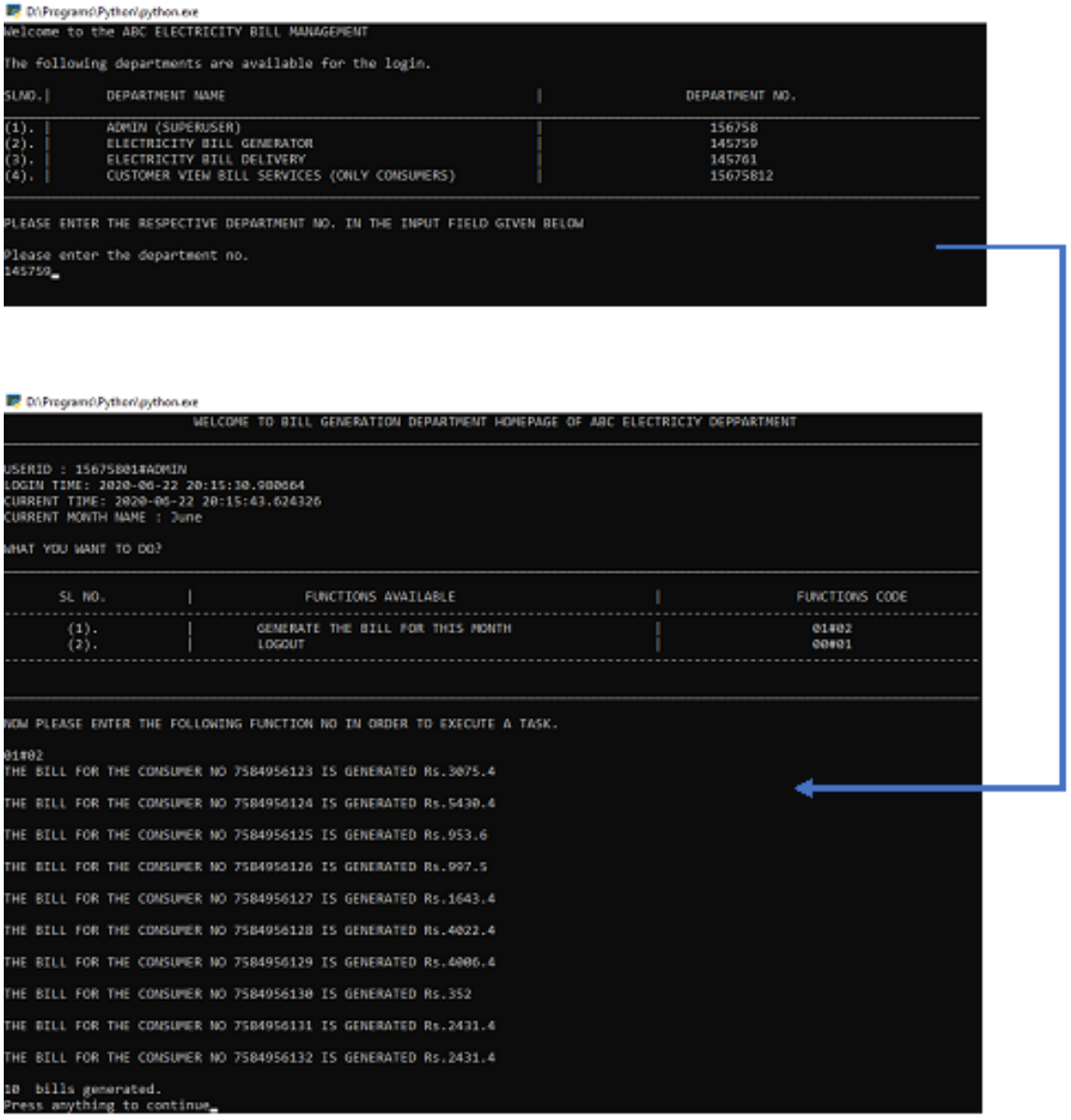
Admin Console Screen Shots



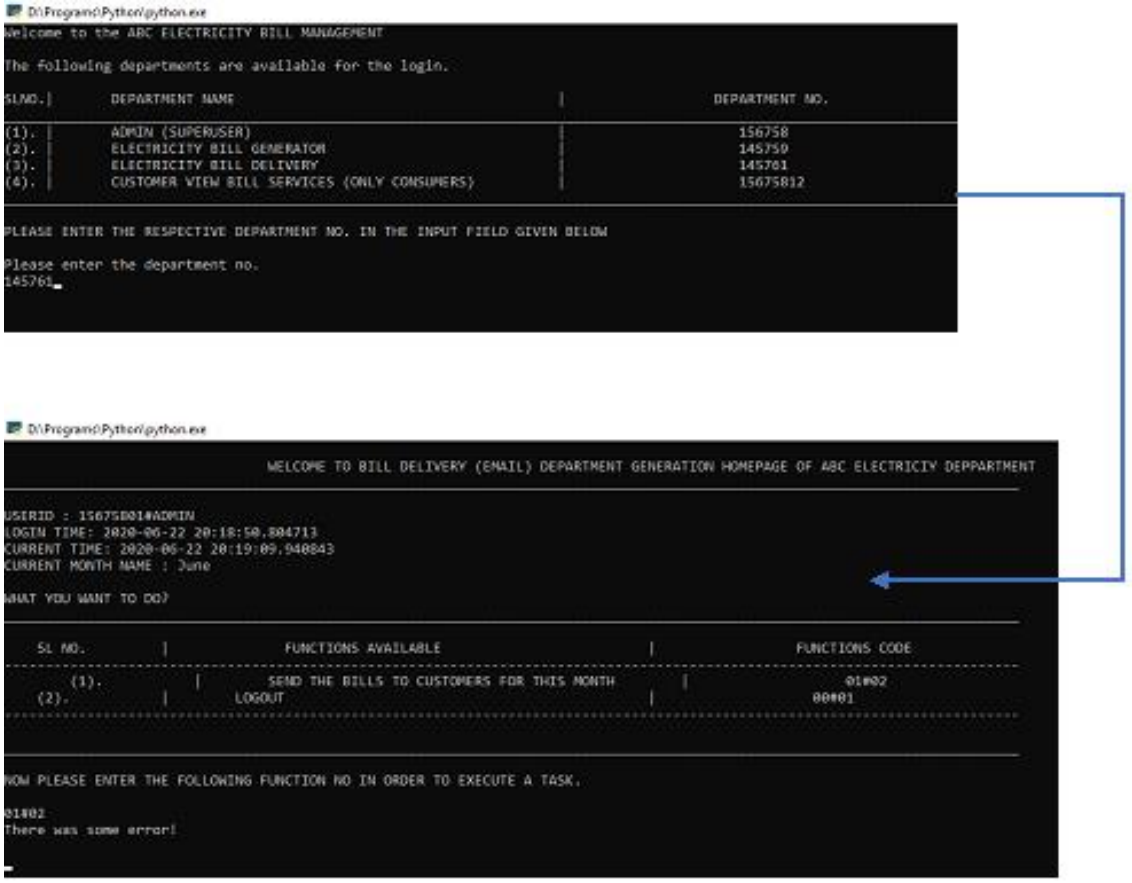
Admin Console Screen Shots (Cont'd)

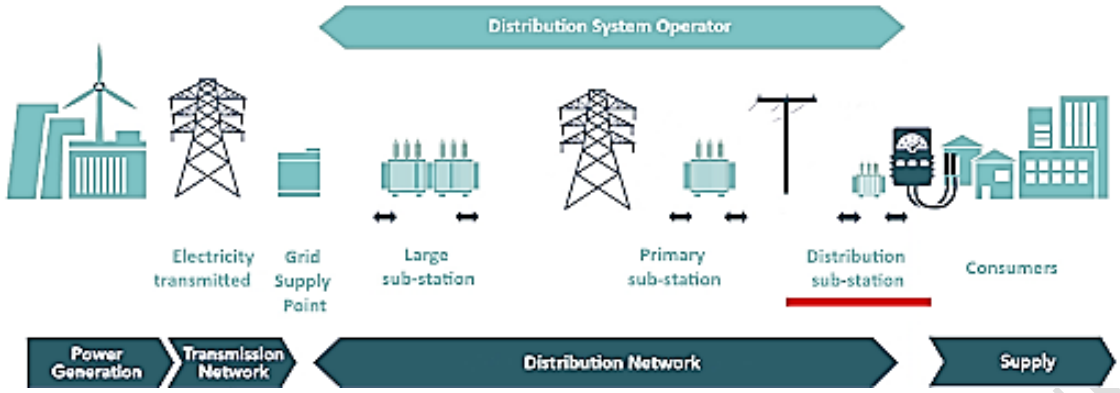


Bill Generation Console Screenshot



Bill – Emailing Console Screenshot





Dhruva Shaw dhruvashaw@gmail.com

Ayush Saha madhabisaha@gmail.com

Smyan Kotkar smyankotkar123@gmail.com