Dharmsinh Desai University, Nadiad

Faculty of Technology

Department of Computer Engineering

B. Tech. CE Semester – IV

Subject: System Design Practice

Project title: Online Electricity Billing System

By:

1)Yash Shah, Roll no: CE122, Id: 19CEUEG051

2) Utsav Suthar, Roll no: CE128, Id: 19CEUBS025

Guided by: Prof. Brijesh S. Bhatt

Prof. Pinkal Chauhan

Prof. Jigar Pandya

DHARMSINH DESAI UNIVERSITY

NADIAD-387001, GUJARAT

CERTIFICATE

This is to certify that the project entitled as "**Online Electricity Billing System**" is a bonafide report of the work carried out by

1) **Mr. Yash Shah**, Student ID No: **19CEUEG051**
2) **Mr. Utsav Suthar**, Student ID No: **19CEUBS025**

of Department of Computer Engineering, semester IV, under the guidance and supervision of Prof. Brijesh S. Bhatt for the subject System Design Practice during the academic year 2020-2021.

Project Guide	Head of the Department
Assistance Professor	Dr. C.K. Bhensdadia
Prof. Brijesh S. Bhatt	Prof. & Head,
Department of Computer	Department of Computer
Engineering,	Engineering,
Faculty of Technology,	Faculty of Technology,
Dharmsinh Desai University,	Dharmsinh Desai University,
Nadiad	Nadiad
Date:30/3/2021	Date:30/3/2021

Contents:

1. Abstract 4

2. Introduction.....5

3. Software Requirement Specifications 6

4. Design

I) Use Case diagram..... 13

II) Class diagram.....…… 14

III) Sequence diagram 15

IV) Activity diagram..... 17

V) Data Flow diagram 19

VI) Structured Chart 22

5. Implementation Detail

I) Modules..... 23

II) Major Functionality 24

6. Work-Flow/Layouts.....27

7. Conclusion 30

8. Limitation and Future extension..... 31

9. Bibliography 32

Abstract

The electricity meters are installed on consumer's place and the power consumption information is collected by meter-reader. Consumers will stand in long queue to pay bill, it is quite embarrassing and time wasting. Nowadays there are so many software for bill payment and they are heavily used in abroad for regular bill payment. This project undertakes the effort of users to keep record of their bill and to pay bill.

Introduction

Online Electricity Billing System has mainly four users Administration, consumer of electricity, Electricity Department, UPI Service Provider.

There is a login page which allows registered used to enter in the system. An administrator has permission to add new costumer's records. Only the administration can allow to saw the details of each costumer due to privacy issue. Administration generate bill via entering the amount of power consumed by consumers.

Electricity consumers can check their own details. They can check their bill, so they have idea about how much amount to pay. They can check their bill of any month of the current year. They can pay bill via Paytm, BHIM, UPI, Credit card, Debit card. After selecting appropriate method of payment, new web page will be open where user should enter their details according to selection of method of payment. After bill payment done successfully "Payment Done" prompt will arrive from system.

There will be some utilities like calculator to calculate amount to pay according to consumed power. There is search button to search bill of any month of the current year.

Technologies/tools used:

- Programming Languages Used: Python
- Framework Used: Django Framework
- Database: phpMyAdmin

Software Requirement Specification:

Online Electricity Billing System

1 Login Management: -

R.1.1 Administration login: -

Description: -

An administration has unique id and password for logging in to the software. If entered id and password by administration is correct then system allowed them to enter. Now administrator can add, update and delete consumer's details.

Input: - administration id and password

Output: -Conformation of login into the system

Process: - It will check and match the data of administration id and password if not match the print message "invalid ", Otherwise it will continue process.

Next: -Redirecting to the Management Page.

R.1.2 Customer's login: -

Description: -

Every customer has their own customer id provided by administration. If entered costumer id and password by consumer is correct then system allowed them to enter.

Input: -Costumer id and password

Output: - Conformation of login into the system

Process: - It will check and match the data of Customer's id and password if not match the print message "invalid ", Otherwise it will continue process

Next: - Home Page of System.

R.1.3 Log Out: -

Description: - There is a log out option provided to log out from particular account.

Input: - User Selection.

Process: - Conformation for logout.

Next: - Login Page.

2. Customer Management: -

→ it will manage by administration.

R.2.1 Register Customer: -

Description: -

To register new customer, details of him/her such as name, address, email id, password will be taken from user. This will store in database and unique customer id is provided to user.

Input: -Customers details

Output: -Conformation message.

Next: - log in page.

R.2.2 Update Customer Detail: -

Description: -

To update details of any customer (details like address, email id). Customer id and new details will be asked to user and those new details will update in database on the basis of customer id.

Input: -Customer id and new detail to update

Process: - check customer by conforming OTP.

Output: - Conformation of changing details.

R.2.3 Delete Customer: -

Description: -

There is option for delete customer.

Input: - Customer id and password for conform the user.

Process: - fetch the data and delete it.

Output: - Conformation message.

Next: - Redirect to login page.

3. Account Management: -

➔ There must be login with system (as customer or administration).

R.3.1 View Details: -

Description: -

There is an option provided to user for showing their details stored in database. Administration has facility to watch every customer's details.

Input: - User selection.

Process: - fetch the data in database and display.

Output: -Details will be shown.

R.3.2 Change Password: -

Description: -

There is an option for changing password for both customers and administration. Though user will be logged in for changing password but he/she need to re-enter old password of their account for security purpose. If entered old password is correct then system asked for new password, after entering it re-enter new password.
If both new password and re-enter new password will match, then that account's password will be changed otherwise "Passwords does not match" message will be arrived.

Input: -Old password, New password, Re-enter new password

Process: - check customer by conforming OTP.

Output: -Conformation of password changed or not

4. Bill Management: -

R.4.1 Generate Bill: -

Description: -

There is an option called 'view bill'. In which user can view his/her electricity bill which she/he will pay. There is a search option to view bill of any month in current year so user need to provide month name to view that months bill. If entered month is greater than or equal to current month then "Bill not found!!" message will arrive.

Input: -User selection and Month name in provided search bar.

Output: -Generate bill

R.4.1.1 Enter Details:

Description: - In this requirement the customer must have their unique customer number which will provided by electricity department.

Input: - Enter customer number.

Process: - Fetch the data from electricity department.

Output: - Continue the payment process.

R.4.1.2 Automatic Display:

Description: - It will automatic display the name of customer which is linked with customer number.

Process: - Fetch the data from electricity department.

Output: - Display the name.

R.4.2 Upload / Update bill: -

Description: -

This function is provided to only administration. Administration can upload a bill of every customer in customer's 'view bill' portal. If there some mistake in bill, then administration can change it.

Input: - User selection

Output: -Conformation of bill uploaded

5. Payment Management: -

R.5.1 Payment method: -

Description: -

There is an option to pay bill. When user select that option then user has option to choose payment method. There are many payment

methods provided to user like net banking, mobile banking, BHIM or UPI. There is another option for pay through second way is pay through Debit or credit card.

Input: -Select suitable payment method and provide E-payment requirement detail for payment. (User selection)

Process: -Process of banking side and fetch the account details provided by user.

Output: - Print Message "Bill Payment Successfully".

Next: - It will redirect to download or print bill.

R.5.2 Download / Print: -

Description: -There Is option to download or print of paid bill

Input: - User selection

Output: - it will give to customer a copy of bill (soft copy or hard copy).

6. Utility Management: -

R.6.1 Calculator: -

Description: -User has facility of calculator to calculate bill. Click on calculator and calculator will be open.

Input: -User selection

Output: -Calculator will open.

R.6.2 Close option: -

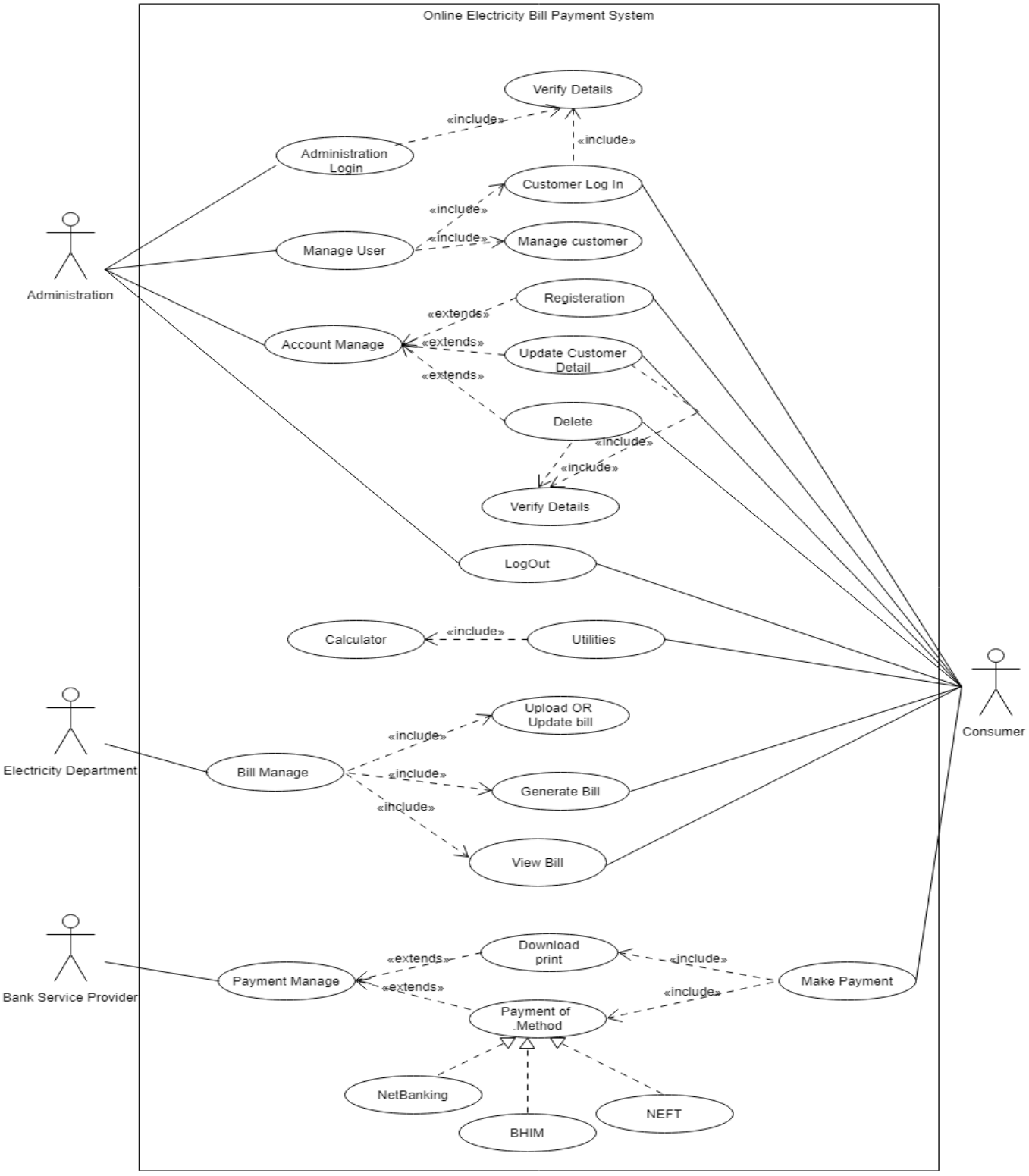
Description: -There is a close option provided to any functionality. So if anything is opened by mistake, then there is an option to close it.

Input: -User selection

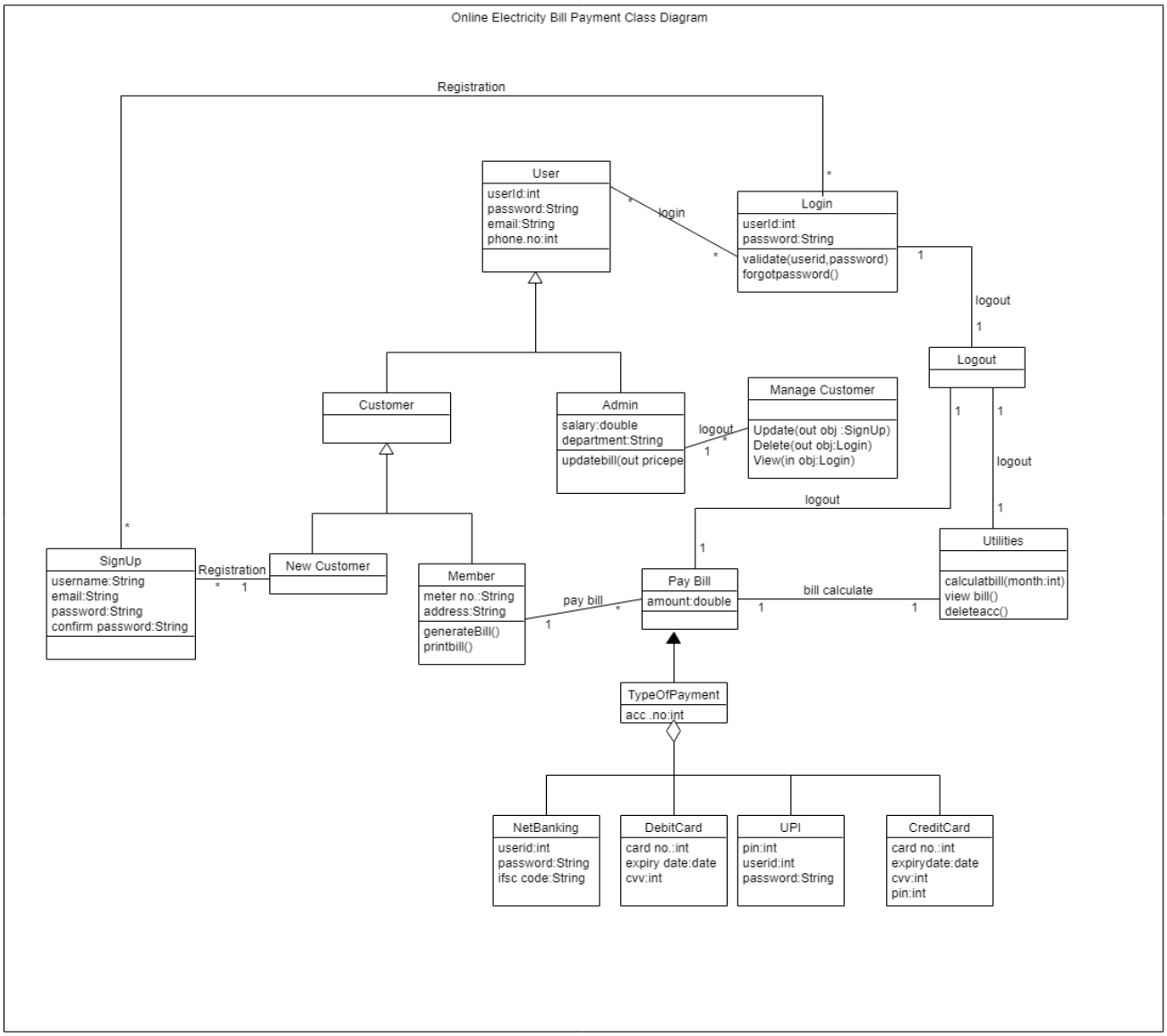
Output: -That functionality will be closed

Design:

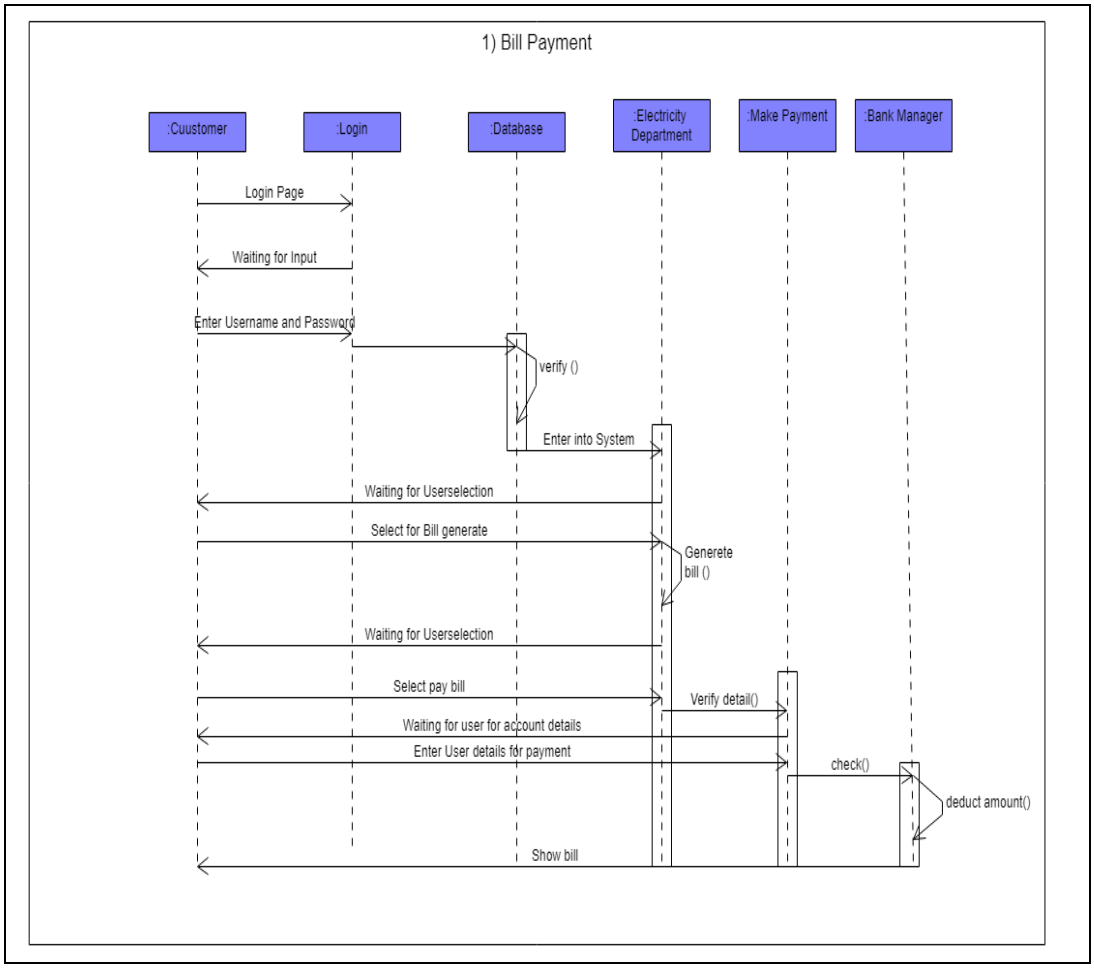
- Use Case diagram

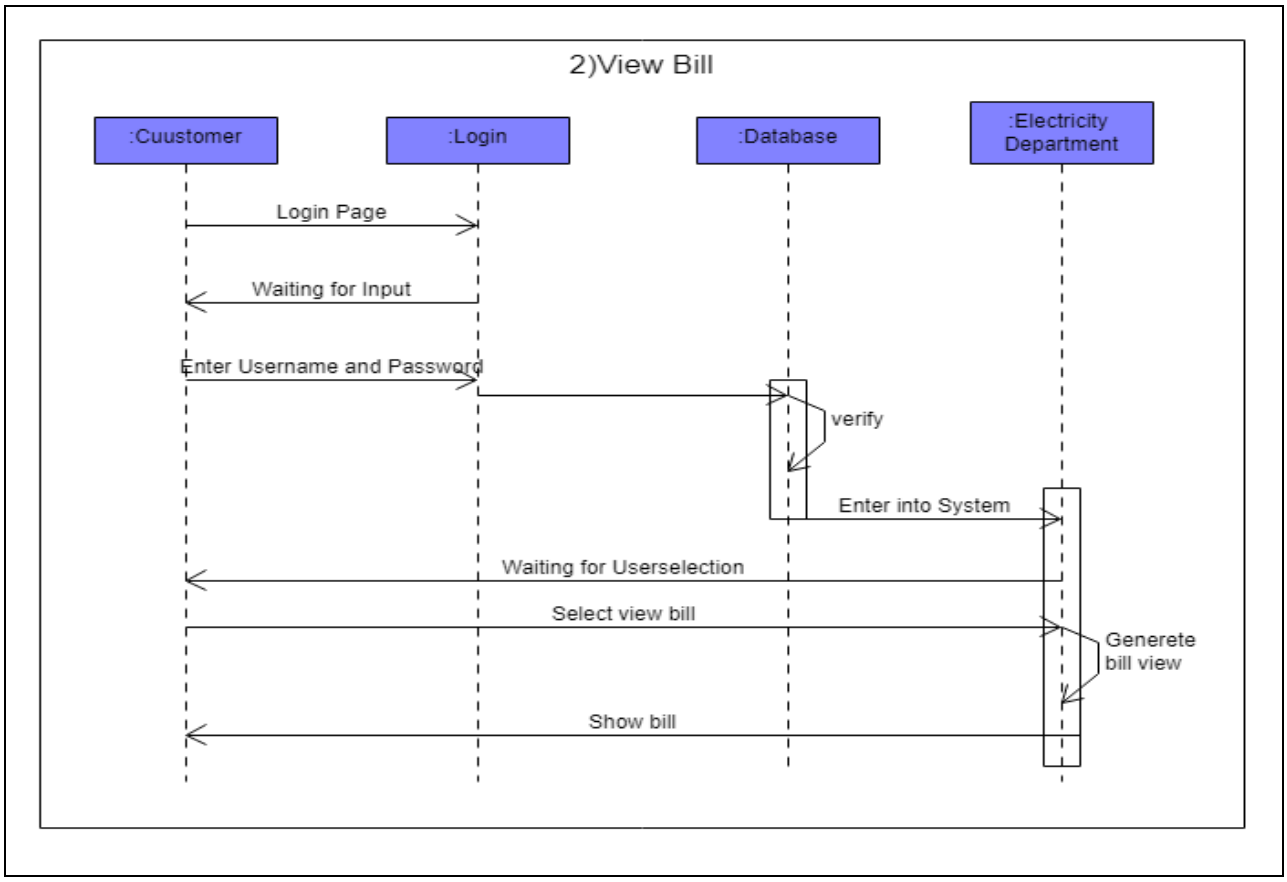


- Class Diagram

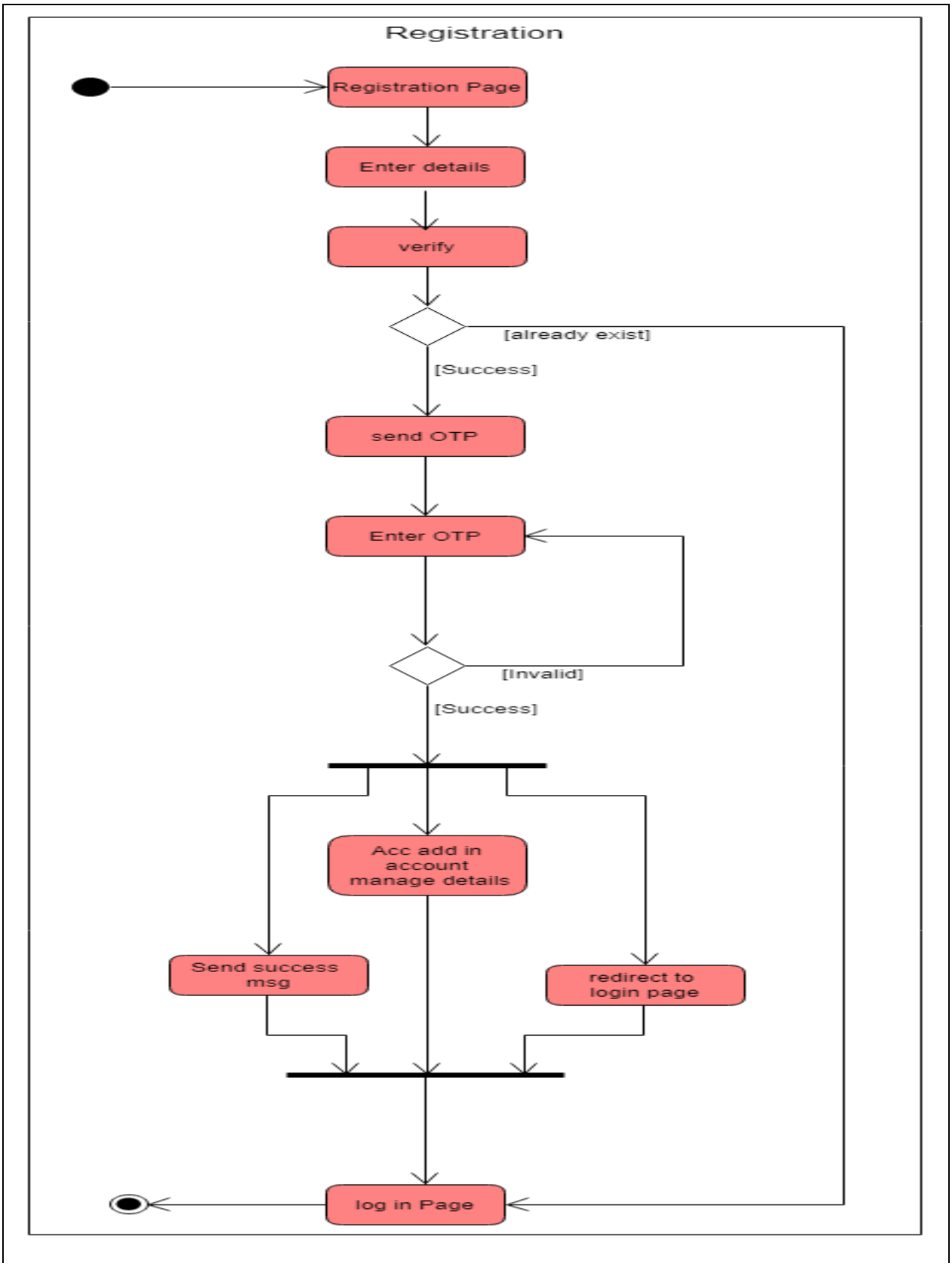


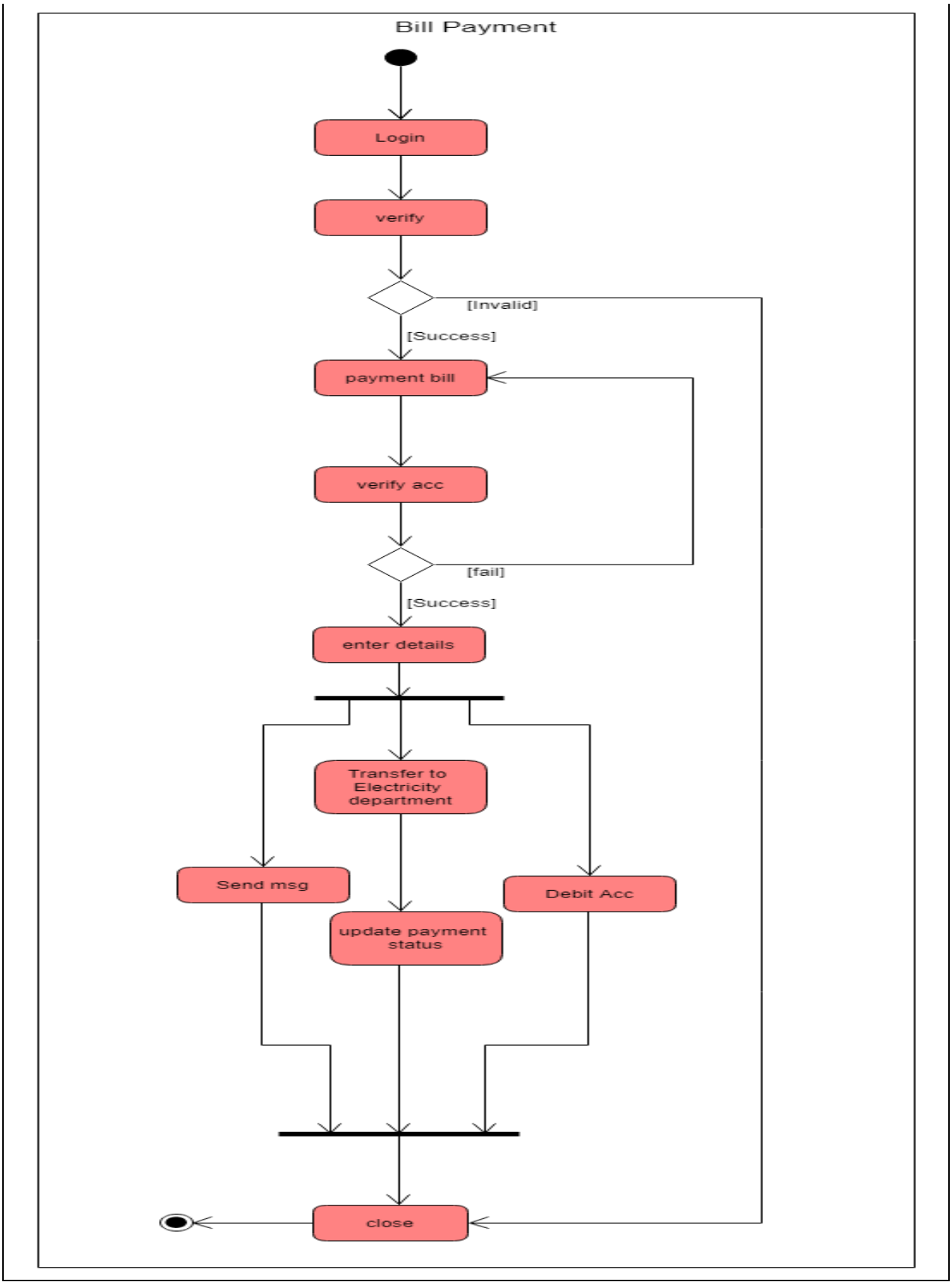
- Sequence Diagram



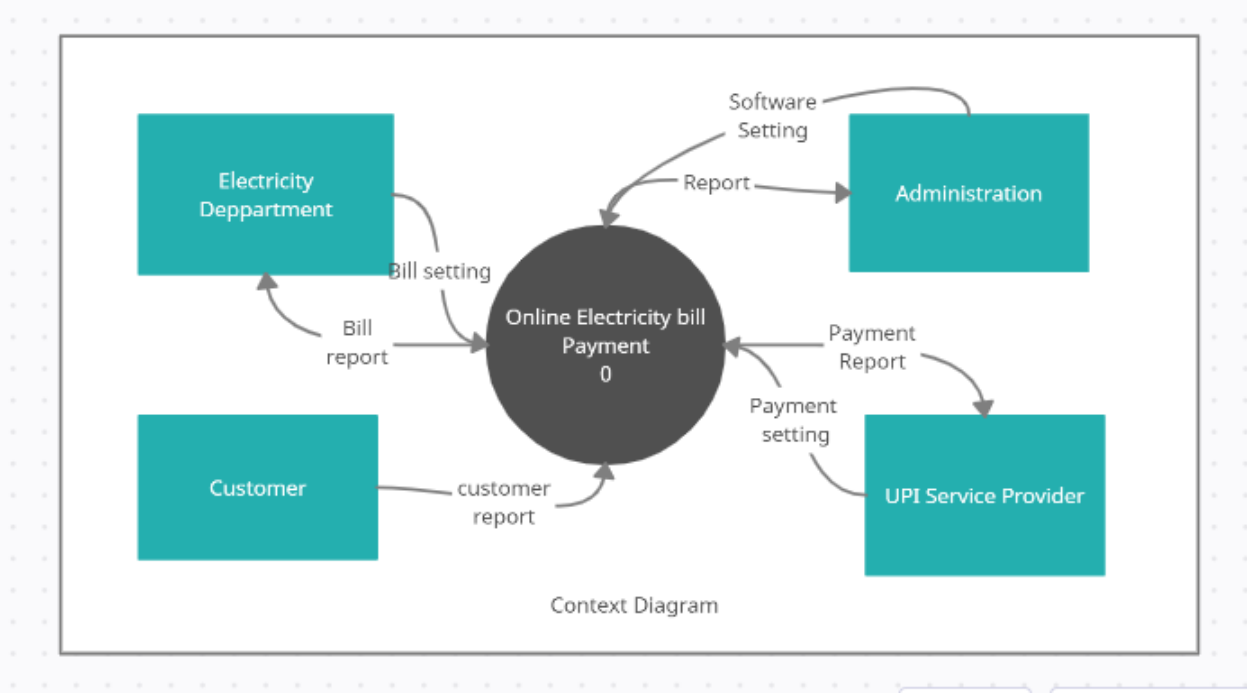

2)View Bill
:Cuustomer
:Login
:Database
:Electricity Department
Login Page
Waiting for Input
Enter Username and Password
verify
Enter into System
Waiting for Userselection
Select view bill
Generete bill view
Show bill

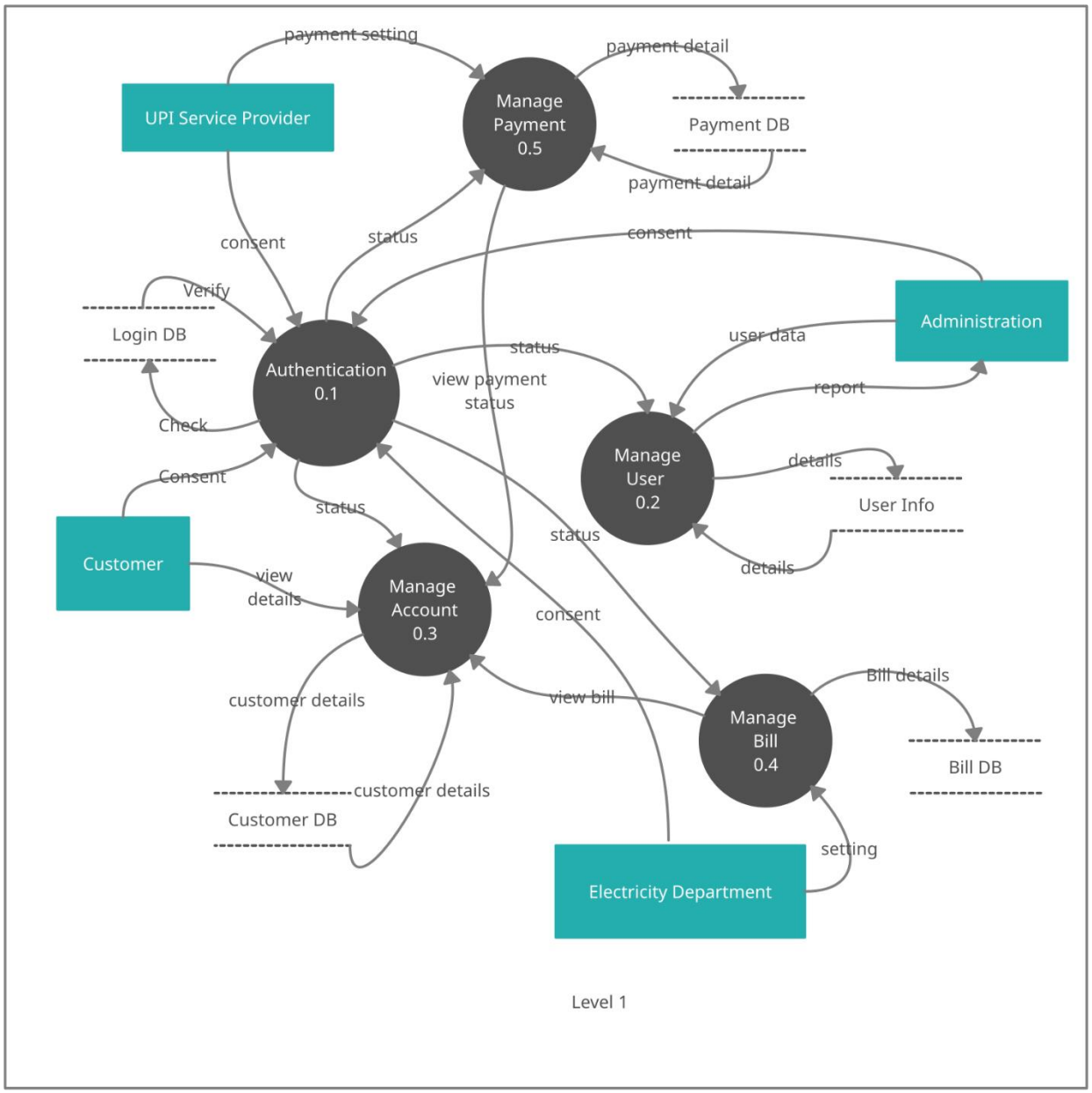
- Activity Diagram



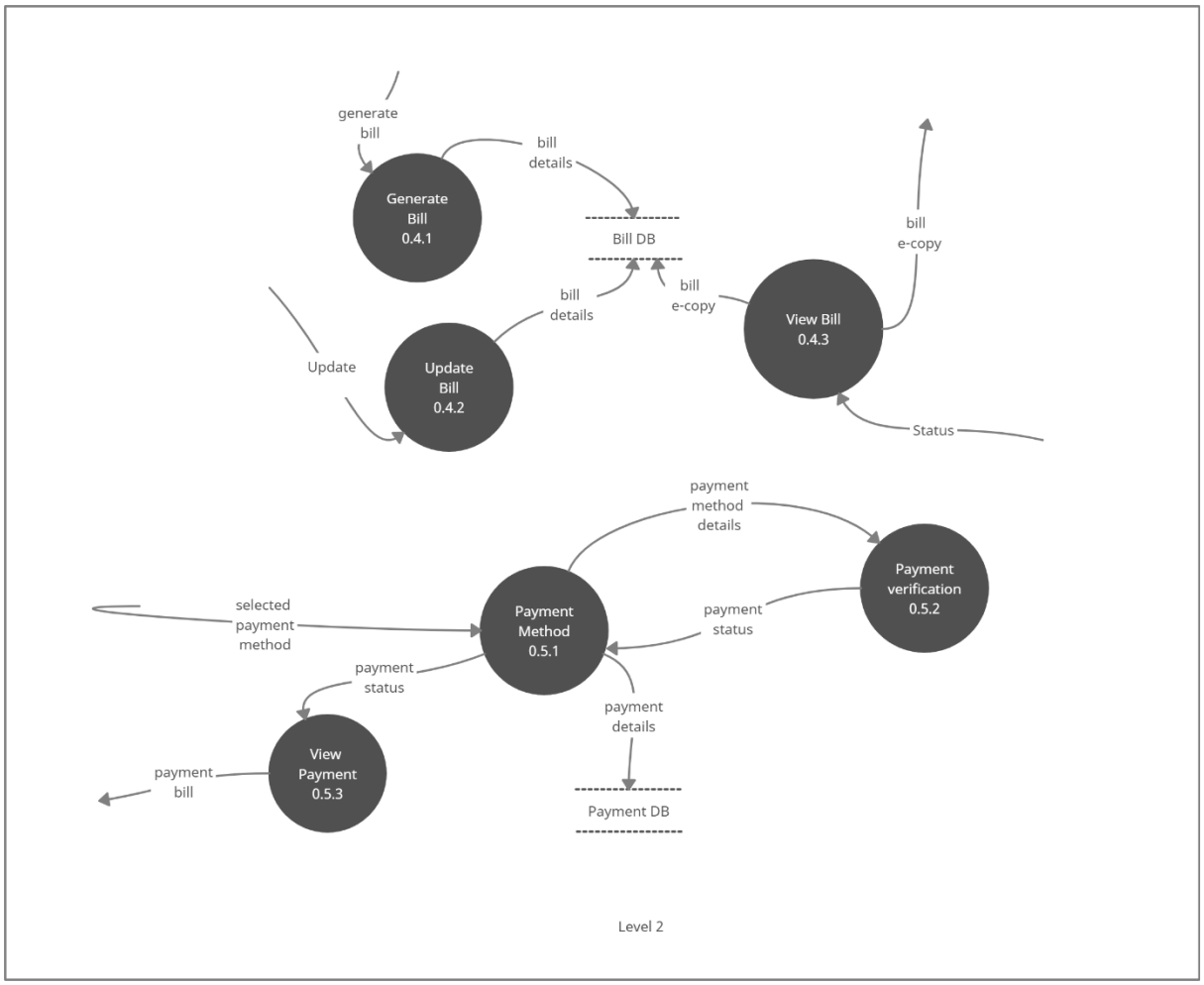

Bill Payment
Login
verify
[Invalid]
[Success]
payment bill
verify acc
[fail]
[Success]
enter details
Transfer to Electricity department
Send msg
Debit Acc
update payment status
close

- Data Flow Diagram

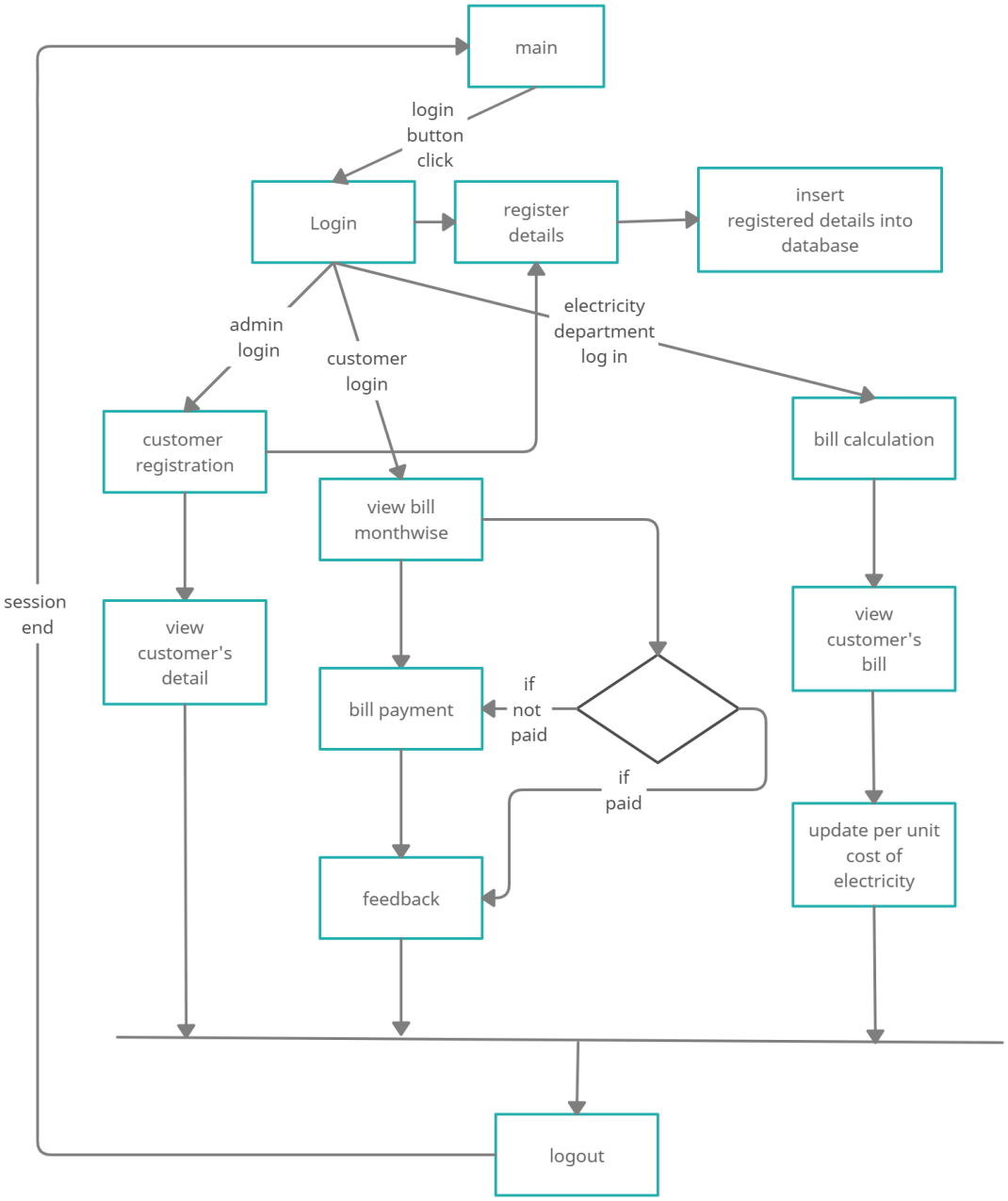



payment setting
payment detail
UPI Service Provider
Manage Payment 0.5
Payment DB
payment detail
consent
status
consent
Verify
Login DB
Authentication 0.1
status
view payment status
user data
Administration
report
Check
Manage User 0.2
details
Consent
User Info
status
Customer
status
details
view details
Manage Account 0.3
consent
Bill details
customer details
view bill
Manage Bill 0.4
Bill DB
customer details
Customer DB
setting
Electricity Department

Level 1


generate bill
bill details
Generate Bill 0.4.1
Bill DB
bill e-copy
bill details
bill e-copy
View Bill 0.4.3
Update
Update Bill 0.4.2
Status
payment method details
Payment verification 0.5.2
selected payment method
Payment Method 0.5.1
payment status
payment status
payment details
View Payment 0.5.3
payment bill
Payment DB
Level 2

- Structured Chart



Implementation Detail:

1. Modules:

In the following section a brief description of each module is given.

Login Management –module:

In Login Management Module user can login as an administrator or as a consumer of electricity. User can perform different tasks according to their login.

Customer Management-module:

In Customer Management Module administration can add new customer, delete existing customer or update existing customer's detail. Consumers can also register themselves.

Account Management-module:

In Account Management Module consumer can view their detail, update their details and also they can change their passwords.

Bill Management-module:

In Bill Management Module consumer can generate their bill according to their bill id. They need to enter power consumed by them.

Payment Management-module:

In Payment Management Module Consumer can pay their bill by UPI, Net Banking, Mobile Banking, Credit/Debit Cart. They need to enter payment details according to their selected payment method. Status of bill changed to paid after successful payment and "payment successful" message arrives.

Utility Management-module:

In Utility Management Module Consumer have facility to calculate their bill amount according to their power consumed units.

2. Major Functions Prototypes

1. Generate Bill:

Generate Bill Function will generate bill of consumer according to his/her bill no., unit of power consumed.

```
def generate_bill(request):
    context = {}
    user = request.user
    if request.method == 'POST':
        bill_no = request.POST['bill_no']
        try:
            unit  = request.POST['unit']
            if float(unit) > 100 :
                amount = float(unit) * 2 * cost_per_unit #float(request.session['cost_per_unit']) #double amount
            else:
                amount = float(unit) * cost_per_unit #float(request.session['cost_per_unit'])
            gb = g( bill_no=bill_no,amount=amount,unit=unit,key= L.objects.get(pk=request.user.id))
            gb.save()
            status = ps(key_id=user,date=gb,status=False)
            status.save()
            messages.success(request, 'Bill Generated Now please pay the bill')
            return redirect('view_bill');
        except:
            messages.success(request, 'Invalid Unit unit can not more than 1250')
            return redirect('view_bill');
    else:
        if user.is_authenticated:
            try:
                bill = g.objects.latest('key')
                bill_no = bill.id+1
            except g.DoesNotExist:
                bill_no = 1
            context = {
                'meter_no' : request.user.meter_no,
                'bill_no'  : bill_no,
                'username' : request.user.username,
            }
            return render(request,'generate_bill.html',context)
        else:
            return redirect('login');
def utility(request):
```

2. Pay Bill:

Pay Bill has four methods to pay (Net Banking, UPI, Mobile Banking, Credit /Debit Card). It will take details of payment according to consumer's selected payment method.

```
def bhim(request):
    context = {}

    user = request.user
    if request.method == 'POST':
        error_message = set([]);
        customer_no = request.POST['customerNo']
        meter_no = request.POST['meterNo']
        bill_no = request.POST['billNo']
        bill_amount = request.POST['billAmount']
        upi_id = request.POST['UPI_ID']
        date    = request.POST['date']

        bill = g.objects.filter(key_id=request.user.id).latest('bill_no')

        if not upi_id:
            error_message.add("Required Field UPI ID")
        elif upi_id.find("@upi") == -1 and upi_id.find("@bhim") == -1:
            error_message.add("Invalid UPI ID")
        if not error_message:
            ins = bhimUPI(customer_no = customer_no, bill_no = bill_no, bill_amount = bill_amount, upi_id = upi_id)
            ins.save()
            if ps.objects.filter(date=bill).exists():
                    p = ps.objects.get(date=bill)
                    p.status = True
                    p.detail = "Mode = Bhim \nUpi id ="+upi_id
                    p.save(update_fields=["status","detail"])
            else:
                status = ps(key_id=user,date=bill,status=True,detail="Mode = Bhim \nUpi id ="+upi_id)
                status.save()

            messages.success(request, 'Payment Successful')
            return redirect('login_home');
        else:
```

```
                'meter_no': request.user.meter_no,
                'bill_no':bill_no,
                'amount':bill_amount,
                'error_message':error_message,
                'date' : date,
                }
                return render(request,'bhim.html',context)
    else:
        if user.is_authenticated:
            customer_no = request.user.id
            meter_no = request.user.meter_no
            try:
                bill = g.objects.filter(key_id=customer_no).latest('bill_no')
                bill_no = bill.bill_no
                p = ps.objects.get(date=bill)
                if p.status:
                    messages.success(request, 'Payment is already Done')
                    return redirect('login_home')
                amount = bill.amount
                date = bill.month
            except g.DoesNotExist:
                messages.success(request, 'There are no bill generated so first generate bill')
                return redirect('generate_bill')
            amount = bill.amount
            context = {
                'customer_no': request.user.id,
                'meter_no': request.user.meter_no,
                'bill_no':bill_no,
                'amount':amount,
                'date' : date,
            }
            return render(request,'bhim.html',context)
        else:
            return redirect('login');
```

Work Flow/Layouts:

➔ Below is given files use as back end purpose in application:
- Online_Electricity_Billing_System\Billing_System\models.py
- Online_Electricity_Billing_System\Login\models.py
- Online_Electricity_Billing_System\payment\models.py

➔ Below is given files to manage urls of application:
- Online_Electricity_Billing_System\Online_Electricity_Billing_System\urls.py
- Online_Electricity_Billing_System\pdf\urls.py
- Online_Electricity_Billing_System\Billing_System\urls.py
- Online_Electricity_Billing_System\payment\urls.py

➔ Below is given files to manage front end purpose in application:
- Online_Electricity_Billing_System\Billing_System\templates\base.html
  \home.html
  \index.html
  \bhim.html
  \mobilebanking.html
  \netbanking.html
  \payment.html
  \register.html
  \utility.html
  \view_bill.html

\calculate.html

\check_status.html

\cradit.html

\display.html

\generate_bill.html

- Online_Electricity_Billing_System\Billing_System\templates\Billing_System\

\account.html
\base.html
\login.html

- Online_Electricity_Billing_System\Billing_System\templates\Billing_System\snippets\footer.html

\header.html

- Online_Electricity_Billing_System\Billing_System\templates\pdf\

\base.html

\main.html

\pdf1.html

\pdf2.html

\view.html

- Online_Electricity_Billing_System\Billing_System\templates\registration

\password_change.html
\password_change_done.html

\password_reset_complete.html
\password\reset_done.html
\password_reset_email.html
\password_reset_form.html
\password_reset_subject.html

Conclusion

Hence-forth in this project we have successfully implemented the Admin-side & Consumer-side functionality, Admin will add the new consumer or delete consumer or update the records of existing consumer. Electricity department can change unit per cost and they have control of billing related stuff. UPI service provider has details of consumer's payment detail. Now Consumer can generate bill and can pay that generated bill.

Limitations

- This project is implemented in small level. I think this project is not implemented in large level like (UGVCL, PGVCL etc.)
- This project has no relation with bank accounts of consumer, so no money has been included in this project. So there is not actual process for bill payment is included but dummy message is included for successful bill payment

Future Extensions

To take over the limitations we are planning this future extension in our system.

- Include PayPal or any other software for payment process.
- Include more dynamic feature in project.

Bibliography

References/resources used for developing project:

- Basic Django tutorial:

  - https://docs.djangoproject.com/en/3.1/
  - https://youtu.be/OTmQOjsl0eg

- For Pdf Download and view:

  - https://www.javatpoint.com/django-pdf-output
  - https://www.youtube.com/watch?v=J3MuH6xaDjI

Online Database:

- http://localhost/phpmyadmin/db_structure.php?server=1&db=project_db