Programming Applications and Frameworks

IT3030

3rd Year, 1st Semester

Proposed System for ElectroGrid (EG) to Develop a High Scalable Online Platform

Group Number: WE/REG/156

Batch: Y3S1.WD.IT.2.1

Group Members:

Registration Number	Name
IT20127428	P.M Meddaduwage
IT20142346	Sandeepa U.H.A. N
IT20187132	G. A. Asahara
IT20194758	Dissanayaka D.M.L.D

Contents

01)Work Distribution

Member	Web Service	Functions
IT20127428 P.M Meddaduwage	Payment handling Service	• Add payment details • Update necessary payment details • Delete payment details • Retrieve /View payment details • Mange payment validations.
IT20142346 Sandeepa U.H.A. N	Consumer support service	• Add the consumer details • Update consumer • Delete consumers • View consumer • Resolve the issues
IT20187132 G. A. Asahara	Bill handling service	• Create the billing details • Update the bill • Delete the unwanted bills • View the bills • Display the bill
IT20194758 Dissanayaka D.M.L.D	Login authentication and user handling service	• User login • User authentication • Update the user profile • Delete users • View the user list

02) Introduction

- ElectroGrid is a renowned company through which they maintain the power grid of the country, the company has a system through which they monitor the power consumption of the users, they generate the bill, and it is send automatically to the users and not only that they accept online payments of the users of their system, but due to the prevailing situation they have faced a problem regarding the scalability of the system. So, through this research we are bring out some solutions, the main solutions are to build up an online platform where it would be more efficient the users, the team has come up with four main sections that are needed to be developed in this system, namely

02.1 Payment handling

- Payment handling is one of the most important part of this system this function is connected to the system in many ways and it helps the users to do their payment in a very easy way just through adding their payment details to the system, and not only that the system gives them the ability to update ,delete and view the necessary and the specific data of the payments, this function is handled by the admin of the system where the authentication of the payment is done, through this function it makes the work of the company more efficient , it makes the system more user friendly where the users customers do not need to do the payment manually but the admin can add up their payment details.

02.02 Consumer support services

- This function is a must in a developing system and with the system scope being huge and developing each day it is a must to keep a record on the system users and about how they feel about it so here the consumer manger is the person who is involved regarding this function, all the details regarding the consumer problems are recorded in the system by adding a description to the system database. Through this function the web application or the company can have a good idea about its customers and develop the system according to the needs of the consumers.

02.03 Bill handling service

- The above function in one of the main functions in the proposed system all the bill details are stored by the system , this function is handled by the finance manager, the calculation of the units used in a month by a consumer is the main part that calculated here the bill displays all the information regarding the payment that need to be done, the bill is printable and also can be sent as digital receipt to the customer , through which it makes the system more efficient and also it makes the system more user friendly.

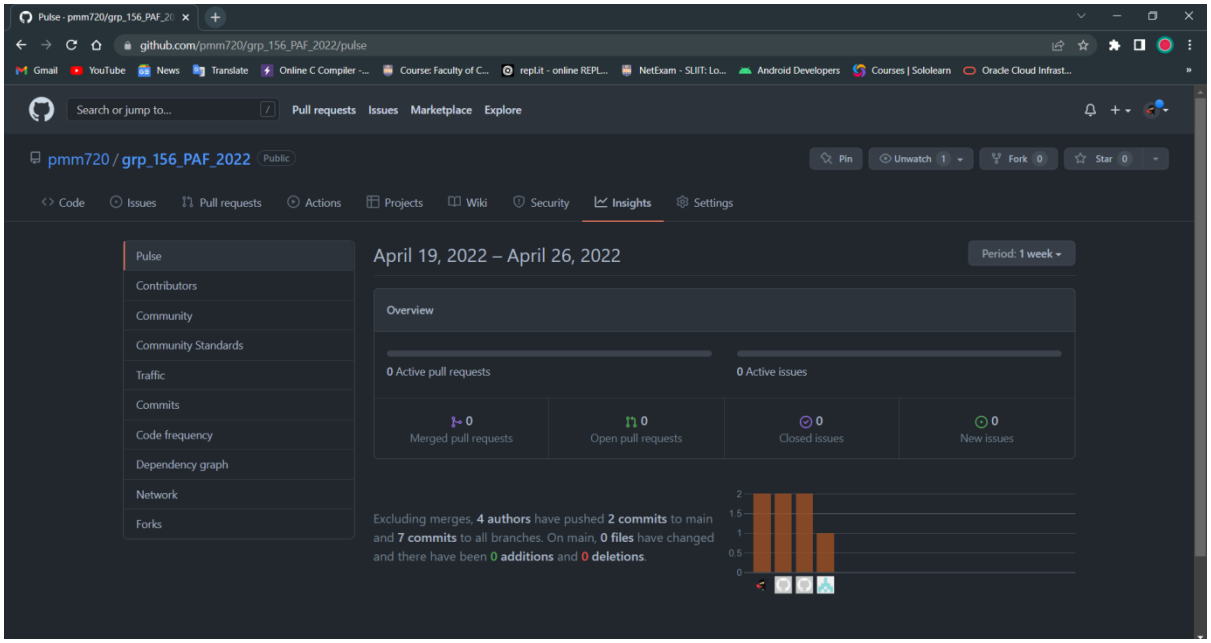
02.04 Login authentication and user handling service

- All the user authentications and other user related services are done through this function, admins, consumers mangers, finance managers and the other users who use the web application is handled through this, all the users that log into the system are registered through and authentication process and they will be added to the system. All the user handling parts are also done through this function, like creating user profiles, updating them, and having a proper database. this is also one of the main functions in the proposed system.

03) GitHub Details

- https://github.com/pmm720/grp_156_PAF_2022.git

GIT commit



04) SE Methodologies

04.1 Introduction to Agile Methodology

- Agile methodologies are product development methodologies that are consistent with the Agile Manifesto's values and principles for software development. Agile techniques attempt to produce the proper product through small cross-functional self-organizing teams that supply small pieces of functionality on a regular basis, allowing for frequent customer input and course correction as needed. This is a method of project management that divides a project into numerous phases. Every phase includes continual development, and the development cycle includes planning, implementing, executing, and assessing. As a result, we will be able to uncover software defects sooner rather than later. A functioning system was developed quickly using this process.

04.2 Uses of Agile Methodology

- Superior quality product
- Customer satisfaction
- Better control
- Improved project predictability
- Flexibility
- Convenient

04.3 Advantages of Agile Methodology

- Accordance changes at any point
- Effective for dynamic environments
- Success can be achieved through freedom
- Rapid value
- Early identification of bugs and errors

04.4 Disadvantages of Agile Methodology

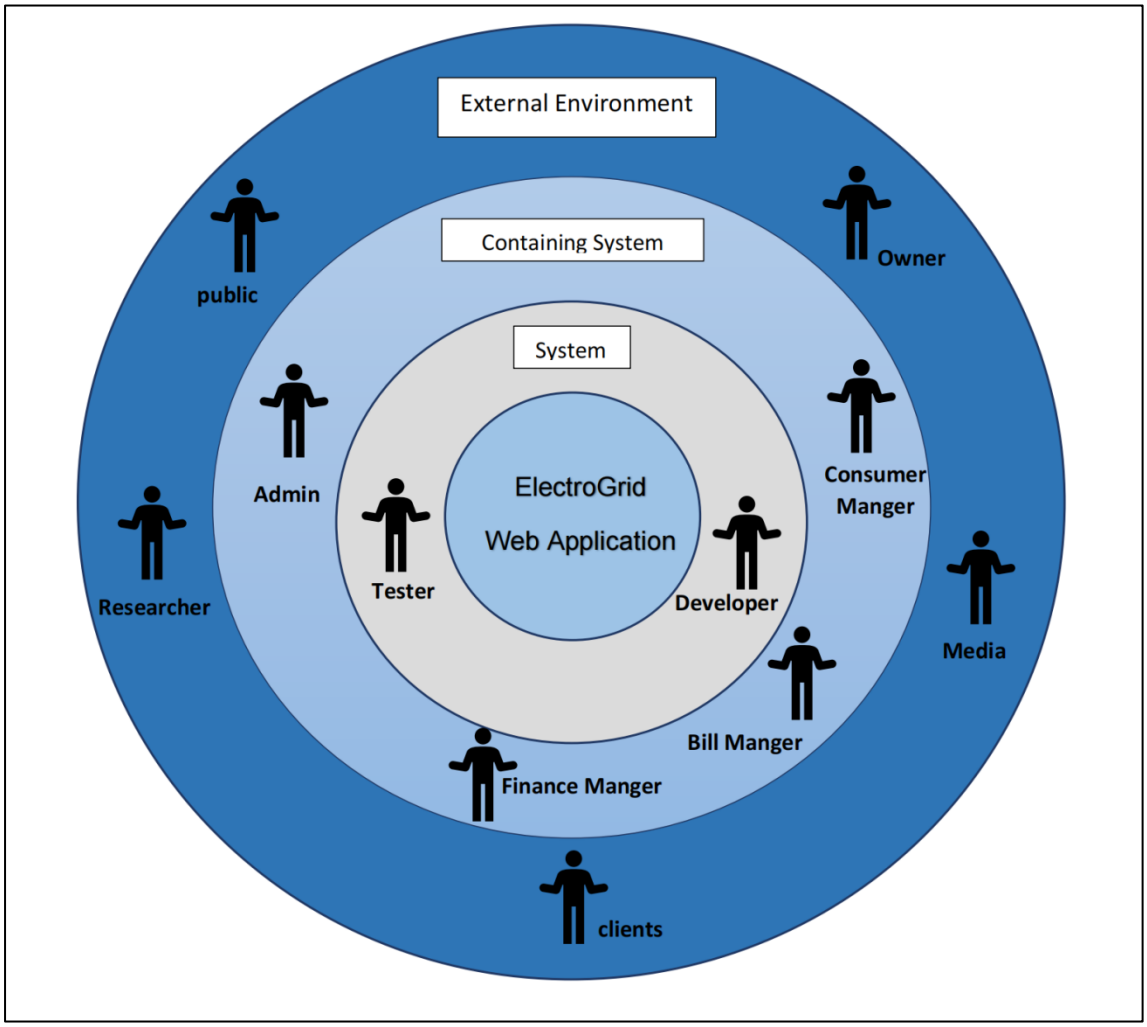
- Not used in developing large and critical systems
- Costly for stable development
- Difficult to predict the efforts (cost, time)
- Difficulty in measuring the progress

05) <u>Time schedule (Gantt chart)</u>

	Week 01	Week 02	Week 03	Week 04	Week 05	Week 06	Week 07	Week 08	Week 09	Week 10
01-Gathering information/ requirements										
02-Intsallation of the software										
03- Designing the database										
04-Iplementaion										
05-Feedback										
06-Interatgration and testing										
07-finalizing the system and making the reports										

06) Requirements

06.01 Stakeholder analysis (Onion Diagram)



06.02 Technical Requirements

- Technical requirements define the technical features and difficulties that must be addressed for the project or product to run smoothly. Performance-related concerns, software reliability, and accessibility are examples of technological considerations.
    1. User authentication and authorization
    2. Privacy – all the data details should be encrypted

06.02 Functional Requirements

- Payment handling
- Add payment details
- Update necessary payment details

- Delete payment details
- Retrieve /View payment details
- Mange payment validations.

- Consumer Support service
- Add the consumer details
- Update consumer
- Delete consumers
- View consumer
- Resolve the issues

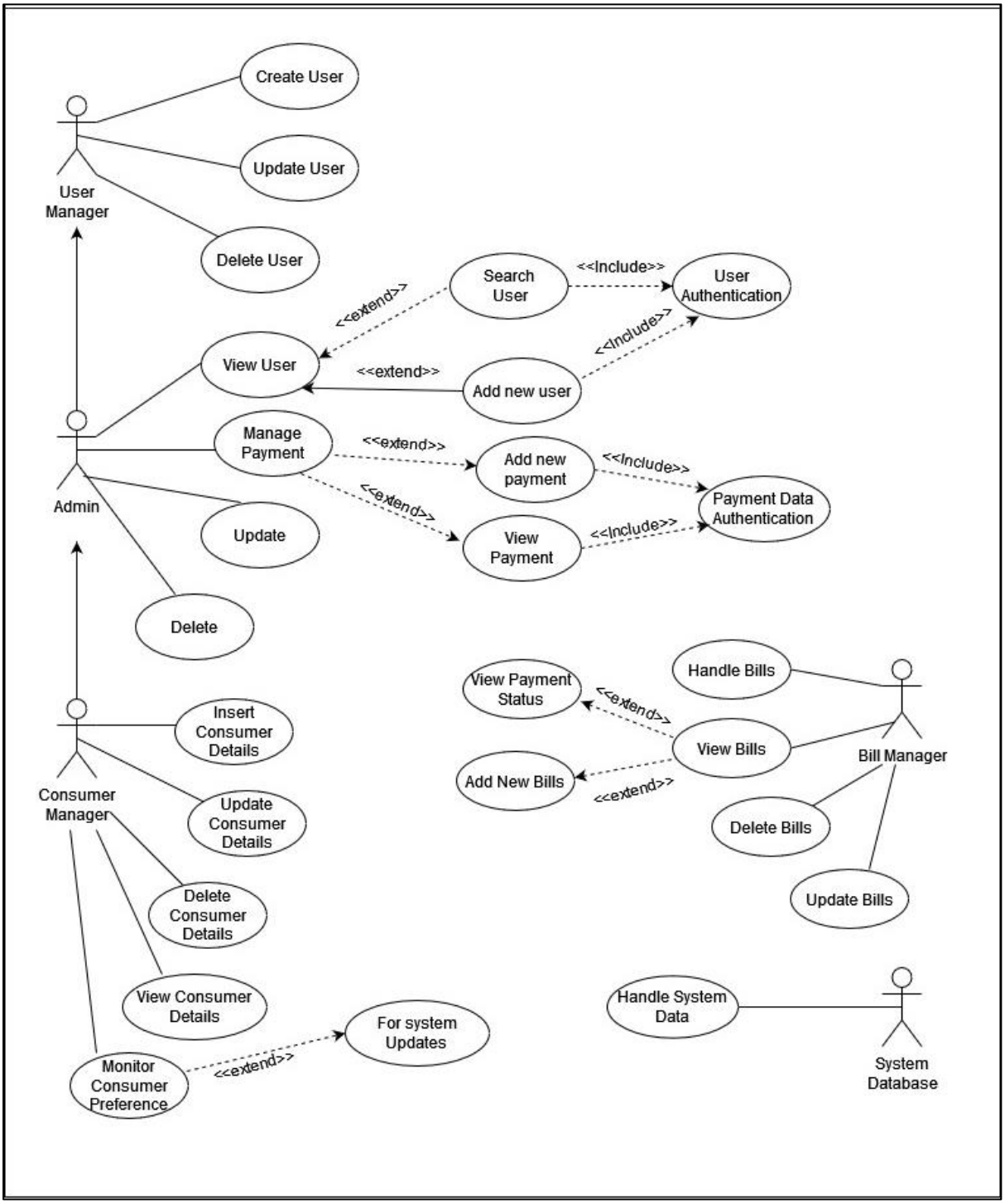
- Bill Handling
- Create the billing details
- Update the bill
- Delete the unwanted bills
- View the bills
- Display the bill

- Login authentication and User handling
- User login
- User authentication
- Update the user profile
- Delete users
- View the user list

06.03 Functional Requirements

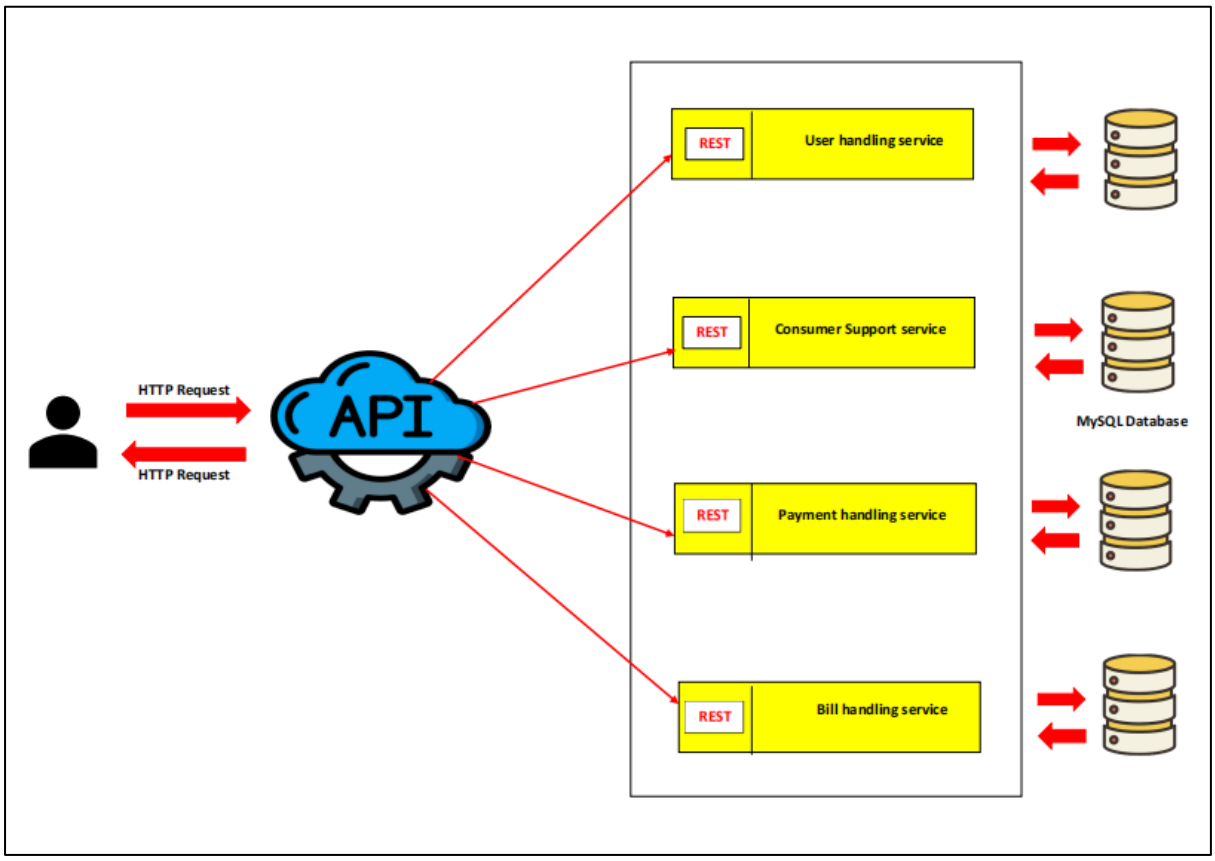
1. Authentication and Authorization-the user must be validated and verified
2. Performance- determines the wait time and load time of the application
3. Productivity- allows the system users to be more productive
4. Reliability- averages the time for applications and service downfalls

06.04 Requirements Modelling (use case diagram)



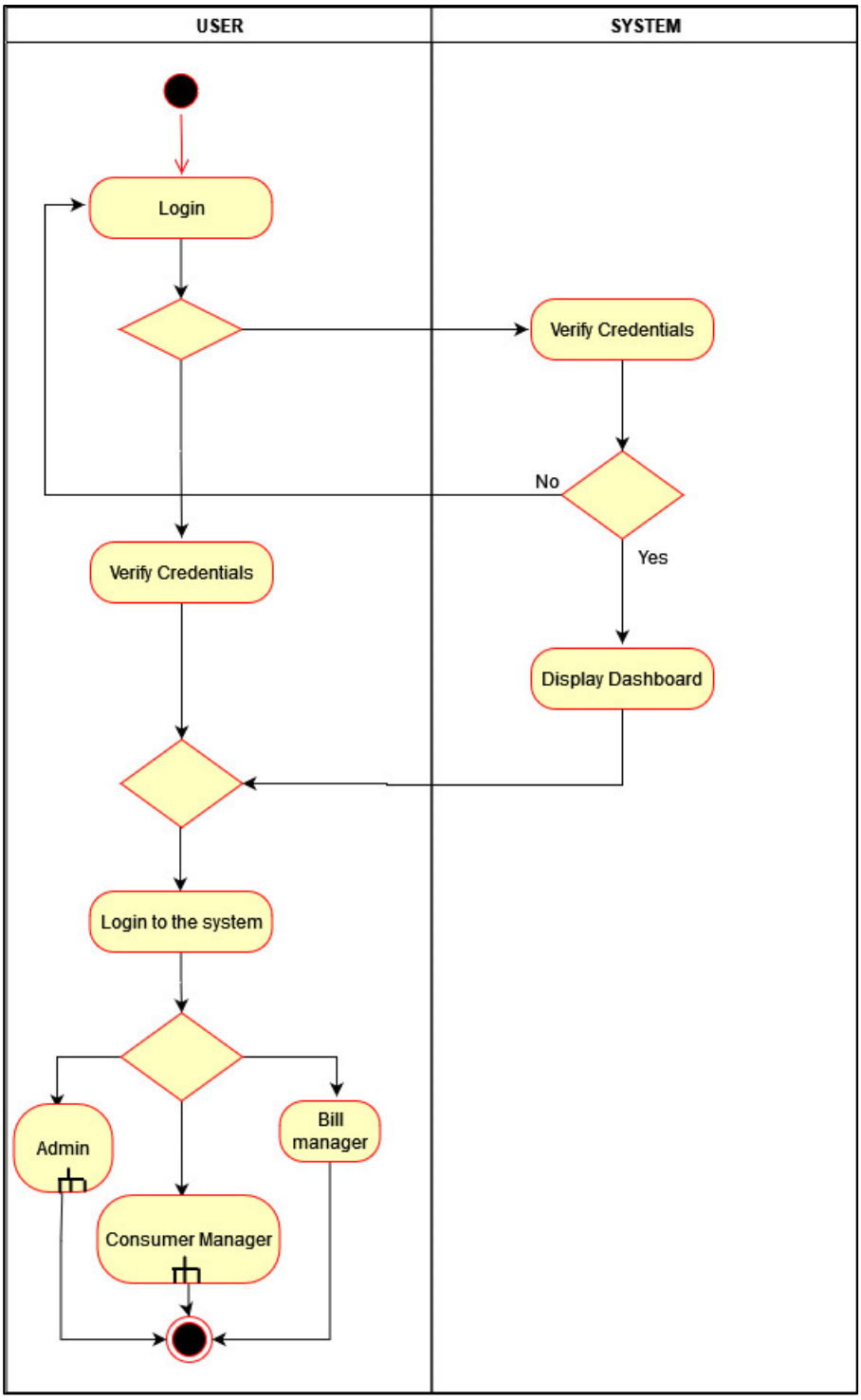
07) System's Overall Design

07.01 Overall Architecture

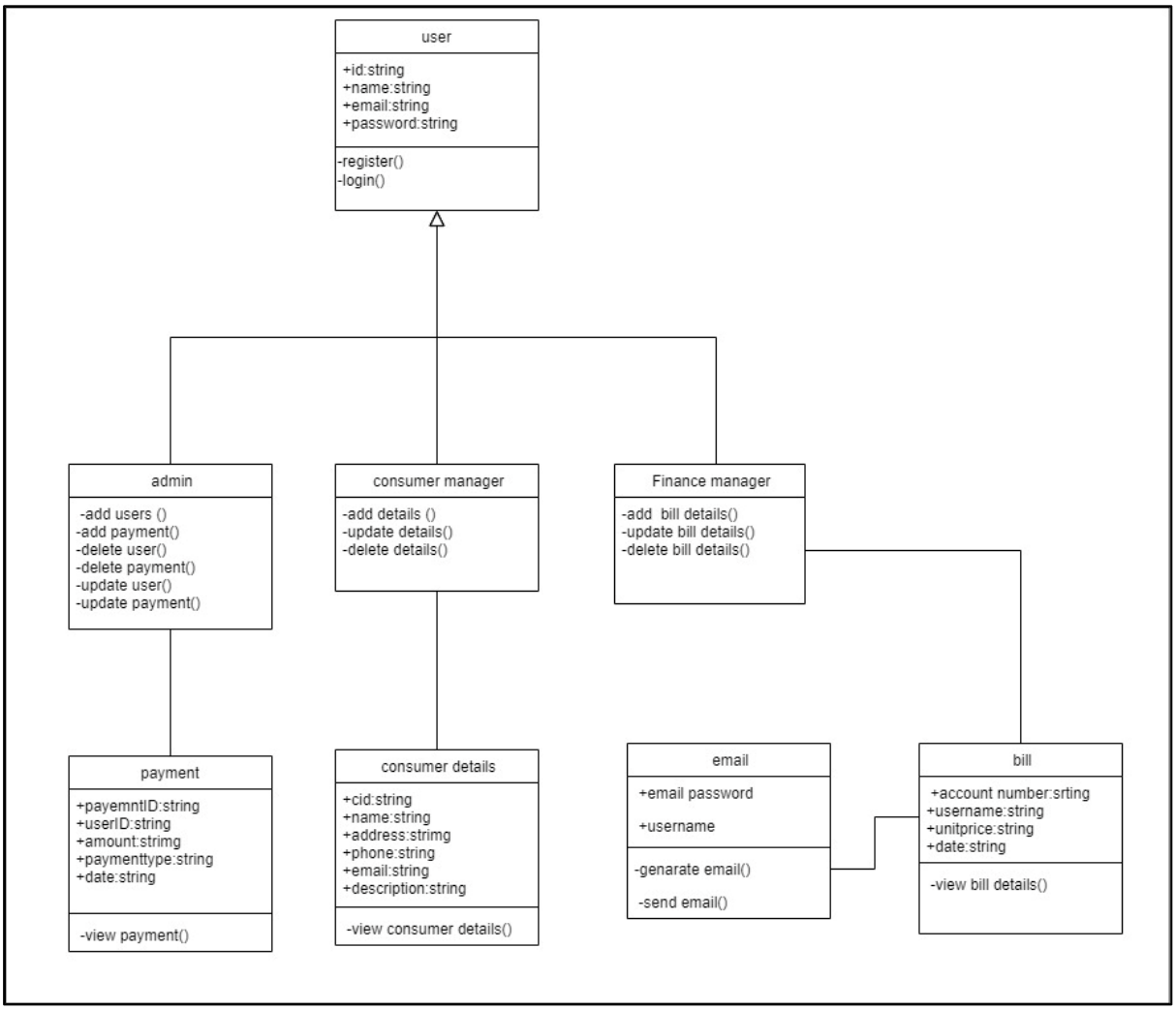


- The proposed system for Electro grid is designed based on the new technology where the system gives many options to its users, according to the overall system architecture, there are four main services that the system is mainly focused on the above diagram describes it, the user can navigate to the web application by using the HTTP protocol address provide, through which they would be navigated to the application. MySQL database is used the main database of the system where the data of each function is stored, separated databases were used for each of the services , the data is passed to the database when a user is attached with the system , by using API the system is developed so that the data is passed by being filtered through a gate way, Tomcat Server was used as the server environment to the application to run on.

07.02 Overall Activity diagram



07.04 Any Other Relevant Design Diagrams for the Overall System(class diagram)



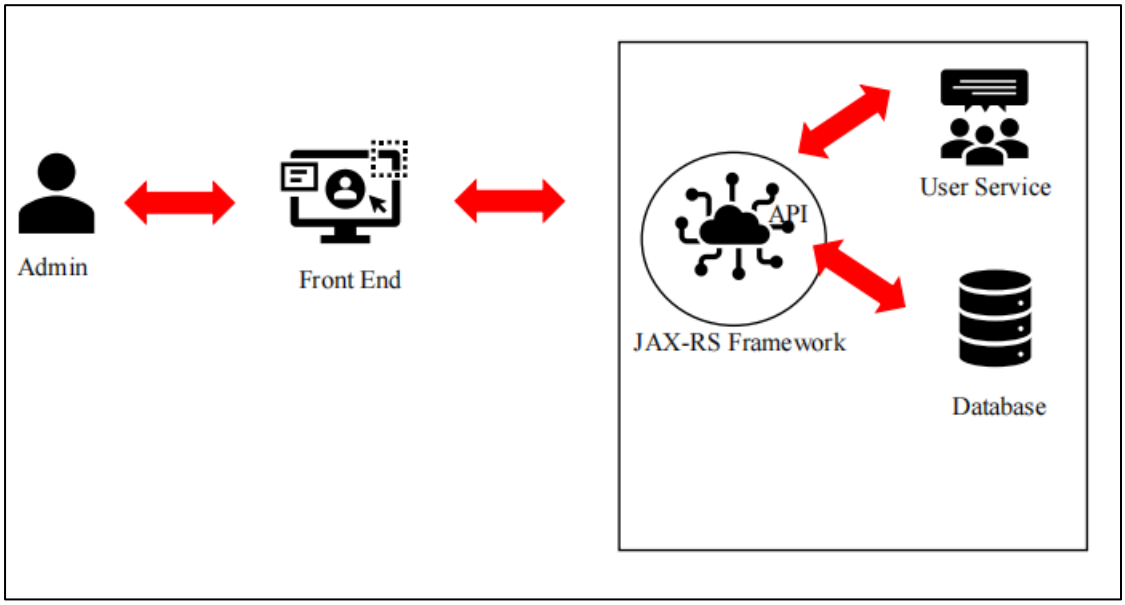
08)Individual Sections

08.01 Payment Handling Service – IT20127428

1) Service design

- The system admin handles this service, after the system admin logs to the web application he can access the payment handling service, where the admin can add, delete, view, and update the payment details.

A) API of the service



i. **Create payment (POST)**
Resource: Admin
Request: POST paymentfunction/payService/pays
Media: Form data -URL encoded
Data: userID:9930, Amount:2300, paymenttype: cash, Date:2022-05-06
Response: Inserted Payment Data Successfully
URL:http://localhost:8080/paymentfunction/payService/pays

ii. **Update payment (PUT)**
Resource: Admin
Request: PUT paymentfunction/payService/pays

Media: Form data -Application JSON
Data:

{
"paymentID":"2",
"userID":"9930",
"Amount":"2300",
"paymenttype”: cash",
"Date":"2022.05.06"
}

Response: Payment Data Updated Successfully
URL:http://localhost:8080/paymentfunction/payService/pays

iii. **View payment (GET)**
Resource: Admin
Request: GET paymentfunction/payService/pays
Media: Form Data
Response: HTML table will be displayed with all the payment data
URL: http://localhost:8080/paymentfunction/payService/pays

iv. **Delete payment (DELETE)**
Resource: Admin
Request: DELETE
Media: Application XML
Data:

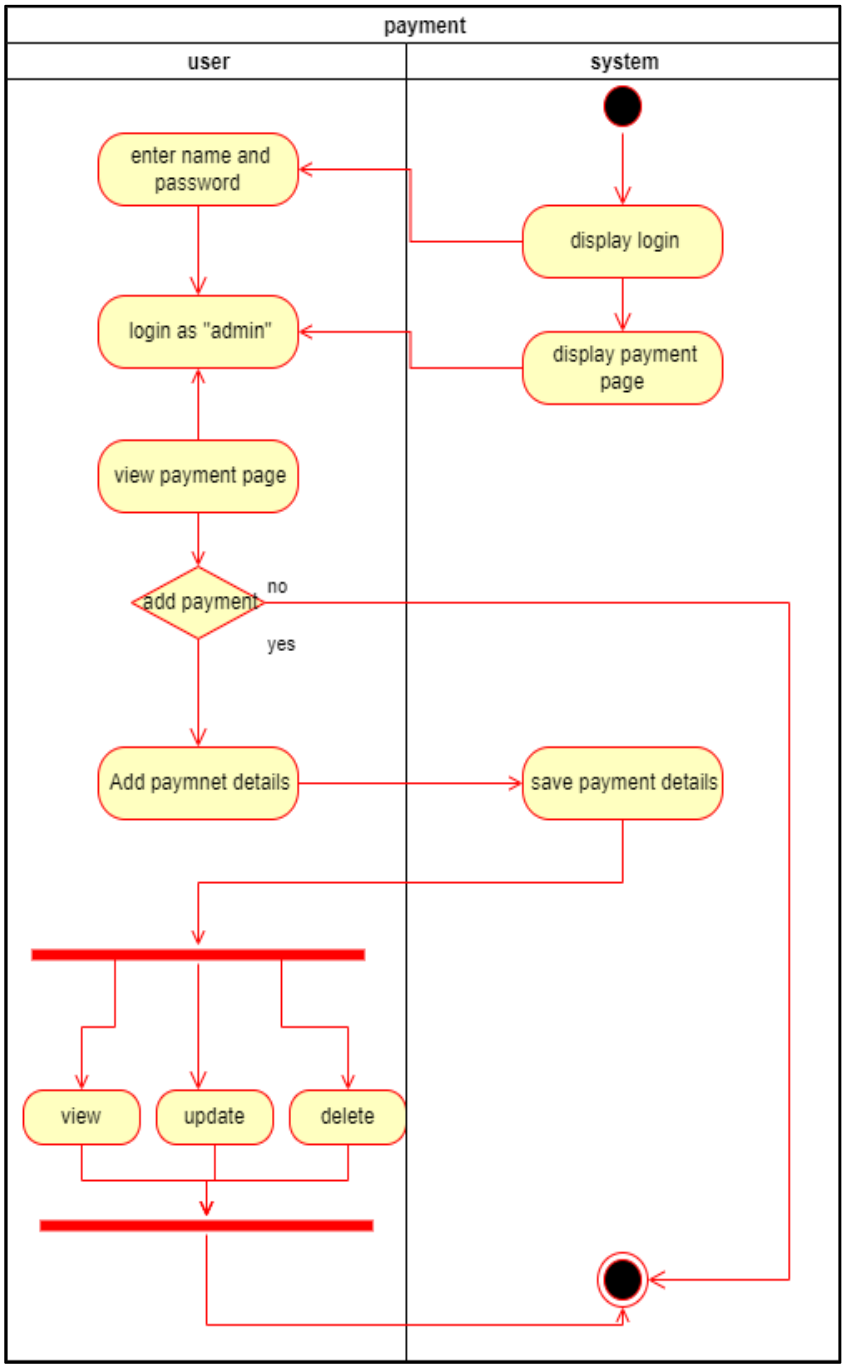
<itemData>
<paymentID>1</paymentID>
</itemData>

Response: Payment Deleted Successfully

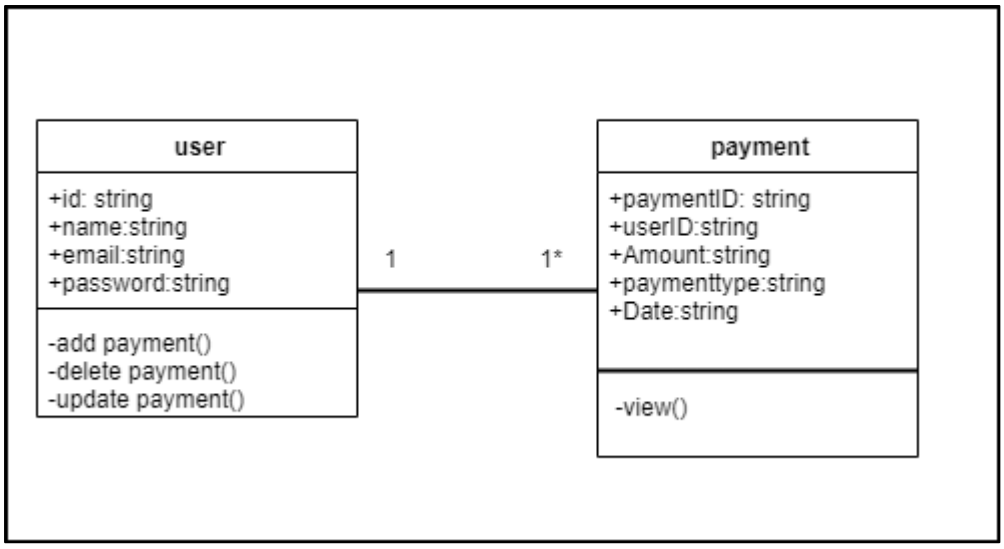
B) Internal logic (Class diagram, activity diagrams, flowcharts)

- The admin can log to the main web application where he will be navigated to the payment page, through that page the admin can do all the CRUD operations like insert, delete, view and update.

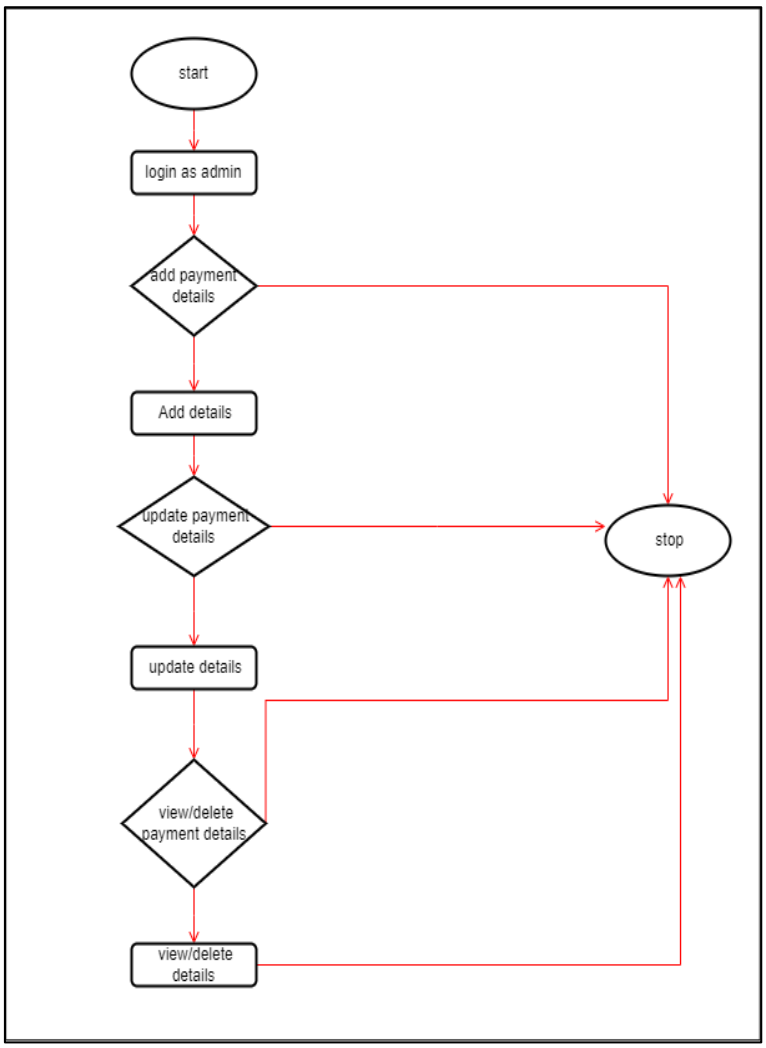
1) Activity Diagram



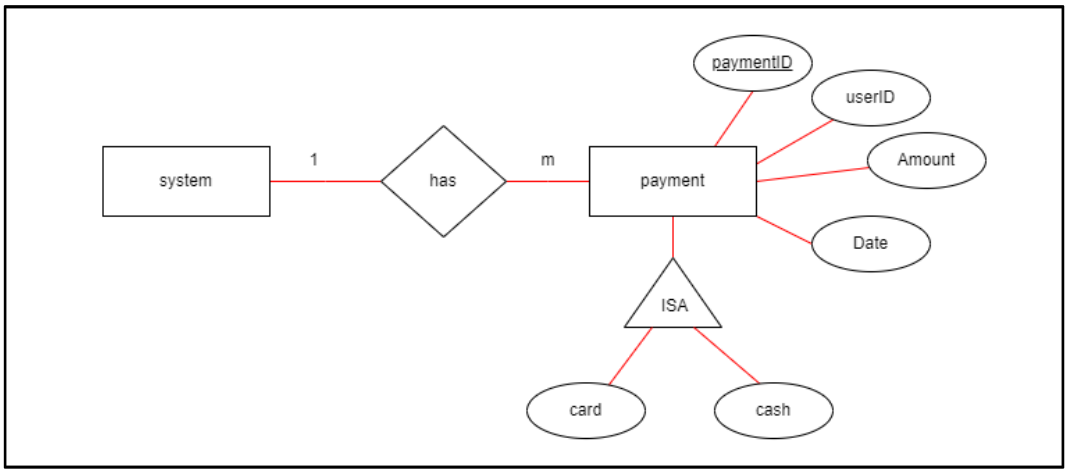
02) class diagram



03) flow chart



04) Database for Payment (ER)



2) Service development and testing

a) Tools Used

- Dependency Management Tool: Maven
- Testing Tool: Postman
- Version Control System: GIT
- IDE: eclipse
- Programming Language: Jersey framework (JAX-RS)
- Programming Language: Java
- Database: phpMyAdmin (MySQL)
- Server: Apache Tomcat Server

b) Testing Methodology and result

Test ID	Description	Input	Expected output	Actual output	Result
1	Create payment	userID:9930, Amount:2300, paymenttype: cash, Date:2022-05-06	Payment Data Updated Successfully	Payment Data Updated Successfully	pass
2	View payment		HTML table will be displayed with all the payment data	HTML table will be displayed with all the payment data	pass

3	Update payment	PaymentID:2, userID:9930, Amount:2300, paymenttype: cash, Date:2022-05-06	Payment Data Updated Successfully	Payment Data Updated Successfully	pass
4	Delete payment	PaymentID="2"	Payment Deleted Successfully	Payment Deleted Successfully	pass

3) **Assumptions**

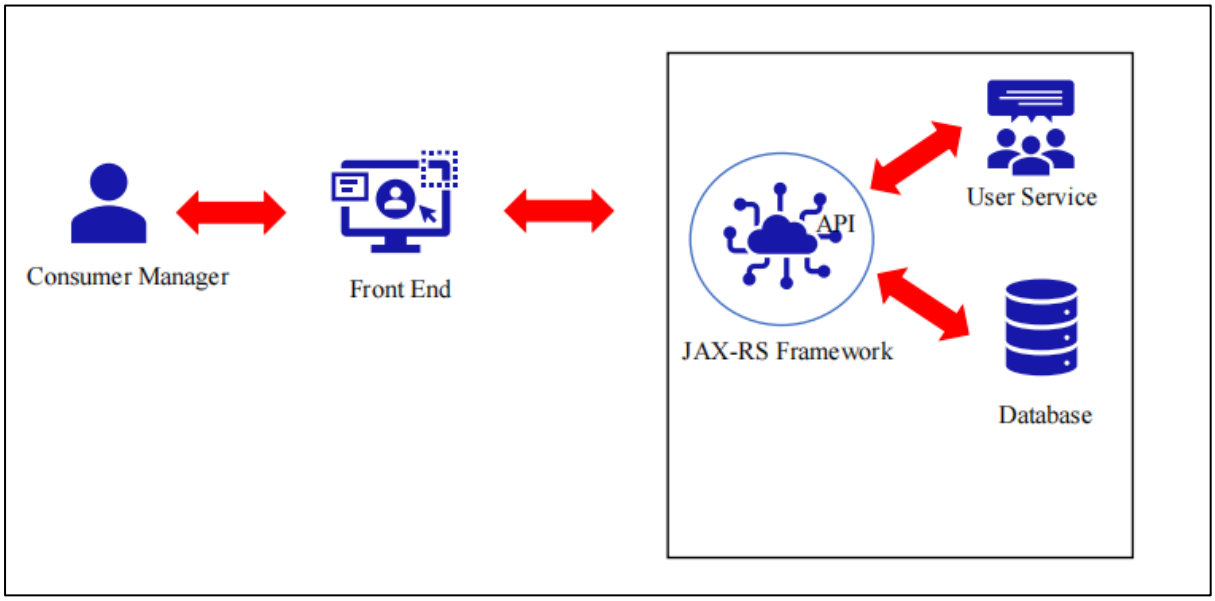
- System has the payment function
- Payment details can be added by the admin
- Admin can delete, update, and view the details
- All the fields should be filled

08.02 Consumer Support Service – IT20142346

1) **Service design**

- Consumer support manger manages this function, by which after logging in to the application the manger can insert, view, update and delete any data regarding the consumer support function.

A) API of the Service



i. Create consumer (POST)
Resource: manager
Request: POST ConsumerSS/ConsumerService/Cons
Media: Form data -URL encoded
Data: name: Niluka, address: Tangalle, Sri Lanka, phone:0472242735, email:nil@gmail.com,description:XXXXXXXXXX
Response: Data Inserted successfully!
URL: http://localhost:8080/ConsumerSS/ConsumerService/Cons

ii. Update consumer (PUT)
Resource: manager
Request: PUT ConsumerSS/ConsumerService/Cons
Media: Form data -Application JSON
Data:

{

"cid": "1",
"name":"Lalith",
"address": Malabe, Kaduwela",
"phone":"0703487266",
"email":"lali@gmail.com",
"description":"xxxxxxxx"

}

Response: Data Updated successfully!
URL: http://localhost:8080/ConsumerSS/ConsumerService/Cons

iii. **View consumer (GET)**
Resource: manager
Request: GET ConsumerSS/ConsumerService/Cons
Media: Form Data
Response: HTML table will be displayed with all the consumer data
URL: http://localhost:8080/ConsumerSS/ConsumerService/Cons

iv. **Delete consumer (DELETE)**
Resource: manager
Request: DELETE
Media: Application XML
Data:

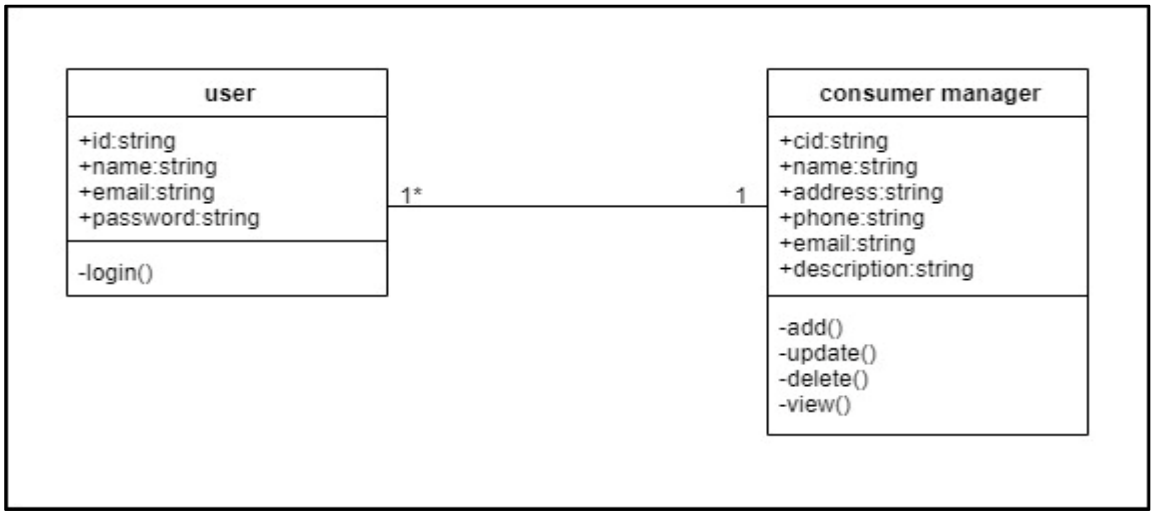
```
<itemData>
  <cid>3</cid>
  </itemData>
```

Response: Data Deleted successfully!

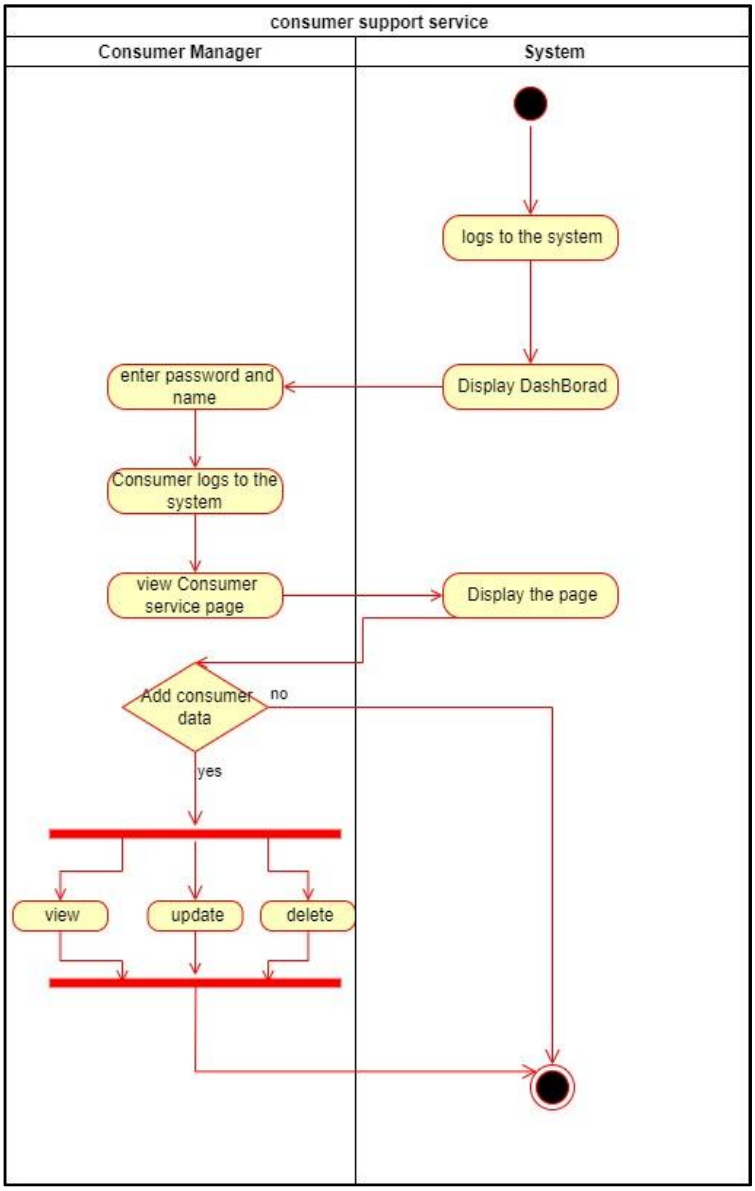
B) Internal Logic (Class Diagram/Activity Diagram/Flow Chart/ER)

- Consumer manager can directly add the details through the consumer management page after login into the system, the consumer manager can enter all the required data and can view, update ad deletes them according to fulfil the system requirements.

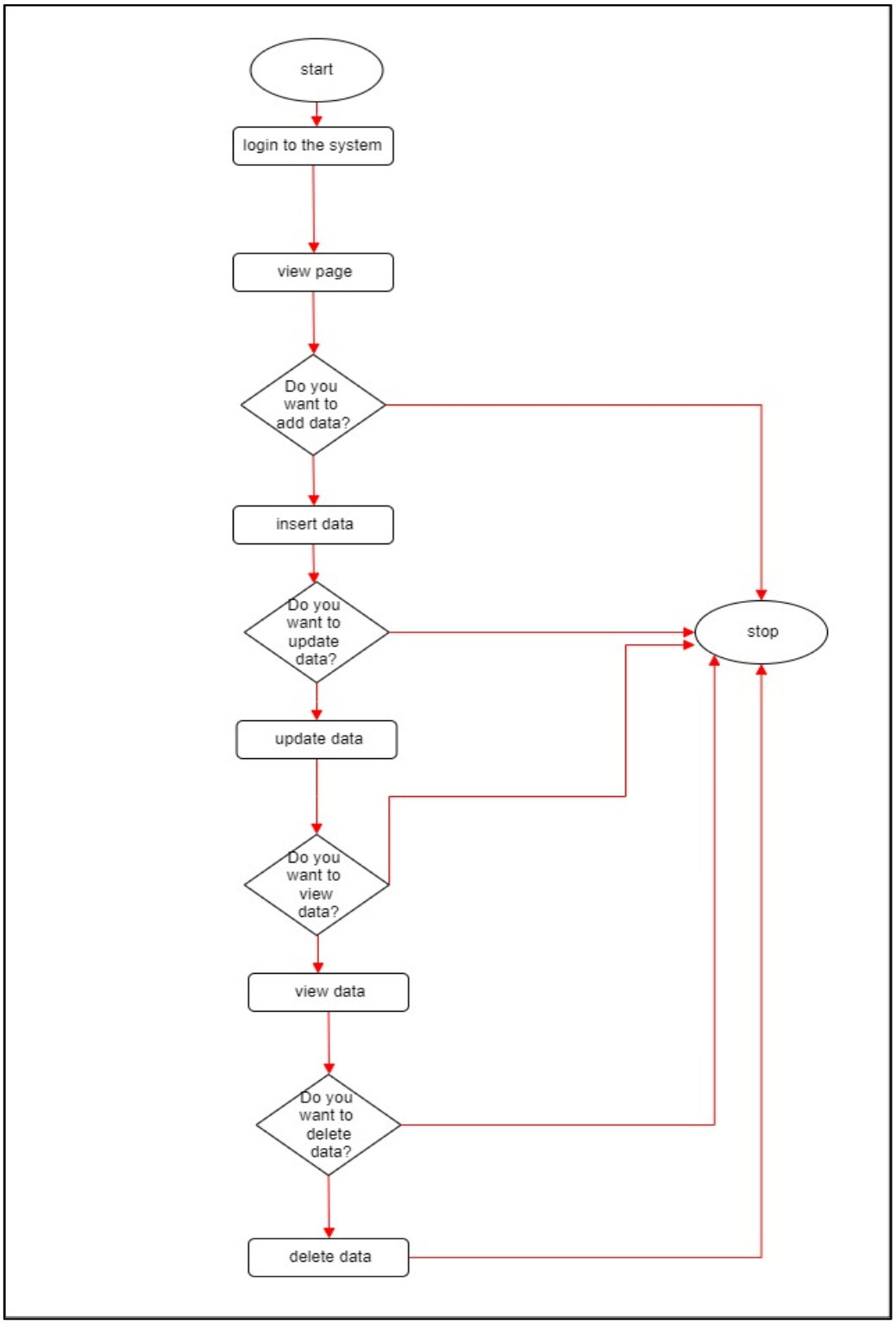
1) Class Diagram



2)Activity Diagram

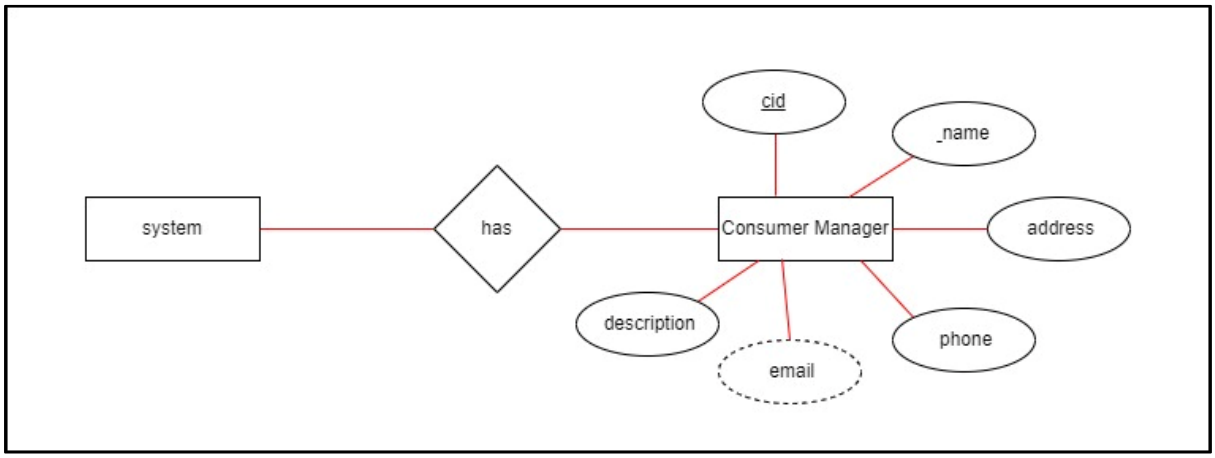


3)Flow Chart


start
login to the system
view page
Do you want to add data?
insert data
Do you want to update data?
stop
update data
Do you want to view data?
view data
Do you want to delete data?
delete data

4) Er diagram

2) Service development and testing



a) Tools Used

- Dependency Management Tool: Maven
- Testing Tool: Postman
- Version Control System: GIT
- IDE: eclipse
- Programming Language: Jersey framework (JAX-RS)
- Programming Language: Java
- Database: phpMyAdmin (MySQL)
- Server: Apache Tomcat Server

b) Testing Methodology and result

Test ID	Description	Input	Expected output	Actual output	Result
1	Create consumer	cid:1, name: Niluka, address: Tangalle, Sri Lanka, phone:0472242735	Data Inserted successfully!	Data Inserted successfully!	pass
2	View consumer		HTML table will be displayed with all the consumer data	HTML table will be displayed with all the consumer data	pass
3	Update consumer	cid:1, name: Niluka, address: Tangalle, Sri	Data Updated successfully!	Data Updated successfully!	pass

		Lanka, phone:0472242735			
4	Delete payment	cid="1"	Data Deleted successfully!	Data Deleted successfully!	pass

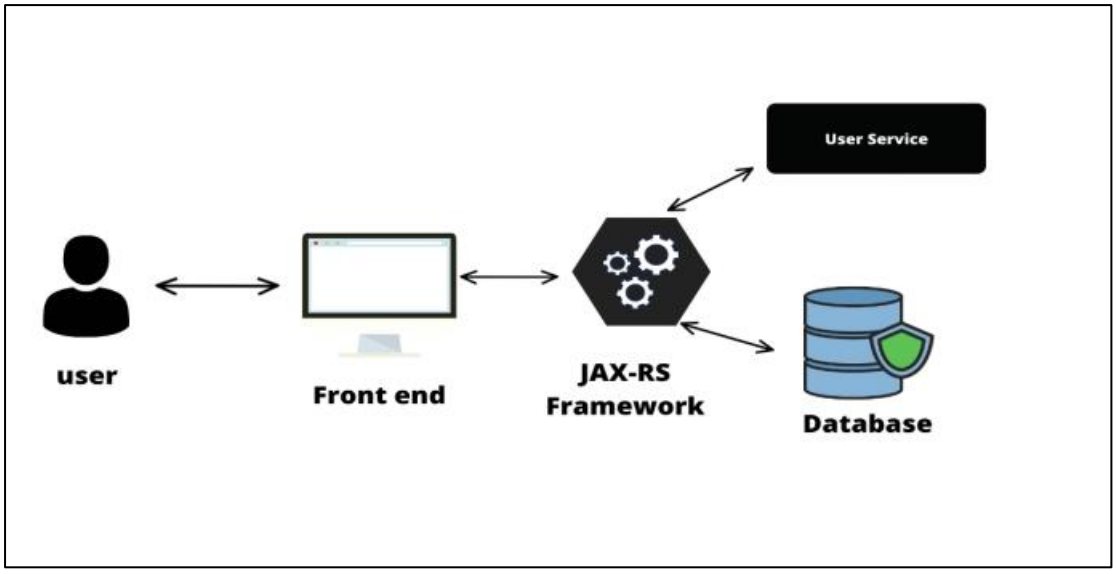
3) <u>Assumptions</u>

- Consumer manager handles the function
- All the consumer related data can be inserted
- The inserted data can be updated, deleted, and viewed

08.03 User Service– IT20194758

1. **Service Design**

- Users can register to the system using registration form by providing their details. After that user can login to the system as a valid user by providing valid credentials. The access to the system varies according to the user type.



- **API of the service**

I. **Create User (POST)**

Resource: Users
Request: POST userService/UserService/User
Media: Form data - URL encoded
Data: username: Malka, name: Malka, phone:0714526395, email: malka@gmail.com, password:123456
URL:http://localhost:8090/Lab5Rest/UserService/Users
Response: Inserted successfully

II. **Update User (PUT)**

Resource: Users
Request: PUT userService/UserService/User
Media: Form data – Application JSON
Data: {
"userID":"1",
"username": "malka",
"name":"malkaJayalee",
"phone":"0724526395",
"email”: malkaj@gmail.com",
"password":"123956"
}

III. URL:http://localhost:8090/Lab5Rest/UserService/Users
Response: Updated successfully

IV. **View User (GET)**

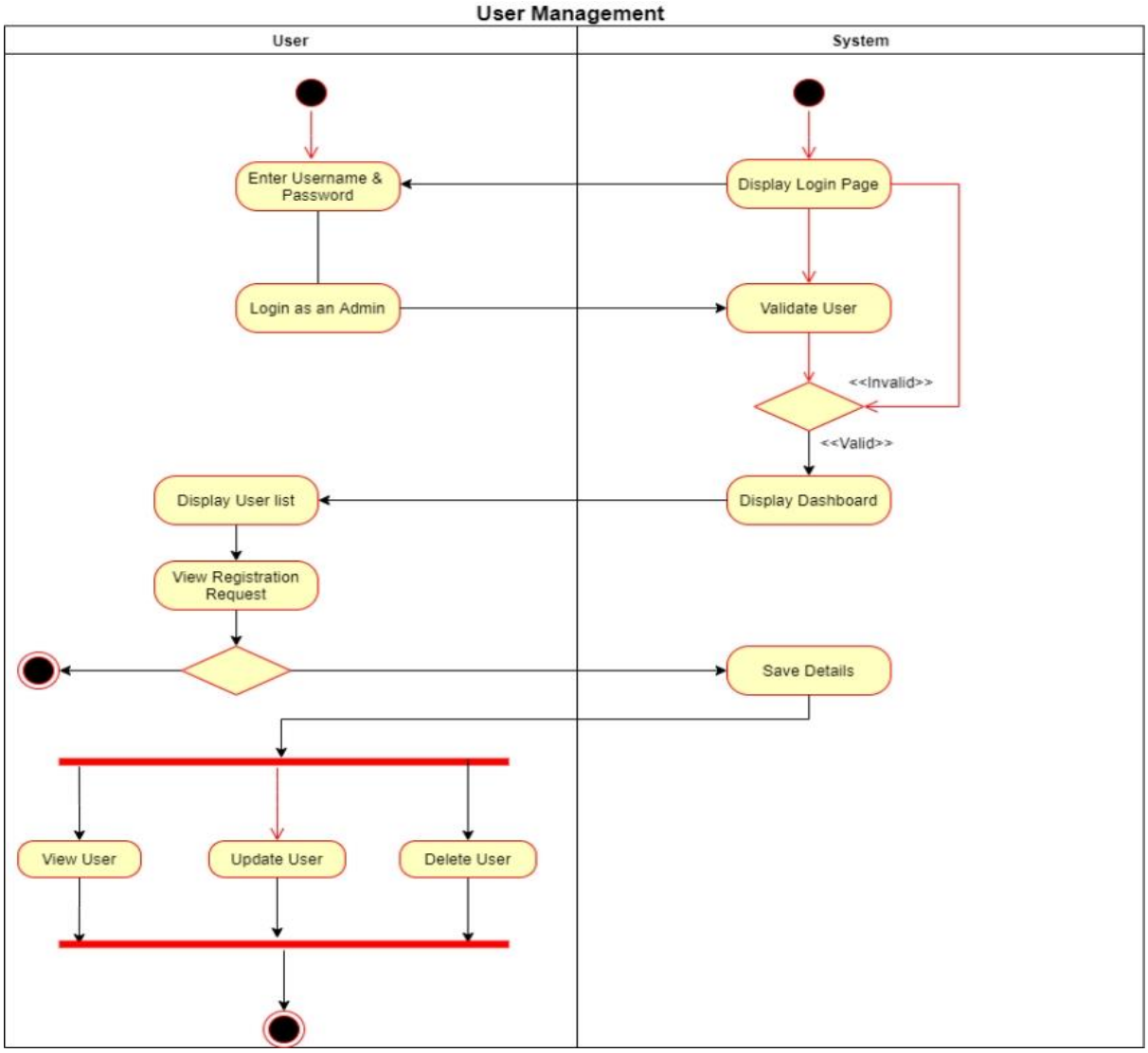
Resource :Users
Request: GET userService/UserService/User
Media: Form Data
Response: HTML table with all attributes in the User table

V. **Delete User (DELETE)**

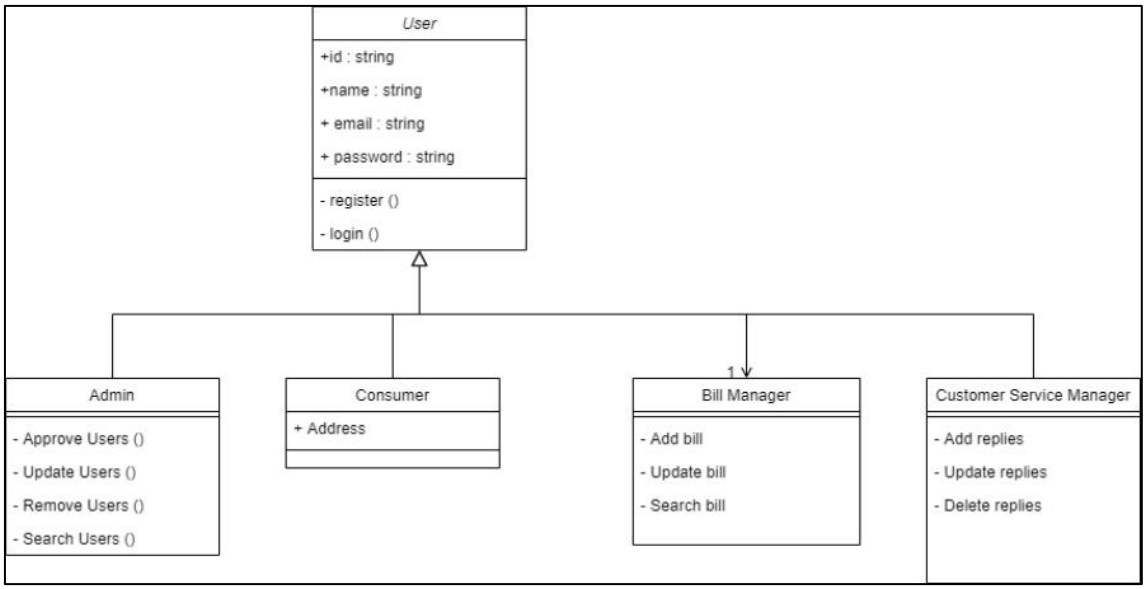
Resource: - User
Request: - DELETE
Media: - Application XML
Data: -
<userData>
<userID>1</userID>
</userData>
Response: - Deleted successfully

- **Internal logic (Class Diagram / Activity Diagram/Flow Chart)**
  The responsibility of the user service is to manage the details of the users who register to the system. Mainly there are 4 types of user's consumer, bill manager, and customer service manager. When registering to the system as a researcher they should provide valid certification to prove that they are customer service managers. The administrator of the system will check the certification and accept their registering request. Then they were given the access to customers inquiry section and give answers for questions raised by the customers. The customer can register to the system using normal procedure and get the monthly bill and pay that on online payment system. Finance manager is added by the administrator by providing credentials for login.

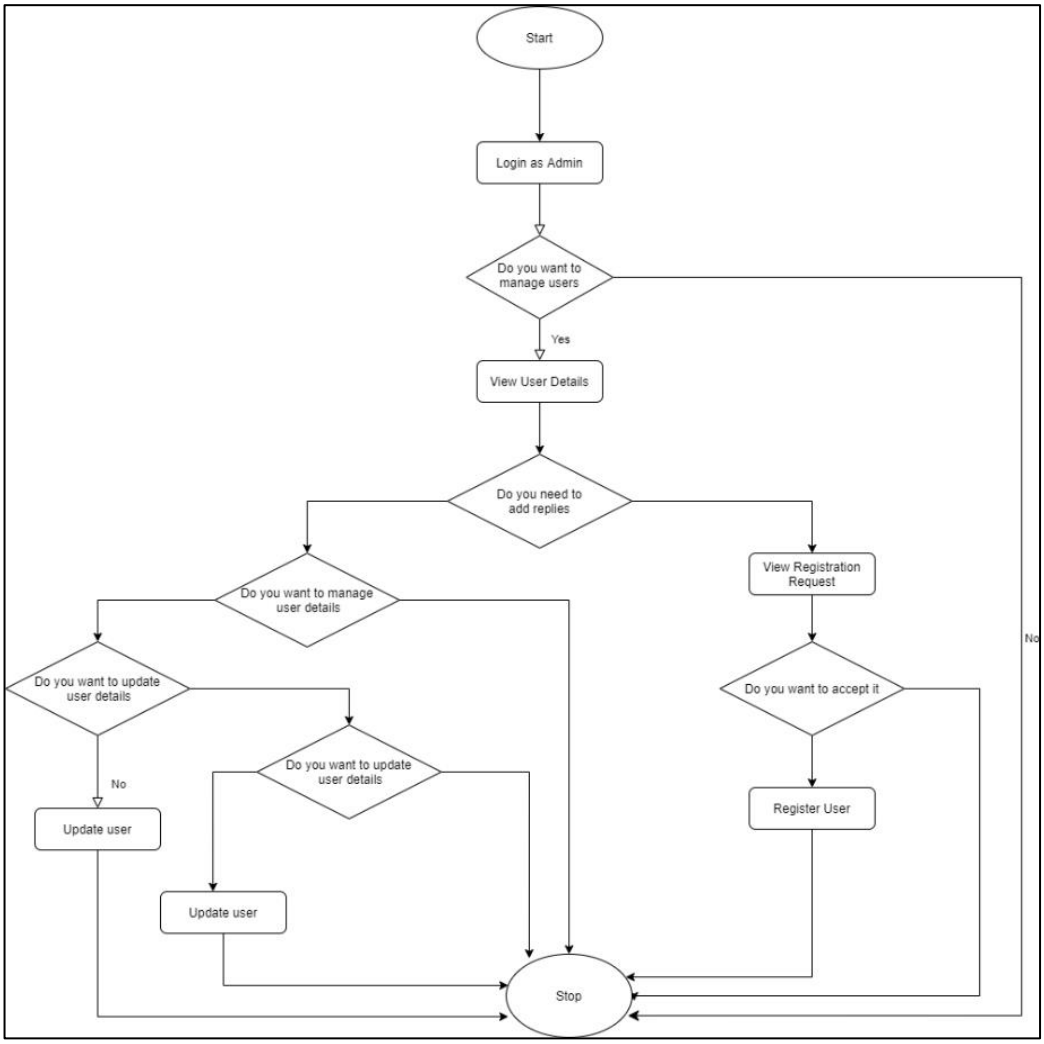
- Activity Diagram



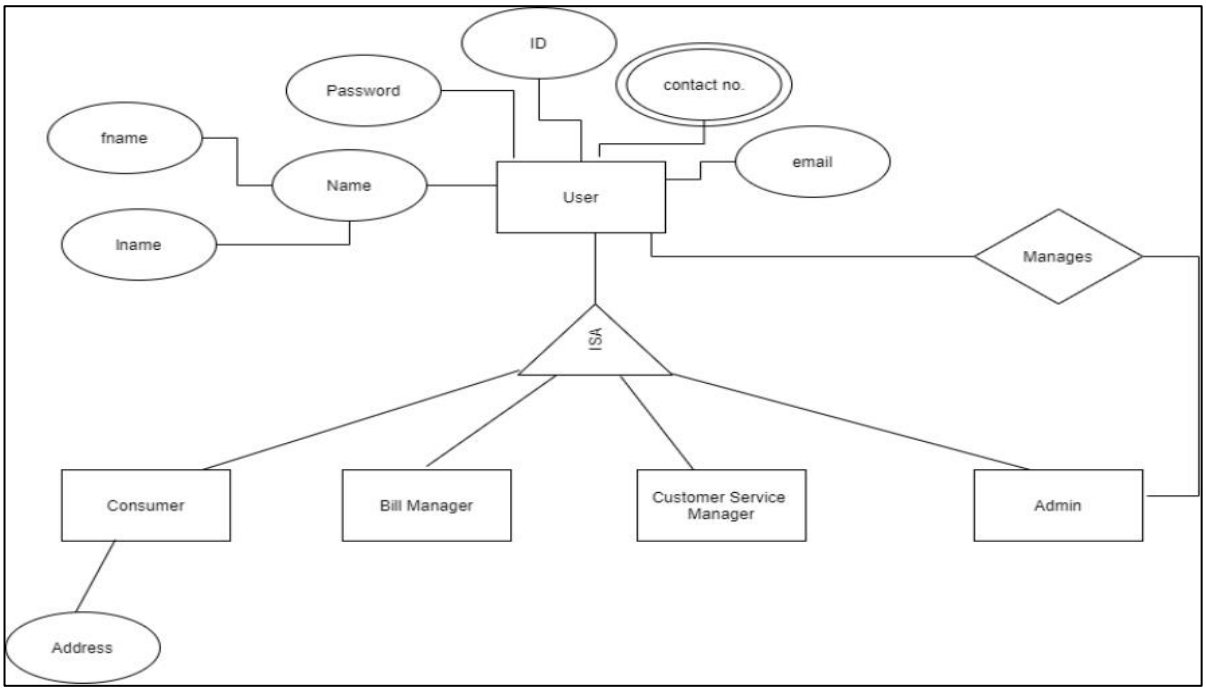
- Class Diagram



- Flow Chart



- Database for the service (ER)



2) Service Development & Testing

1) Tools Management

- Dependency Management Tool: Maven
- Testing Tool: Postman
- Version Control System: GIT
- IDE: eclipse
- Programming Language: Jersey framework (JAX-RS)
- Programming Language: Java
- Database: phpMyAdmin (MySQL)
- Server: Apache Tomcat Server

2) Testing methodology and results

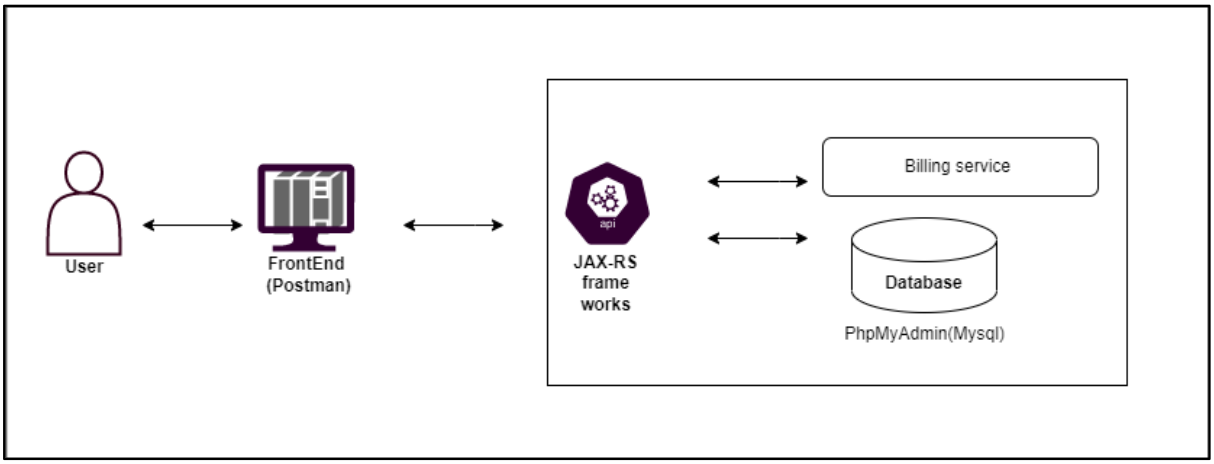
Test ID	Description	Input	Expected Output	Actual Output	Result
1	Create User	username: Hashan userNIC:7376649209 contactNumber:0784532134, userType: a,	Inserted Successfully	Inserted Successfully	Pass
2	View User		Display a HTML table with all the attributes in user table	Display a HTML table with all the attributes in user table	Pass
3	Update User	userID: 12 username: Hashan userNIC:7376649209 contactNumber:0784532134, userType: a,	Update Successfully	Update Successfully	Pass
4	Delete User	userId – “1”	Deleted Successfully	Deleted Successfully	Pass

2. Assumptions
    - There are mainly three types of users in the system researcher, consumer, and financial manager.
    - Admin has the responsibility to manage all the users of the system
    - Researchers only can send registration requests, After the approval of the administrator they will be considered as a valid user.
    - Financial manager will be added by the administrator to maintain the transactions of the system.

08.04 Bill Handling– IT20187132

API Of the Service

Admin should be able to add users' bill details. If users' bill details are incorrect, He should ability to edit /delete these. Then generate bill, and User should ability to view his Total Electricity Monthly bill amount by online and Admin should be able to send a bill using email.



1. **Create Bill (POST)**

   Resource: Bill
   Request: POST billingService/BillService/Bill
   Media: Form data - URL encoded
   Data:
   accno: 28918229115, customername: kapila, email: kapila98@gmail.com, totalnoofprice:10, date:2021/08/03,totalamount:30000
   Response: Inserted successfully
   URL:

2. **Update Bill (PUT)**

   Resource: Bill
   Request: PUT billService/BillService/Bill
   Media: Form data – Application JSON Data:
   {
   "accNo":"1",
   "cusName": " Kapila",
   "email":"kapila98@gmail.com",
   "totalNoOfPrice": "10"
   " date ":"2021/08/13",
   " totalAmount ":"20000"

   }
   Response: Successfully updated
   URL:

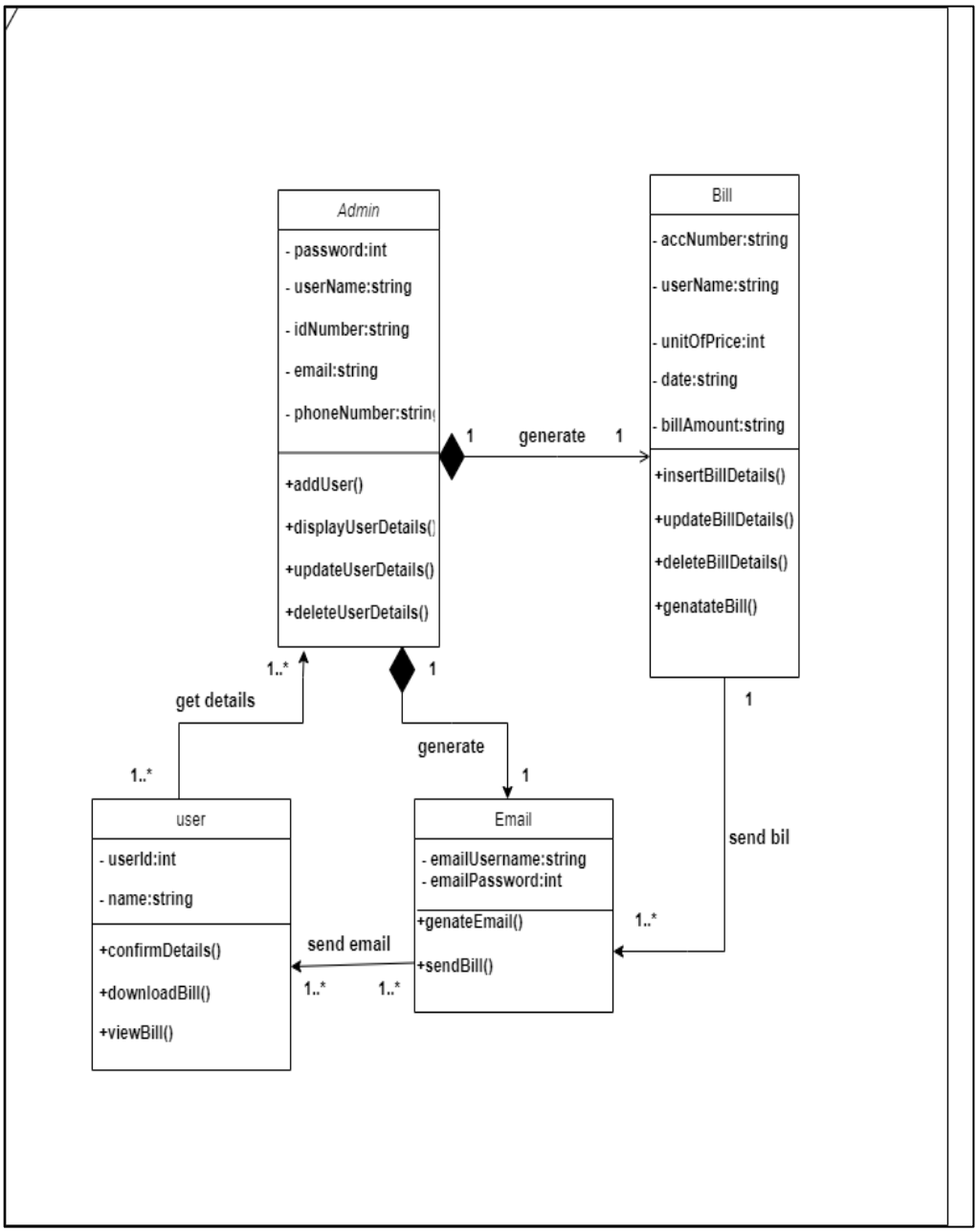
3. **View Bill(GET)**

   Resource: Bill
   Request: GET billService/BillService/Bill
   Media: Form Data
   Response: HTML table with all attributes in the Bill table
   URL:

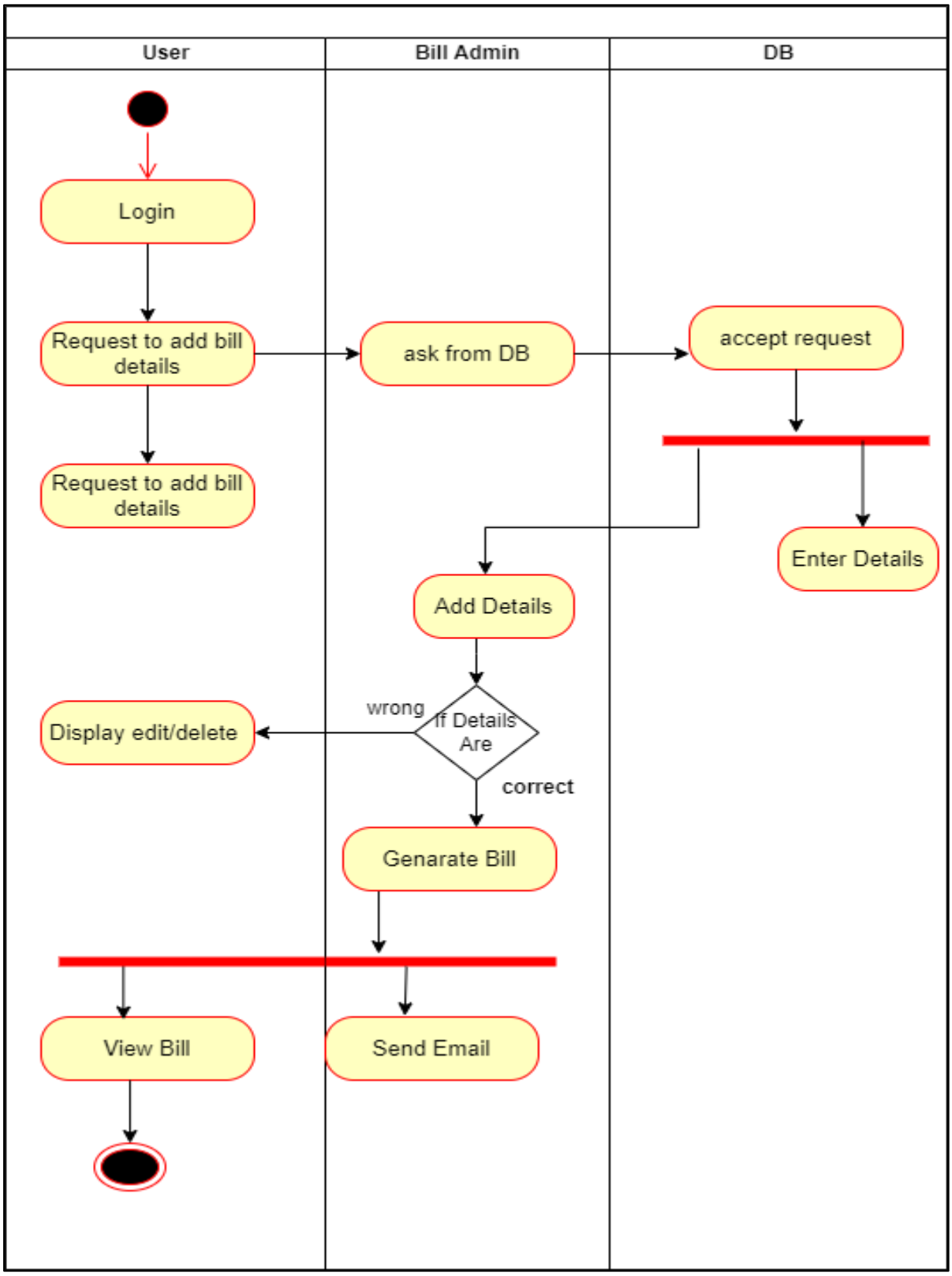
4. **Bill (DELETE)**

   Resource: - Bill
   Request: - DELETE
   Media: - Application XML
   Data: -
   <billData>
   <accNo>1 </accNo>
   </billData>
   Response: - successfully Deleted
   URL:

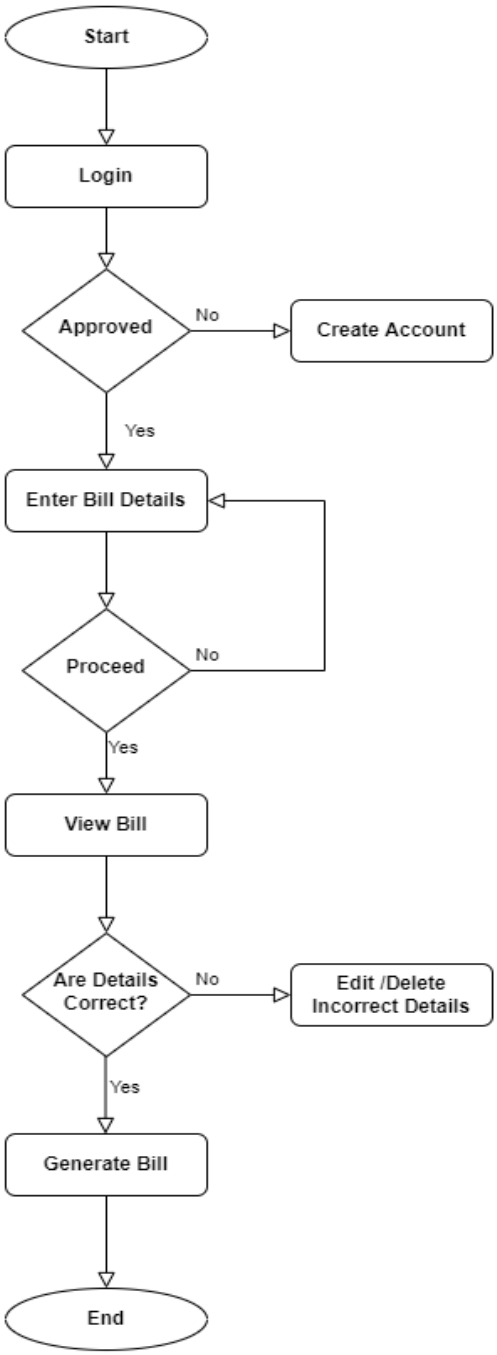
Class Diagram



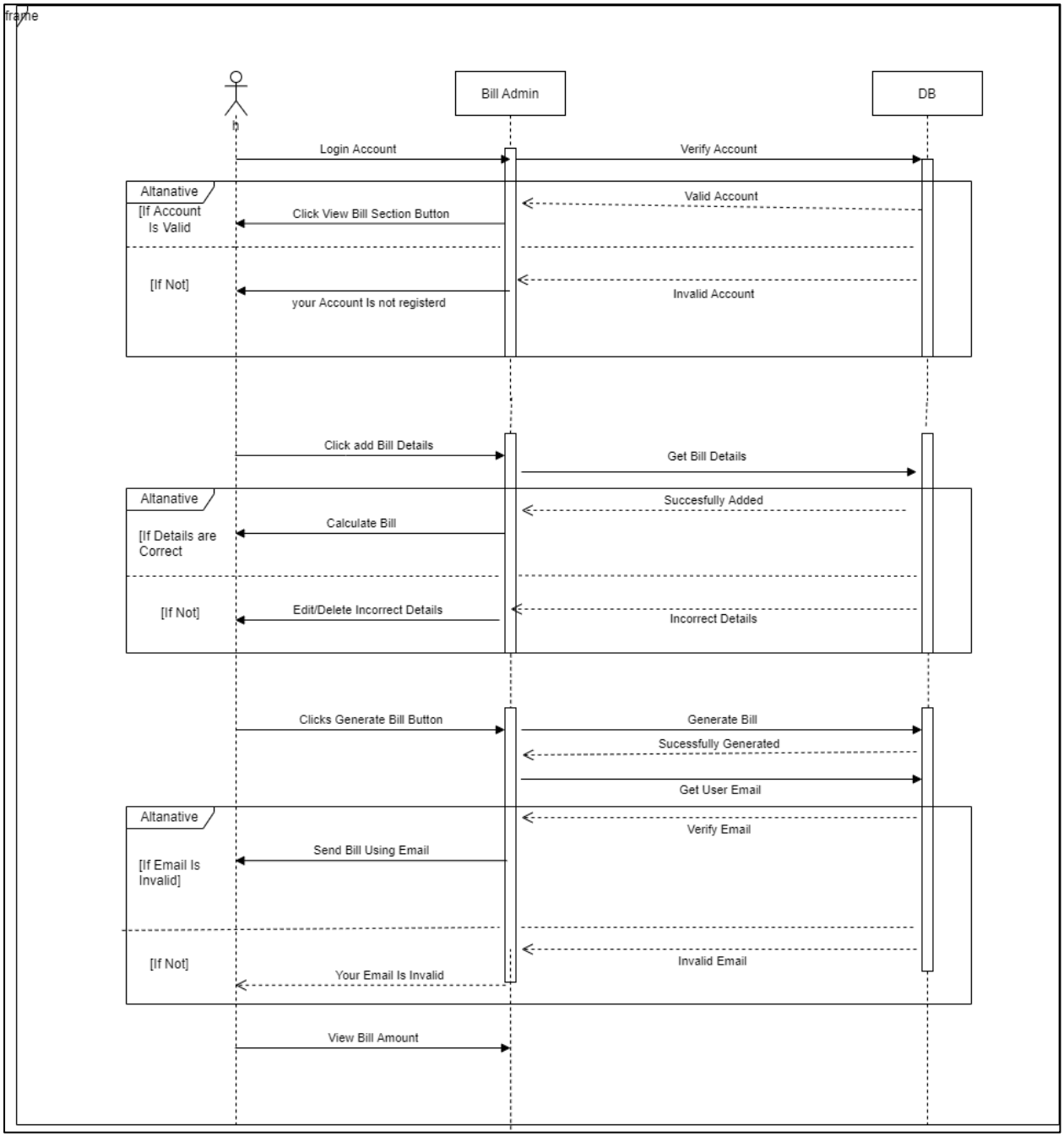
Activity Diagram



Flow Chart Diagram



Sequence Diagram



Testing

2) Service development and testing

- Tools used
  - Dependency Management Tool: Maven
  - Testing Tool: Postman
  - Version Control System: Git
  - IDE: eclipse
  - Programming Language: Jersey framework (JAX-RS)
  - Programming Language: Java
  - Database: (MySQL)
  - Server: Apache Tomcat Server

Test ID	**Description**	**Input**	**Expected Output**	**Actual Output**	**Result**
1	Create Bill	accNo: 2891829115, cusName: supun, email: kapila01@gmail.com, totalNoOfPrice:10, date: 2021/09/08, totalAmount:30000	Inserted Successfully	Inserted Successfully	Pass
2	View Bill Details		Display a html table with all the attribute in bill table	Display a html table with all the attribute in bill table	Pass
3	Update Bill Details	accNo: 2789052009, cusName: kapila, email: kapila20@gmail.com, totalNoOfPrice:5, date: 2020/09/08, totalAmount:15000	Update Successfully	Update Successfully	Pass
4	Delete Bill	accNo- “1”	Delete Successfully	Delete Successfully	Pass

09) System's integration details

1) Tools Used, Testing Methodology and Results & API Documentation

- The following tools were used to develop the project.

1. Dependency Management Tool: Maven
2. Testing Tool: Postman
3. Version Control System: Git
4. IDE: eclipse
5. Programming Language: Jersey framework (JAX-RS)
6. Programming Language: Java
7. Database: phpMyAdmin (MySQL)
8. Server: Apache Tomcat Server

- For testing purpose postman was used.
- For integration GitHub was used.

2) The Architecture used to Design the System

- The high-level architecture diagram was used to design the overall architecture of the system.
- Use case diagram was used to identify the use cases.
- ER diagram to identify the tables of the database.
- Activity diagram and flow charts to identify the flow of the system.
- Class diagram to identify the classes for the identification

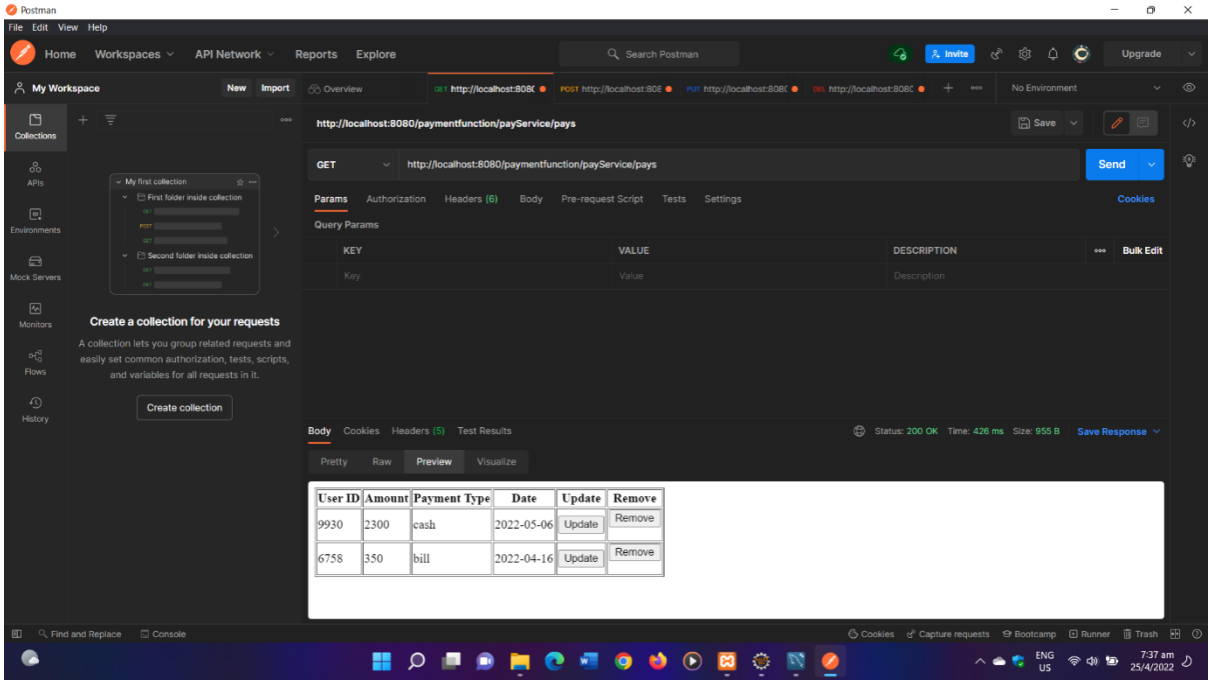
10)References

- SE Methodologies
https://acodez.in/12-best-software-development-methodologies-pros-cons/
- Maven Documentation
https://maven.apache.org/guides/

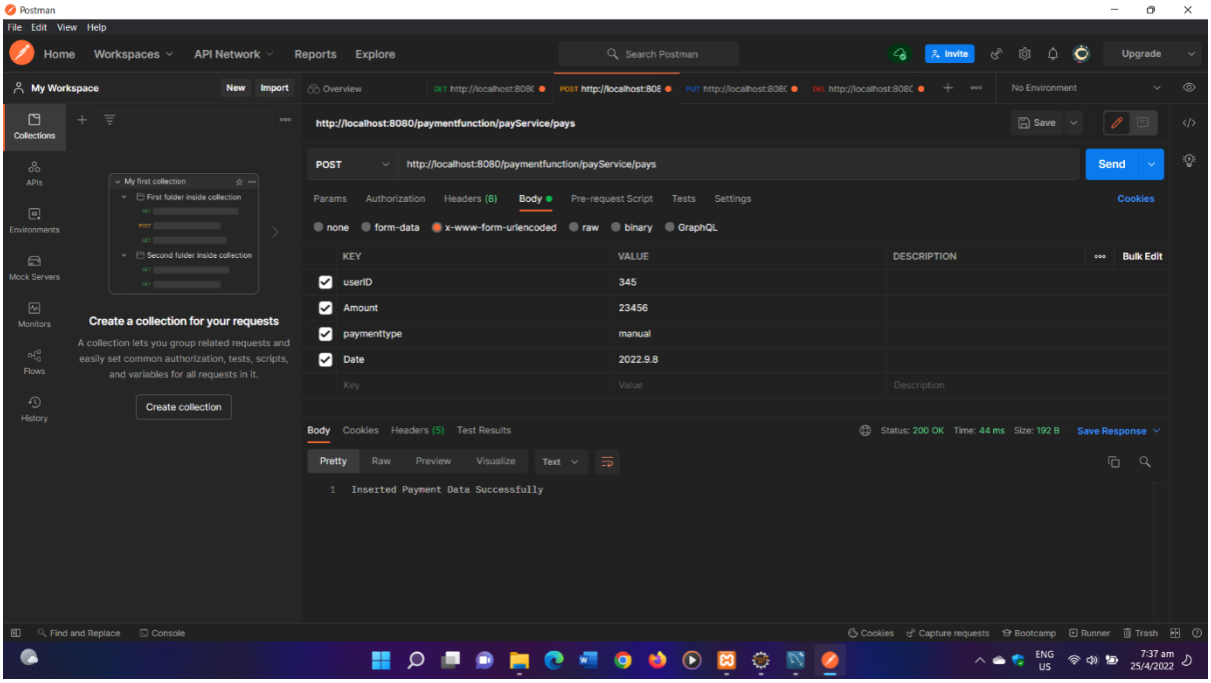
Appendix

01) Screen Shots of IT20127428

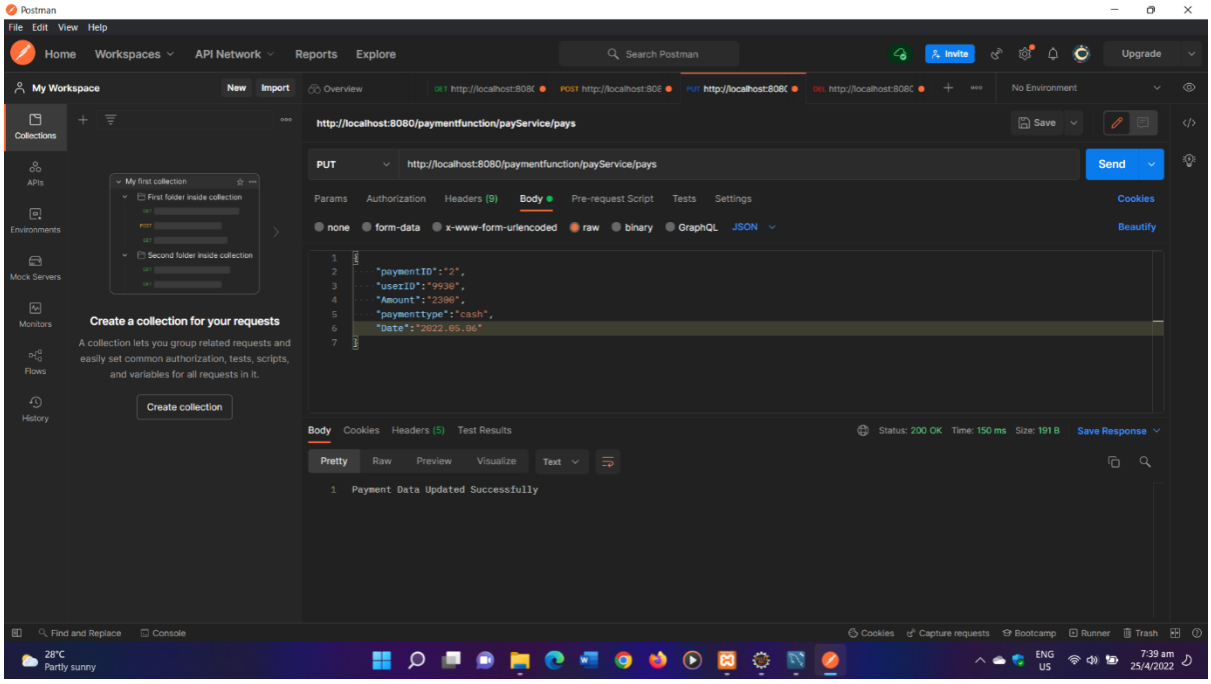
GET method



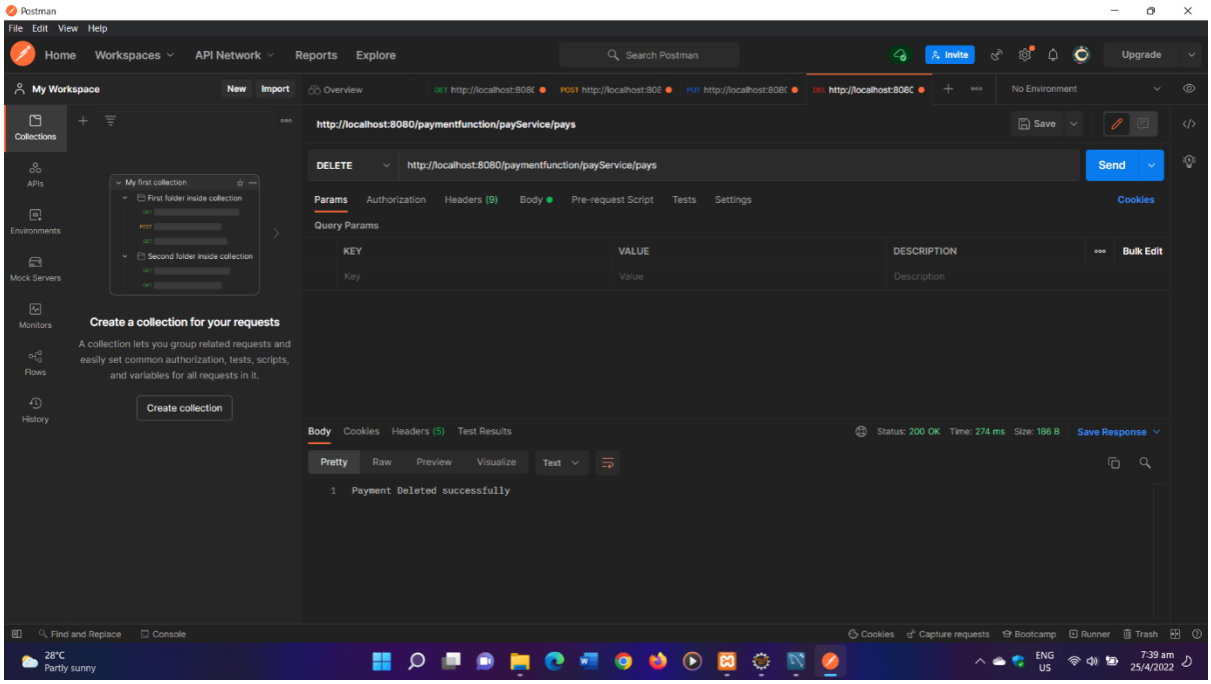
POST method



PUT method

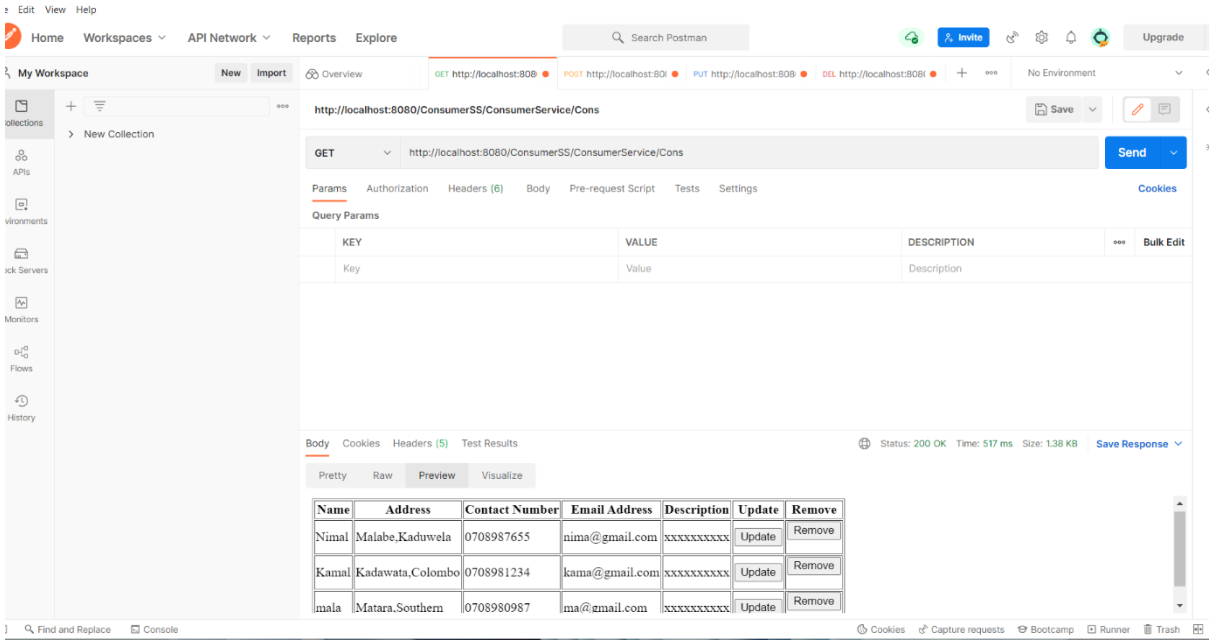


DELETE method

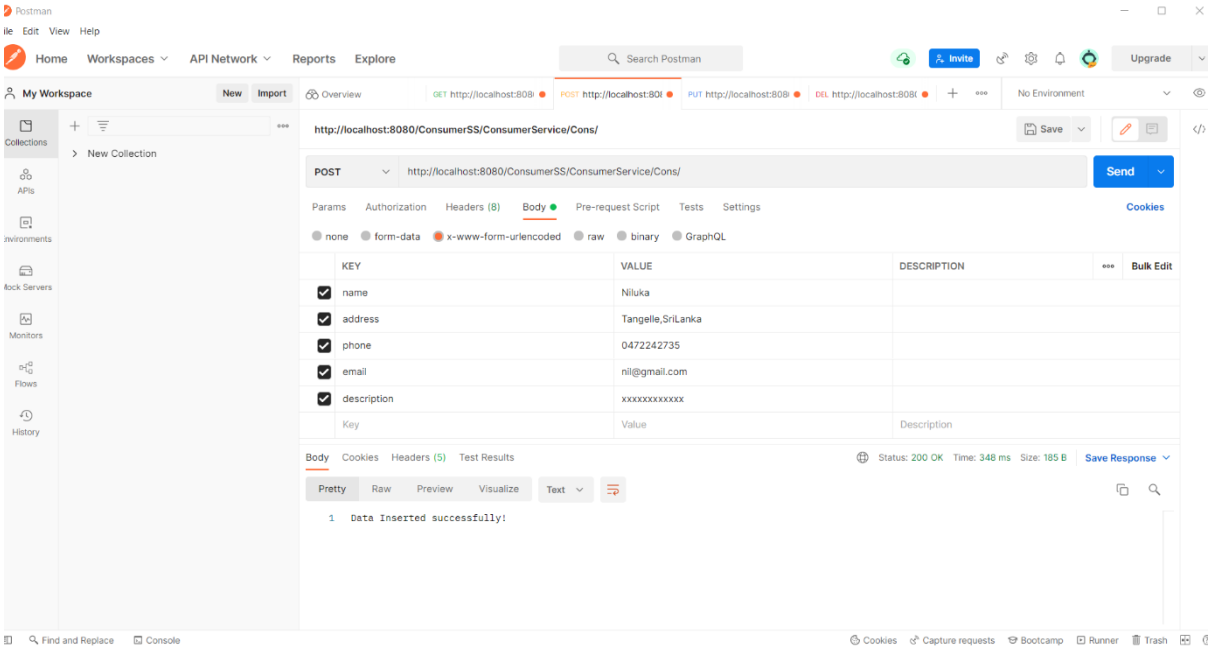


02) Screen Shots of IT20142346

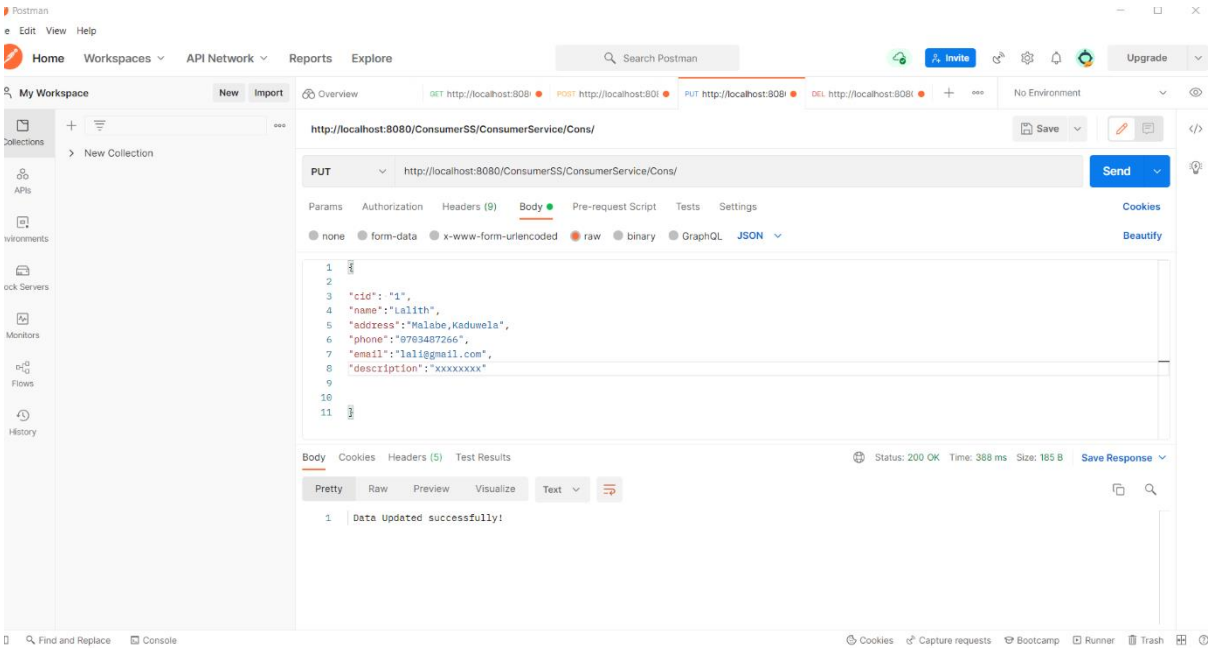
GET method



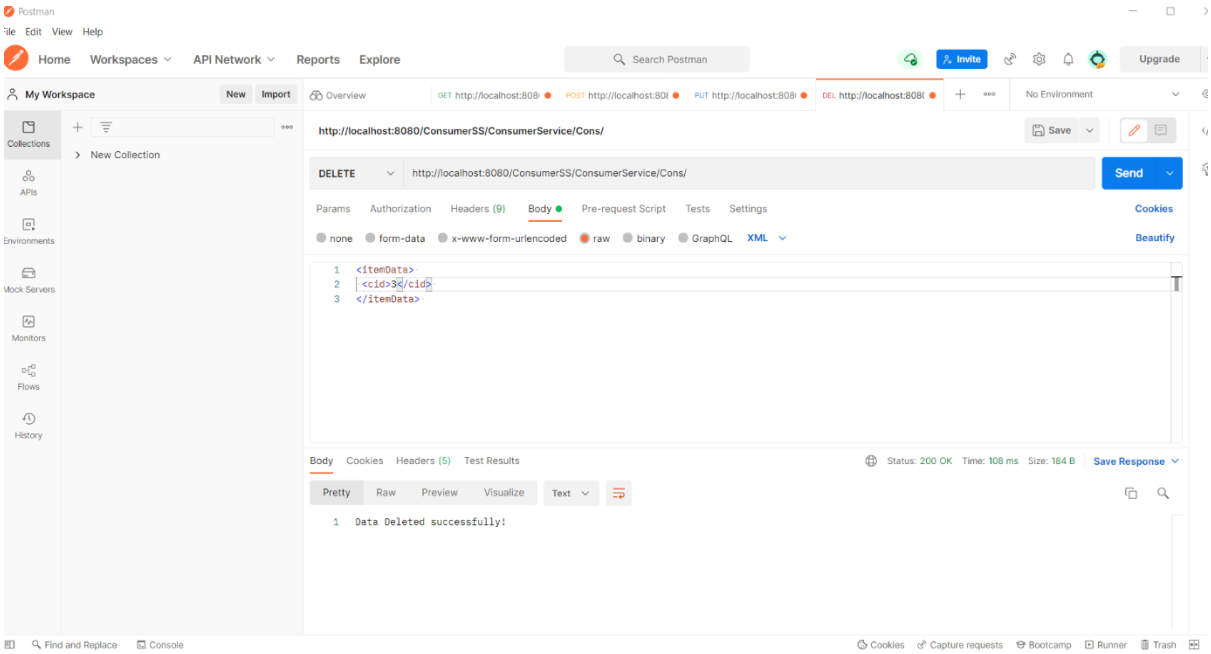
POST method



PUT method

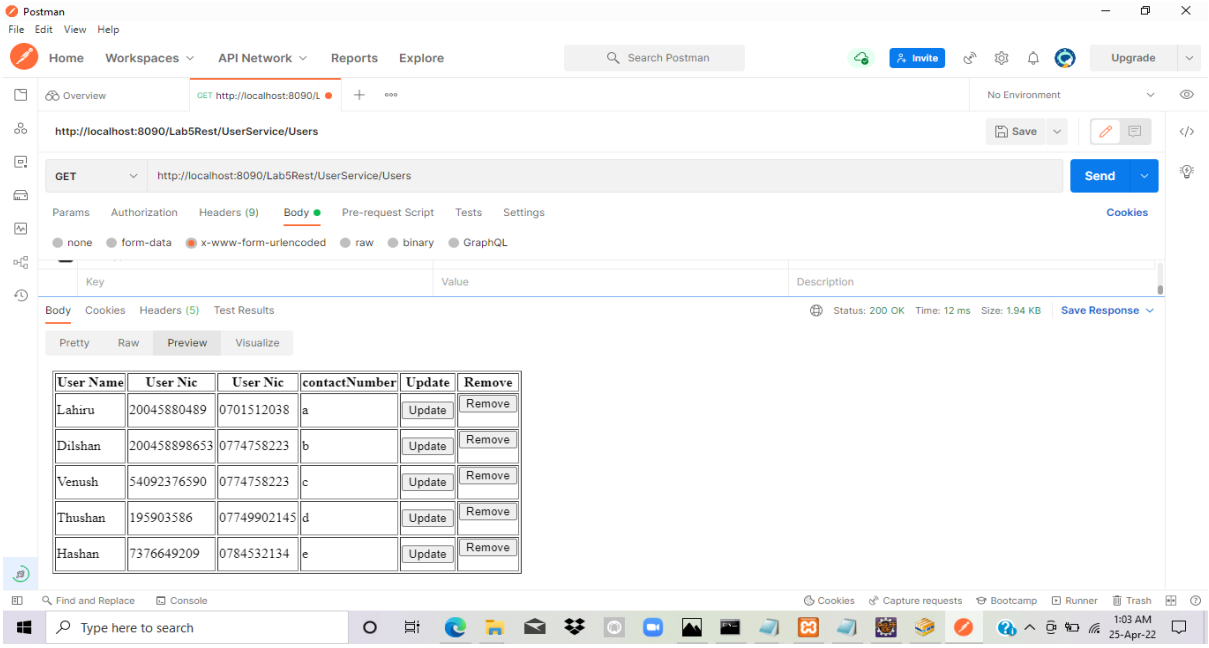


DELETE method



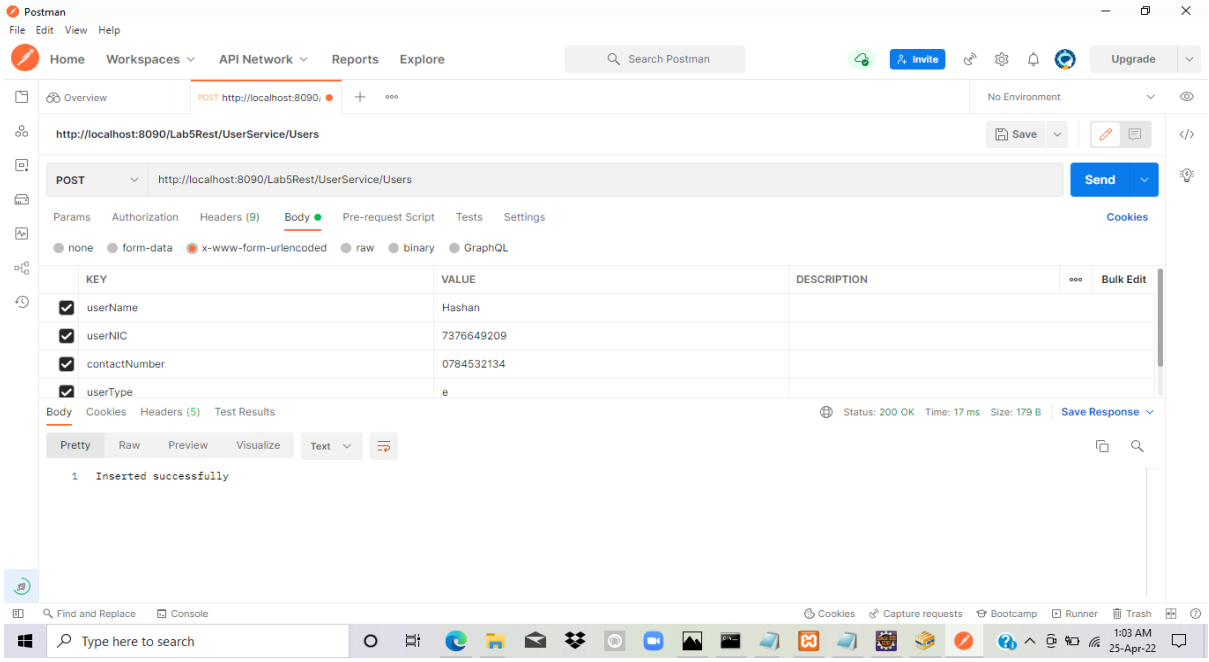
03) Screen Shots of IT20194758

GET method

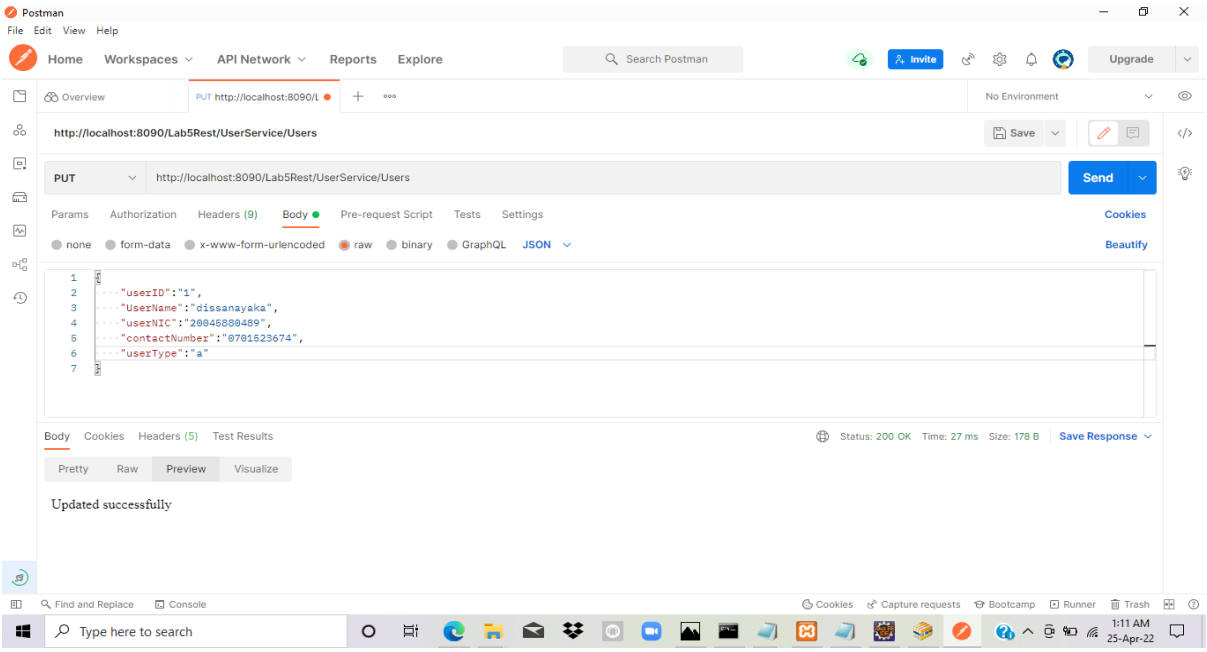


User Name	User Nic	User Nic	contactNumber	Update	Remove
Lahiru	20045880489	0701512038	a	Update	Remove
Dilshan	200458898653	0774758223	b	Update	Remove
Venush	54092376590	0774758223	c	Update	Remove
Thushan	195903586	07749902145	d	Update	Remove
Hashan	7376649209	0784532134	e	Update	Remove

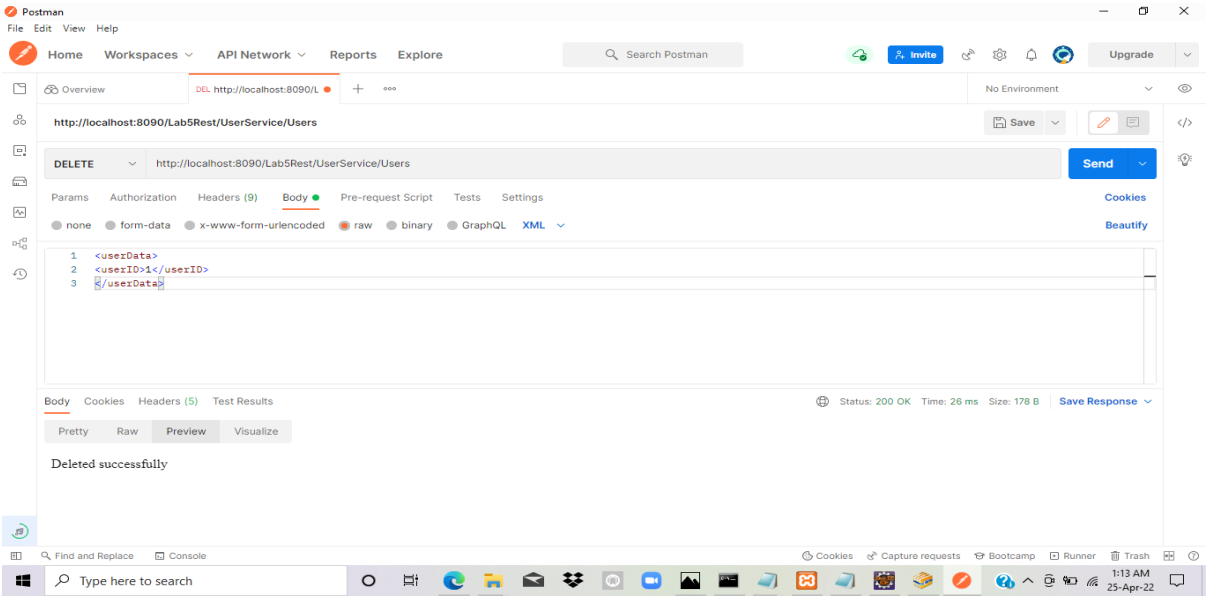
POST method



PUT method

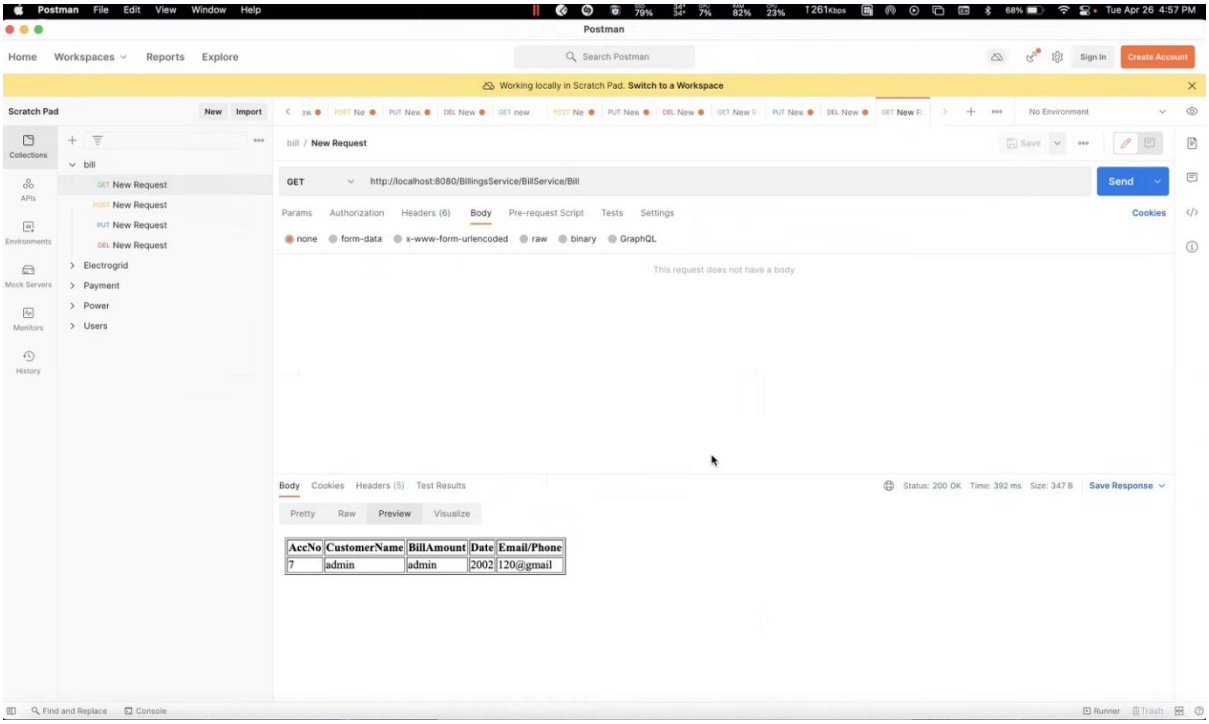


DELETE method

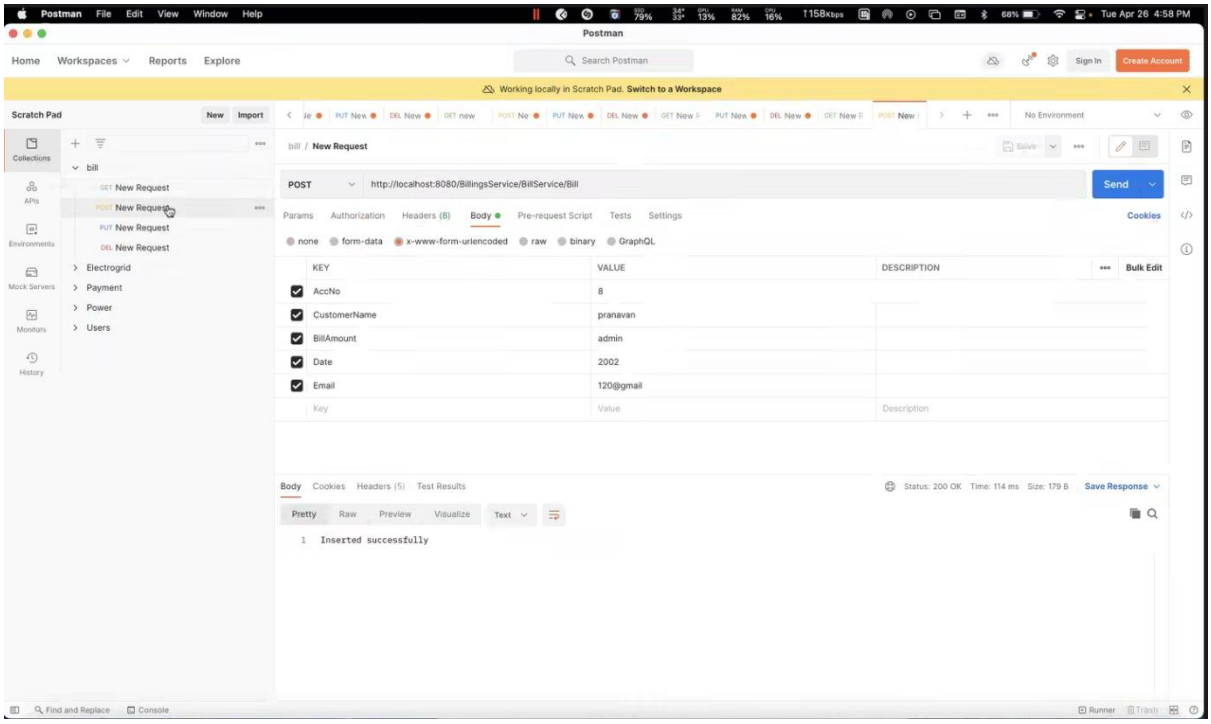


04) Screen Shots of IT20187132

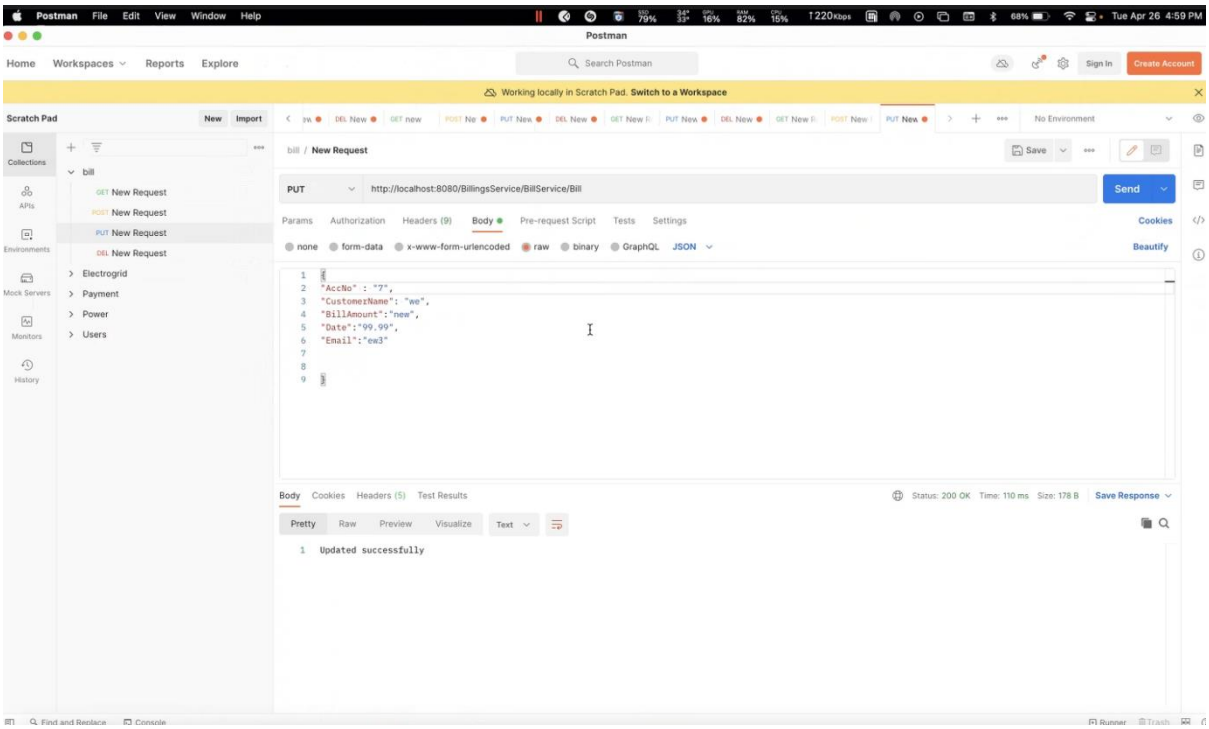
GET method



POST method



PUT method



DELETE method