

Sri Lanka Institute of Information Technology

ElectroGrid (EG) System

# Project Report

Programming Applications and Frameworks Project 2022

Project ID: Y3.S1.WD.IT.02.02

Submitted by:

1. IT20212018 – R.R.W.G.B.P.P.M.Hulangamuwa
2. IT20068400 – Bihash. A
3. IT20110734 – O.G.K.G.Perera
4. IT20253158 – U.R.S.N.Udapolawatta

Submitted to:

………………………….. Mr. Nalaka Dissanayake

2022.04.26

# Table of Contents

[Table of Contents 2](#_TOC_250004)

1. Member Details & Workload Distribution 2
2. Version Controlling - Github Repository 2
3. [SE Methodology/ Methods 3](#_TOC_250003)
4. Time Schedule (Grantt Chart) 3
5. [Requirements 4](#_TOC_250002)
   1. Stakeholder Analysis 4
   2. Requirements Analysis 4
   3. Requirements Modeling 5
6. [System’s overall design 5](#_TOC_250001)
   1. [Overall Architecture 6](#_TOC_250000)
   2. Activity Diagrams 6
   3. Other Relevant Diagrams 6
7. **Individual Sections** Error! Bookmark not defined.
8. **System Integration details** Error! Bookmark not defined.
9. **Appendix** Error! Bookmark not defined.

# 1.Member Details & Workload Distribution

|  |  |
| --- | --- |
| Student ID | IT20212018 |
| Student Name | R.R.W.G.B.P.P.M.Hulangamuwa |
| Workload Distribution | Payment Microservice |

|  |  |
| --- | --- |
| Student ID | IT20068400 |
| Student Name | Bihash.A |
| Workload Distribution | Units & Bill Microservices |

|  |  |
| --- | --- |
| Student ID | IT20110734 |
| Student Name | O.G.K.G.Perera |
| Workload Distribution | Admin Microservice |

|  |  |
| --- | --- |
| Student ID | IT20253158 |
| Student Name | U.R.S.N.Udapolawatta |
| Workload Distribution | User Microservice |

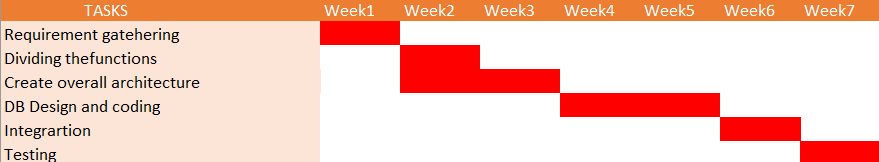
1. **VCS Repository Link:** <https://github.com/PiyumaliHulangamwa/PAF_2022>

## SE Methodology/ Methods

Incremental Development Model is a Software Engineering Methodology where the main requirements are broken down in to multiple modules. Incremental development is done in steps from analysis design, implementation, testing/verification, maintenance. Each module goes through the requirements, design, coding and testing phases.

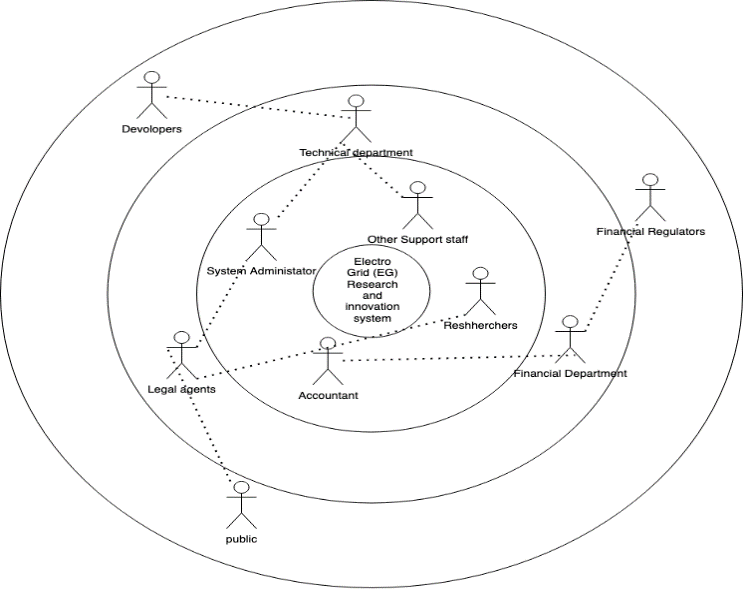
For this project we have used incremental development model since all the requirements are clearly understood and the development team is fairly new microservice are developed separately to minimize errors and fix the easily. So for our project, incremental development method is the best software engineering methodology.

## Time Schedule (Gantt Chart)



**5.Requirements**

### . Stakeholder analysis (onion diagram)



* 1. **Requirements Analysis (Functional, Non-functional, Technical requirements)**

### Functional Requirements

* **User Microservice**

By the admin user can enter to the system. User enter profile details and user has a chance to sign in and edit his/her profile. User can delete his paid bills if they not needed.

* **Admin Microservice**

Admin can manage (add, edit, delete) users by login to the system. Admin can search the payment history by user id.

### Units & Bill Microservices

### Here the admin can enter the particular costs per unit and update the cost. Admin search cost by unit or admin can delete the units. Bill is generated according to the units he/she earned.

* **Payment Microservice**

Admin can enter payment information or update payment information. He/she can search payment receipt by user id and if dues are exceeded the bill amount can delete the payment receipt.

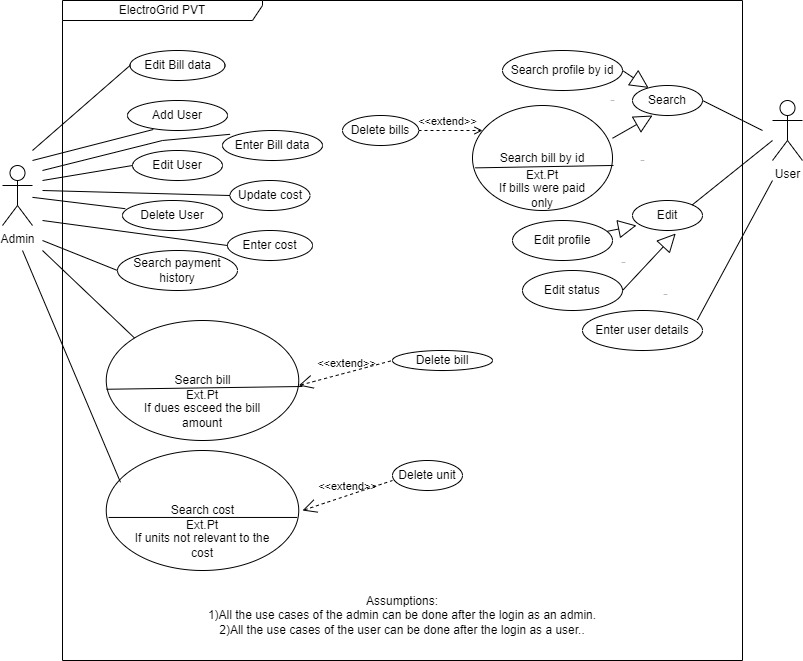
### 5.1.2.Non Functional Requirements

* + - 1. Performance – Response time, throughput, utilization
      2. Security – All the Users end points are protected using a token
      3. Reliability
      4. Scalability
      5. Maintainability

### 5.1.3.Technical Requirements

New browser like Google Chrome, Mozilla Firefox and Opera will support a lot better than the old browsers. (E.g : Internet Explorer)

**5.3. Requirements modelling (Use case diagram)**



**Individual Sections**

|  |  |
| --- | --- |
| Student No : | IT20212018 |
| Student Name : | R.R.W.G.B.P.P.M.Hulangamuwa |
| Microservice: | Payment Microservice |

# Service Design

## API Design Rationale

The Payment microservice is an important section in this ElectroGrid (EG) power grid system.

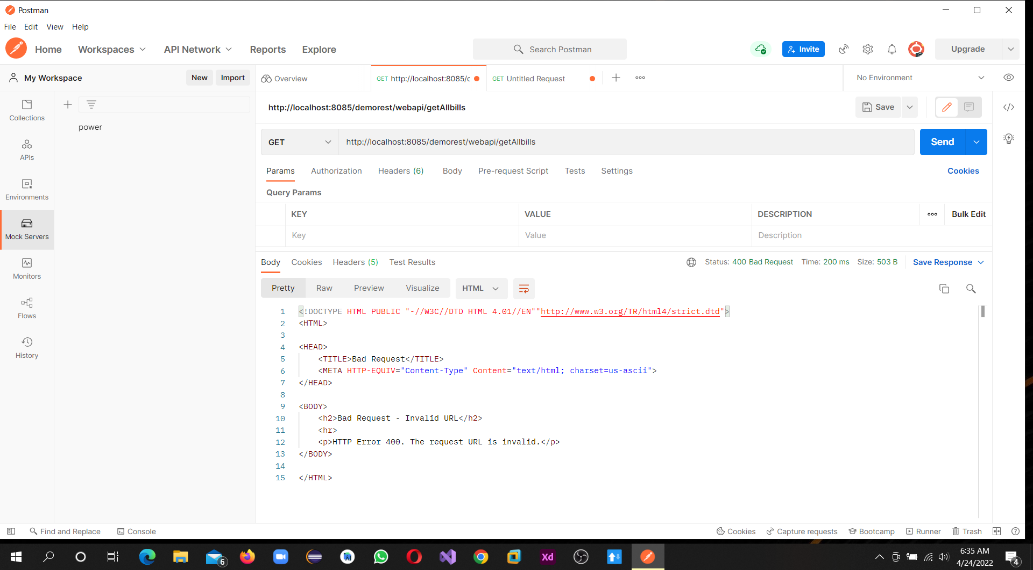
In this Service,

* + Admin can add payment information.
  + Admin can update the payment information.
  + Admin can view the payment receipt by user id.
  + Admin can delete the payment receipt if it exceed the dues over bill amount.
  + If need, Admin can retrieve all the payment Details.

## API Of The Service

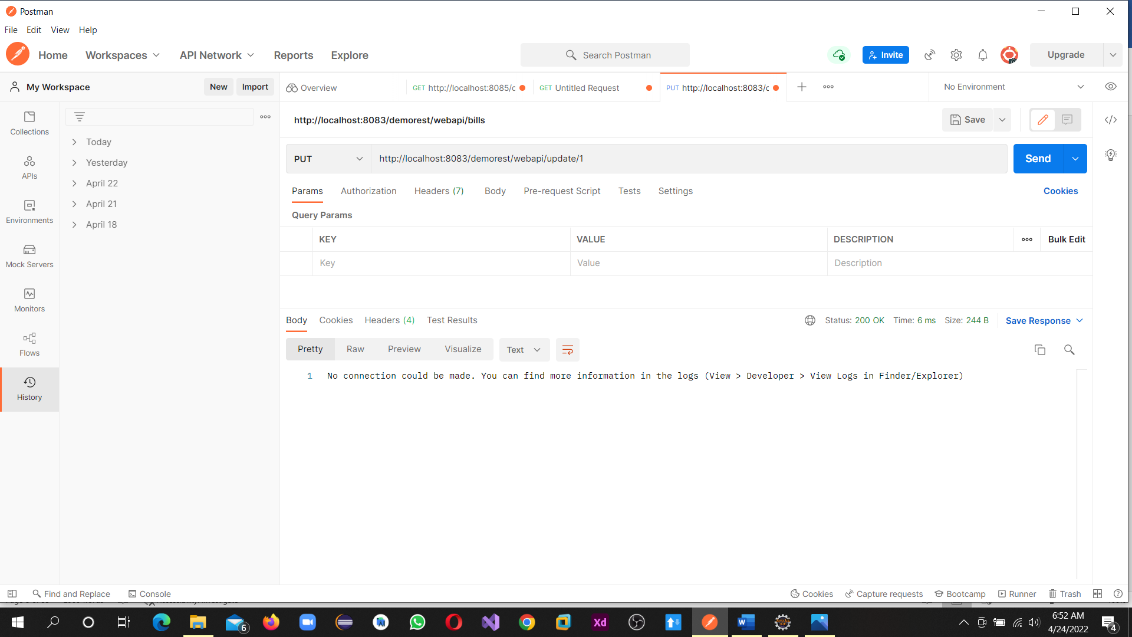
### API for view Payment Details -GET Request

URL : http://localhost:8083/demorest /webapi/getAllbills



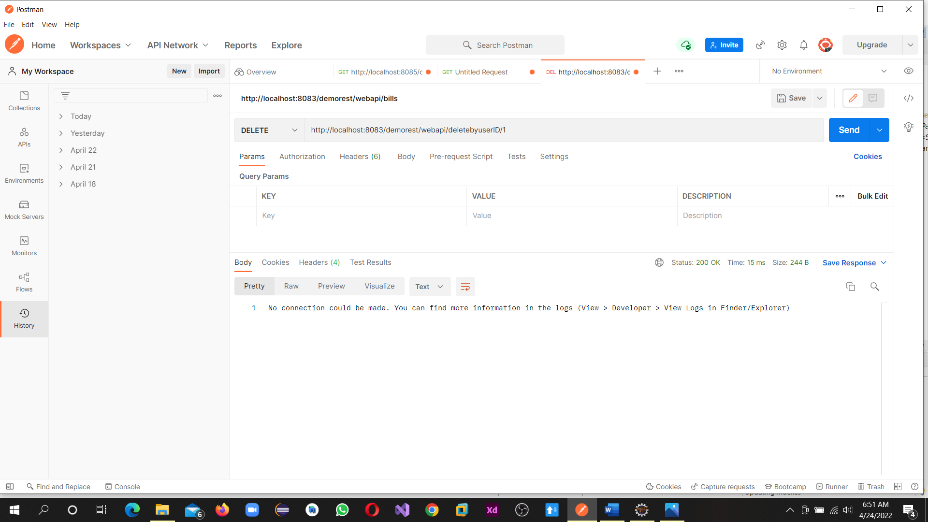
### API for update Payment details -PUT Request

### URL : http://localhost:8083/demorest /webapi/update /1

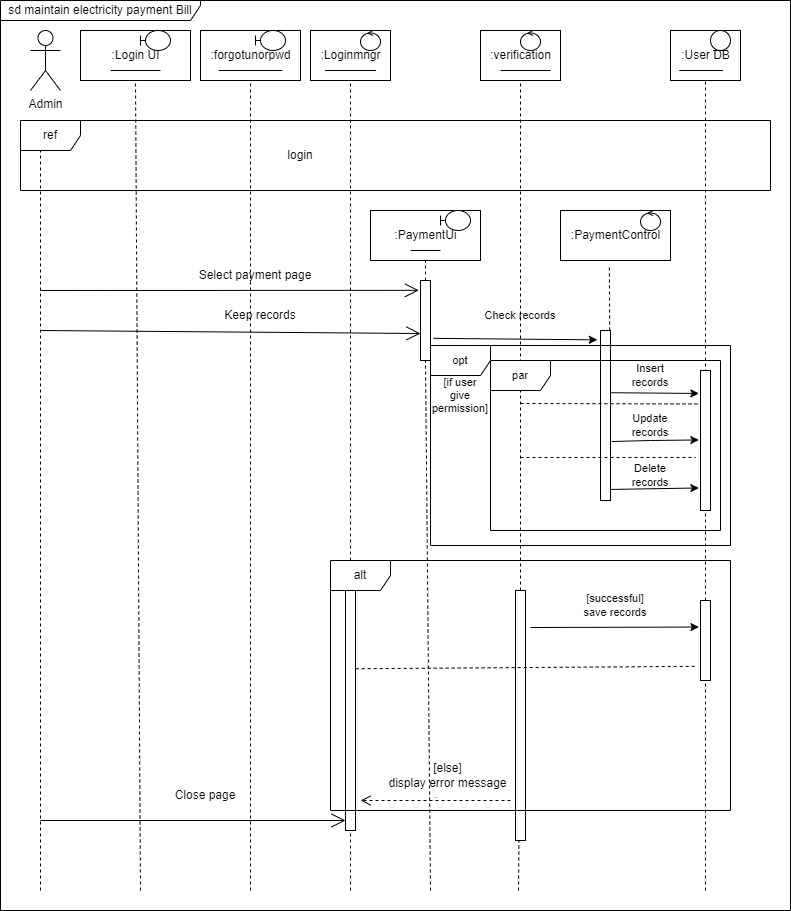
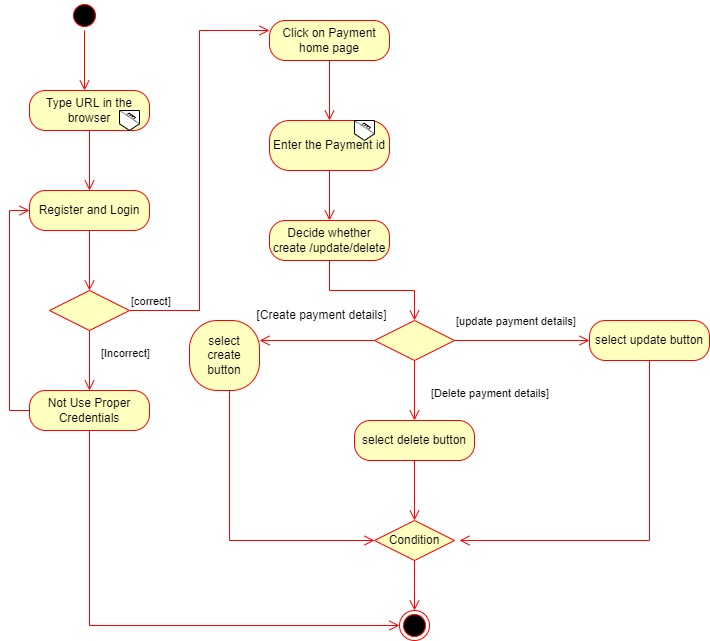


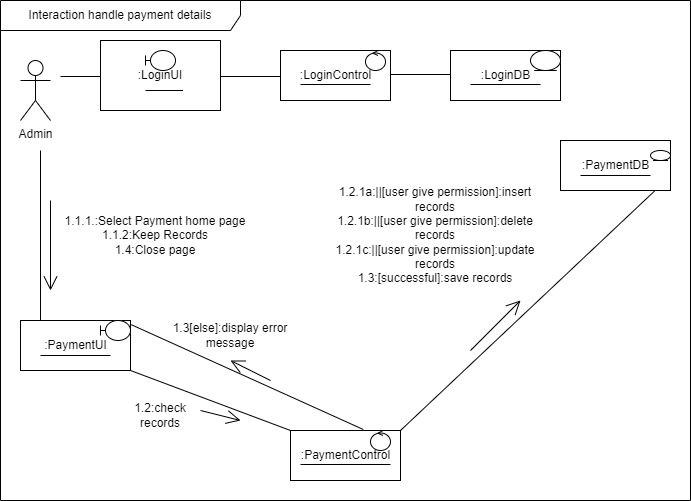
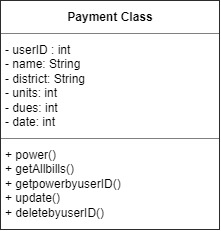
* 1. **API for Delete Payment receipt -DELETE Request**

URL : http://localhost:8083/demorest /webapi/deletebyuserID/1



## Internal Logic/ Database Design Diagrams of Payment Microservice

* 1. **Service Development**
  2. **Tools Used**
     + Technologies Used : Java - JAX-RS (Jersey) on Tomcat

Reason: -Familiarity with the language, easy server configuration

* + - IDE : Eclipse 4.18

Reason: -User friendly IDE

* + - Database : MySQL

Reason: - Less complex to implement the DB connection

* Testing : POSTMAN
  1. **Testing Methodology and Results**

|  |  |
| --- | --- |
| Test Id | 01 |
| Test Description | Insert payment details |
| Test Input(s) | userID=5,name=sunil,district=kandy,units=200,dues=2000,date=2012-09-08 |
| Expected output(s) | Inserted successfully |
| Actual output(s) | Bad Request |
| Result(PASS/FAIL) | FAIL |

|  |  |
| --- | --- |
| Test Id | 02 |
| Test Description | Retrieve payment details |
| Test Input(s) | userID=5 |
| Expected output(s) | userID=5,name=sunil,district=kandy,units=200,dues=2000,date=2012-09-08 |
| Actual output(s) | No Connection |
| Result(PASS/FAIL) | FAIL |

|  |  |
| --- | --- |
| Test Id | 03 |
| Test Description | Update payment details |
| Test Input(s) | userID=5,name=sunil,district=kandy,units=200,dues=2000,date=2012-09-08 |
| Expected output(s) | userID=5,name=Ann,district=Colombo,units=200,dues=2000,date=2012-09-08 |
| Actual output(s) | No Connection |
| Result(PASS/FAIL) | FAIL |

|  |  |
| --- | --- |
| Test Id | 04 |
| Test Description | Delete payment receipt which exceeded the dues over the bill amount |
| Test Input(s) | Payment id=2 |
| Expected output(s) | Deleted successfully |
| Actual output(s) | No Connection |
| Result(PASS/FAIL) | FAIL |

|  |  |
| --- | --- |
| Student No : | IT20068400 |
| Student Name : | Bihash.A |
| Microservice : | Units & Bill Microservices |

# Service Design

## API Design Rationale

## Units Service-

API Design Rationale the Unit Part is an important section in this PowerGrid innovative

project Electricity system. User have to login first and then go to the unit system. Then

they can do this kind of functions. That functions are below in the section,

·Admin can insert new Unit Details.

 · Admin can update unit details.

· Admin can view their Unit details with User ID.

· Admin can delete their Units.

## Billing Service-

API Design Rationale the billing Part is also an important section in this PowerGrid innovative

project Electricity system. User have to login first and then go to the billing system. Then they

can do this kind of functions. That functions are below in the section,

·Admin can insert new Bill Details.

  · Admin can update Bill details.

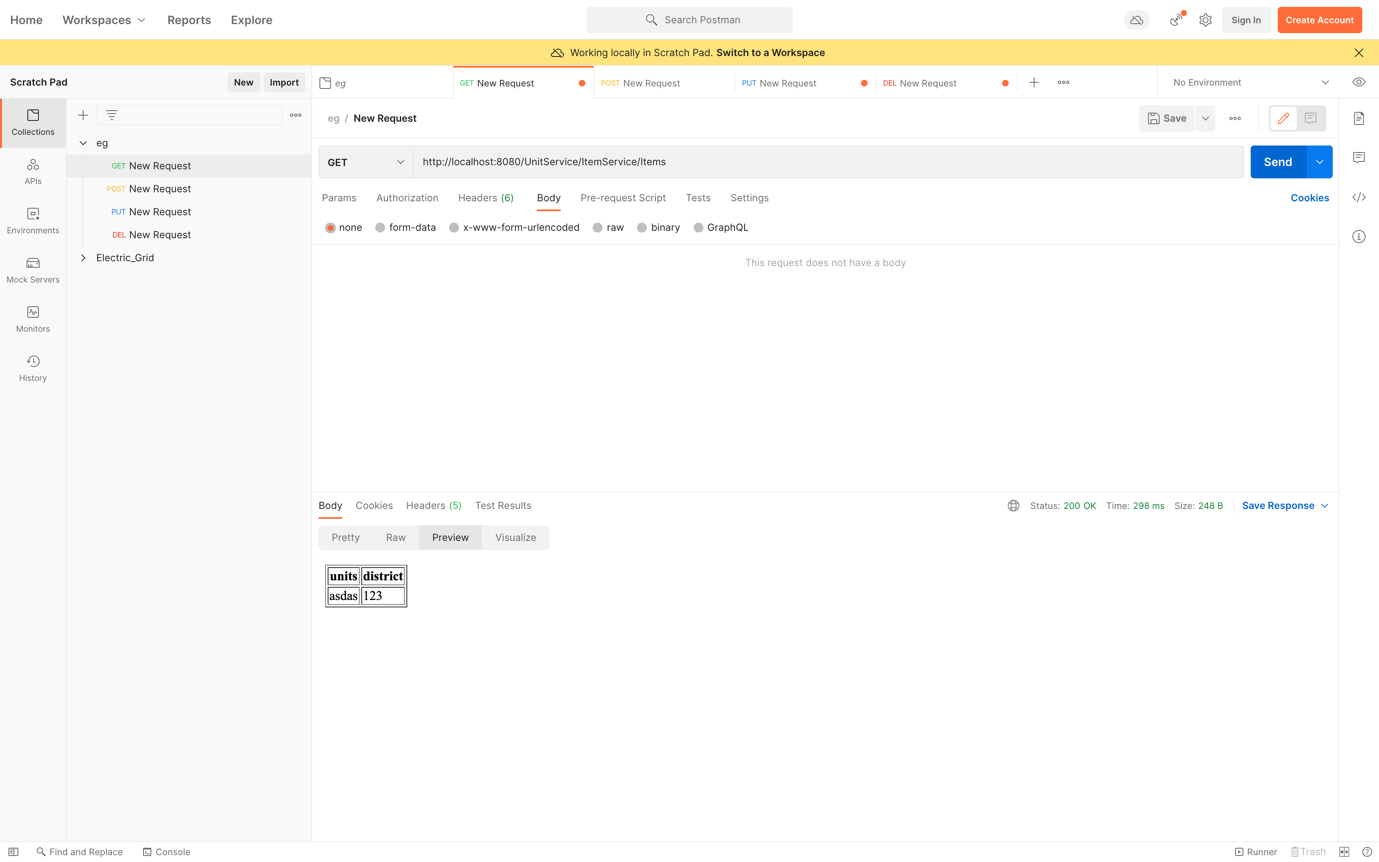
· Admin can view their bill details with User ID.

· Admin can delete their Bills

## API Of The Service

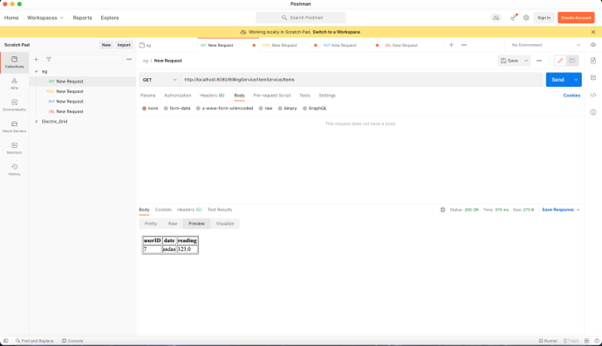
### API for view unit details (GET Request)

### URL : http://localhost:8080/UnitService/UnitService/Unit



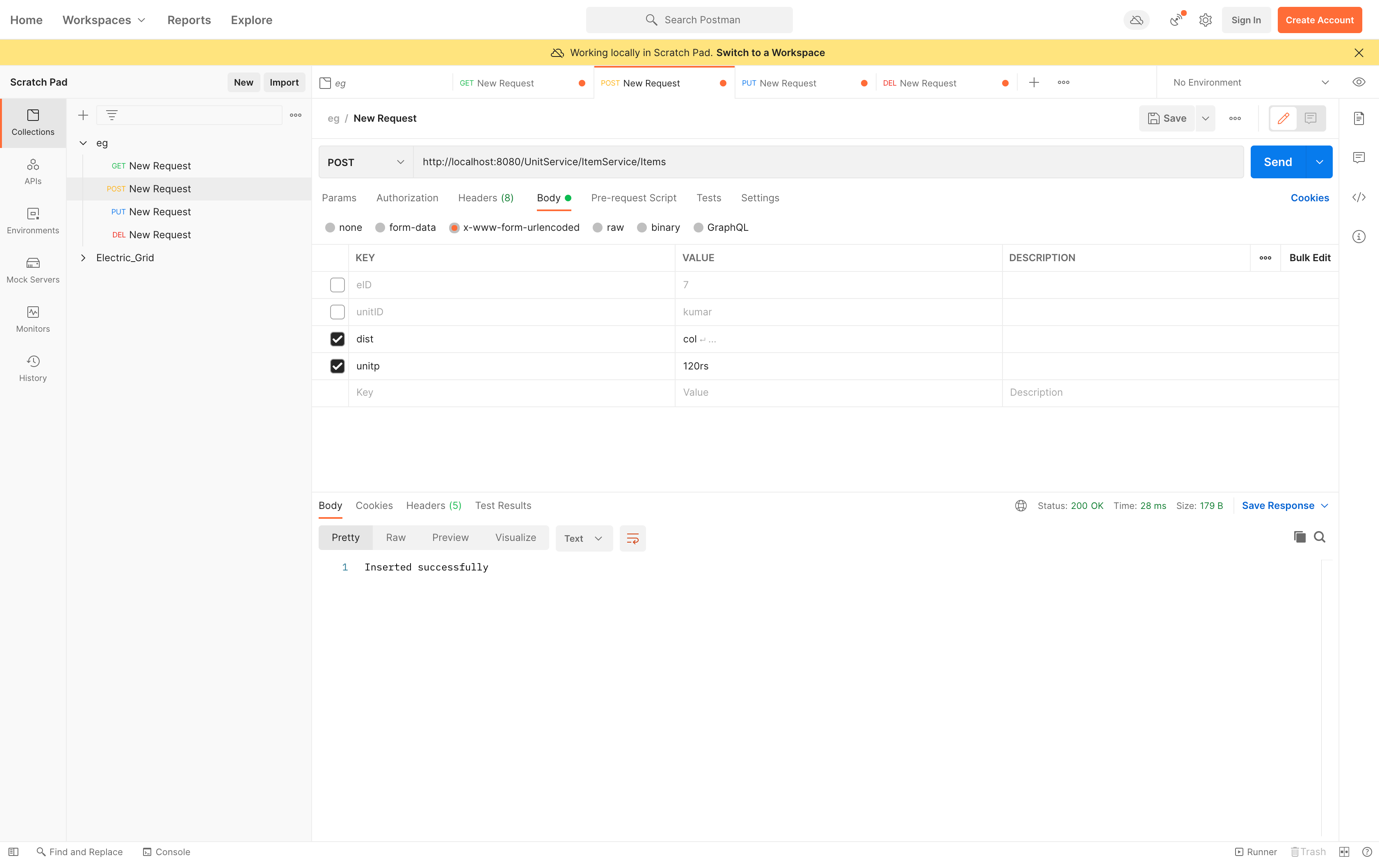
### 2.2. API for view billing details (GET Request)

### url: http://localhost:8080/BillingService/BillService/Bill



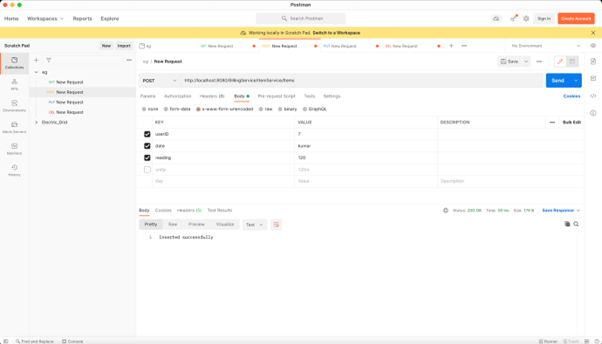
### API for add unit details (POST Request)

### URL : http://localhost:8080/UnitService/UnitService/Unit



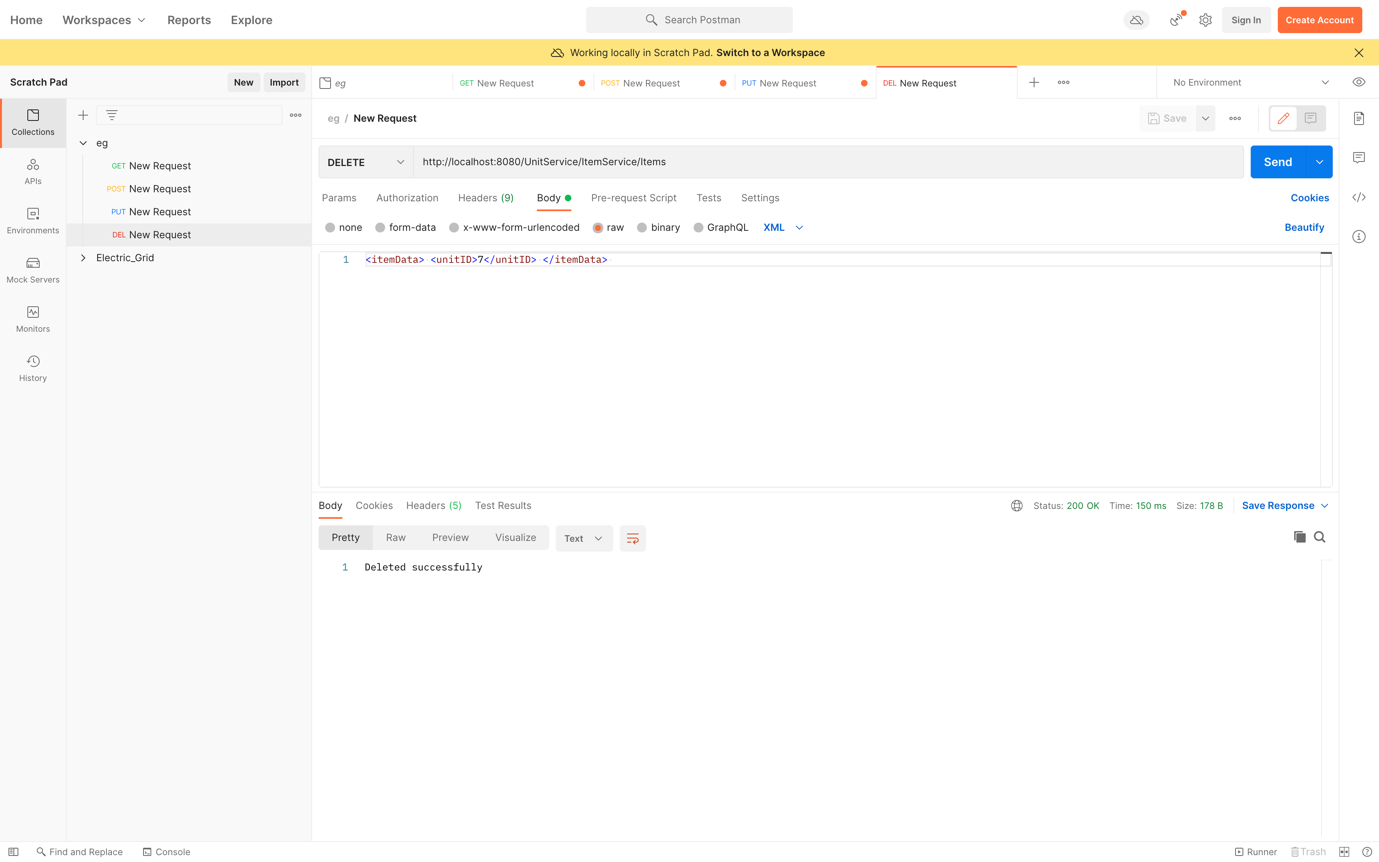
### 2.4.API for add billing details (POST Request)

### url: http://localhost:8080/BillingService/BillService/Bill



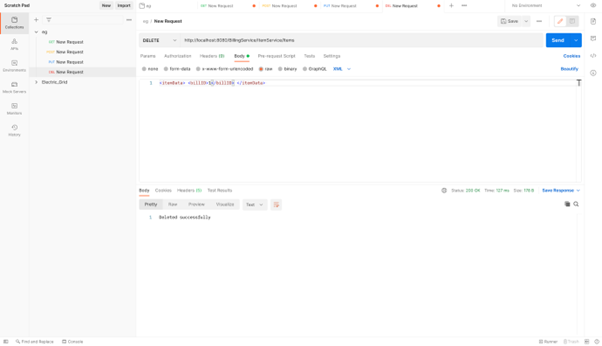
### 2.5.API for remove unit details (DELETE Request)

### URL : http://localhost:8080/UnitService/UnitService/Unit



### 2.6.API for remove billing details (DELETE Request)

### url: http://localhost:8080/BillingService/BillService/Bill



### 2.7.API for update unit details (PUT Request)

### URL : http://localhost:8080/UnitService/UnitService/Unit

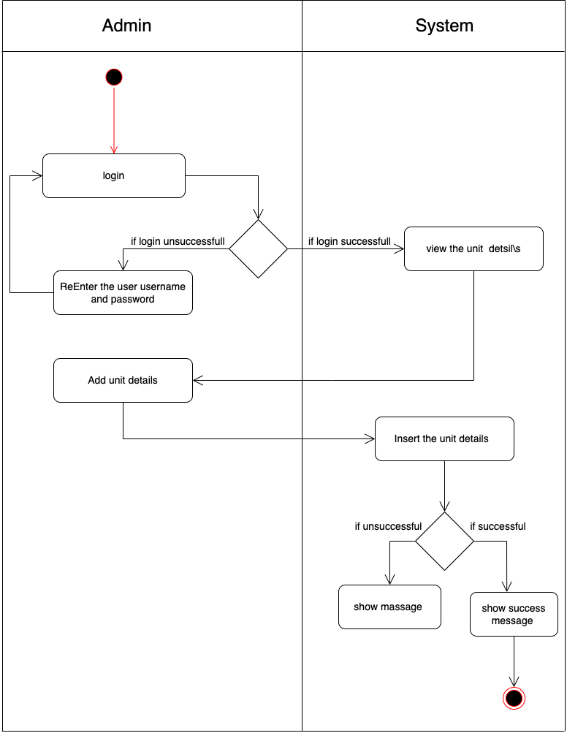
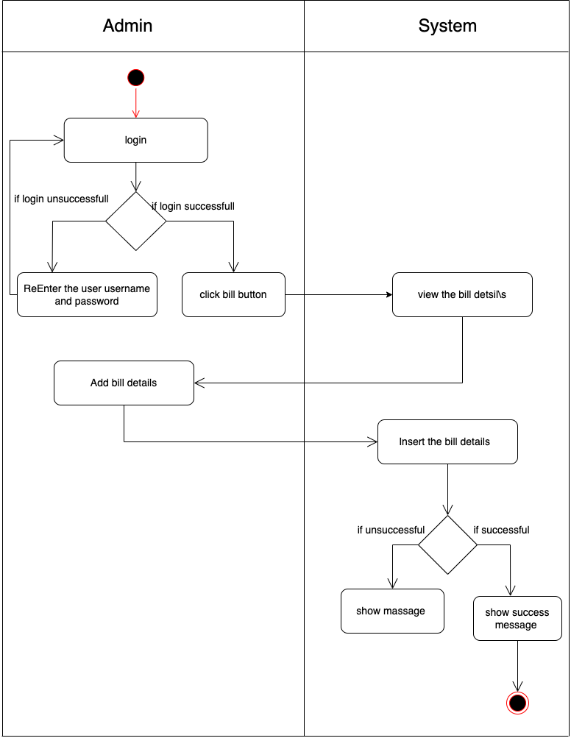
### 

### 2.8.API for update billing details (PUT Request)

### url: http://localhost:8080/BillingService/BillService/Bill

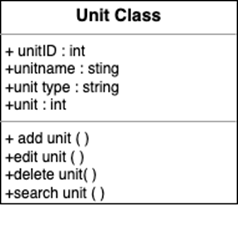
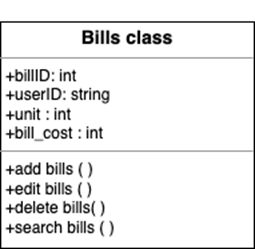
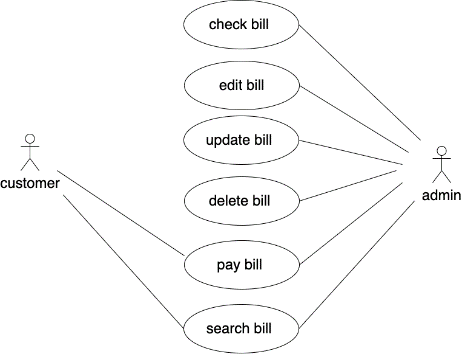
### 

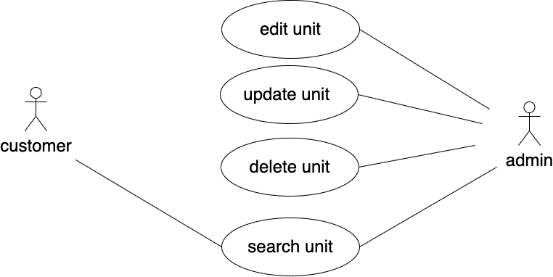
## Internal Logic/ Database Design Diagrams of Units & Bill Microservice

** **

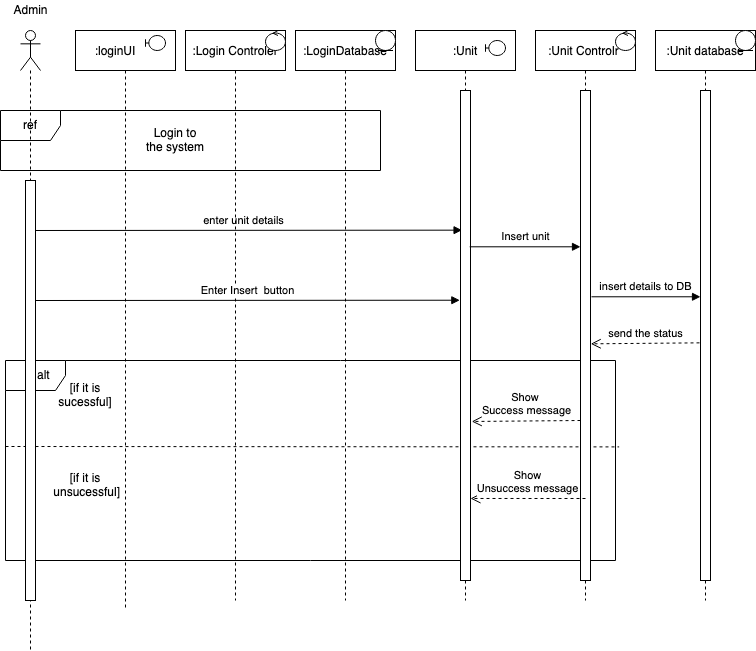
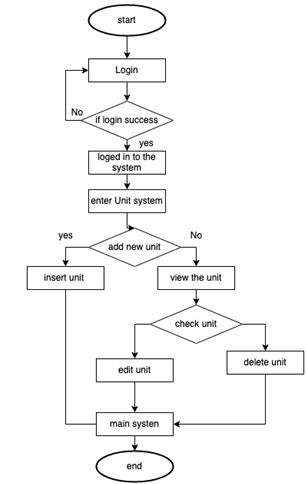
**Activity diagram-Insert Unit Details Activity diagram-Insert Bill Details**

**Insert Bill Details**

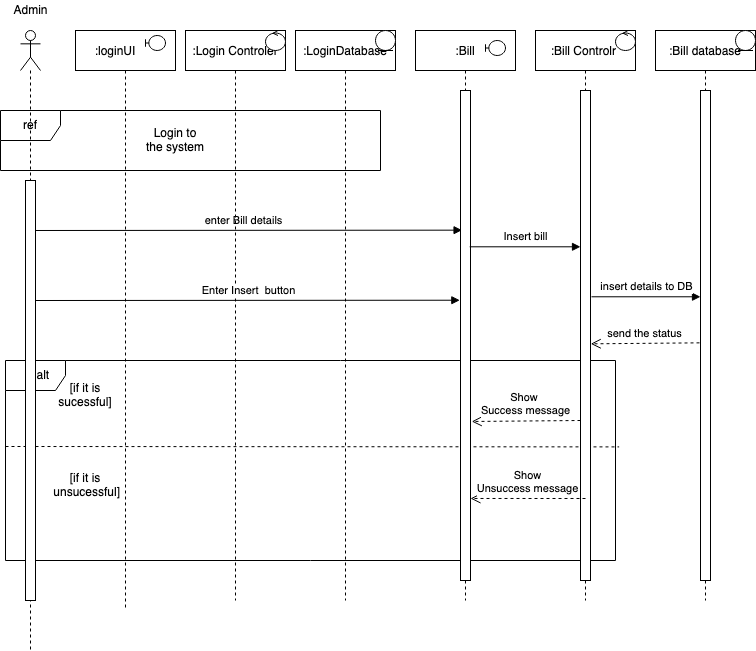
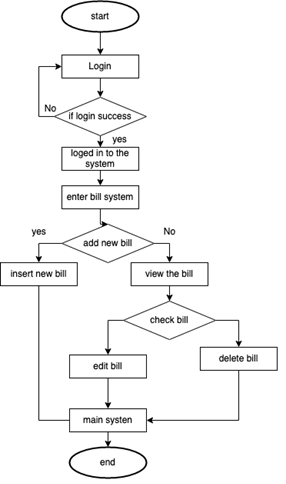
  



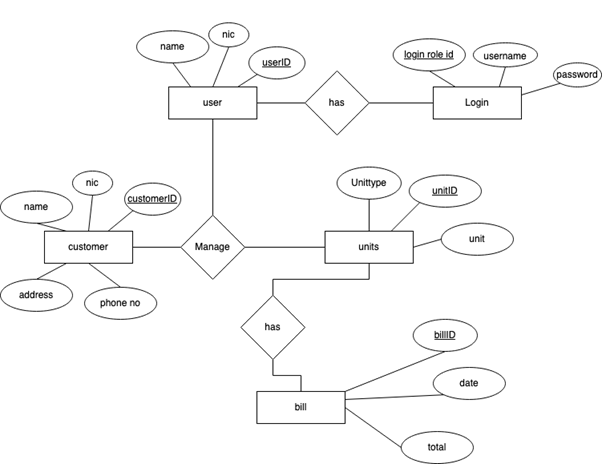
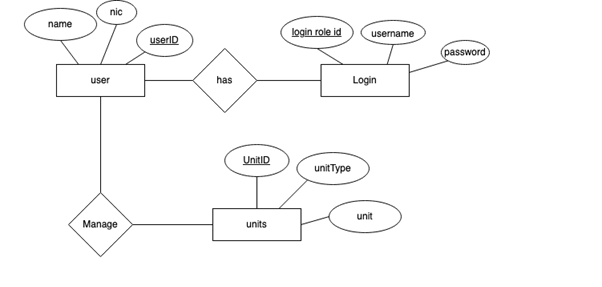
**Insert Unit Details**

**Sequence Diagram-Unit Service Flow chart-Unit Service**

**Sequence Diagram- Billing Service Flow chart- Billing Service**



**ER Diagram- Unit Service ER Diagram- Billing Service**

## Service Development And Testing

### Tools Used

* + - Technologies Used : Java - JAX-RS (Jersey) on Tomcat
    - IDE : Eclipse 21.12
    - Database : MySQL
    - Testing : POSTMAN
  1. **Testing Methodology and Results**

**Unit Service**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test description | Test input | Expected output | Actual output | Result |
| 01 | Adding a new Unit details in to the DB (testing postman) | { "unittype": "12", "unit": "120" } | { "unittype": "12", "unit": "120" } | { "unittype": "12", "unit": "120" } | Pass |
| 02 | Update the  unit details it show the success message. | { "unittype": "12", "unit": "100" } | { "unittype": "12", "unit": "100" } | { "unittype": "12", "unit": "100" } | Pass |
| 03 | Delete the account using the particular Unitid | <unitId>1</unitId> | Show Message as  “ Deleted Successfully” | Unit Details was Deleted Message : “ Deleted Successfully” | Pass |
| 04 | Display  the details from the entering the UnitID. | { "unittype": "12", "unit": "100" } | |  |  |  | | --- | --- | --- | | unitID | unittype | unit | | 3 | 12 | 100 | | |  |  |  | | --- | --- | --- | | unitID | unittype | unit | | 3 | 12 | 100 | | Pass |

**Billing Service**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| est ID | Test descr T iption | Test input | Expected output | Actual output | Result |
| 01 | Adding a new Bill details in to the DB (testing postman) | { "date": "12.05.2022", "total": "1430.00""unit": "120" } | { "date": "12.05.2022", "total": "1430.00""unit": "120" } | { "date": "12.05.2022", "total": "1430.00""unit": "120" } | Pass |
| 02 | Update the Bill details it show the success message. (change the date) | { "date": "20.05.2022", "total": "1430.00""unit": "120" } | { "date": "20.05.2022", "total": "1430.00""unit": "120" } | { "date": "20.05.2022", "total": "1430.00""unit": "120" } | Pass |
| 03 | Delete the account using the particular Billid. | <billId>1</billId> | Show Message as  “ Deleted Successfully” | Unit Details was Deleted Message : “ Deleted Successfully” | Pass |
| 04 | Display  the details from the entering the Billid. | { "date": "20.05.2022", "total": "1430.00""unit": "120" } | |  |  |  |  | | --- | --- | --- | --- | | unitID | date | total | unit | | 3 | 20.05.2022 | 1430.00 | 100 | | |  |  |  |  | | --- | --- | --- | --- | | unitID | date | total | unit | | 3 | 20.05.2022 | 1430.00 | 100 | | Pass |

|  |  |
| --- | --- |
| Student No : | IT20110734 |
| Student Name : | O.G.K.G Perera |
| Microservice: | Admin Microservice |

## Service Design

### API Design Rationale

Rationale The Funding body is an important section in this PoweGrid innovative project

Electricity system. User can register using login if the user is new to the system they can

signup and made an account.

· Users can register using login id user is the new for the system they can sign up from the system.

· Admin can update their profile details.

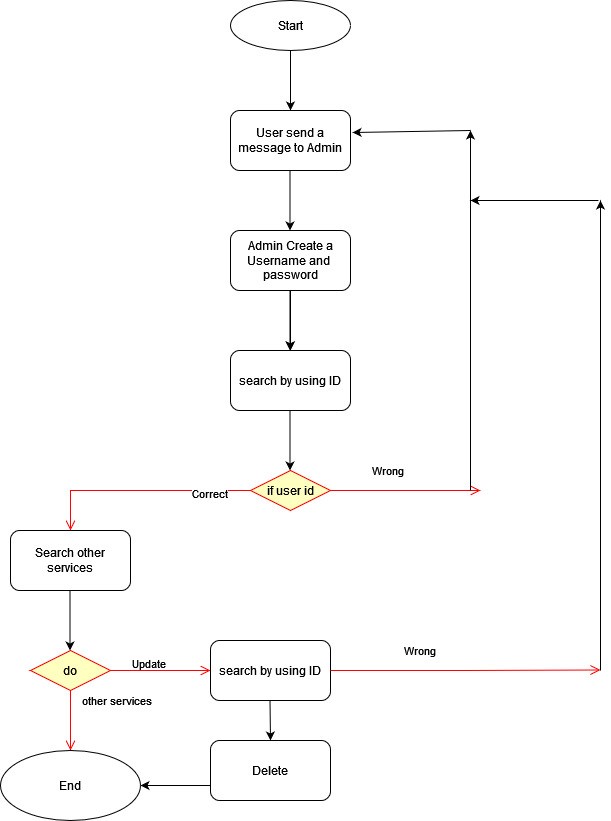
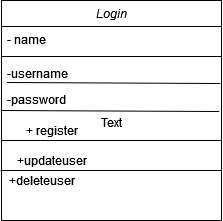
· Admin can view their profile details using user ID.

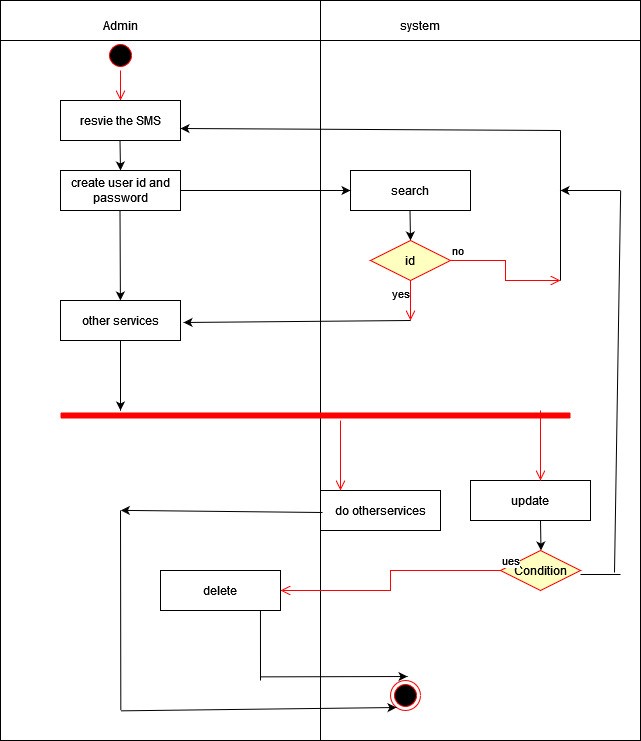
· Admin can delete their user.

**2.Internal logic of Admin Microservice**

Diagram, schematic

Description automatically generated



## 3.Service Development And Testing

### 3.1.Tools Used

* + - Technologies Used : Java - JAX-RS (Jersey) on Tomcat
    - IDE : Eclipse 21.12
    - Database : MySQL
    - Testing : POSTMAN

4.**Testing Methodology and Results**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test ID | Test description | Test input | Expected output | Actual output | Result |
| 01 | Adding a new user name and password (Register) to the Database | { "userName": "Mahendra Costa", "password": "mahi#123",  } | { "password": "mahi#123", "regiteredAt": "A few seconds ago", "id": 8 "userName": "Mahendra Costa", , "updatedAt": null } | { "password": "mahi#123", "regiteredAt": "A few seconds ago", "id": 8 "userName": "Mahendra Costa", , "updatedAt": null } | Pass |
| 02 | Update the user name or a password it show the successful popup message. | { "userName": "Mahendra Munasinghe", "password": "mahendra@1343"  } | { "password": "mahendra@1343", "regiteredAt": null, "id": 8, "userName": "Mahendra Munasinghe", , "updatedAt": "A few seconds ago" } | { "password": "mahendra@1343", "regiteredAt": null, "id": 8, "userName": "Mahendra Munasinghe", , "updatedAt": "A few seconds ago" } | Pass |
| 03 | Delete the account using the particular user id | User Id {4} | Show Message as “User Account Deleted Successfully” | User Details are Deleted Message : “Funding Body Account Deleted Successfully” | Pass |
| 04 | Display the details from the entering the Username. | {“UserName”:”Mahendra Munasinghe”} | { "password": "67hb435gf", "regiteredAt": "2021-04-24 00:29:36",  "id": 1,  "FullName": "Amila Sandaruwan", "email": "amila@gmail.com", "updatedAt": null }, | { "password": "67hb435gf", "regiteredAt": "2021-04-24 00:29:36", "id": 1, "Full Name": "Amila Sandaruwan", "email": "amila@gmail.com", "updatedAt": null }, | Pass |
| 05 | View of the all users | View of all users | {"password": "67hb435gf", "regiteredAt": "2021-04-24 00:29:36", "id": 1, "Full Name": "Amila Sandaruwan", "email": "amila@gmail.com", "updatedAt": null },  { "password": "mahendra@1343", "regiteredAt": "2021-04-25 16:40:43", "id": 8, "Full Name": "Mahendra Munasinghe", "email": "mahendra@gmail.com", "updatedAt": null }  Etc … | {"password": "67hb435gf", "regiteredAt": "2021-04-24 00:29:36", "id": 1, "Full Name": "Amila Sandaruwan", "email": "amila@gmail.com", "updatedAt": null },  { "password": "mahendra@1343", "regiteredAt": "2021-04-25 16:40:43", "id": 8, "Full Name": "Mahendra Munasinghe", "email": "mahendra@gmail.com", "updatedAt": null }  Etc … | Pass |

|  |  |
| --- | --- |
| Student No: | IT20253158 |
| Student Name: | U.R.S.N.Udapolawatta |
| Microservice: | User Microservice |

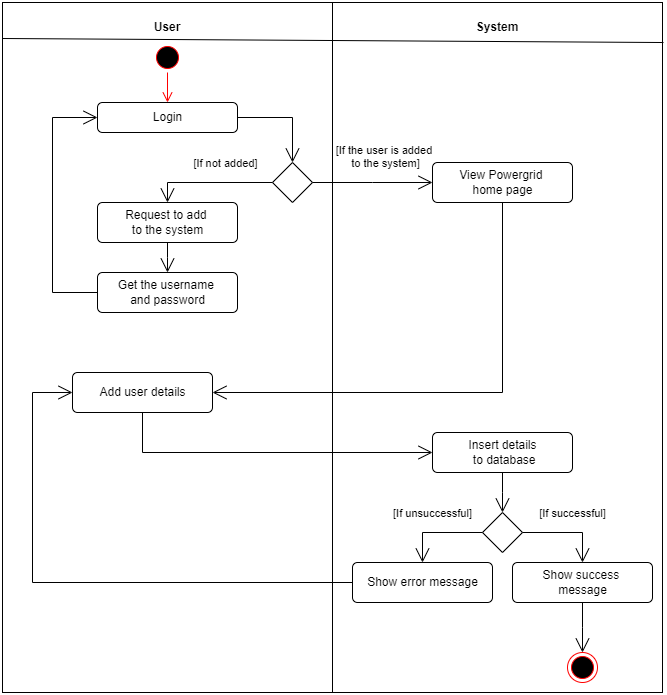
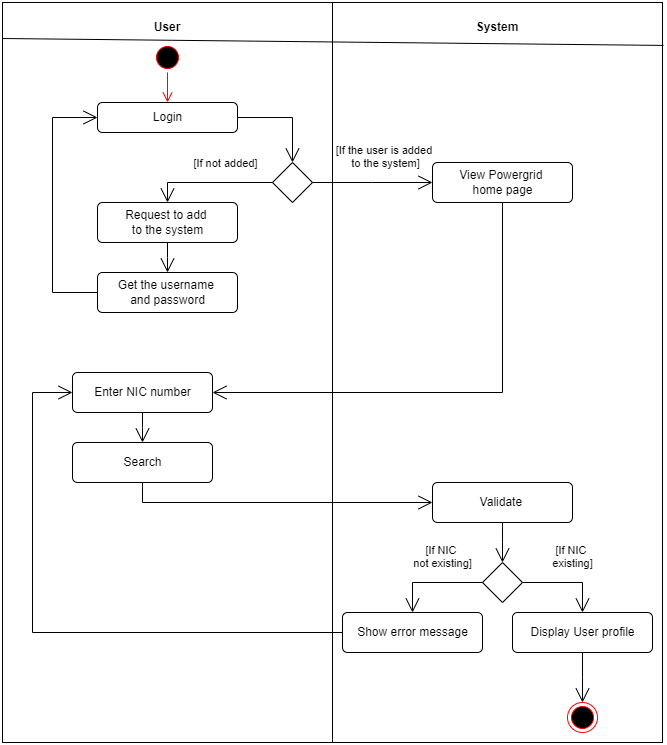
# Service Design

## API Design Rationale

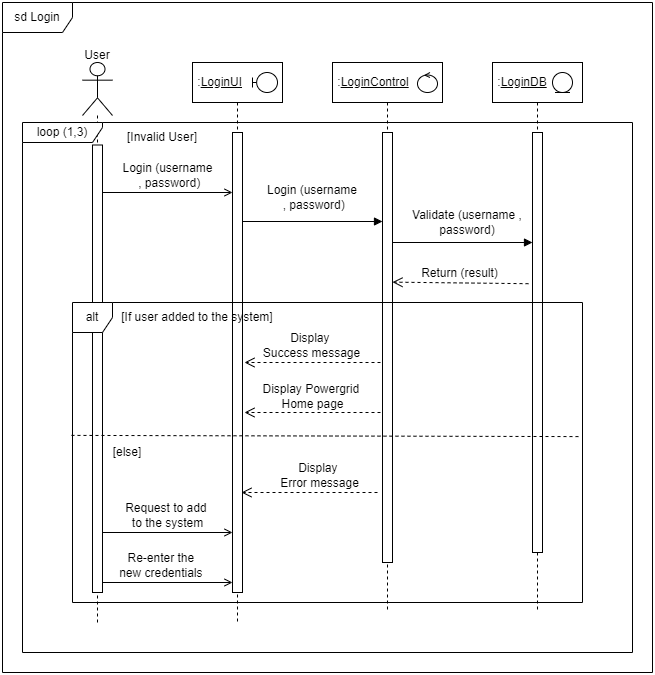
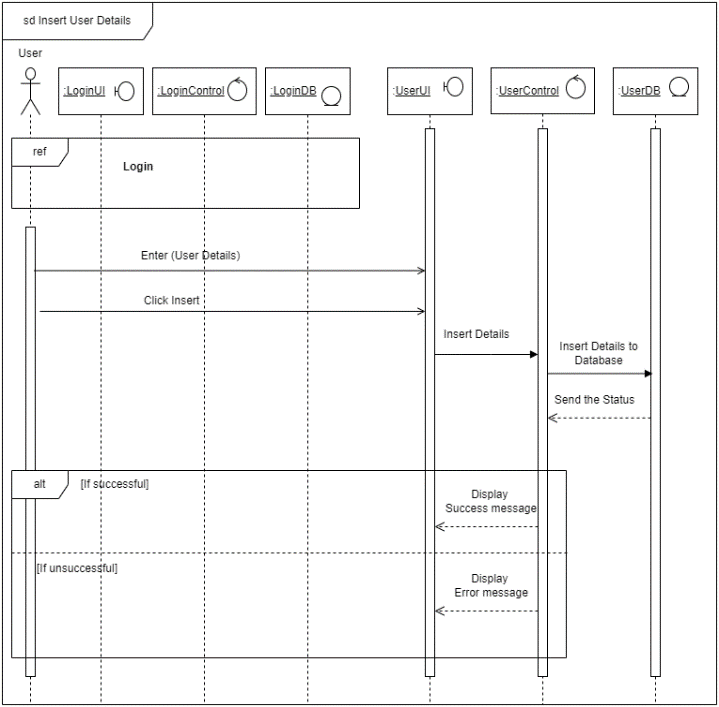
In this Service,

* Users can login to the system entering the username and password created by the admin.
* Users can also add and update their profile.
* Users can view their profile after they search it by id.
* Also, users can view the paid bills when search by bill id.
* Users can delete the paid bills.

**2**.**Internal Logic/ Database Design Diagrams of User Microservice**

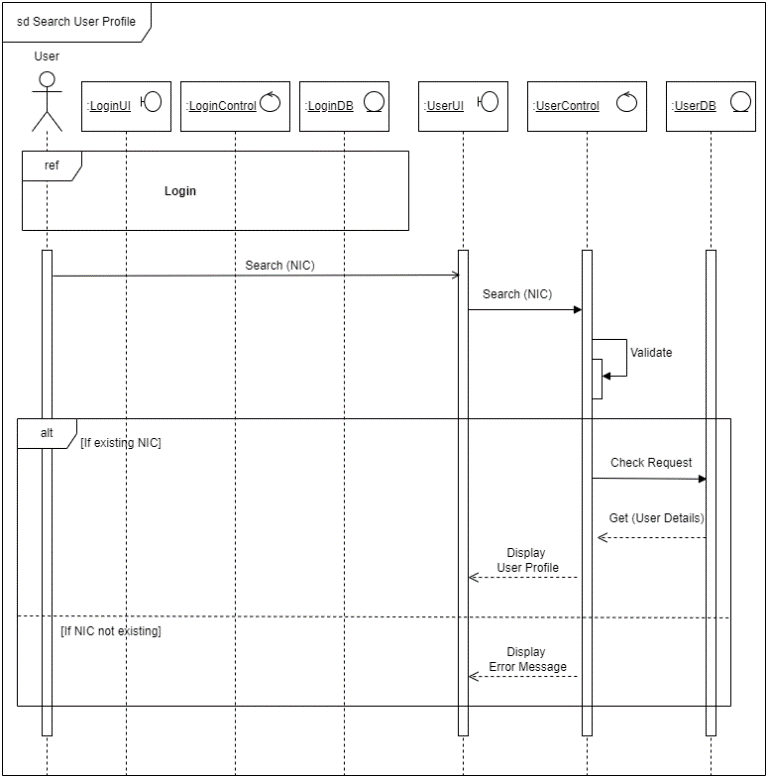
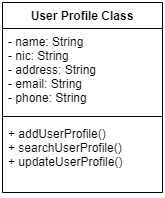
 

**Activity Diagram** - Insert User Details **Activity Diagram** – Search profile by Id

** **

### Sequence Diagram – Login Sequence Diagram -Insert User Details

### 

** **

**Sequence Diagram** – Search User Profile

## 3.Service Development And Testing

### 3.1.Tools Used

* + - Technologies Used : Java - JAX-RS (Jersey) on Tomcat
    - IDE : Eclipse 21.12
    - Database : MySQL
    - Testing : POSTMAN

**3.2**.**Testing Methodology and Results**

|  |  |
| --- | --- |
| Test ID | U01 |
| Description | Add user details to a new profile |
| Test Inputs | {   "name": “Nimal Amarasena”,   "nic": “8526351456V”,   "address": "No 822/A, Mahawa Road, Kurunegala",   "email": “nimalamara@gmail.com”,   "phone": “0775684522”   } |
| Expected Output | User details are added to the database and display result    {   "name": “Nimal Amarasena”,   "nic": “8526351456V”,   "address": "No 822/A, Mahawa Road, Kurunegala",   "email": “nimalamara@gmail.com”,   "phone": “0775684522”   } |
| Actual Output | User details are added to the database and display result    {   "name": “Nimal Amarasena”,   "nic": “8526351456V”,   "address": "No 822/A, Mahawa Road, Kurunegala",   "email": “nimalamara@gmail.com”,   "phone": “0775684522”   } |
| Result (Pass/Fail) | Pass |

|  |  |
| --- | --- |
| Test ID | U02 |
| Description | Update user profile details |
| Test Inputs | {  "name": “Nimal Sanju Amarasena”,  "address": "No 822/A, Uduyaya Road, Kurunegala",  "phone": “0714550023”  } |
| Expected Output | User details are updated in the database and display result    {   "name": “Nimal Sanju Amarasena”,   "nic": “8526351456V”,   "address": "No 822/A, Uduyaya Road, Kurunegala",   "email": “nimalamara@gmail.com”,   "phone": “0714550023”   } |
| Actual Output | User details are updated in the database and display result    {   "name": “Nimal Sanju Amarasena”,   "nic": “8526351456V”,   "address": "No 822/A, Uduyaya Road, Kurunegala",   "email": “nimalamara@gmail.com”,   "phone": “0714550023”   } |
| Result (Pass/Fail) | Pass |

|  |  |
| --- | --- |
| Test ID | U03 |
| Description | Search user profile by Id |
| Test Inputs | {  “nic”:”6623884801V”  } |
| Expected Output | User details are retrieved from the database and display result    {   "name": “Rasika Ranathunga”,   “nic”:”6623884801V”,   "address": "No 27/A, Temple Road, Dehiwala",   "email": “rasikarr@gmail.com”,   "phone": “0715521131”   } |
| Actual Output | User details are retrieved from the database and display result    {   "name": “Rasika Ranathunga”,   “nic”:”6623884801V”,   "address": "No 27/A, Temple Road, Dehiwala",   "email": “rasikarr@gmail.com”,   "phone": “0715521131”   } |
| Result (Pass/Fail) | Pass |
| Test ID | U04 |
| Description | Delete paid bills |
| Test Inputs | {  “bill id “:4  } |

|  |  |
| --- | --- |
| Expected Output | Paid bill  is deleted and show result as    "Successfully Deleted the Bill" |
| Actual Output | Paid bill  is deleted and show result as    "Successfully Deleted the Bill" |
| Result (Pass/Fail) | pass |