ElectroGrid (EG) Project
Batch Group Y3.S1.WD.IT.01.01
Assignment Group 62

Registration No	Name
IT20145934	Wanasooriya W.M.D.C.
IT20143886	Samarawickrama H.N.
IT20146474	Liyanage A.L.D.K.S
IT20154912	Sahan A.K

Group work

1. **Member details and role**

Student ID	Name	Web service	Description
IT20145934	Wanasooriya W.M.D.C.	Payment Management	❖ Add a Payment Details ❖ View all Payment Details ❖ Update the Payment Details ❖ Delete a Payment Details
IT20143886	Samarawickrama H.N.	Employee Management	❖ Add an Employee Details ❖ View all Employee Details ❖ Update the Employee Details ❖ Delete an Employee Details
IT20146474	Liyanage A.L.D.K.S	Registration Management	❖ Add a Registration Details ❖ View all Registration Details ❖ Update the Registration Details ❖ Delete a Registration Details
IT20154912	Sahan A.K	Billing Management	❖ Add a Billing Details ❖ View all Billing Details ❖ Update the Billing Details ❖ Delete a Billing Details

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

Micro-service	Functional Requirement	Non-Functional Requirement	Technical Requirement
Payment Management	- Add a Payment Details - View all Payment Details - Update the Payment Details - Delete a Payment Details	- Security and Privacy - Performance - Response in time - Maintainability - Data Integrity	After confirming the relevant payment, Customer can add payment details such as payment method(visa/master), card details (card number, name on card, CVC, expire date), bill amount to go ahead with procedure. System will make sure to validate payment details by avoiding null values and inaccurate data formats. Consumer can view all his/her payment details by payment id. Admin can manage payment details by updating payment details and deleting unnecessary payment details of the system. Payment details will be saved after payment get authorized.
Employee Management	- Add an Employee Details - View all Employee Details - Update the Employee Details - Delete an Employee Details	- Security and privacy - Scalability - High performance	The administrator can enter employee details. Each employee has a uniquely identifiable identity number. The administrator can view, update, delete and enter details about an employee. Only the administrator can perform the details entry, update, and deletion operations.
Registration Management	- Add a Registration Details - View all Registration Details - Update the Registration Details - Delete a Registration Details	- Security and privacy - Scalability - High performance - Recoverability - Availability	There are mainly 3 types of users in the system. Customer, admin, and manager. To check the bill balance, monthly electricity usage, overdue payment information through the system Customer must be registered to the system by the admin. When registering to the system as a valid Customer should provide name, address, email, phone number. The administrator of the system will check the details and accept their registering request. After that they can login to the system by providing valid credentials and check their electricity usage and do payments. As same the Customer, the manager also must register to the system to calculate bills, accept payments, and manage overdue information. The managers are added by the administrator of the system. Moreover, the administrator of the system can search for users, update details of the users and remove inactive users from the system.
Billing Management	- Add a Billing Details - View all Billing Details - Update the Billing Details - Delete a Billing Details	- Security and privacy - Scalability - High performance	The administrator can enter the billing details to the system. Each bill has a unique bill id. If the administrator wants to update or delete bill details, he can do it through the billing management function.

3. Clickable link for GitHub Repository

https://github.com/IT20146474/ElectroGrid-EG-G62.git

4. SE Methodology/Methods - Agile Software Development

What is Agile?

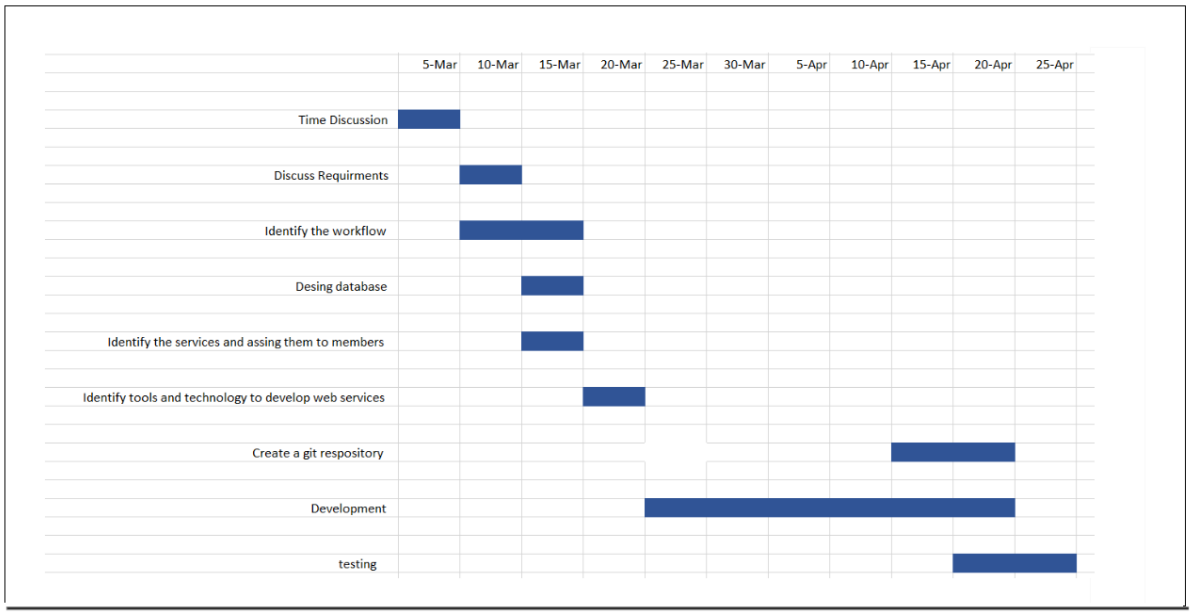
Agile Software Development is a methodology for creating a disciplined software management process that also allows for rapid changes to the development project. This methodology reduces risk by building software in short time periods known as iterations, which typically run one week to one month.

Advantages of Agile development methodology	Disadvantages of Agile Development Methodology
This methodology has an adaptive approach which is able to respond to the changing requirements of the clients and etc.	This methodology focuses on working software rather than documentation, hence it may result in a lack of documentation and etc.

How we organized as a team to work with Agile methodology?

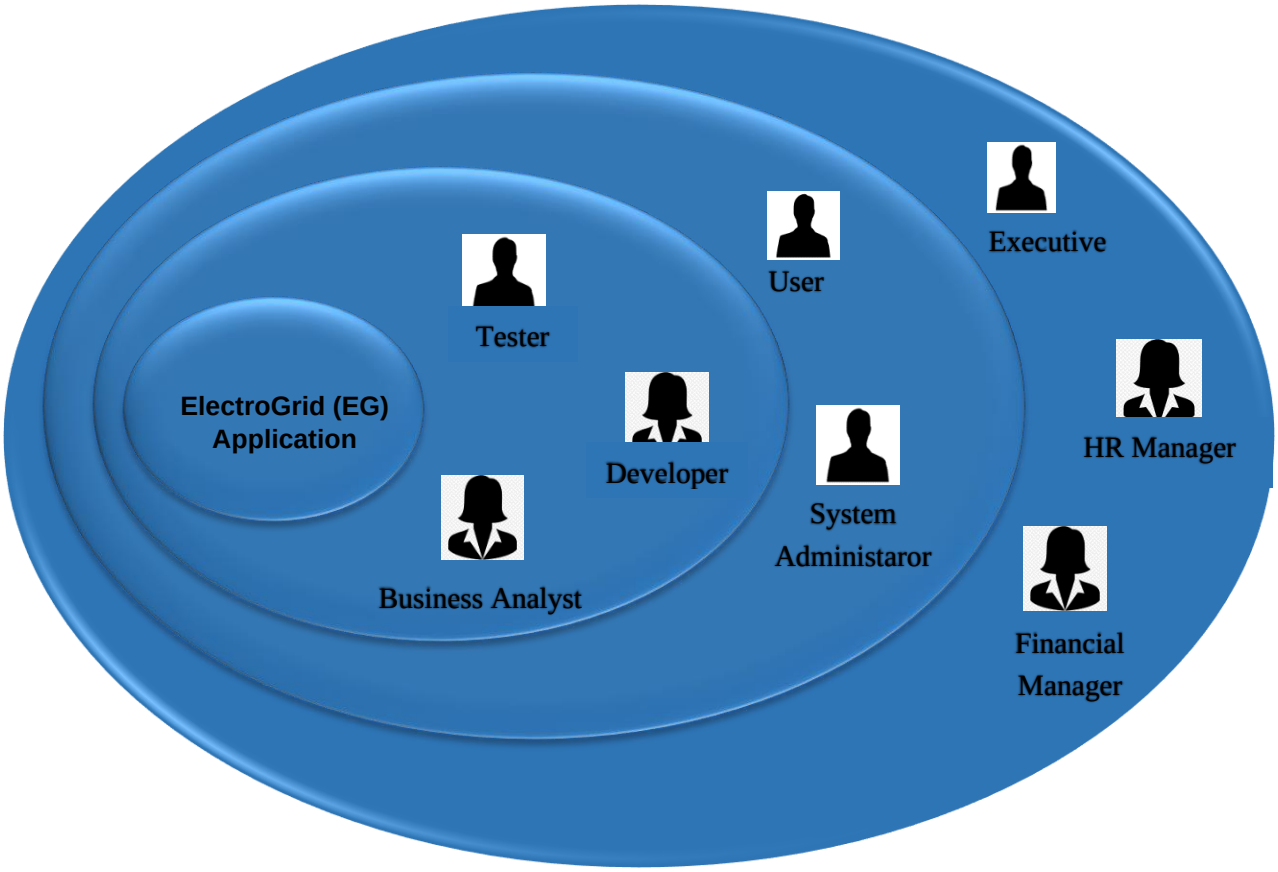
After talks (modify the requirements or accept the team's proposals), the scope of work was occasionally altered to accommodate new requirements. This is referred to as Agile's flexibility advantage. We divided the scenario's requirements into five micro services after recognizing them. This is another benefit of using Agile approach, which allows you to divide down work and consider microservices as tiny cycles (known as Sprints in Scrum). Because of the discussions with team members, they work closely together and have a clear understanding of their roles, and the work done within a cycle is frequently reassessed to improve the final result.

5. Time Link (Gantt Chart)

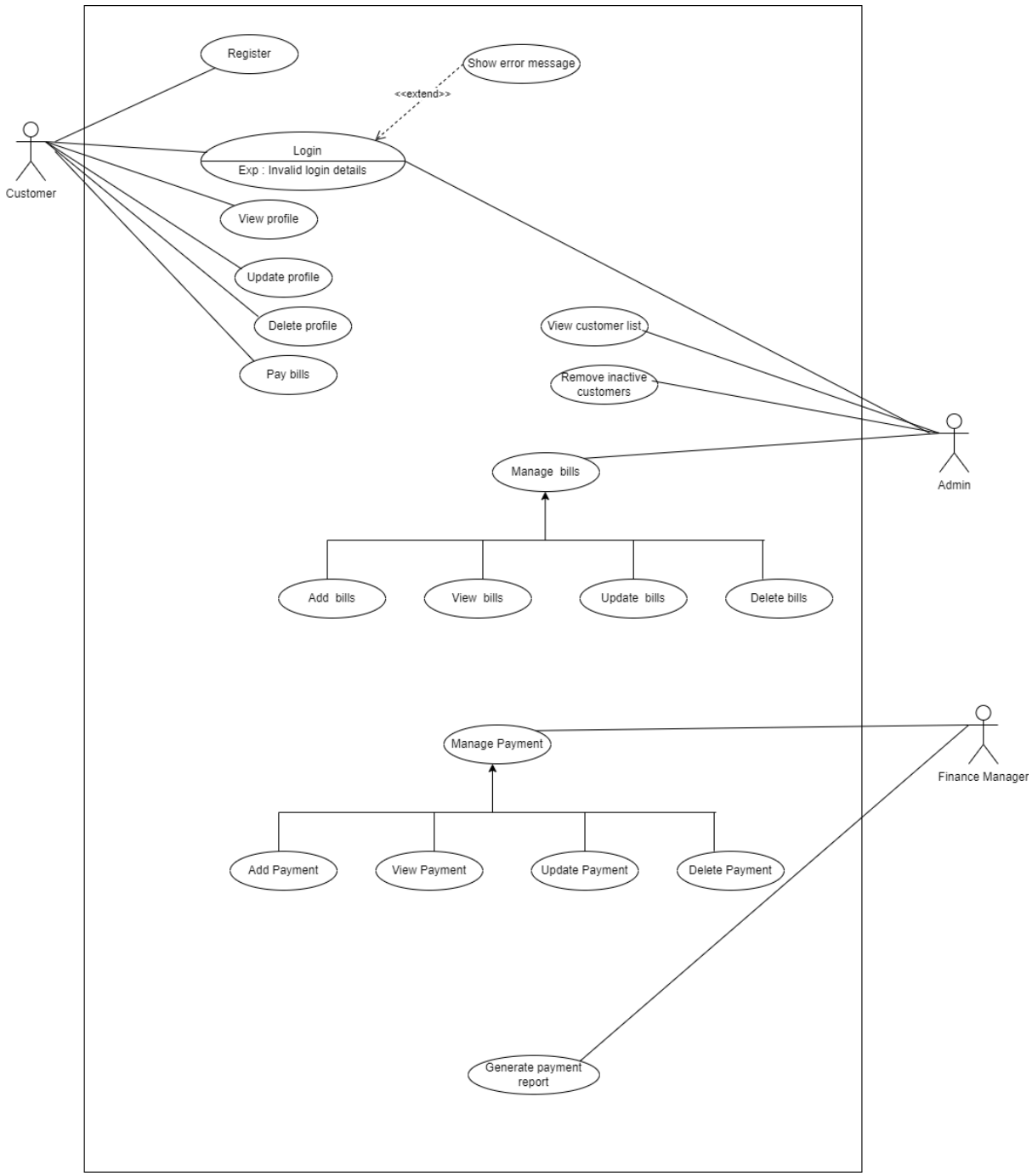


6. Stakeholder Analysis

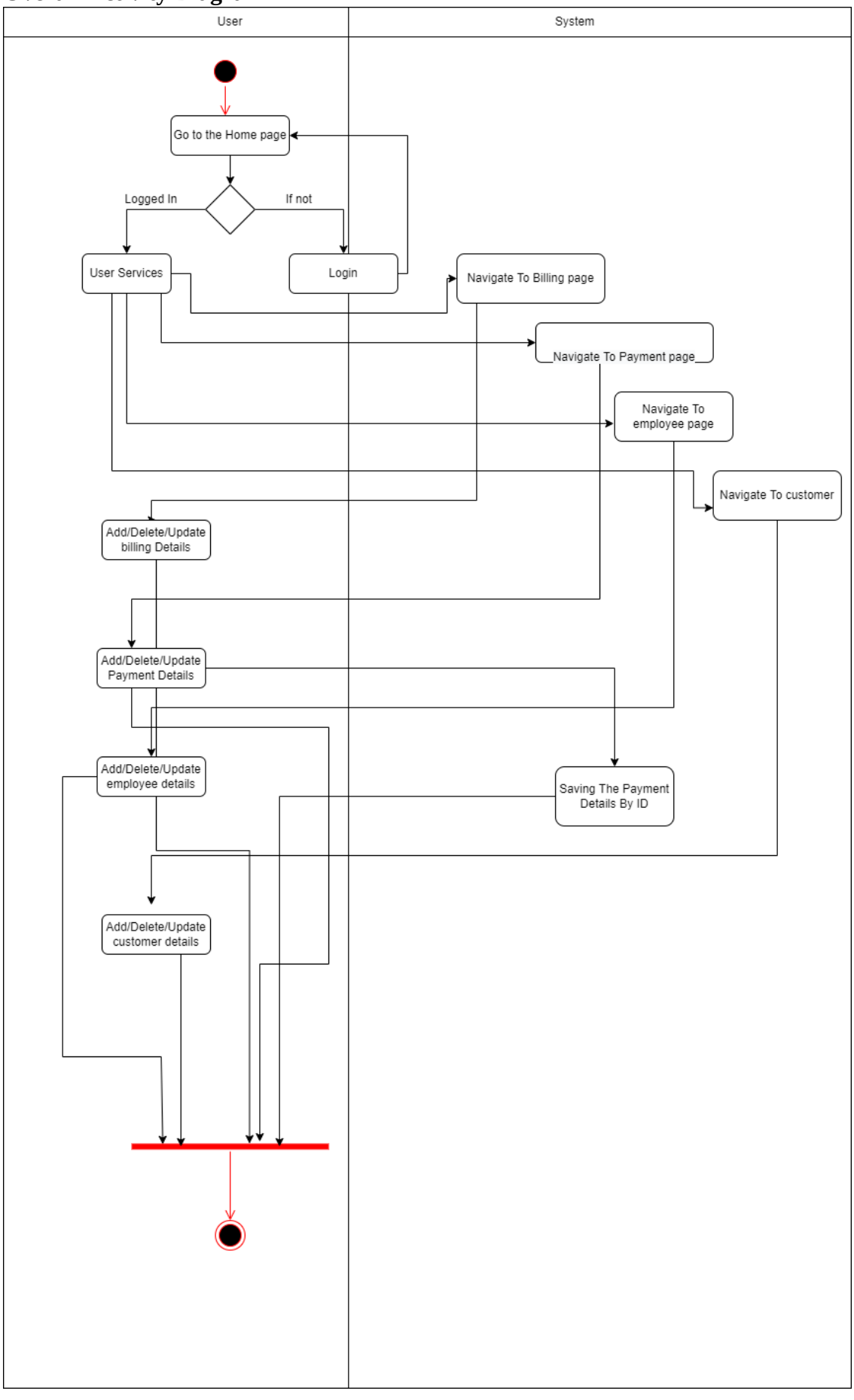
Onion Diagram



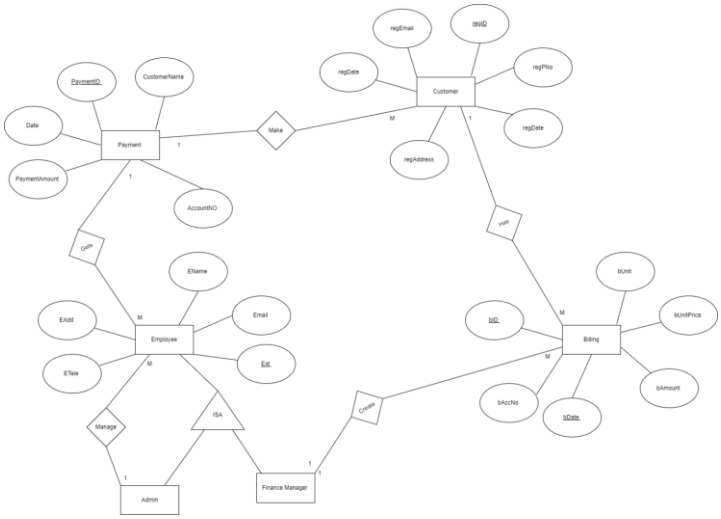
7. Overall Use Case Diagram



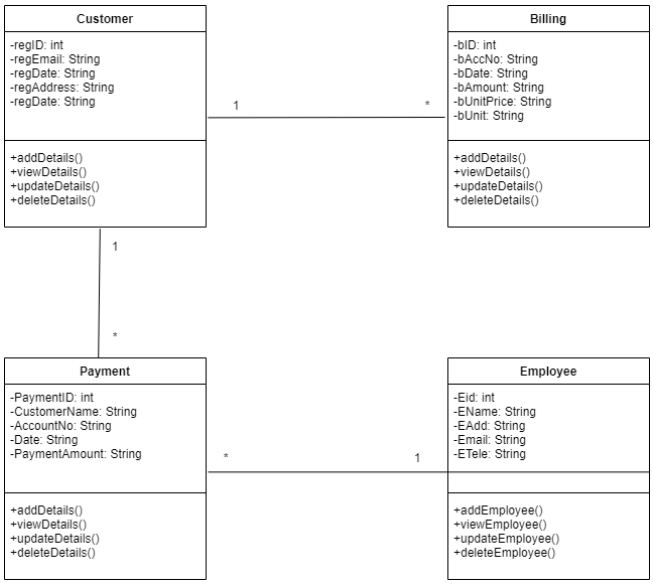
8. Overall Activity Diagram



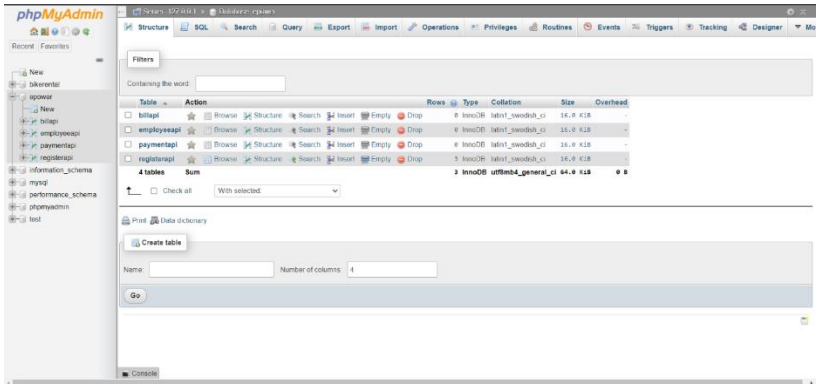
9. Overall ER Diagram



10. Overall, Class Diagram



11. Overall Data Base

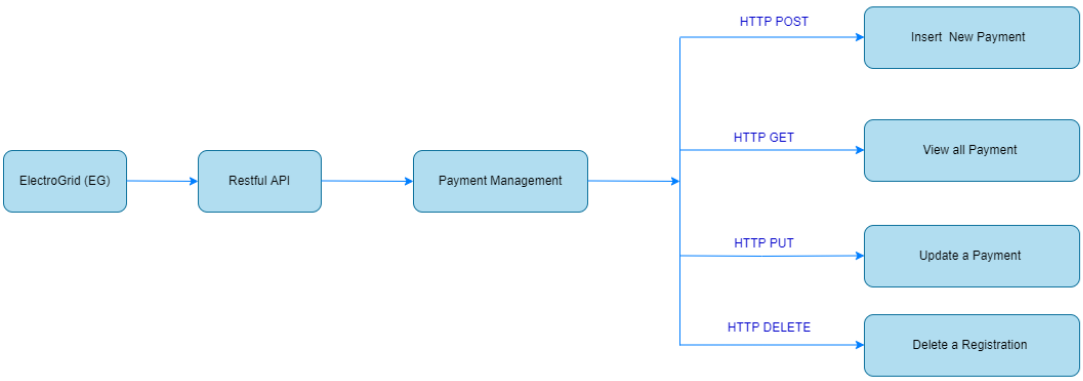


12. Individual Section

IT20145934

Wanasooriya W.M.D.C.

1. API design: Payment Management



Resource	Payment
Request	GET Payment_Management/ Payment API/ Payment (Get all Payment details)
URL	http://localhost:8085/Registration_Management/RegistrationAPI/Registration

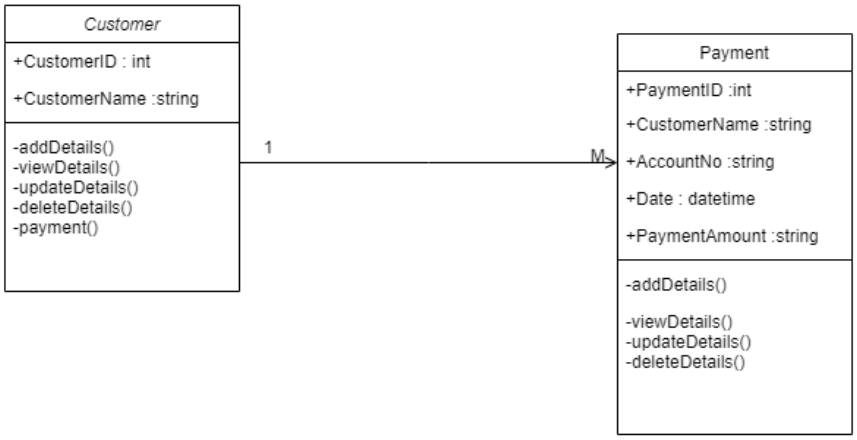
Resource	Payment
Request	POST Payment_Management/ Payment API/ Payment (Insert Payment details)
URL	http://localhost:8085/Registration_Management/RegistrationAPI/Registration

Resource	Payment
Request	PUT Payment_Management/ Payment API/ Payment (Update Payment details)
URL	http://localhost:8085/Registration_Management/RegistrationAPI/Registration

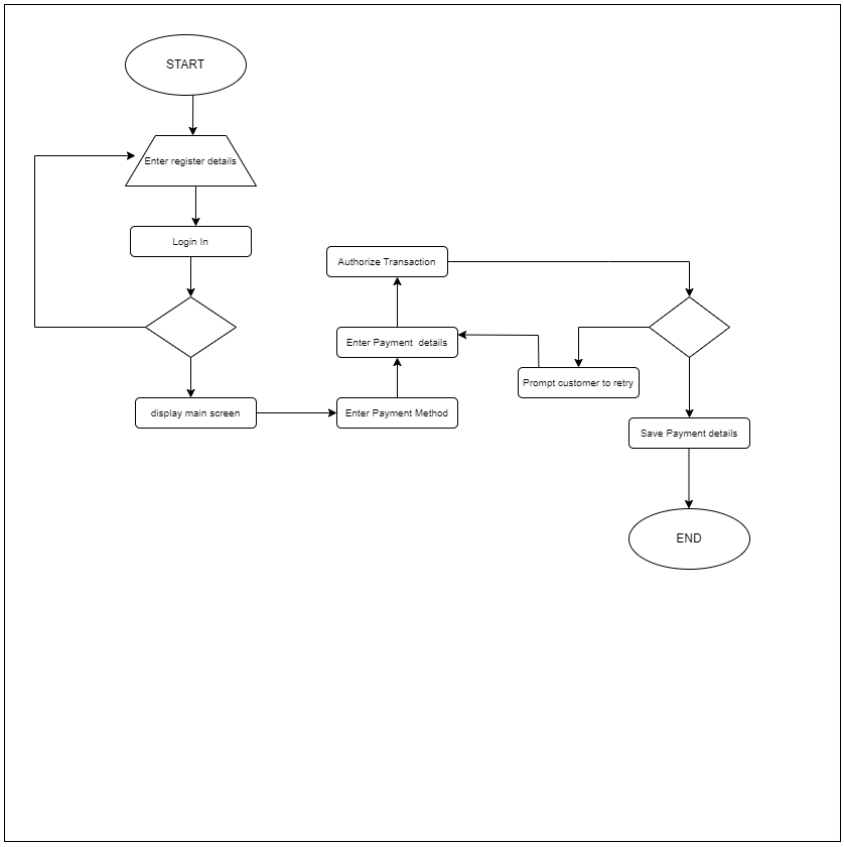
Resource	Payment
Request	DELETE Payment_Management/ Payment API/ Payment (Delete Registration details) (Delete Payment by ID)
URL	http://localhost:8085/Registration_Management/RegistrationAPI/Registration

2.Internal logic design

- Class Diagram

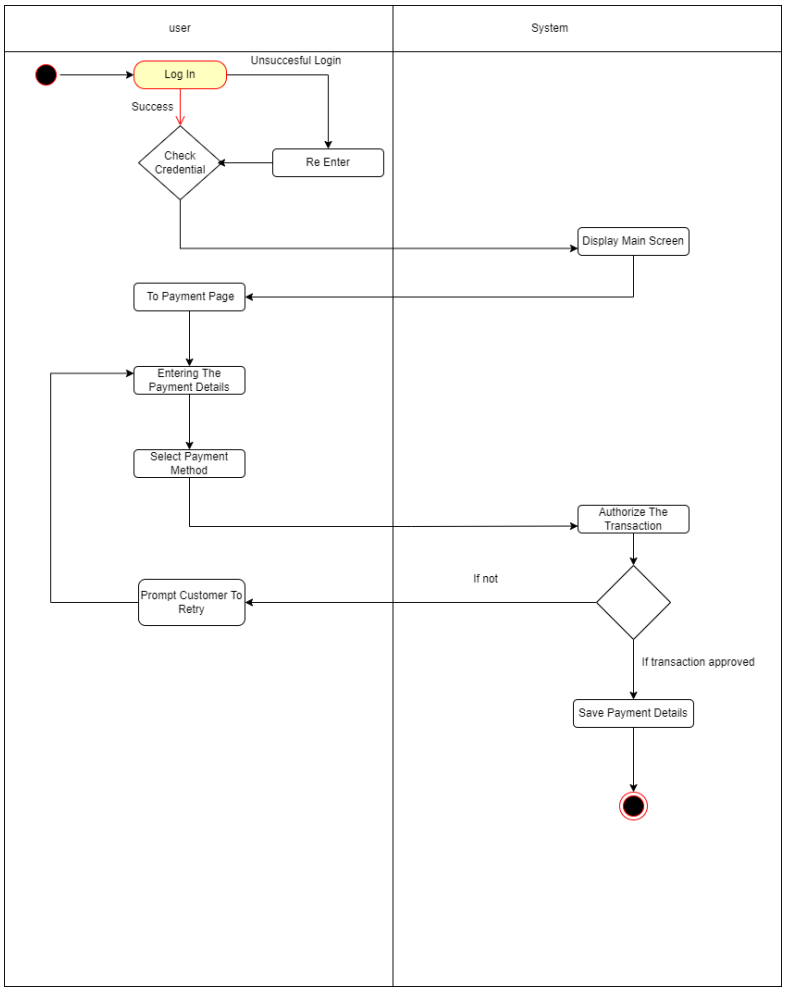


- Flow Chart Diagrams

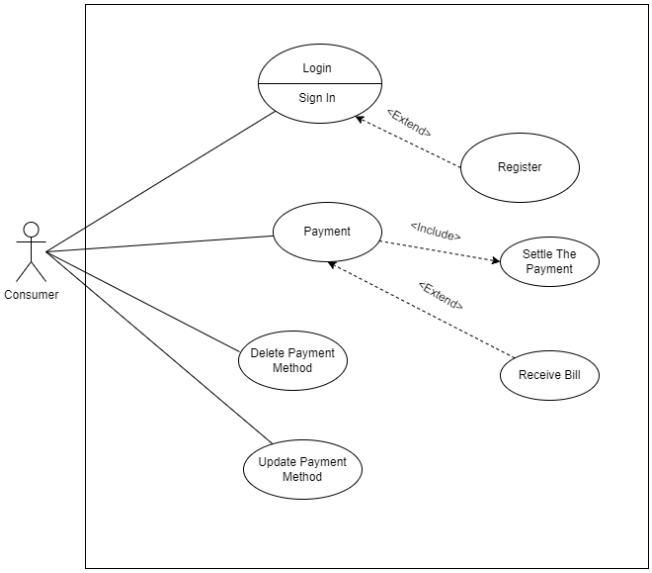


3. Any other relevant design diagrams

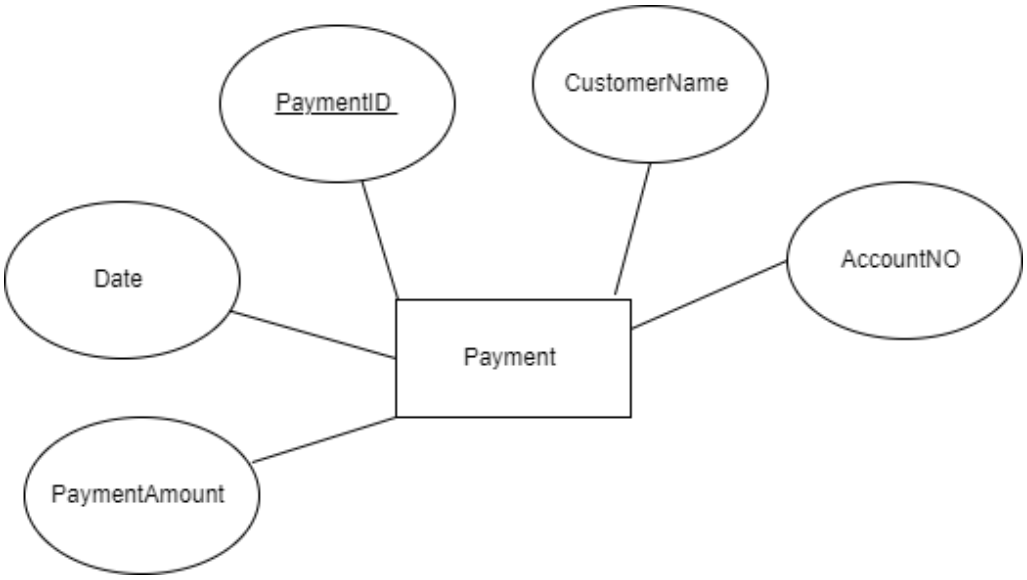
- Activity Diagram



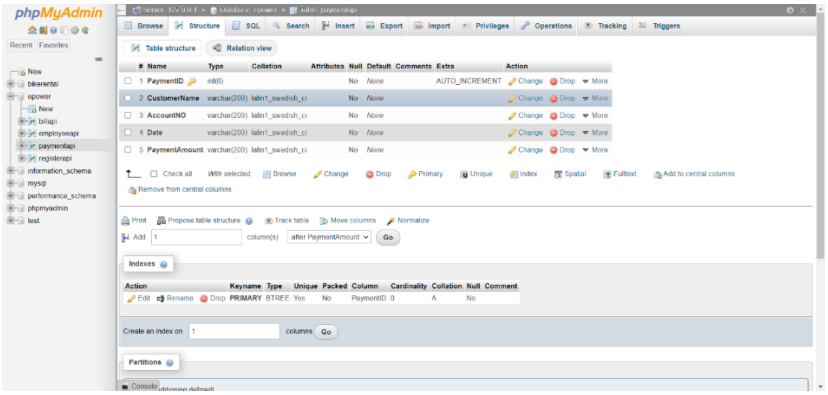
- User case Diagrams



- Er Diagram



4. Data Base



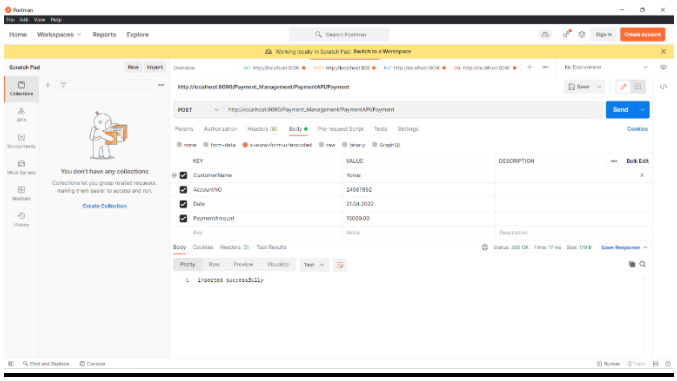
5.Development tools selection and justification.

Tools used

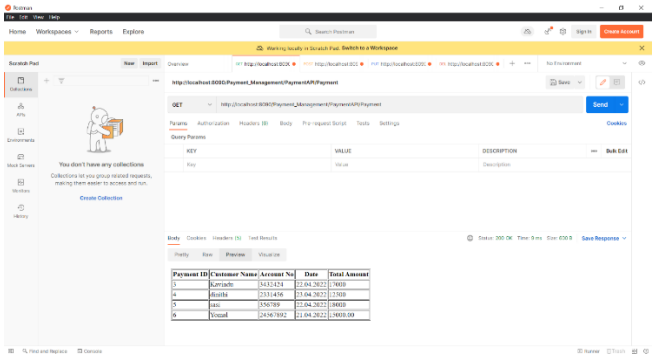
- Dependency Management Tool: - Maven
- Testing Tool: - Postman
- IDE: - Eclipse
- Programming Language: Java
- Framework: - JAX - RS
- Database: - phpMyAdmin (MySQL)
- Server: - Apache Tomcat
- Version Control System: - Git

6.Testing methodology and results.

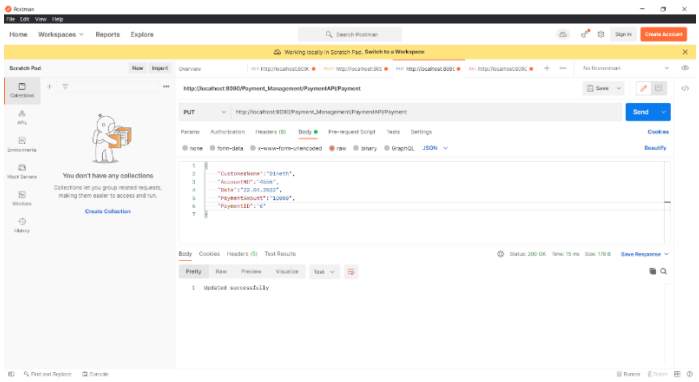
Test ID	Test Description	Test Input(s)	Expected Output	Actual Output	Result (Pass/Fail)
01)	Add a payment	Attributes of payment	Saves into the database and display message " Insert successfully"	Saves into the database and get message "Insert added successfully"	pass
02)	Update payment Details	Attributes to be updated along with the Payment ID	Display message as " updated successfully"	Display message as " updated successfully"	Pass
03)	View payment details by Payment ID	URL for the API and Payment ID	Display payment details	Display payment details	Pass
04)	Delete Payment by id	Payment ID of the Payment want to delete	Display message as "Deleted successfully"	Display message as "Deleted successfully"	Pass



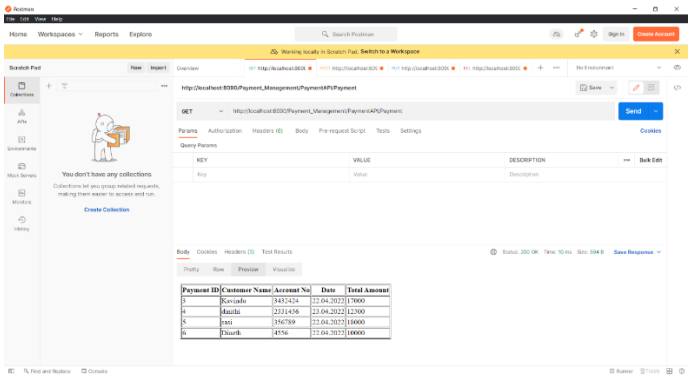
Add a payment



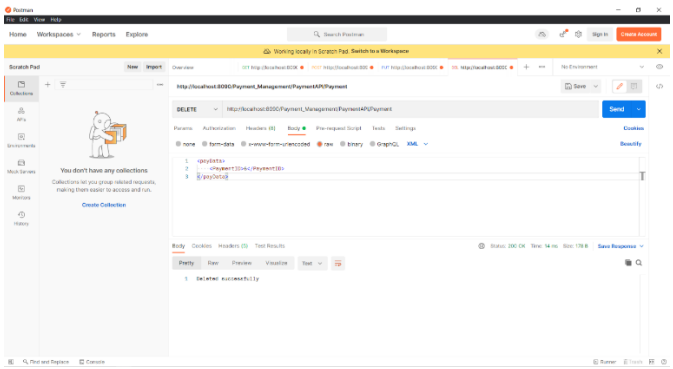
After Insert Payment Details



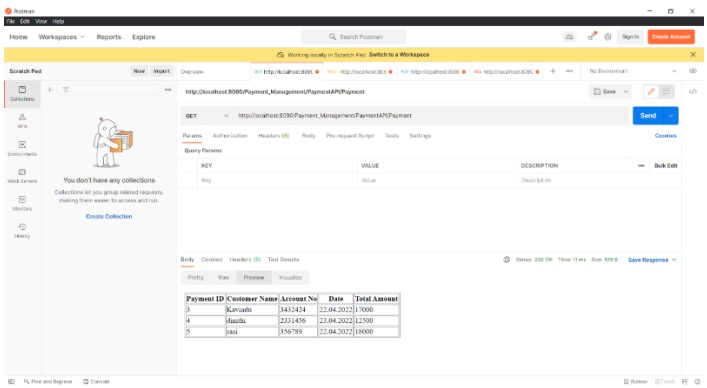
Update payment Details



After Update Payment Details



Delete Payment using PaymentID

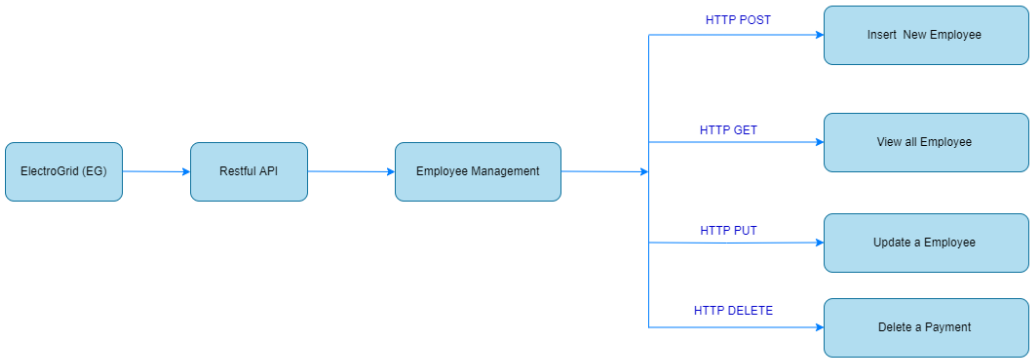


After Delete Payment Details (Delete Payment ID 6)

IT20143886

Samarawickrama H.N.

1. **API design: Employee Management**



Resource	Employee
Request	GET Employee_Management /EmployeeAPI/ Employee (Get all Registration details)
URL	http://localhost:8080/Employee_Management/EmployeeAPI/Employee

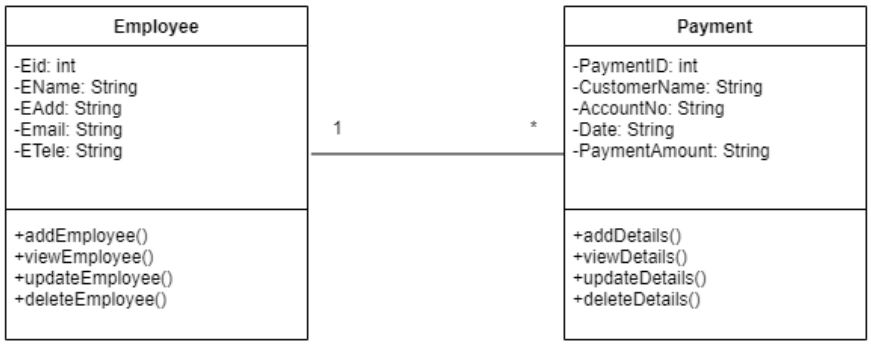
Resource	Employee
Request	POST Employee_Management /EmployeeAPI/ Employee (Insert Registration details)
URL	http://localhost:8080/Employee_Management/EmployeeAPI/Employee

Resource	Employee
Request	PUT Employee_Management /EmployeeAPI/ Employee (Update Registration details)
URL	http://localhost:8080/Employee_Management/EmployeeAPI/Employee

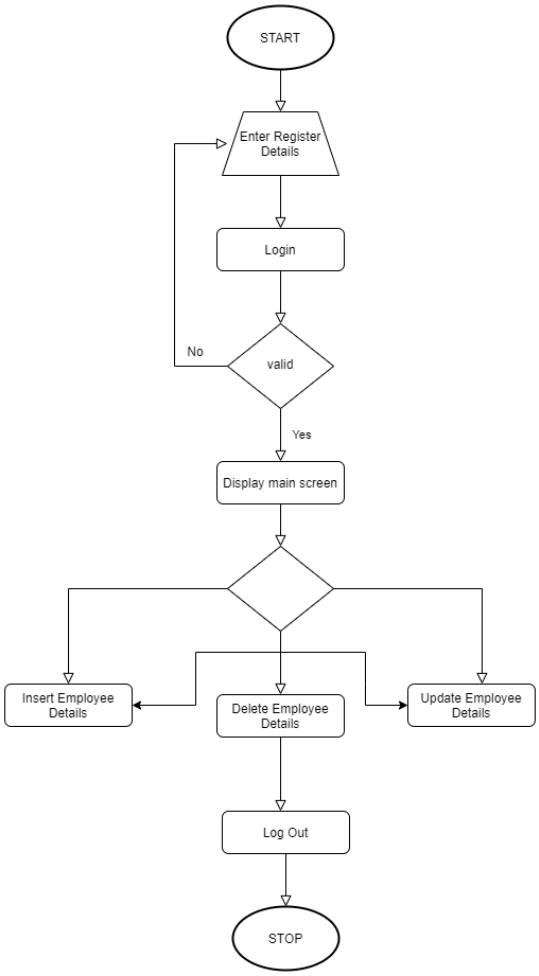
Resource	Employee
Request	DELETE Employee_Management /EmployeeAPI/ Employee (Delete Registration details) (Delete Registration by ID)
URL	http://localhost:8080/Employee_Management/EmployeeAPI/Employee

2.Internal logic design

- Class Diagram

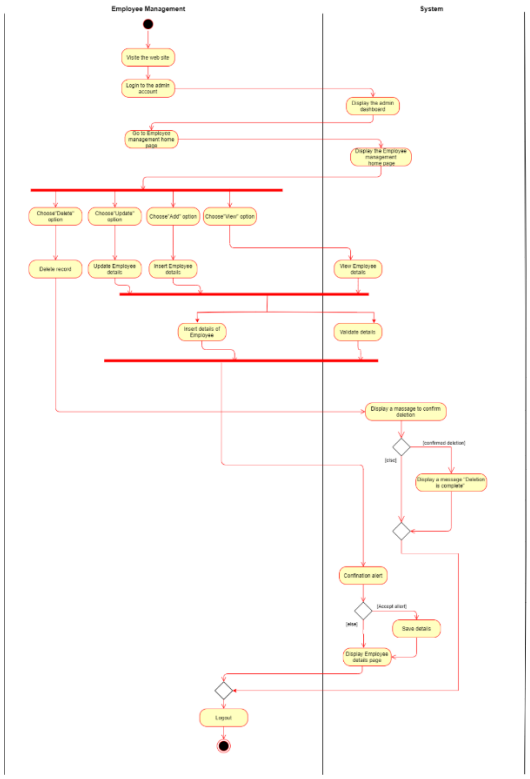


- Flow Chart Diagrams

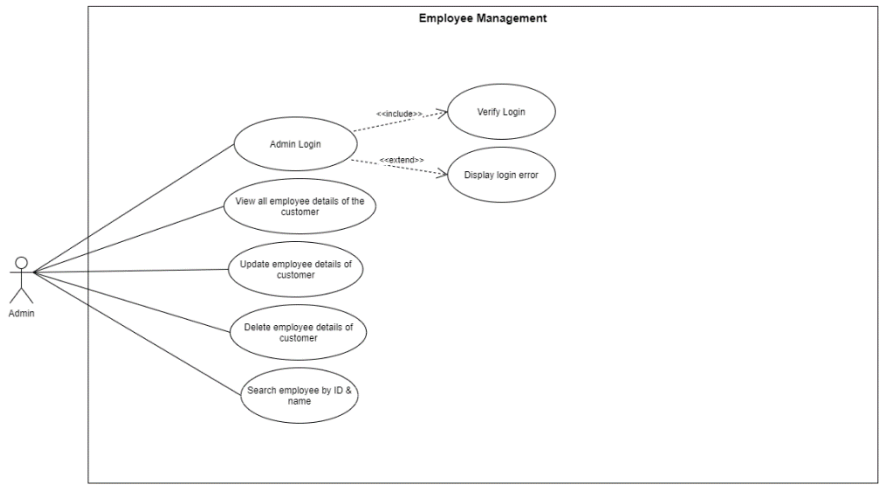


3. Any other relevant design diagrams

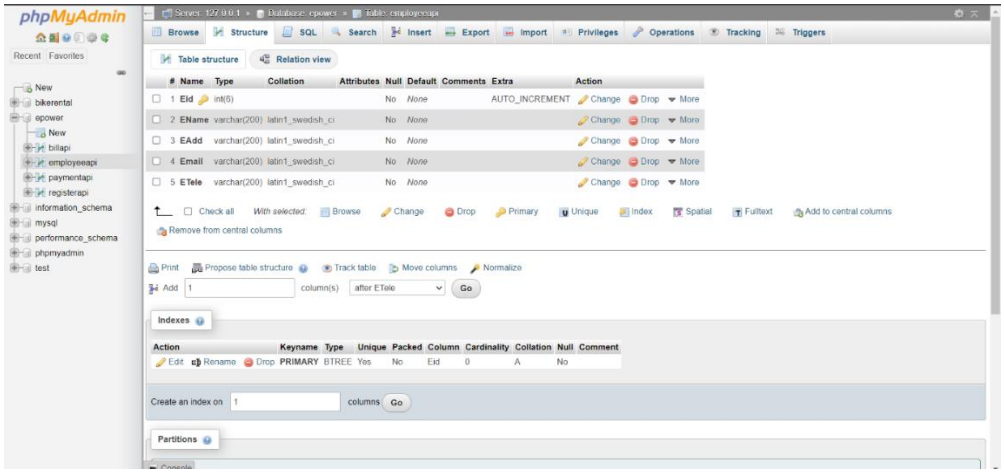
Activity Diagram (Funding Service)



- User case Diagrams



4.Data Base



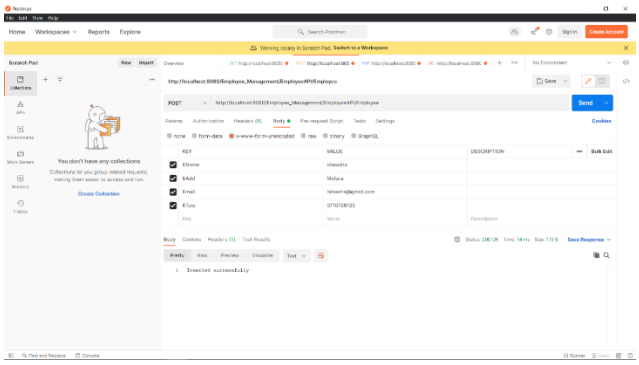
5.Development tools selection and justification.

Tools used

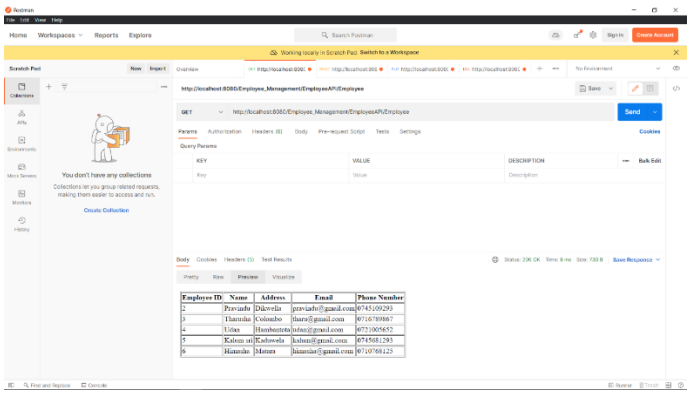
- Dependency Management Tool: - Maven
- Testing Tool: - Postman
- IDE: - Eclipse
- Programming Language: Java
- Framework: - JAX - RS
- Database: - phpMyAdmin (MySQL)
- Server: - Apache Tomcat
- Version Control System: - Git

6.Testing methodology and results.

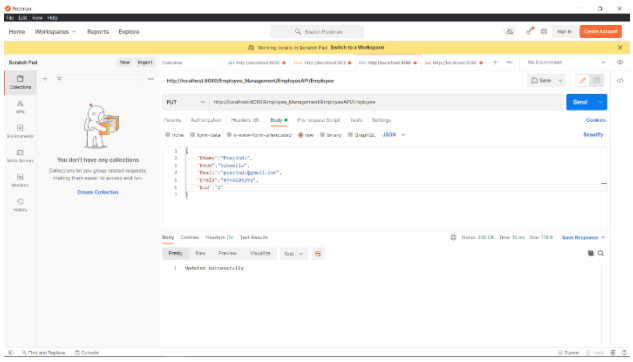
Test ID	Test Description	Test Input(s)	Expected Output	Actual Output	Result (Pass/Fail)
01)	Add an Employee Details	Attributes of Employee	Saves into the database and display message " Insert successfully"	Saves into the database and get message "Insert added successfully"	pass
02)	Update Employee Details	Attributes to be updated along with the Employee ID	Display message as " updated successfully"	Display message as " updated successfully"	Pass
03)	View Employee details by Employee ID	URL for the API and Employee ID	Display Employee details	Display payment details	Pass
04)	Delete Employee by id	Employee ID of the Employee want to delete	Display message as "Deleted successfully"	Display message as "Deleted successfully"	Pass



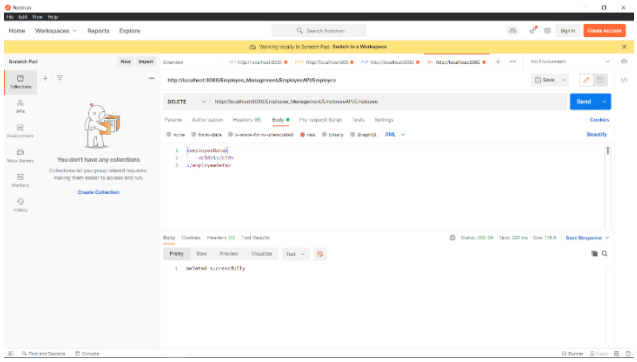
Add an Employee



After Insert Employee Details



Update Employee Details

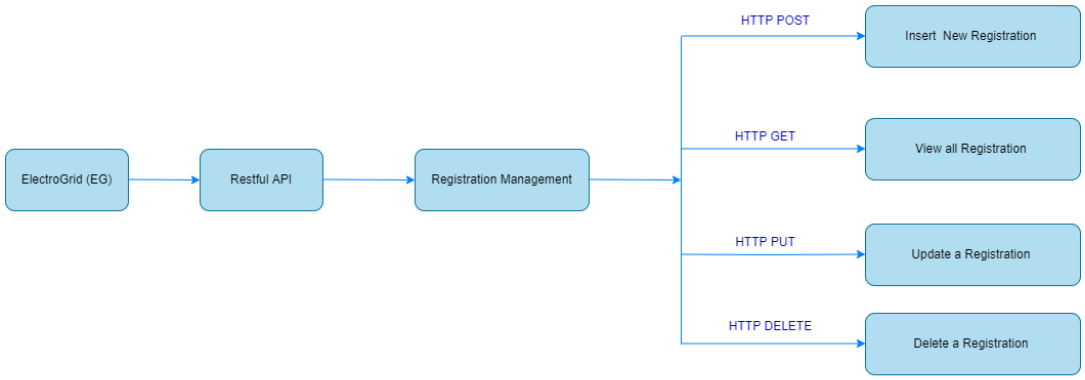


Delete Employee using EmployeeID

IT20146474

Liyanage A.L.D.K.S

1. API Design: Registration Management



Resource	Registration
Request	GET Registration_Management/RegistrationAPI/Registration (Get all Registration details)
URL	http://localhost:8085/Registration_Management/RegistrationAPI/Registration

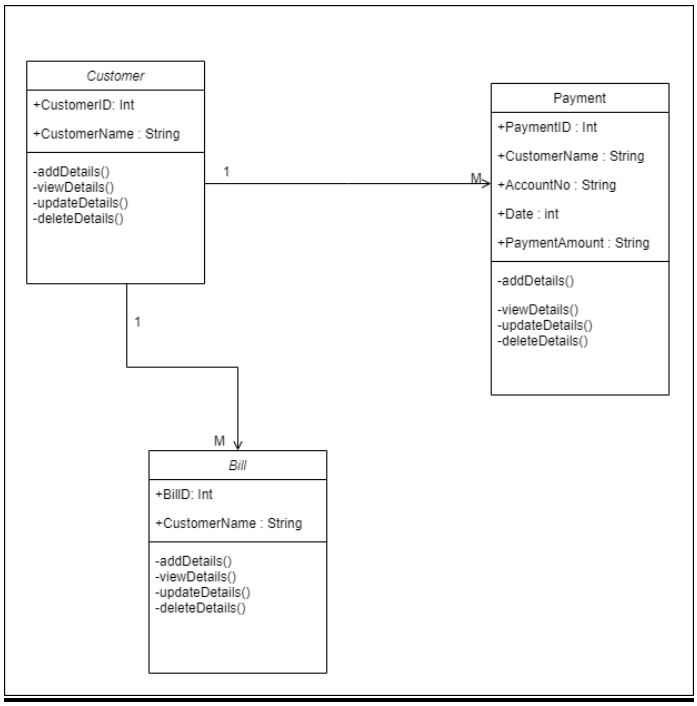
Resource	Registration
Request	POST Registration_Management/RegistrationAPI/Registration (Insert Registration details)
URL	http://localhost:8085/Registration_Management/RegistrationAPI/Registration

Resource	Registration
Request	PUT Registration_Management/RegistrationAPI/Registration (Update Registration details)
URL	http://localhost:8085/Registration_Management/RegistrationAPI/Registration

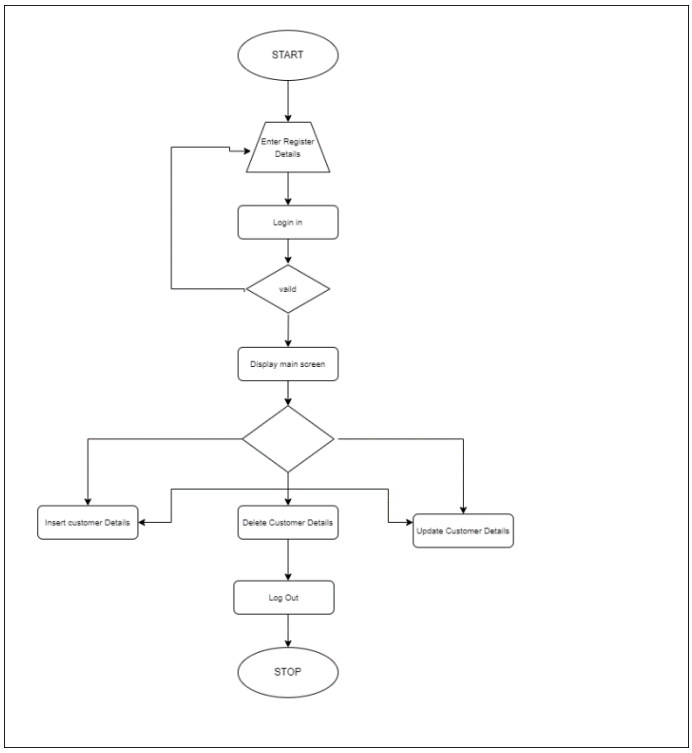
Resource	Registration
Request	DELETE Registration_Management/RegistrationAPI/Registration (Delete Registration details) (Delete Registration by ID)
URL	http://localhost:8085/Registration_Management/RegistrationAPI/Registration

2.Internal logic design

- Class Diagram

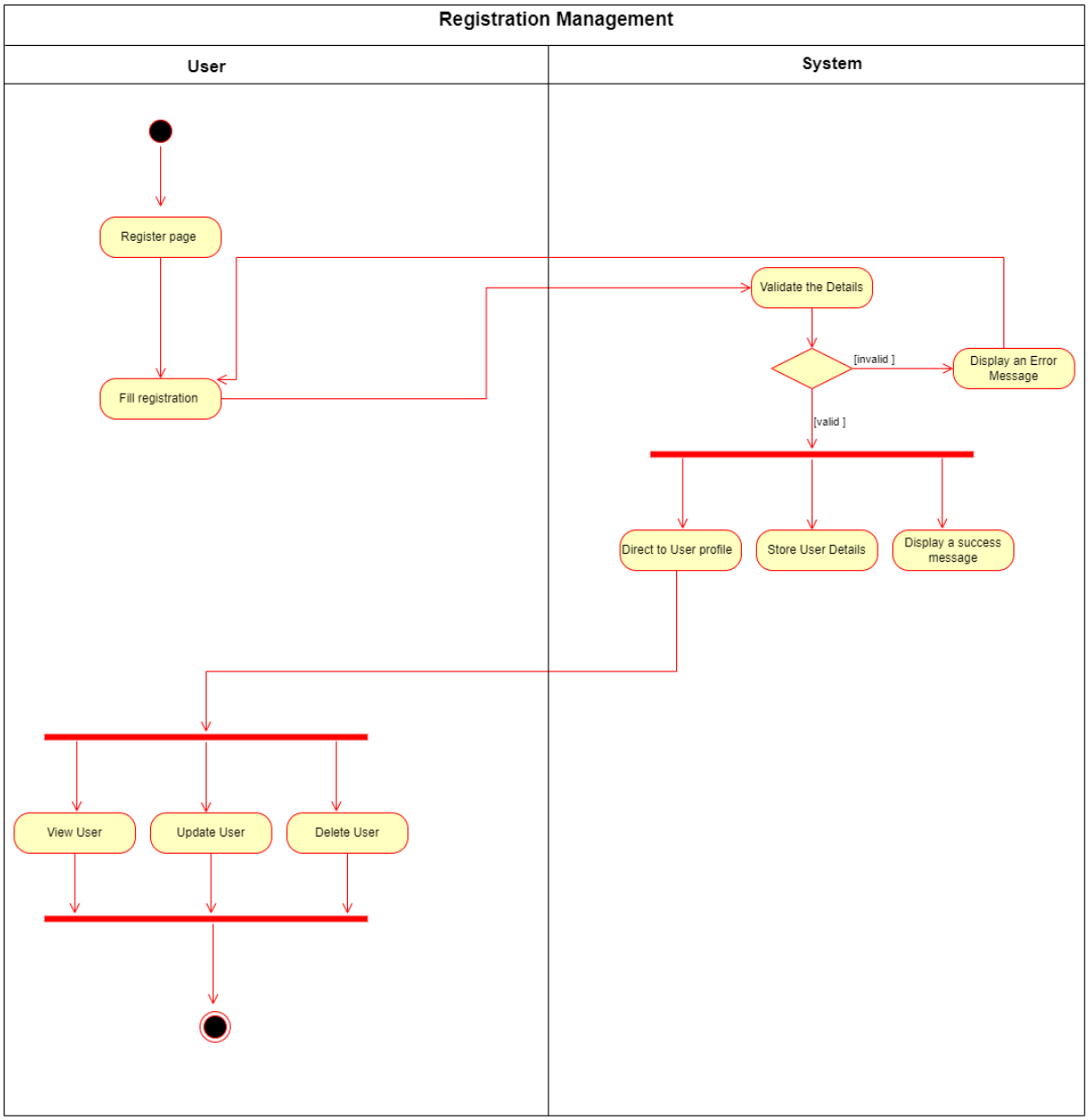


- Flow Chart Diagrams

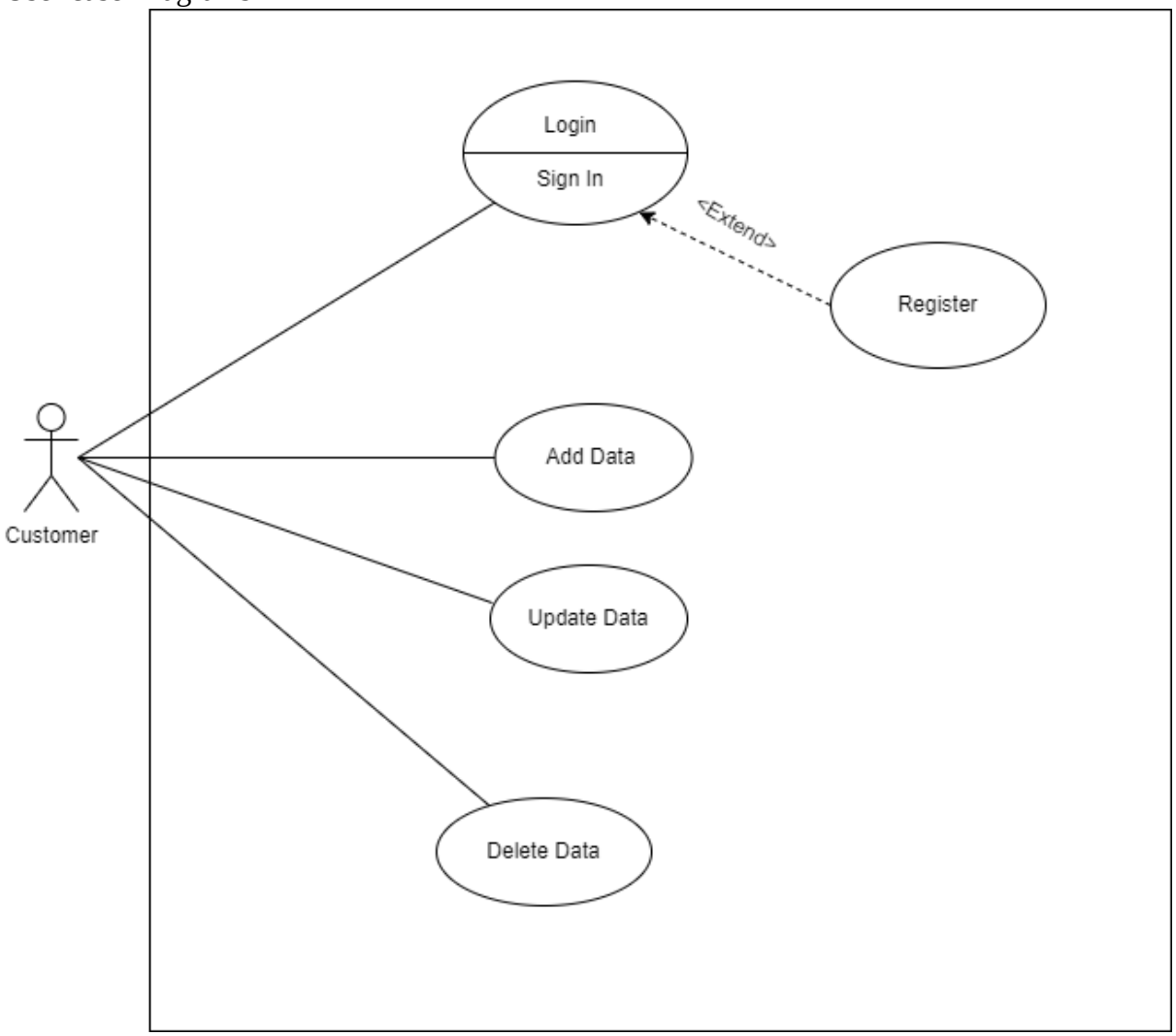


3. Any other relevant design diagrams

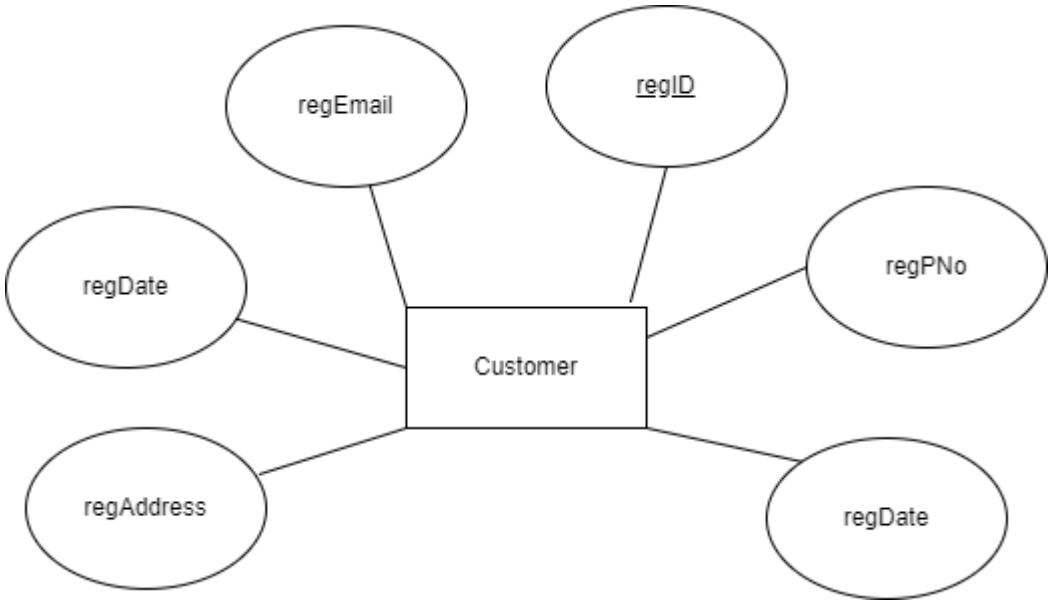
- Activity Diagram (Funding Service)



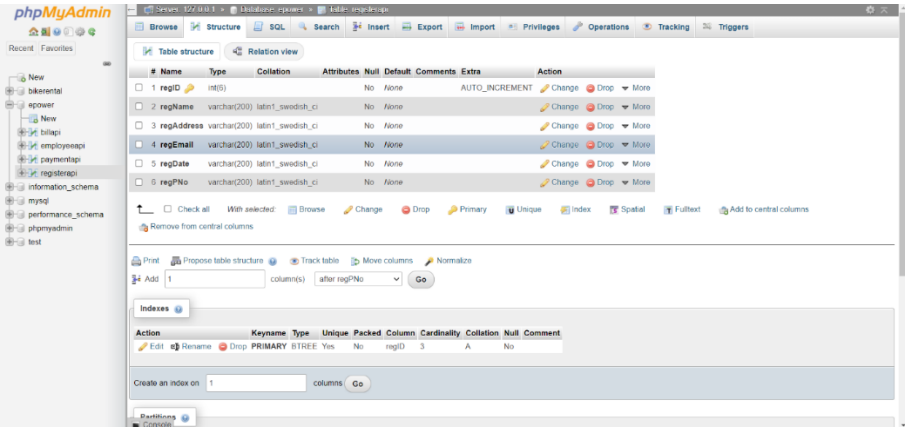
❖ User case Diagrams



❖ Er Diagram



4.Data Base



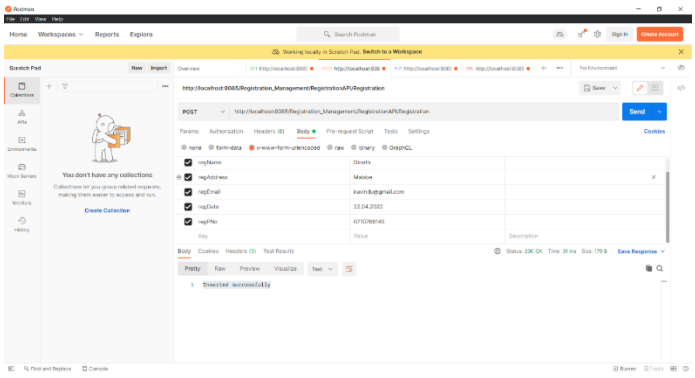
5.Development tools selection and justification.

Tools used

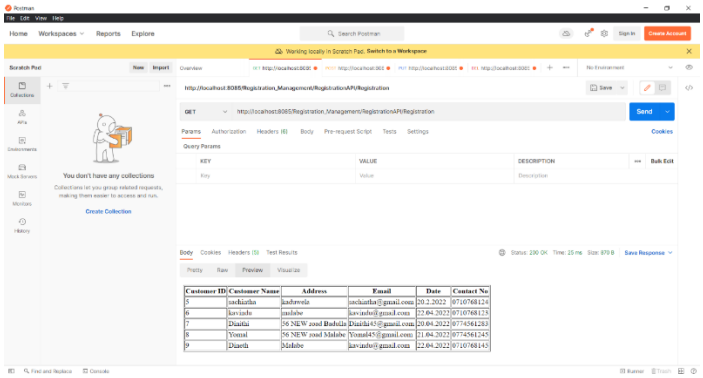
- Dependency Management Tool: - Maven
- Testing Tool: - Postman
- IDE: - Eclipse
- Programming Language: Java
- Framework: - JAX - RS
- Database5: - phpMyAdmin (MySQL)
- Server: - Apache Tomcat
- Version Control System: - Git

6.Testing methodology and results.

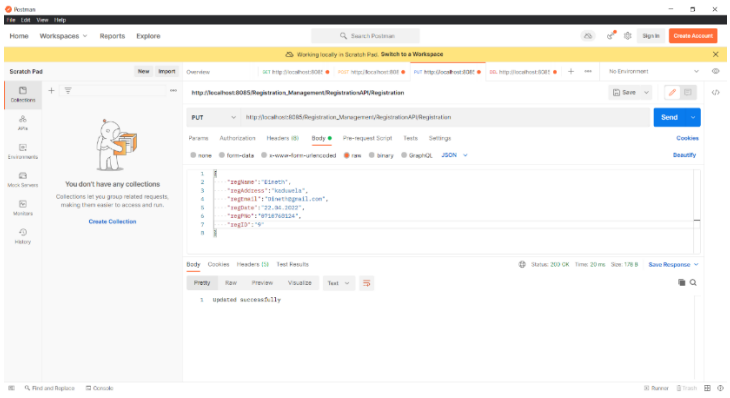
Test ID	Test Description	Test Input(s)	Expected Output	Actual Output	Result (Pass/Fail)
01)	Add Register Details	Attributes of Register	Saves into the database and display message " Insert successfully"	Saves into the database and get message "Insert added successfully"	pass
02)	Update Register Details	Attributes to be updated along with the Register ID	Display message as " updated successfully"	Display message as " updated successfully"	Pass
03)	View Register details by Register ID	URL for the API and Register ID	Display Register details	Display Register details	Pass
04)	Delete Register by id	Register ID of the Register want to delete	Display message as "Deleted successfully"	Display message as "Deleted successfully"	Pass



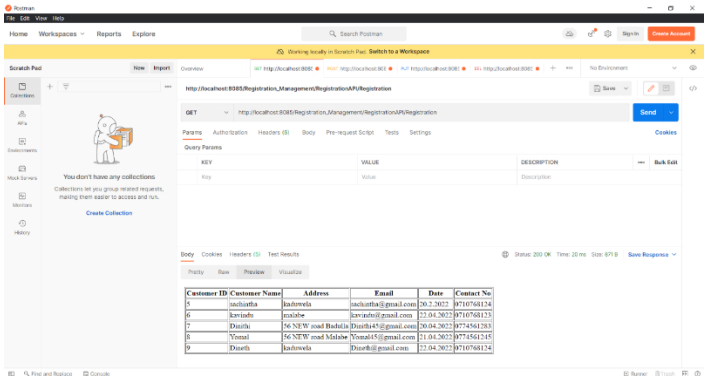
Add a Register Details



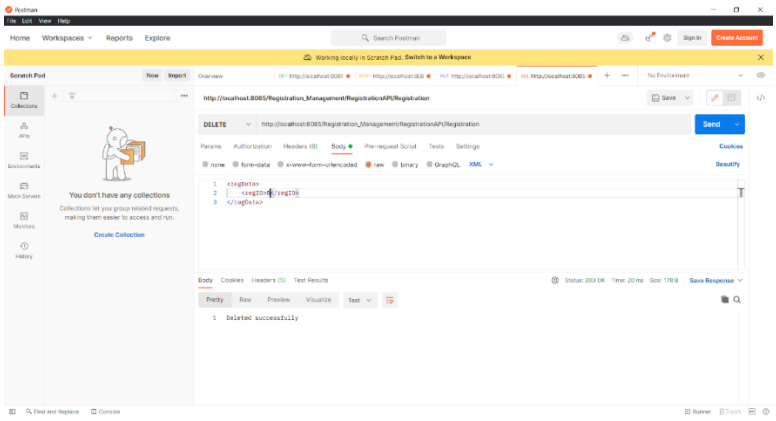
After Insert Register Details



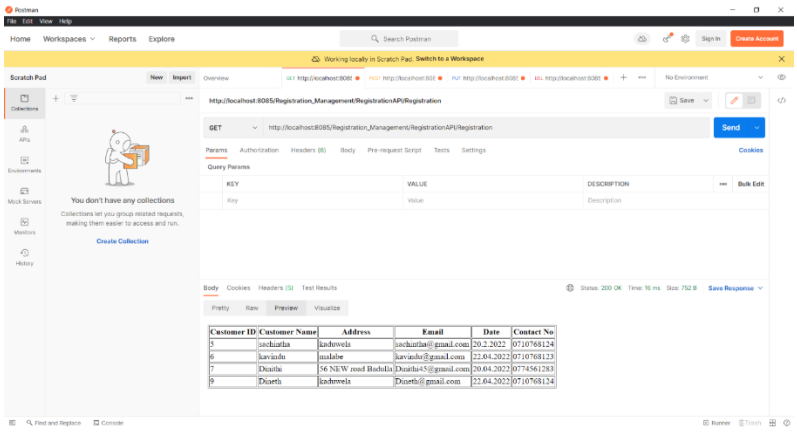
Update Register Details



After Update Register Details (Update RegisterID 9)



Delete Register using RegisterID

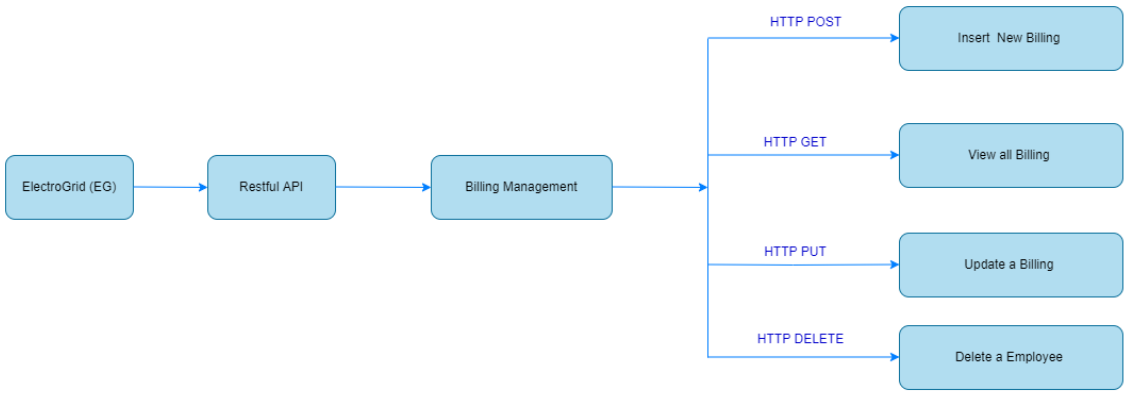


After Delete Register Details (Delete Register ID 8)

IT20154912

Sahan A. K

1.API Design: Billing Management



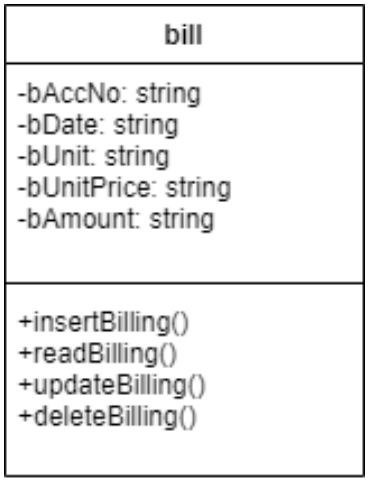
Resource	Funds
Request	GET Billing_Management/BillingAPI/Billing
URL	http://localhost:8085/Billing_Management/BillingAPI/Billing

Resource	Funds
Request	GET Billing_Management/BillingAPI/Billing
URL	http://localhost:8085/Billing_Management/BillingAPI/Billing

Resource	Funds
Request	POST Billing_Management/BillingAPI/Billing
URL	http://localhost:8085/Billing_Management/BillingAPI/Billing

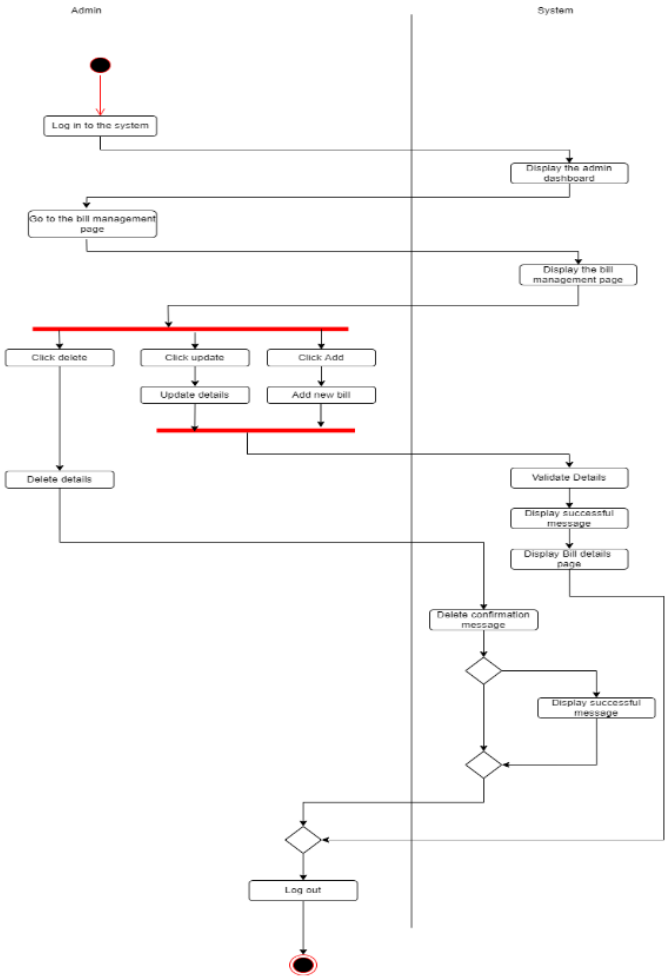
Resource	Funds
Request	PUT Billing_Management/BillingAPI/Billing
URL	http://localhost:8085/Billing_Management/BillingAPI/Billing

2.Internal logic design

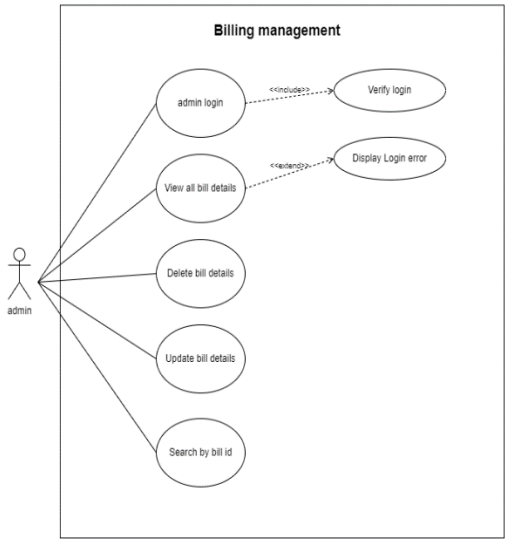


3. Any other relevant design diagrams

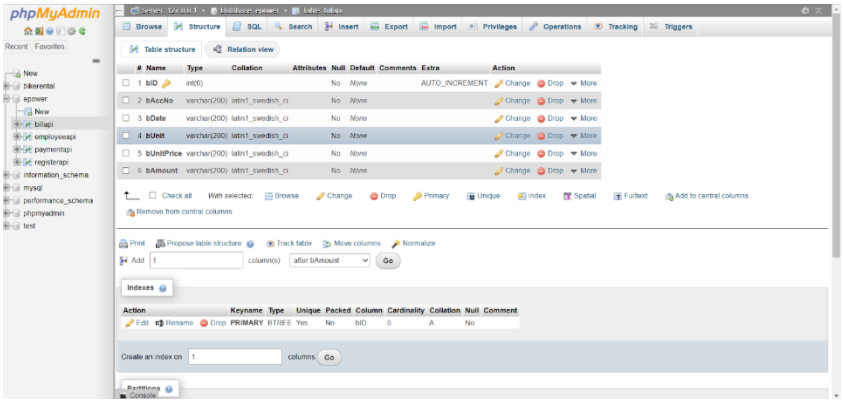
- Activity Diagram (Funding Service)



- User case Diagrams



4.Data Base



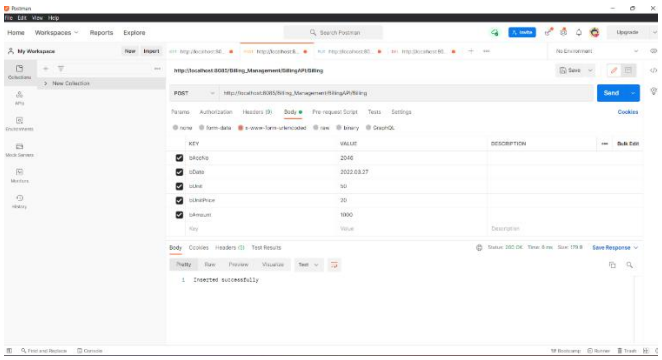
5.Development tools selection and justification.

Tools used

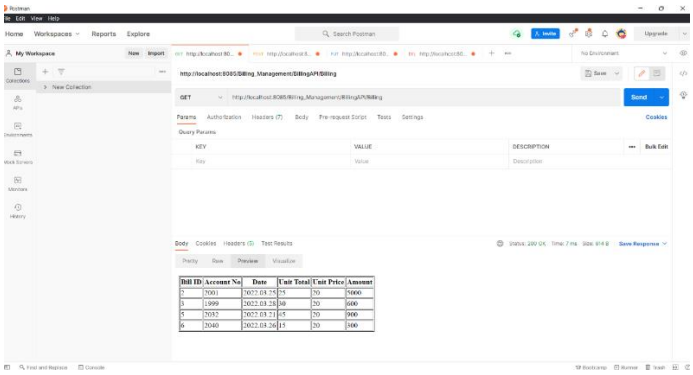
- Dependency Management Tool: - Maven
- Testing Tool: - Postman
- IDE: - Eclipse
- Programming Language: Java
- Framework: - JAX - RS
- Database: - phpMyAdmin (MySQL)
- Server: - Apache Tomcat
- Version Control System: - Git

6.Testing methodology and results.

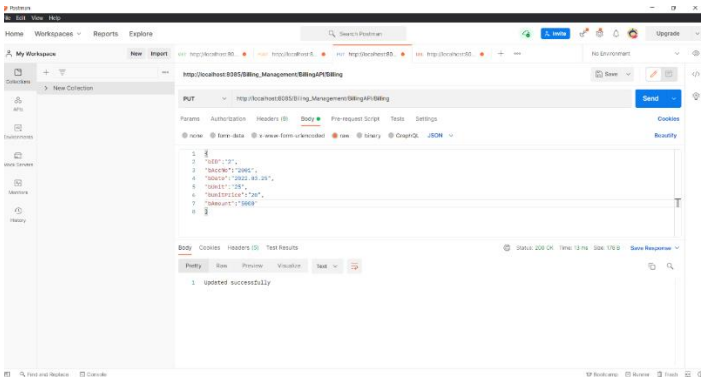
Test ID	Test Description	Test Input(s)	Expected Output	Actual Output	Result (Pass/Fail)
01)	Add Bill Details	Attributes of Bill	Saves into the database and display message " Insert successfully"	Saves into the database and get message "Insert added successfully"	pass
02)	Update Bill Details	Attributes to be updated along with the Bill ID	Display message as " updated successfully"	Display message as " updated successfully"	Pass
03)	View Bill details by Bill ID	URL for the API and Bill ID	Display Bill details	Display Bill details	Pass
04)	Delete Bill by id	Bill ID of the Bill want to delete	Display message as "Deleted successfully"	Display message as "Deleted successfully"	Pass



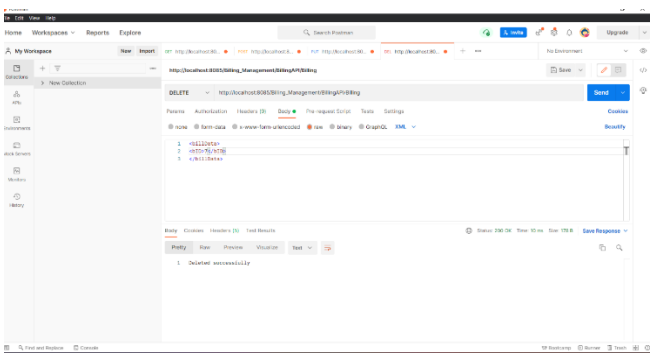
Add a Bill



After Insert Bill Details



Update Bill Details



Delete Bill using BillID