GadgetBadget Project Group
Batch Group 15
Assignment Group 14

Registration No	Name
IT19201160	Wanasinghe W. A. D. B.
IT19153346	Maddugoda C. D.
IT19170176	Fernando W. N. D.
IT19174686	Ramawickrama H. N.
IT18125726	Kumarage L.V.A

Group work

1. Member details and role

Student ID	Name	Web service	Description
IT19201160	Wanasinghe W. A. D. B.	Funding Service	- Add a Funding - View all funding - View a funding by ID - Update the funding amount - Delete a funding
IT19153346	Maddugoda C.D	User Service	- Add a user - Update user details - Delete user details - Get user details by Id - View all user details - Validate user details(in login) - Reset password(password forgotten)
IT19170176	Fernando W. N. D.	Order Service	- Add an Order. - Update an Order. - Cancel an Order. - View all Orders. - View Orders by ID
IT18125726	Kumarage L.V.A	Payment Service	- Add a new payment record - Delete a payment record - Update a payment record - View all payment details - View payment by id details
IT19174686	Ramawickrama H.N	Product Service	- Add a new Product record - Delete a product record - Update a product record - View all product details - View product by id details

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

Micro-service	Functional Requirement	Non-Functional Requirement	Technical Requirement
Funding Service	- Add funding details - View all funding - View a funding by id - Delete a funding - Update funding amount	- Security and privacy - Scalability - High performance	Funder can add funding details using a form. Only four columns to fill. Addition to that update is work only to edit the funding amount. If specific funding is inserted, then funder cannot change other fields. Funder can stop funding by deleting the record. Funder can view all the funding that specific funder provided. In other hand researches can view all the funding that researchers received.
User Service	- Add user details - Update user details - Delete user details - View user details - Authentication of details	- Security and privacy - Scalability - High performance - Recoverability - Availability	User can enter details and create a profile. Username should not be an already taken username. User is uniquely identified using user id. User can view, update or delete only his/her details unless if the user is the admin. Only an existing profile can be updated, viewed or deleted. System validates the username and password when logging into the system. If the password is forgotten User Code sent to the email should match with the provided user Code when resetting the password.
Order Service	- Add Order - Update Order - Delete Order - Retrieve all Order - Retrieve Orders by ID	Scalability, Capacity, Availability, Reliability, Recoverability, Maintainability, Usability, Data Integrity	Customer can make an order according to the requirements of a research project and make the order for a relevant research project. System will check whether the research project which was ordered by the customer is already ordered or not. So, the customer only will be able to order an available research project only with a future order date. The customer can also update the order if he wants another research project or if he wants to change the date of his order. Also, the system will allow the customer to delete an order only with a future date (Which has not been overdue) Also the customer is able to view all the orders which was made by him and also, he can view an order only with an orderID.
Payment Service	- Add Payment - Update Payment - Delete Payment	- Security and Privacy - Performance	After confirming the particular order, Customer can add payment details such as payment method(visa/master), card details

	- Retrieve All payment details - Retrieve payment details by Id	- Response in time - Maintainability - Data Integrity	(card number, name on card, cvc, expire date) to go ahead with procedure. System will make sure to validate payment details by avoiding null values and inaccurate data formats. Customer can view all his/her payment details by payment id including sub amount (generated by adding tax amount (automatically generated through system) and amount of the order). Admin can manage payment details by updating particular payment details and deleting unnecessary payment details of the system
Product Service	- Add Product - Update Product Details - Delete a product - Retrieve all products - Retrieve a product by ID	- Security and privacy - Scalability - High performance	Researcher can enter the details of the product. Each product has a uniquely identified product ID. The researcher can view, update, delete and insert a certain product. Update and delete operations can be done to an existing product only. The date field of the product to be entered is validated in a way that the user cannot enter any past dates

3. Clickable link for GitHub Repository

https://github.com/PAF-GadgetBadget/GB-Version1

4. SE Methodology/Methods - Agile Software Development

What is Agile?

Agile Software Development is an approach that is used to design a disciplined software management process which also allows some frequent alteration in the development project. This methodology is used to minimize risk by developing software in short time boxes which are called iterations that generally last for one week to one month.

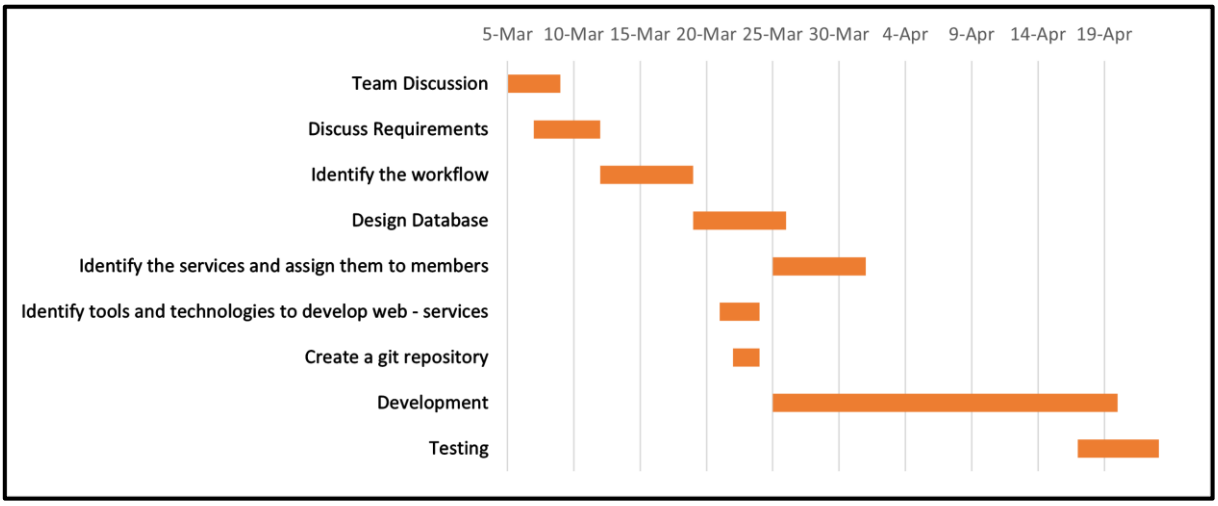
Advantages of Agile development methodology	Disadvantages of Agile Development Methodology
This methodology has an adaptive approach which is able to respond to the changing requirements of the clients and etc.	This methodology focuses on working software rather than documentation, hence it may result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

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

Sometimes the scope of work changed according to new requirements after discussions (change the requirements or accept the team's suggestions). This is an advantage of Agile known as flexibility.

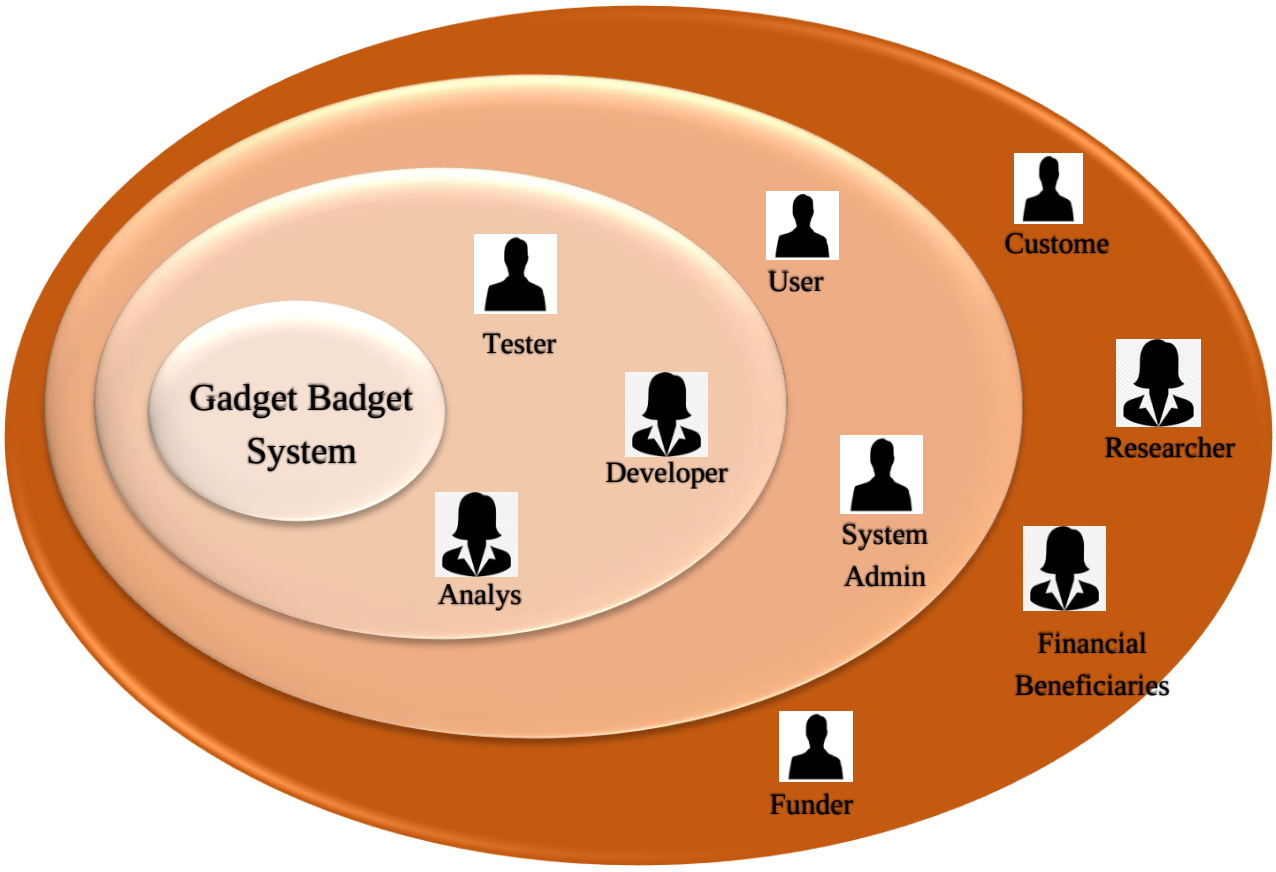
After identifying the requirements from the scenario, we separated them into five micro services. This is also an advantage of following Agile methodology which helps to work breakdown and treat these micro services as small cycles (known as Sprints in Scrum). Because of having discussion with team members work closely together and have clear vision about their responsibilities and there is frequent reassessment of the work done within a cycle to make the final product better.

5. Time Link (Gantt Chart)

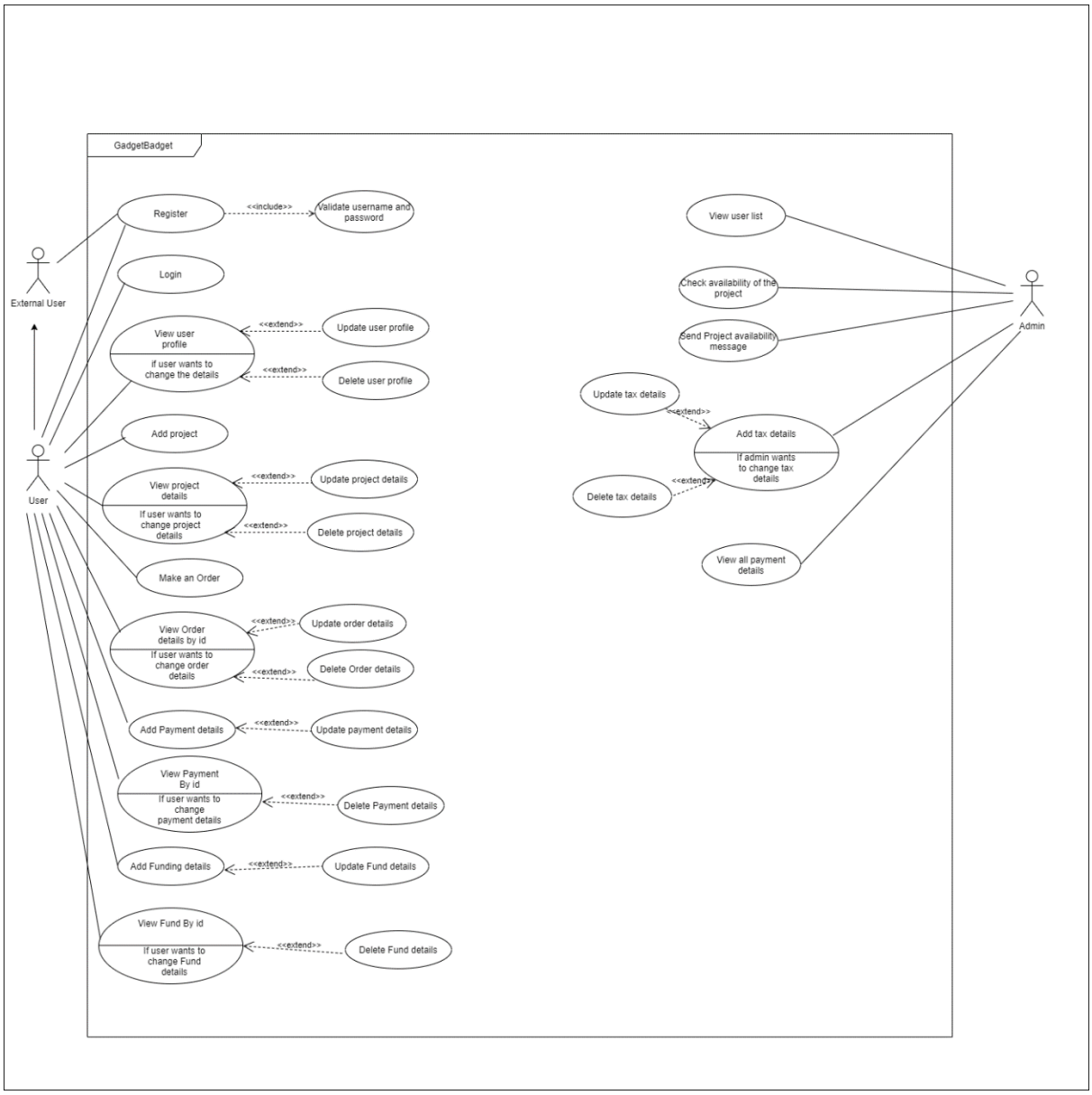


6. Stakeholder Analysis

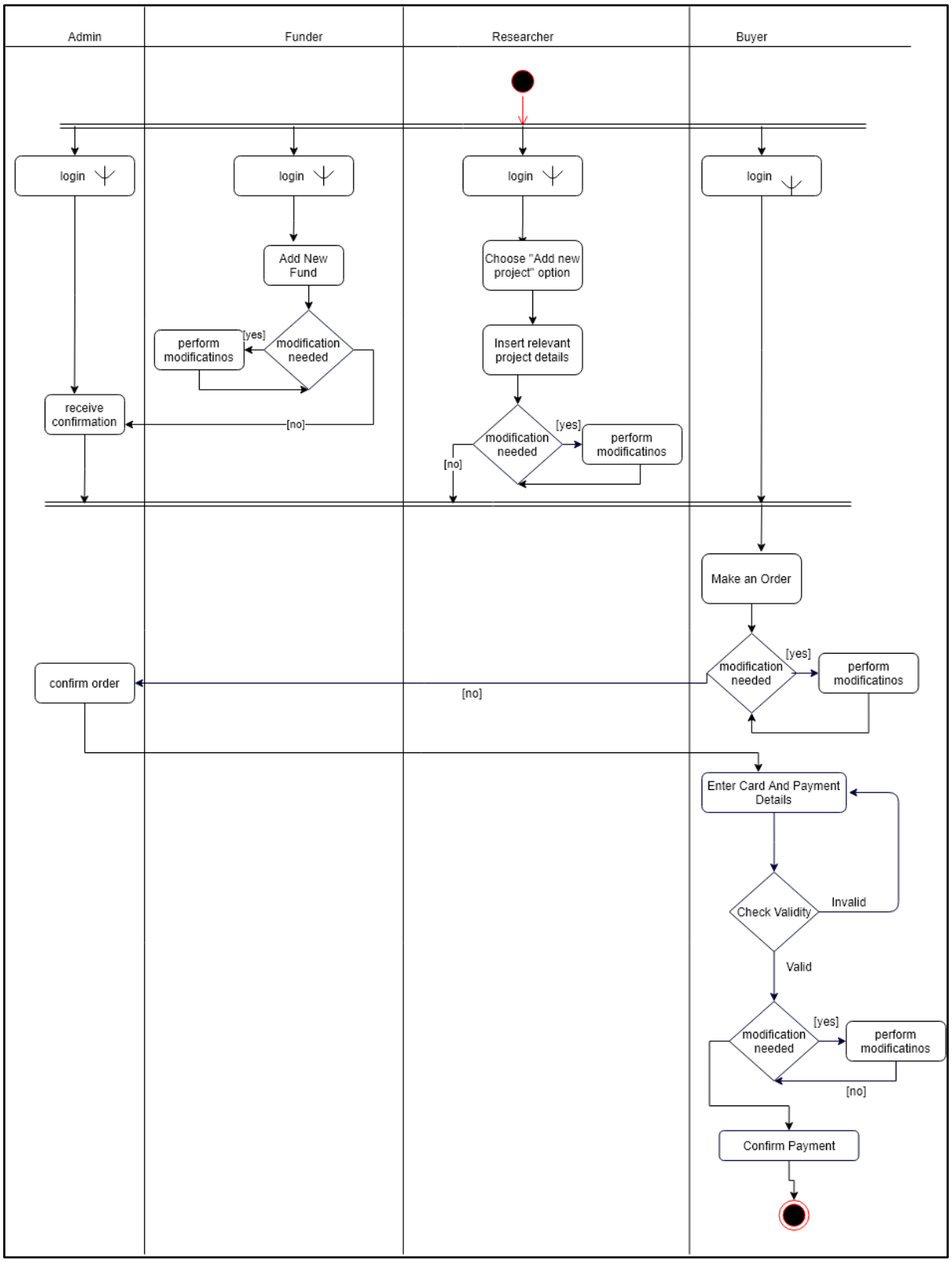
Onion Diagram



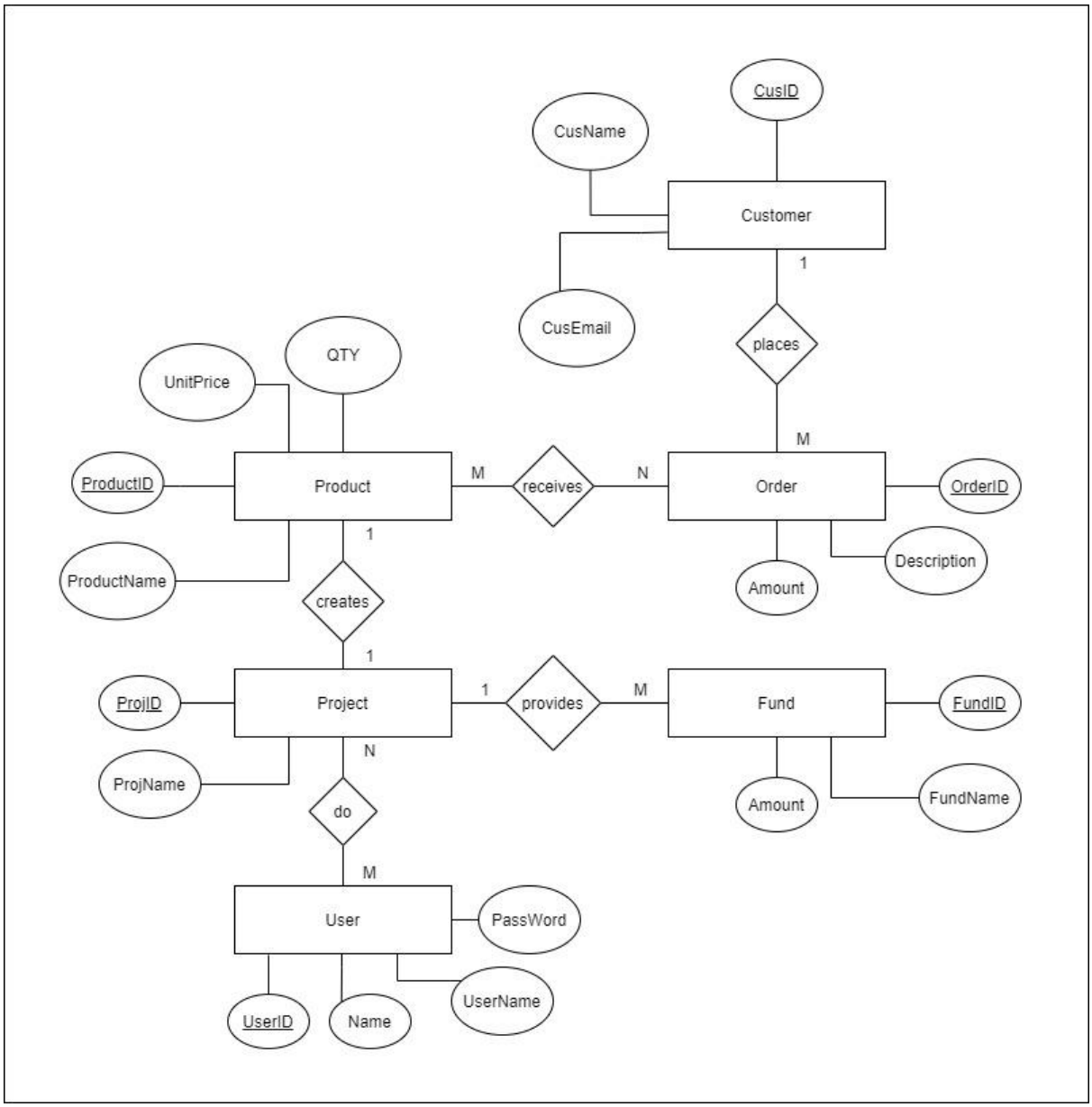
7. Overall Use Case Diagram



8. Overall Activity Diagram



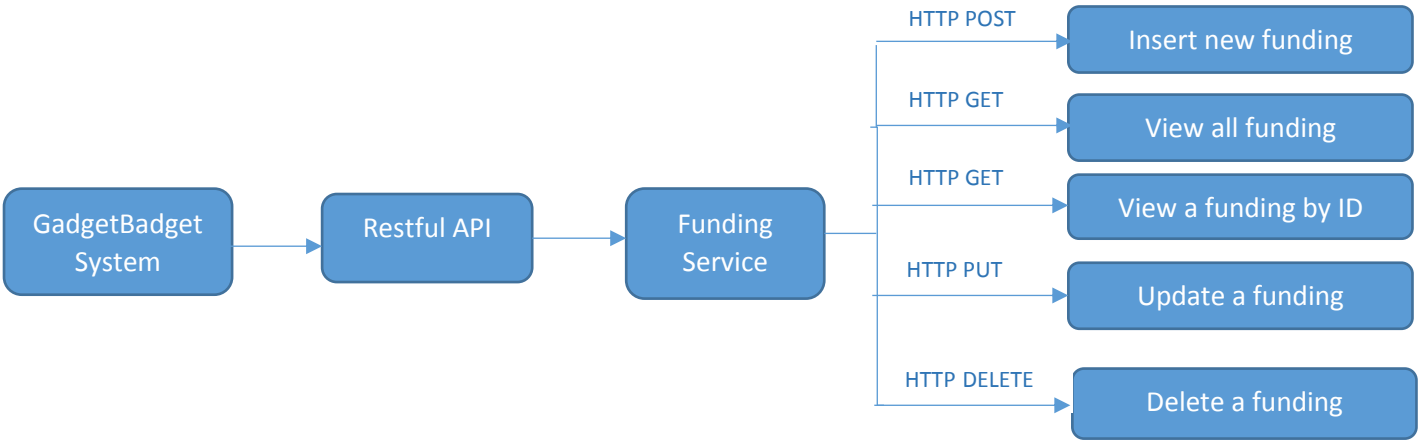
9. Overall ER Diagram



10. Individual Section

IT19201160 Wanasinghe W. A. D. B.

1. API design



Resource	Funds
Request	GET Funding_Service/FundService/Funding (Get all fund details)
URL	http://localhost:8080/Fund_Service/ FundService /Funding/

Resource	Funds
Request	GET Funding_Service/FundService/Funding/{Funder} (Get fund by ID)
URL	http://localhost:8080/Fund_Service/ FundService /Funding/{Funder}

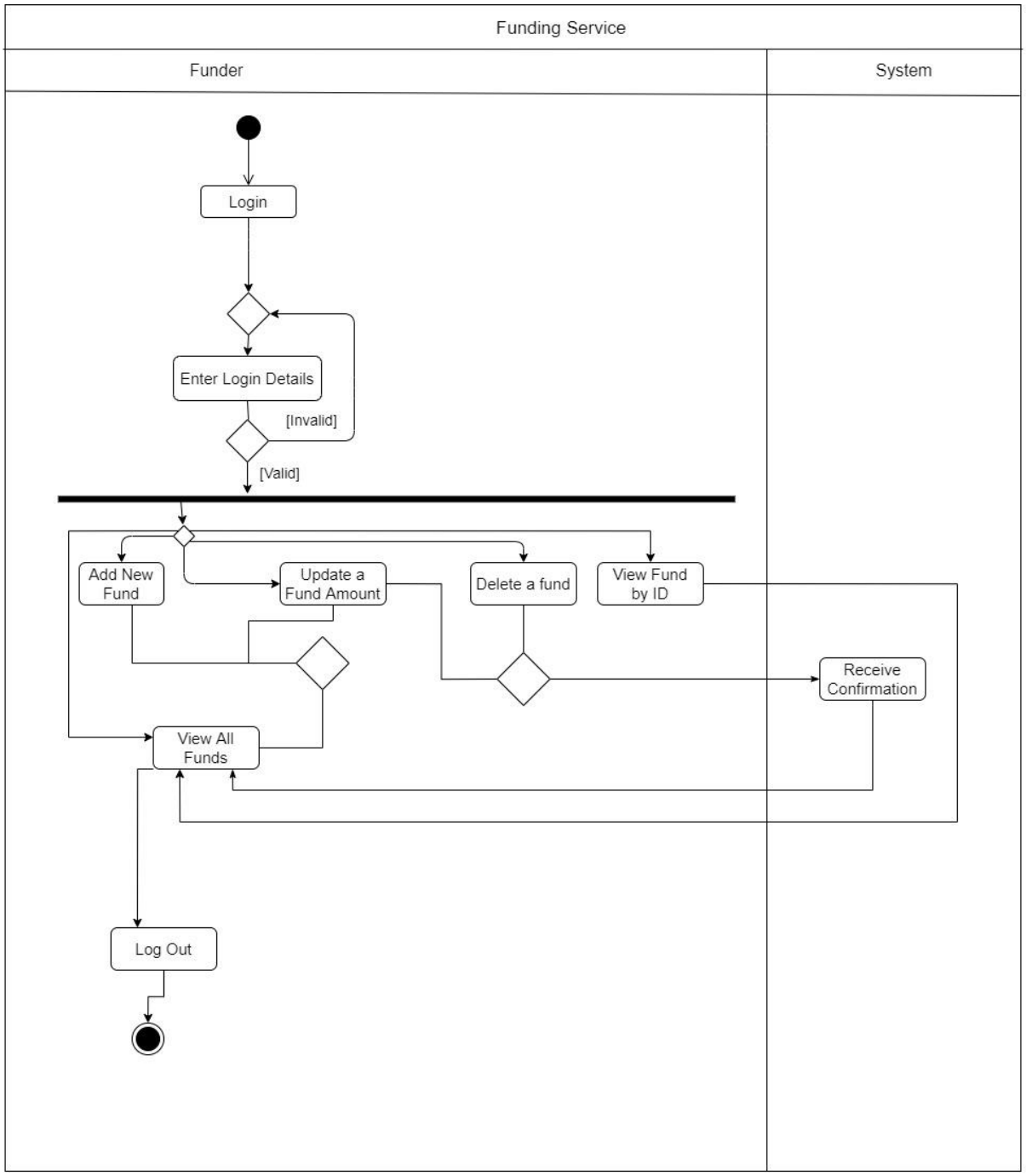
Resource	Funds
Request	POST Funding_Service/FundService/Funding (Post fund detail)
URL	http://localhost:8080/Fund_Service/ FundService /Funding/

Resource	Funds
Request	PUT Funding_Service/FundService/Funding/{Funder} (Update fund by ID)
URL	http://localhost:8080/Fund_Service/ FundService /Funding/{Funder}

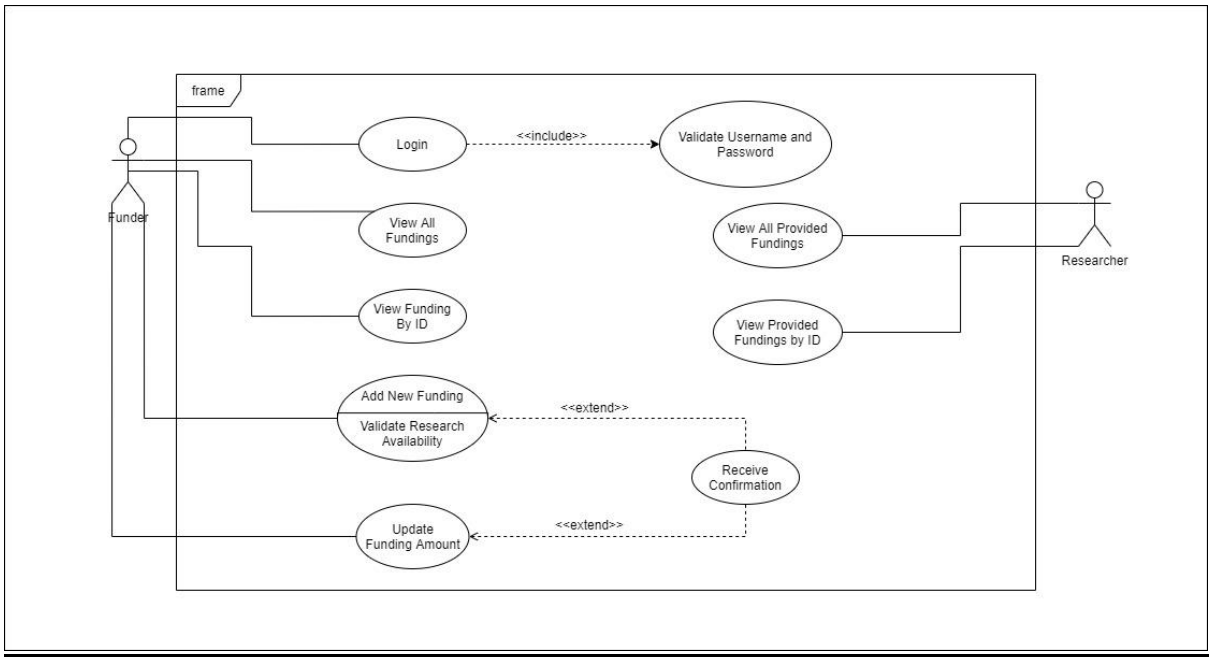
Resource	Funds
Request	DELETE Funding_Service/FundService/Funding/{Funder} (Delete fund by ID)
URL	http://localhost:8080/Fund_Service/ FundService /Funding/{Funder}

2.Internal logic design

Activity Diagram (Funding Service)

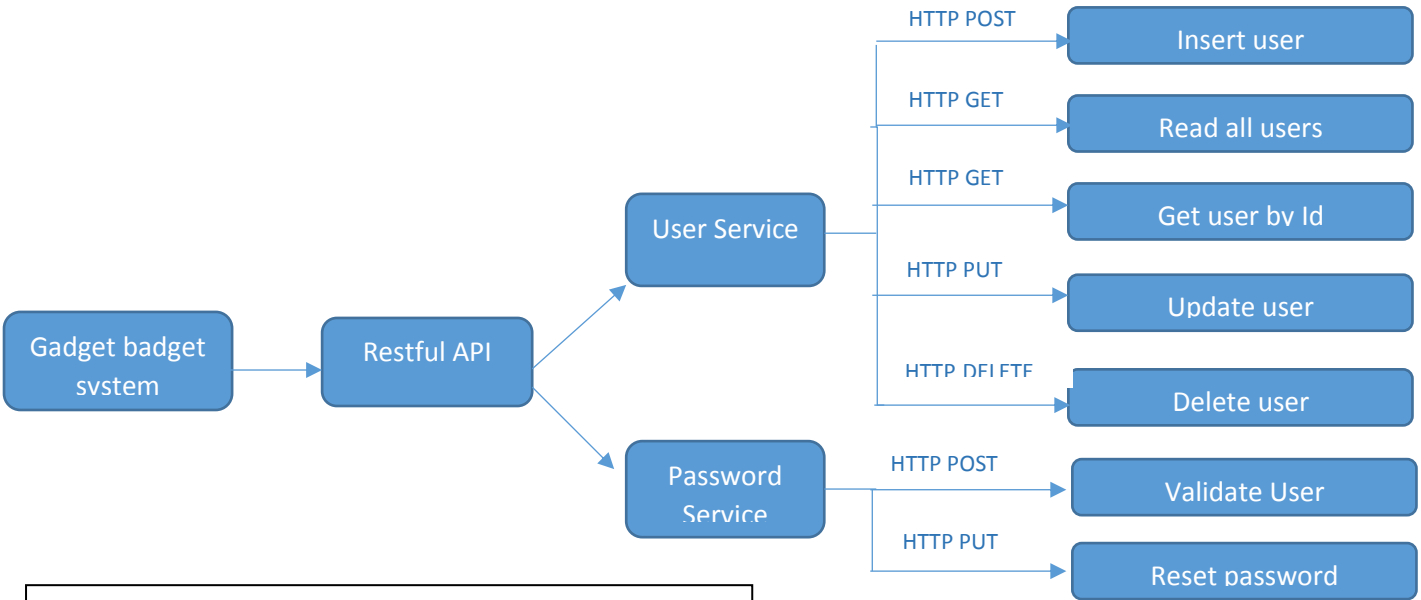


4. Any other relevant design diagrams



IT19201160 Maddugoda C. D.

1. API design



**HTTP DELETE was not used to reset the password since I had not utilized the HTTP Put operation in Password Service. Since Reset is a type of update PUT method was used.

Resource	Users
Request	GET User_Service/myUserService/Users(Get all User details)
URL	http://localhost:8080/User_Service/myUserService/Users/
Media	*APPLICATION_FORM_URLENCODED*
Response	String status message Html table with all user details or `"Error while reading the user details"`

Resource	User
Request	GET User_Service/myUserService/Users/getUserbyID/{userId}(Get user details of a specific user)
URL	http://localhost:8080/User_Service/myUserService/Users/getUserbyId/{userId}
Media	*APPLICATION_FORM_URLENCODED*
Response	String status message Html table with user details with given user ID or `"Error while reading the user details"`

Resource	Users
Request	POST User_Service/myUserService/Users/(insert User details)
URL	http://localhost:8080/User_Service/myUserService/Users/
Media	*APPLICATION_FORM_URLENCODED*
Data	[{"key":" userCode ","value":"002"}, {"key":" name","value": Risal Perera "}, {"key":" NIC","value":"92754638v"}, {"key":" userEmail ","value":" rsal@gmail.com "}, {"key":" userPhone ","value":" 0716028567 "}, {"key":" userType ","value":" admin "}, {"key":" username ","value":" Risali "}, {"key":" password ","value ":" Risal##77 "}]
Response	String status message `"Inserted successfully"` or `"Error while inserting"`

Resource	Users
Request	PUT User_Service/myUserService/Users(update User details)
URL	http://localhost:8080/User_Service/myUserService/Users/
Media	Application/Json
Data	{ "userId":"4", "userCode":"007", "name":"Mubarak", "NIC":"640675645v", "userEmail":"mubarak64@gmail.com", "userPhone":"0719078222", "userType":"Customer", "username":"Mubarak",

	"password":"mubbzz%646" }
Response	String status message `"Updated successfully"` or `"Error while updating the user "`

Resource	Users
Request	DELETE User_Service/myUserService/Users(delete User profile)
URL	http://localhost:8080/User_Service/myUserService/Users/
Media	Application/xml
Data	<userData> <userID>3</userID> </userData>
Response	String status message `"Deleted successfully"` or `"Error while deleting the user "`

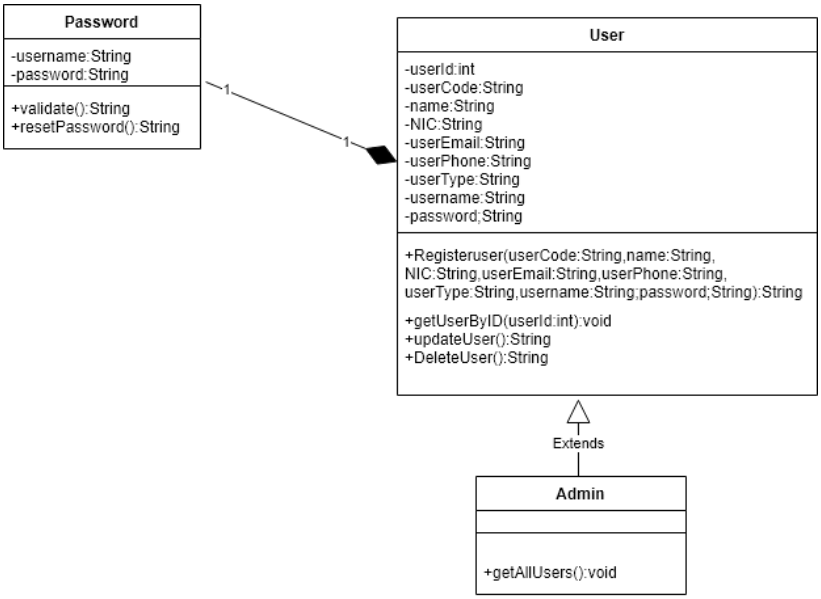
Resource	Password
Request	POST User_Service/myUserService/ Password/validateCredentials(validate User details)
URL	http://localhost:8080/User_Service/myUserService/ Password/validateCredencials/
Media	APPLICATION_FORM_URLENCODED
Data	[{"key":" username ","value":" Risali "}, {"key":" password ","value ":" Risal##77 "}]
Response	String status message `"Welcome "+ userName` or `"Welcome Admin"` or `"incorrect Username or password"`

Resource	Password
Request	PUT User_Service/myUserService/Password/(reset password)
URL	http://localhost:8080/User_Service/myUserService/Password/
Media	*APPLICATION_FORM_URLENCODED*
Response	String status message `"Password reseted Successfully"` or `"Error while Reseting the Password`

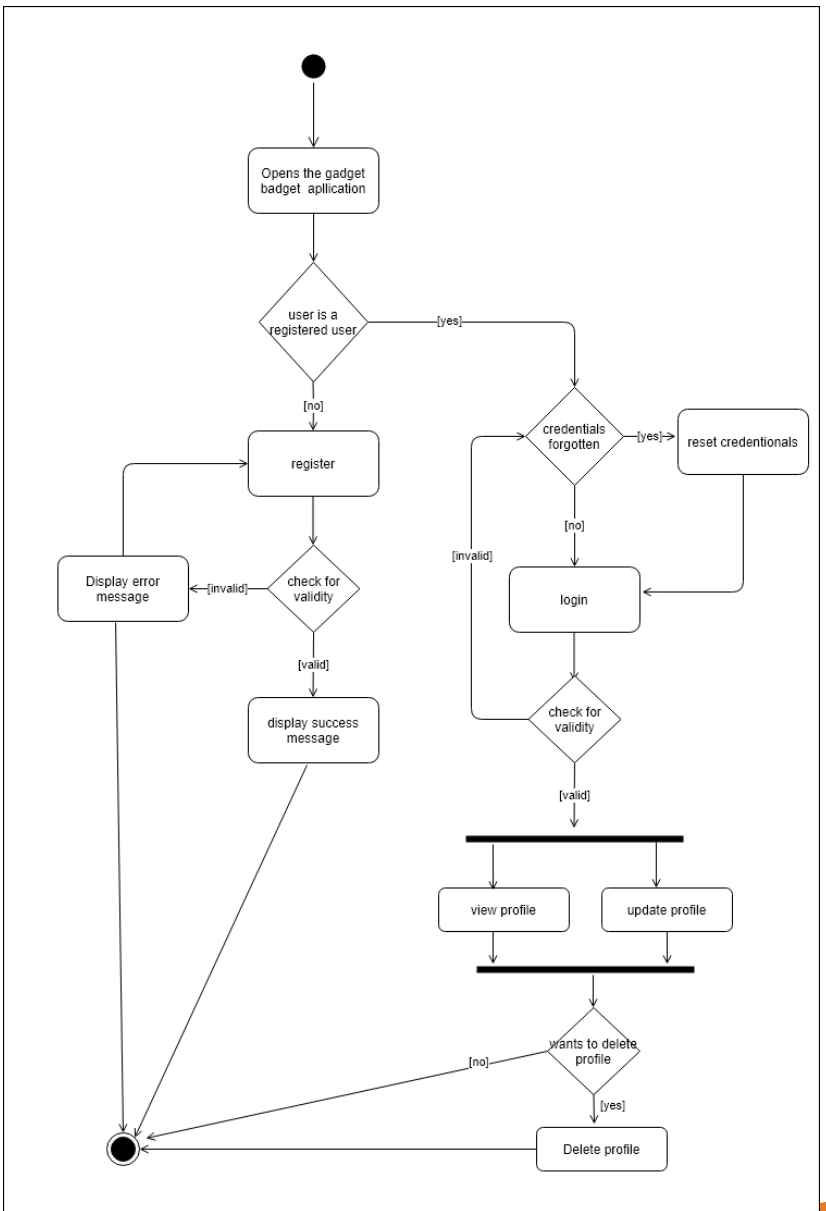
2.Internal logic design

Class Diagram

User can register to the system by providing registration details. Admin inherits all attributes and method of user and contains a special method to view details of all the users. Password class validates the credentials when a user tries to log in to the system as well as enables resetting of the password if password is forgotten. Password class cannot exist without a user.

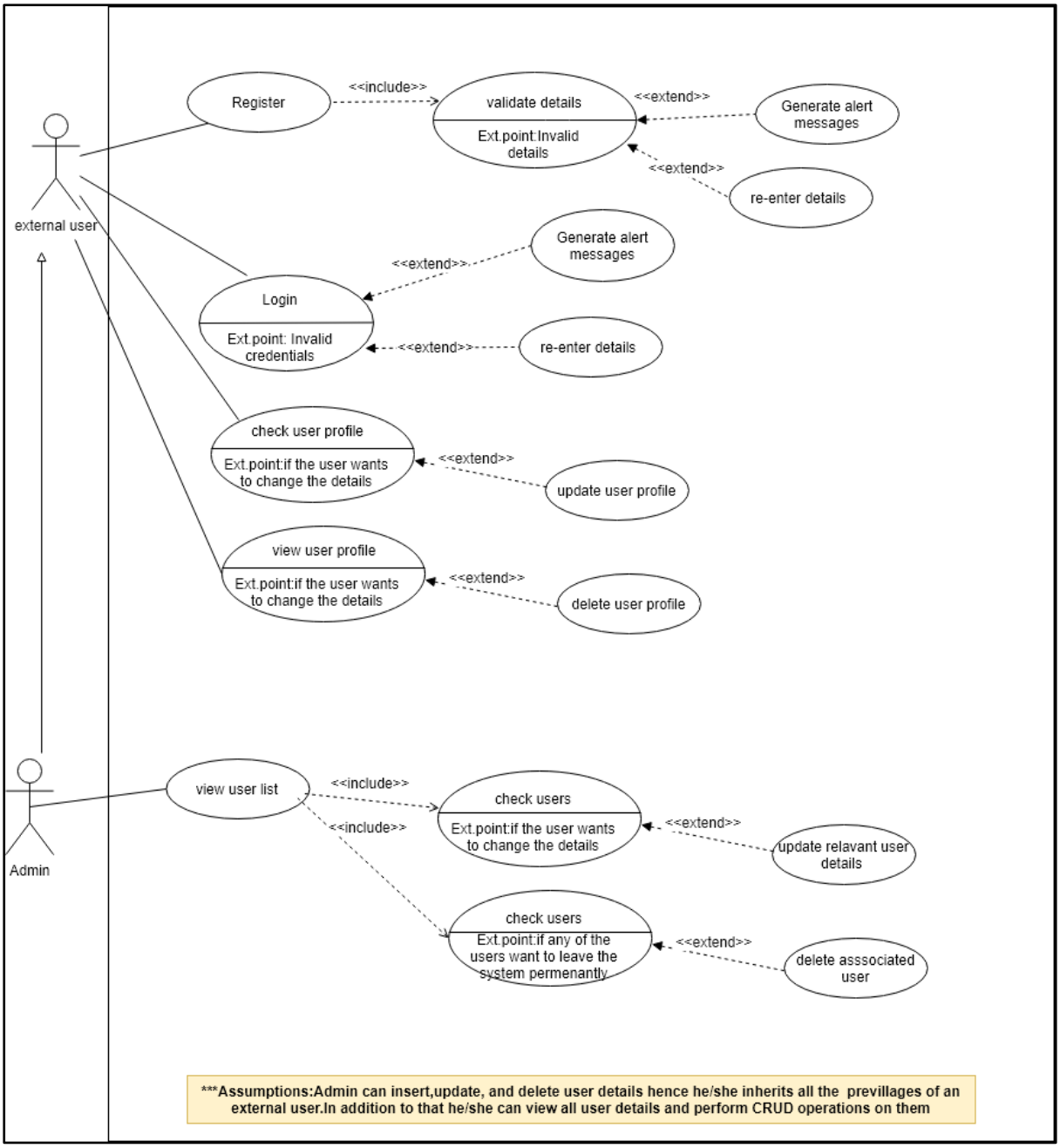


Activity Diagram



Any Other Diagrams

Use Case Diagram



5. Development tools selection and justification

Dependency management tool: Maven –

- Main reason for the selection is its ease to add new dependency. It downloads all the required jar files automatically into eclipse so that it is not required to edit pom.xml manually.
- minimal configuration is required
- Maven is recommended to have best performance with Eclipse by most professionals and since the project is developed using Eclipse, I find it as an excellent dependency managemet tool.

- Maven setup build environment(i.e takes care of the version conflicts in the dependant libraries so that I can get rid of the errors like NoClassDefFoundError).

Testing tool: Postman –

- It is easy to test the backend development without a proper frontend.
- It is recommended to have best performance for restful APIs with postman than any other testing tool by professionals.
- Different formats of data(xml,json,text) can be passed to post requests.
- It generates the response within a very short time
- Postman has a user friendly interface with easy selections(for content type) multiple ways to pass data and also a preview available to view results which makes it easy to test the service based on the response.

Framework: Jersey –

- It is recommended to provide best performance with eclipse IDE
- It supports JAX RS apis
- Jersey is a good development framework for restful web services

6. Testing methodology and results

Functional Testing –

- Form formatting is paradigmatic
- Inserted values are validated and all the validations functions properly
- Test cases are executed providing expected output

Security Testing –

- Data validation and authentication techniques are working properly
- Data access levels are limited according to user level.

Database testing –

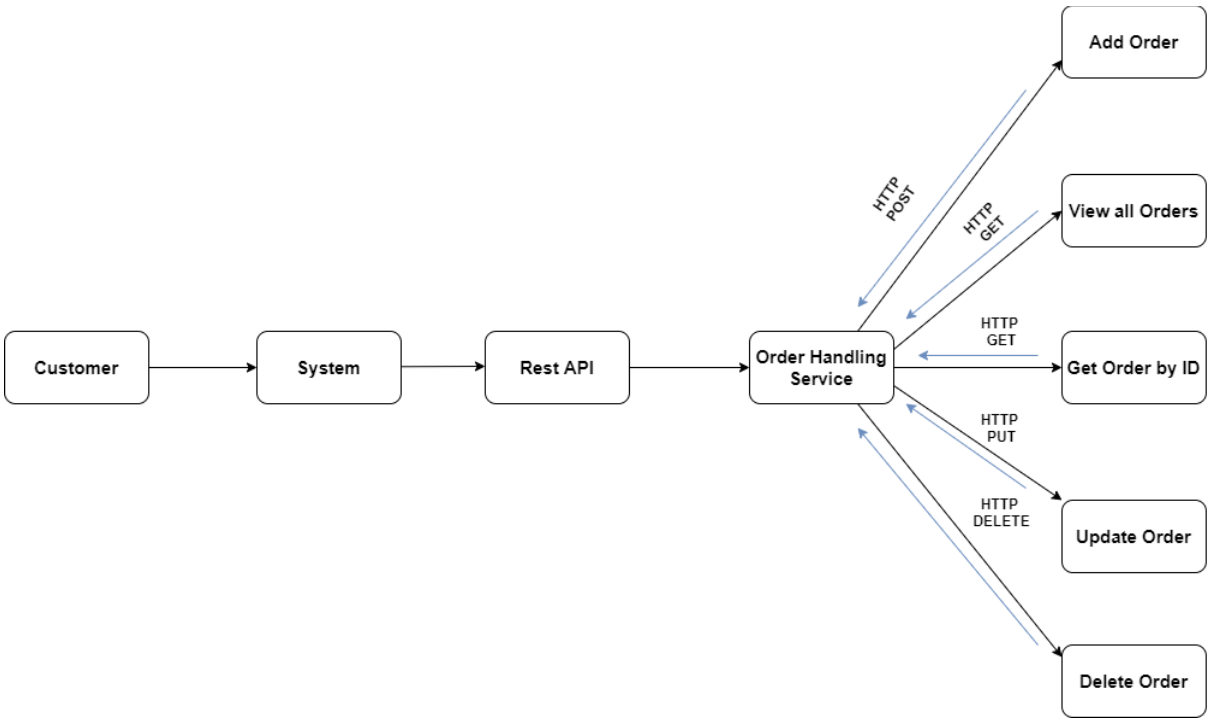
- CRUD operations are performed without any errors.
-

Test Id	Test Case	Inputs	Expected Output	Actual Output	Status(Pass/Fail)
1	Get all the user details	-	All the user details in an html table should be displayed	All the user details in an html table has been displayed	pass
2	insert user details	[{"key":" userCode ","value":"002"}, {"key":" name","value": Risal Perera "}, {"key":" NIC","value":"92754638v"}, {"key":" userEmail ","value":" rsal@gmail.com "}, {"key":" userPhone ","value":" 0716028567 "}, {"key":" userType ","value":" admin "}, {"key":"	Display "Inserted successfully "	"Inserted successfully " has been displayed	pass

		username ","value":" Risali "}, {"key":" password ","value ":" Risal##77 "}]			
3	Update user details	{ "userId":"4", "userCode":"007", "name":"Mubarak", "NIC":"640675645v", "userEmail":"mubarak64@gmail.com ", "userPhone":"0719078222", "userType":"Customer", "username":"Mubarak", "password":"mubbzz%646" }	Display “Updated successfully ”	“Updated successfully ” has been displayed	pass
4	Delete user account	<userData> <userID>3</userID> </userData>	Display “Deleted successfully ”	“Deleted successfully ” has been displayed	pass
5	Validate login credentials	[{"key":" username ","value":" Risali "}, {"key":" password ","value ":" Risal##77 "}]	Display “welcome Admin”	“welcome Admin” has been displayed	pass

IT19170176 Fernando W. N. D.

API for the Service



Resource: OrderService API

Request: GET (Get all orders)

Media :
TEXT_HTML

Response: String status message

"Error occur during retrieving data”

Request: GET
(Get orders by Customer ID)

Media :
TEXT_HTML
Data : OrderID

Response: String status message

"Error occur during retrieving "

Resource: OrderService API

Request: POST (Add orders)

Media :

APPLICATION_FORM_URLENCODED

Data :
customer_customer_id,project_project_id ,date

Response: String status message

"The particular project has been ordered please choose another project."

"You cannot request past dates as order dates Please select a future date"

"Order added successfully."

"Error occurred during adding an Order"

Resource: OrderService API

Request: PUT (Update order)

Media :
APPLICATION_FORM_URLENCODED

Data : date,OrderID

Response: String status message

"The particular project has been reserved please choose another project."

"Order Updated successfully"

"Error occur during updating an Order"

Resource: OrderService API

Request: DELETE
(Delete order)

Media :
APPLICATION_FORM_URLENCODED

Data : OrderID

Response: String status message

“Order Deleted successfully.”

"Error occur during deleting the order"

URL for API request

GET - http://localhost:8093/Order_Service/OrderService/Orders

GET - http://localhost:8093/Order_Service/OrderService/Orders/getOrderbyID/Ord1/1

POST - http://localhost:8093/Order_Service/OrderService/Orders/add

PUT - http://localhost:8093/Order_Service/OrderService/Orders/update/Ord1/1

DELETE - http://localhost:8093/Ord_Service/OrderService/Orders/delete/Ord1/1

Internal logic design

- **Class diagram**

  This shows the relationship between the classes. In this service, there are association and the composition relationships between classes.

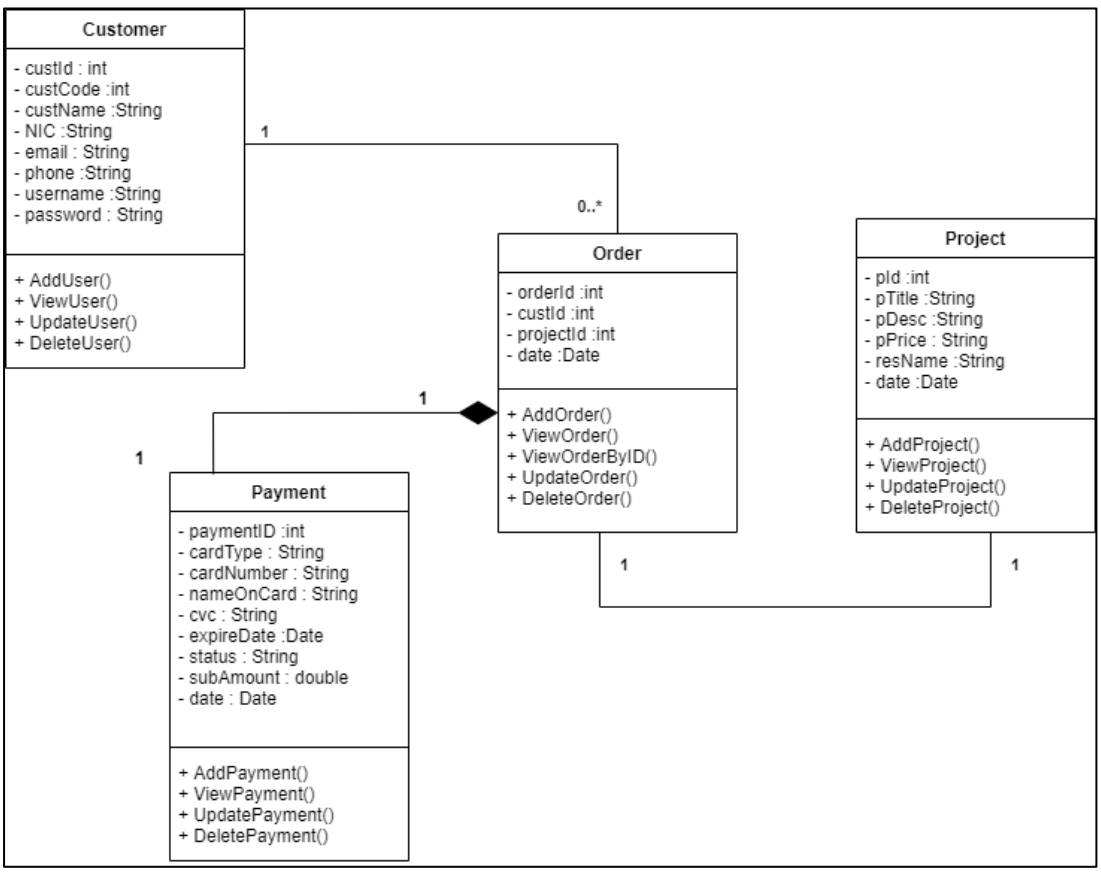
Service Development and Testing

- Dependency management tools
  - IDE: Eclipse

  Database: MySQL

  Back End: Java
  - Maven
  - JAX-RS
- Testing Tools
  - Postman
- Code quality checking tools

Internal Logic – Class Diagram



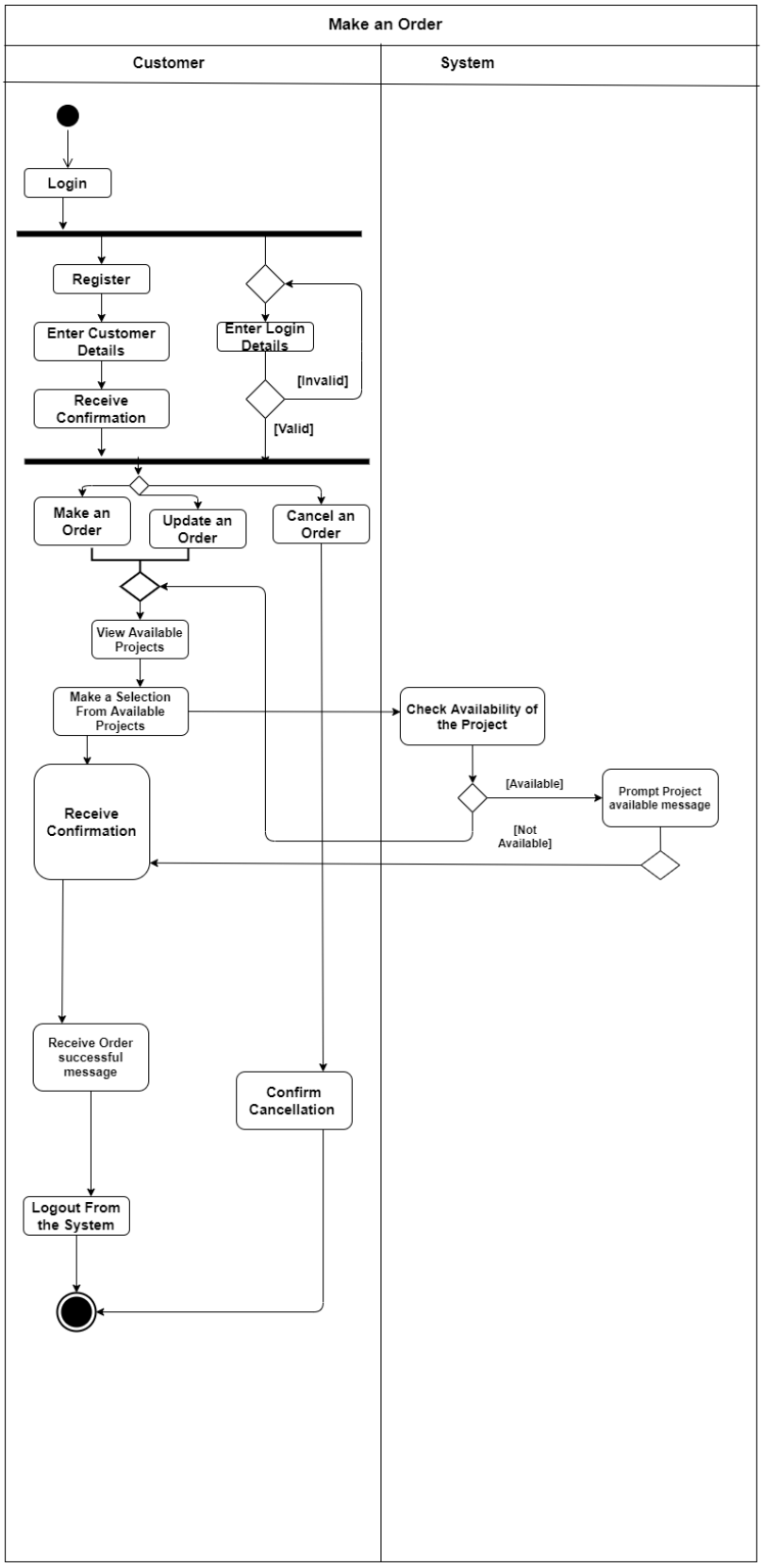
Testing methodology and results

Test ID	Test Description/ Test Steps	Test Input(s)	Expected Output(s)	Actual Output(s)	Result (Pass/Fail)
01	View All Orders.	URL for the API Send Get request.	Display orders details.	Display Orders details.	Pass
02	View Order By ID	OrderID given for the URL with a get request	Display Order details of the given orderID	Display Order details of the given orderID	Pass
03	Add an order.	Attributes to be updated along with the order ID.	"Order Added Successfully."	"Order Added Successfully	pass
04	Update an order.	Date and order ID.	"Order Updated Successfully."	"Order Updated Successfully."	pass
05	Delete an order.	Given order ID of the appointment to be deleted.	"Order Deleted Successfully."	"Order Deleted Successfully"	pass

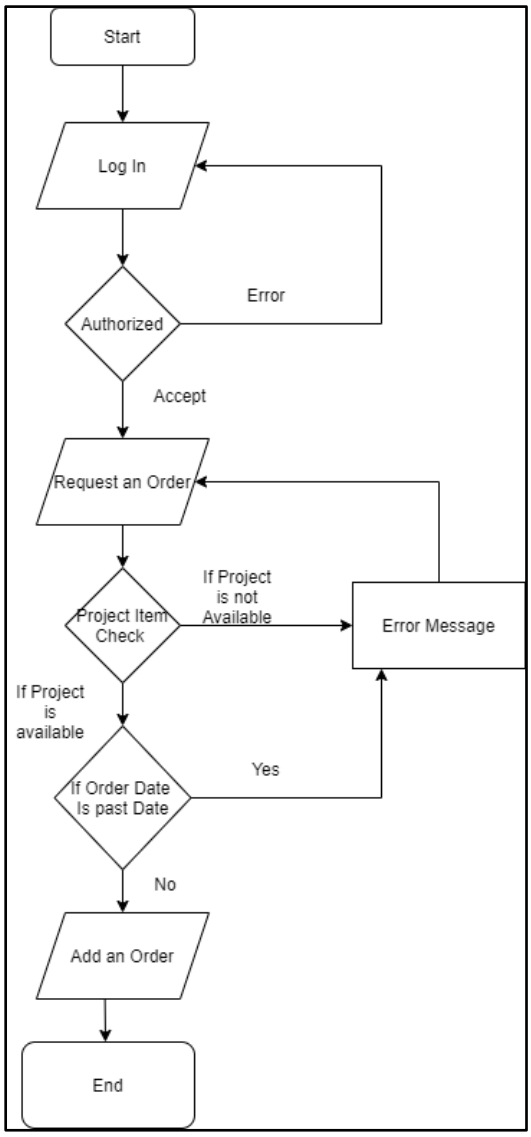
Assumptions and Discussions

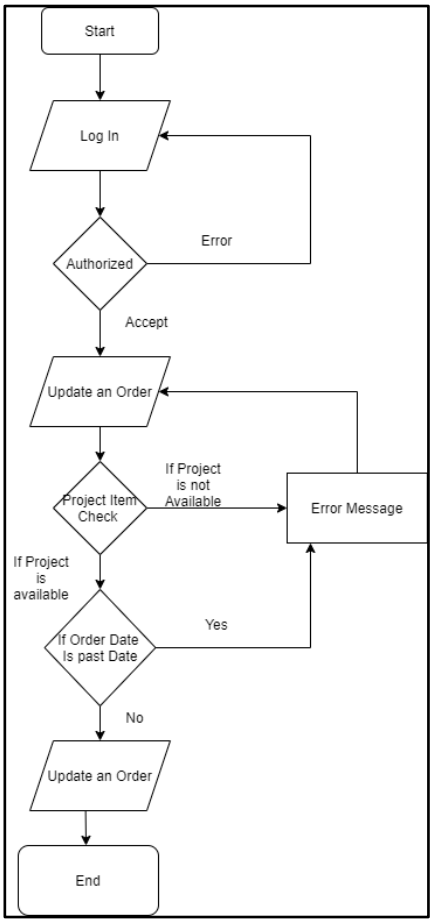
- Only one research project is available from a relevant project (Quantity of a project is only one)
- An order can be made only for a one project at once. (Multiple projects cannot be selected for one order)
- When the customer selects the names from the dropdown such as project name and customer name, the ids will be sent to the service instead of the names through the front end.
- Customer can request orders before current date only.
- Customers can only delete an order after one week from the day initial confirmation of the order.

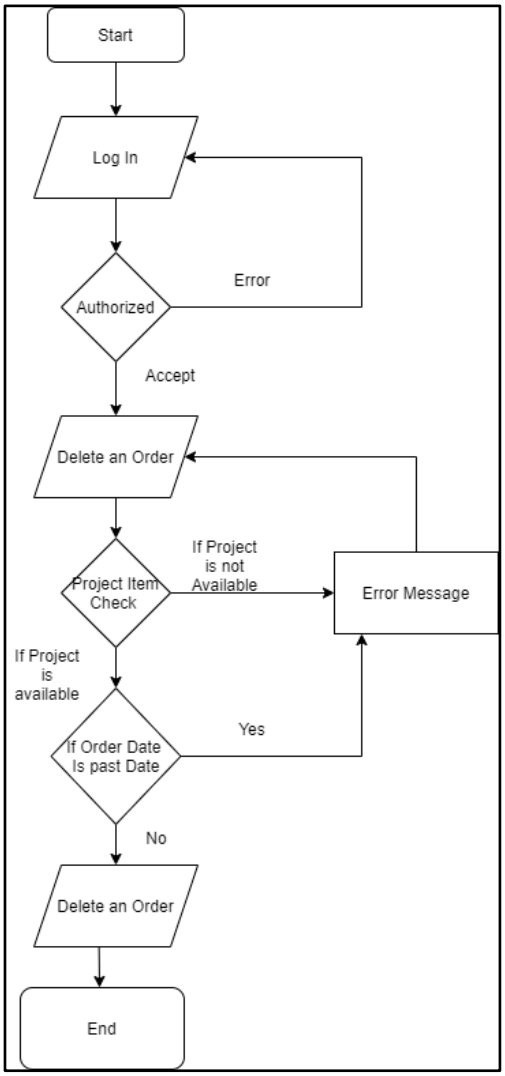
Activity Diagram



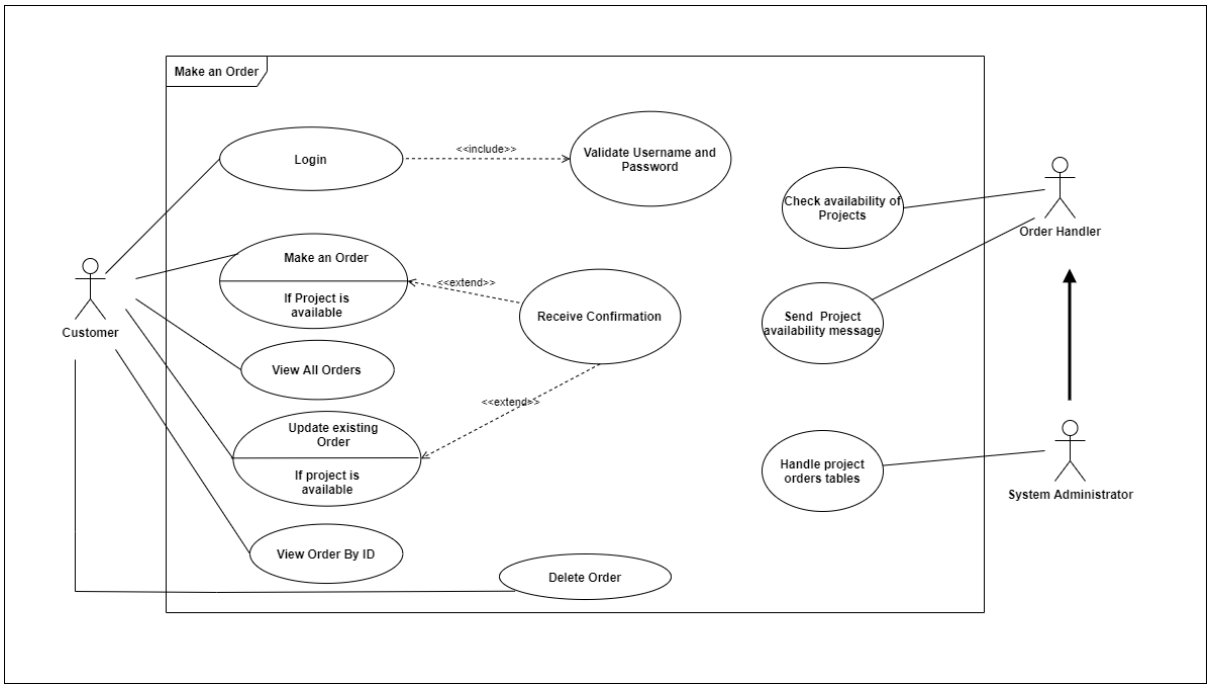
Flow Chart for Make an Order, Update an Order and Delete an order





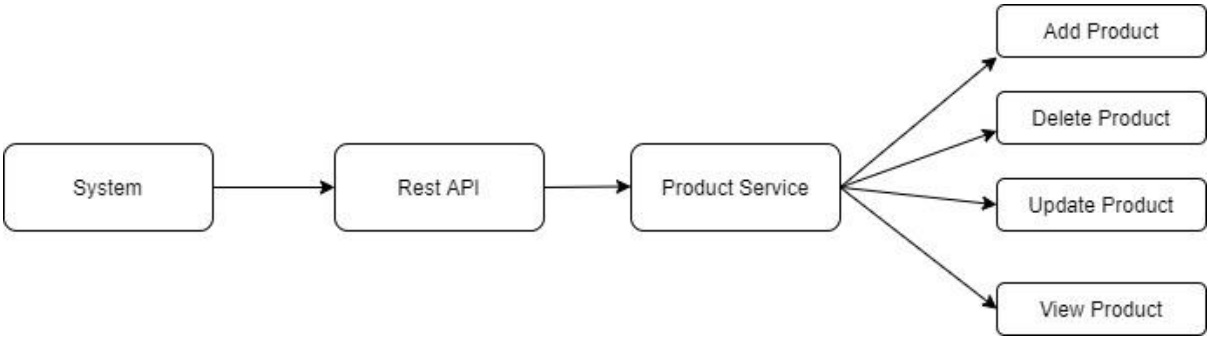


Use case diagram – Order Service



IT19174686 Ramawickrama H.N

1. **API Design : Product Service**



POST – Insert Product	**GET – Read product details**

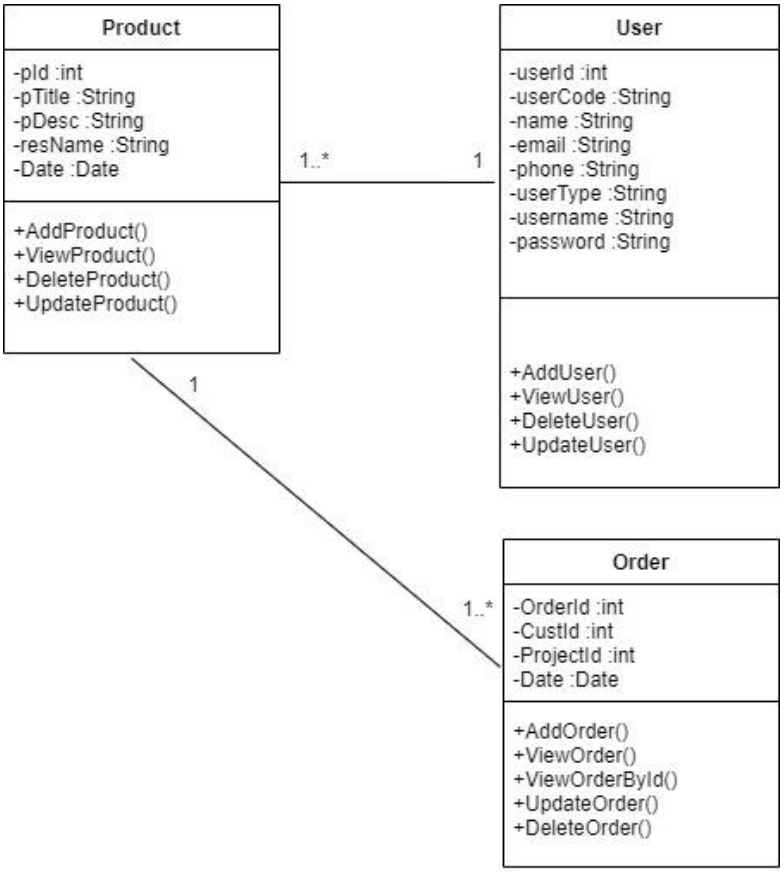
URL:
http://localhost:9881/ProductService/ProductService/Product
 	
Resource : Product
 	
Request :
POST Product_Service/ProductService/Product
 	
Media ***APPLICATION_FORM_URLENCODED***
 	
Data:
{
pTitle : " online election System project"",
pDesc : "Voters as the can get to know the candidate background and choose wise",
pPrice : "13,500.00",
resName : "Rakith Dias",
Date : "2021-02-12"
}
 	
Response : String status message
"Inserted successfully" or
"Error while inserting the product."
	URL:
	http://localhost:9881/ProductService/ProductService/Product
	
	Resource: Product
	
	Request:
	GET Product_Service/ProductService/{pId}
	
	Media : ***TEXT_HTML***
	
	Response : String status message
	"success" or
	"Error while reading the products."

DELETE – Delete Product by ID	**UPDATE – Update product details**
URL: http://localhost:9881/ProductService/ProductService/Product Resource : Product Request : **DELETE** Product_Service/ProductService/{pId}/Product	URL: http://localhost:9881/ProductService/ProductService/Product Resource : Product Request : **PUT** Product_Service/ProductService/{pId}/Product

Media : ***APPLICATION_XML***
 	
Data : <productData>
<pId>7</pId>
</productData>
 	
Response : String status message
`" Deleted successfully " or`
 	
`" Error while deleting product”`
	Media : ***APPLICATION_JSON***
	
	Data:
	{
	"pId" : "7",
	"pTitle": "New online election System project",
	"PDesc" : "Voters will get Unique ID and Passwor d, Using which they can vote for a Candidate only o nce per Election",
	"pPrice" : "12,000.00",
	"resName" : "Rakith Dias",
	"Date" : "2021-02-12"
	}
	Response : String status message
	`" Updated successfully " or`
	`"Error while updating the product”`

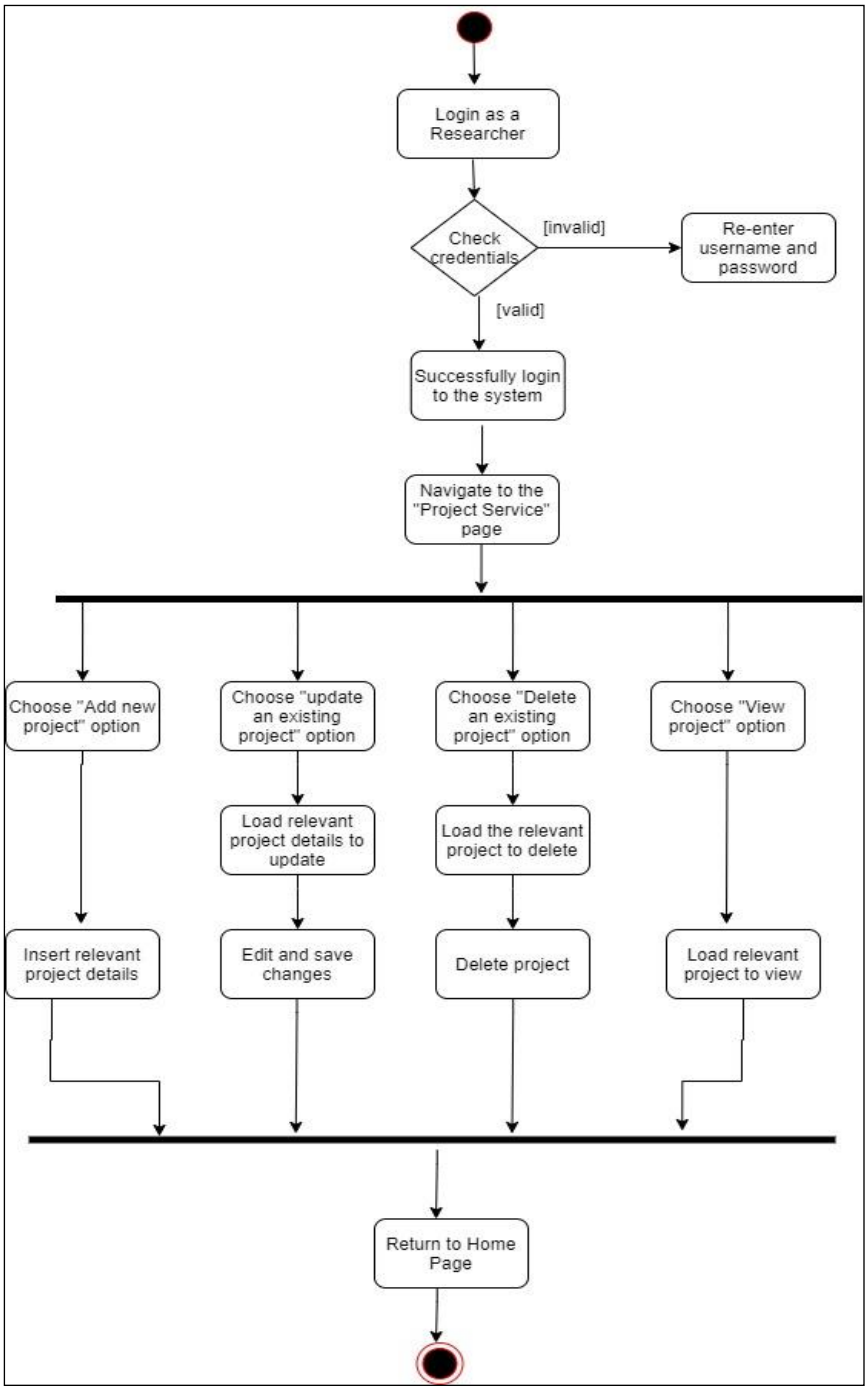
2. Internal Logic Design

Class Diagram

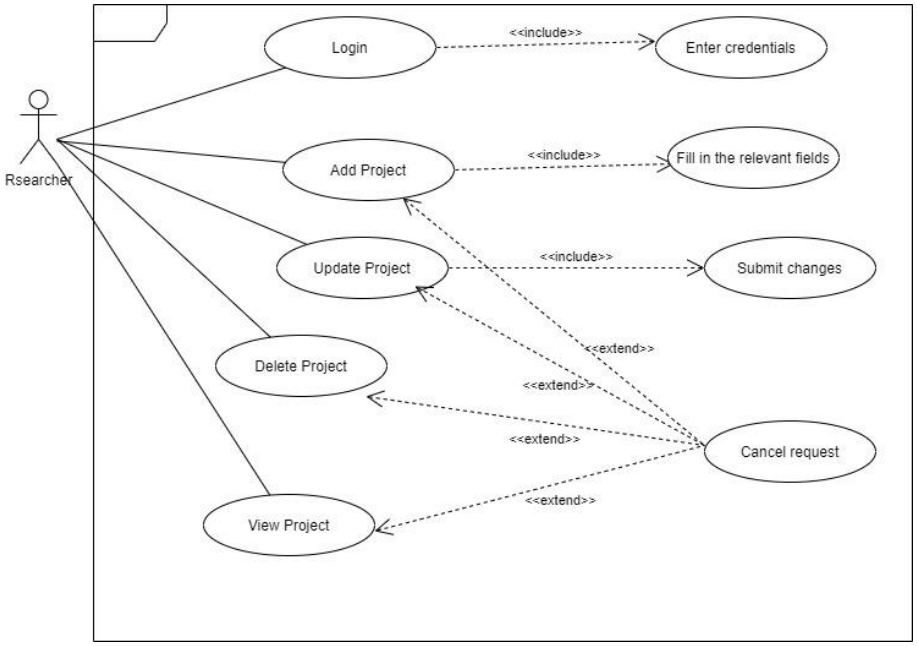


4 .Any other relevant design diagrams

Activity Diagram: Product Service



Use Case Diagram:



5. Development tools selection and justification

Dependency/Package management tool: Maven

- Maven is based on POM and was used to build and manage the project.
- It provided a uniform build system and quality project information such as log document, dependency lists etc..
- It was easier to add jars and other dependencies to the project with the help of Maven.
- Maven was also helpful to integrate our project using Git.
-

Database design tool: MySQL Workbench

- It helps to gain a better visibility on the database.
- The features that the workbench provides, makes the query execution simpler and less complex: This may include features such as auto completion, highlighting of syntax using different colors etc.

Testing tool: Postman

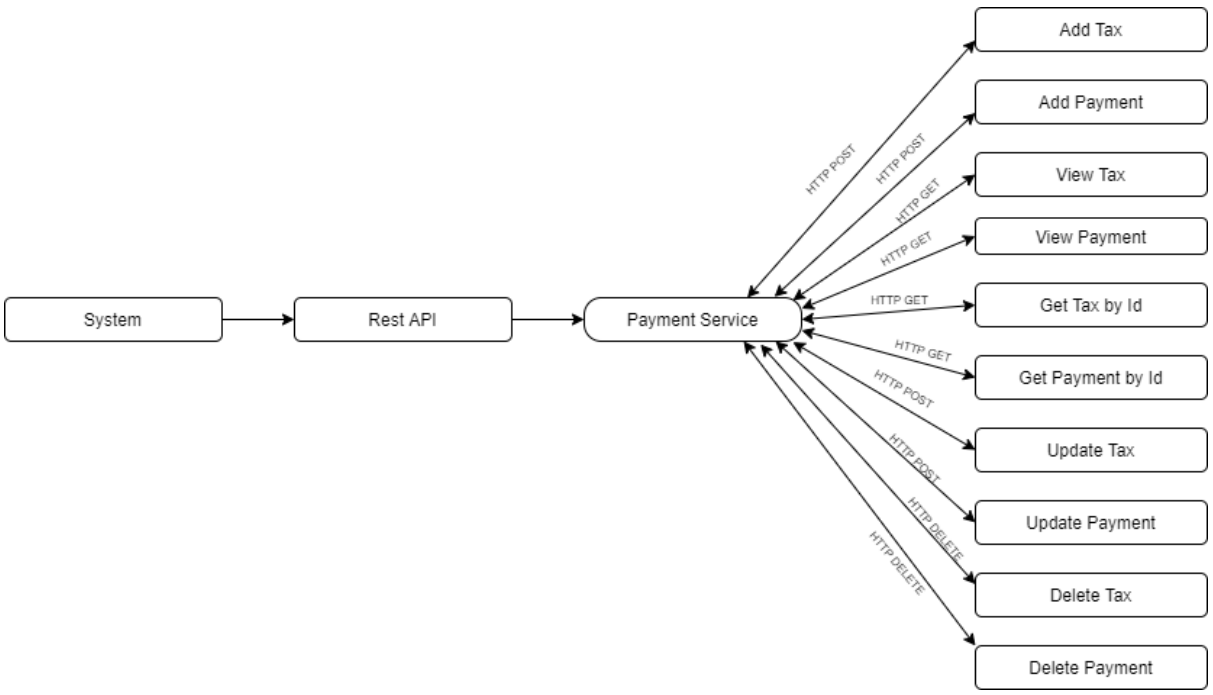
- Postman stores information in an organized way which makes it easier for testing.

6. Testing Methodology and Results

Test ID	Description	Test Inputs	Expected Output	Actual Output	Result(Pass/Fail)
01	Insert Product details	Product ID, Product Title, Product Description, Researcher Name, Date	Display message “Product inserted successfully”	“Product inserted successfully” message is displayed	Pass
02	View Product details by product ID	Product ID	Display product details for the given ID	Product details for the relevant ID is displayed	Pass
03	Update Product details	Product Title, Product Description, Researcher Name, Date	Display message “Product details updated successfully”	“Product details updated successfully” message is displayed	Pass
04	Delete Product	<productData> <pId>7</pId> </productData>	Display message “Product deleted successfully”	“Product deleted successfully” message is displayed	Pass

IT18125726 Kumarage L. V. A.

API for the Service



Resource: PaymentService API

Request: **POST** (Add payment)

Media: *Application _from _Urlencoded*

Data: Card Type, Card Number, Name on Card, CVC, Expire Date, Status, Payment Date, Order Id

Response: String status message

"Payment added successfully"

"Error occur during adding"

Resource: PaymentService API

Request: **GET** (Get payment by Customer ID)

Media*: TEXT_HTML*

Data: Customer Id

Response: String status message

"Error occur during retrieving "

Resource : PaymentService API

Request : **PUT** (Update payment by ID)

Media : *Application _from _Urlencoded*

Data : Card Type, Card Number, Name on Card, CVC, Expire Date, Status, Payment Date, Order Id

Response : String status message

"Payment updated successfully"

"Error occur during updating "

Resource : PaymentService API

Request : **DELETE** (Delete Payment by id)

Media : *Application _form _URL encoded*

Data : Payment Id

Response : String status message

"Deleted successfully"

Resource : PaymentService API

Request **: POST** (Add Tax entry)

Media : *Application _from _URLencoded*

Data : Amount, Valid From, Valid To

Response : String status message

"Tax entry added successfully"

"Error occur during adding"

Resource : PaymentService API

Request **: PUT** (Update Tax entry)

Media : *Application _from _URLencoded*

Data : Tax ID

Response : String status message

"Tax entry updated successfully"

"Error occur during updating"

Resource : PaymentService API

Request : **GET** (Get all Tax Entries)

Media : *TEXT_HTML*

Data : Tax ID, Amount, Valid From, Valid To

Response : String status message

"Error occur during retrieving "

Resource : PaymentService API

Request : **GET** (Get Tax Entry by ID)

Media : *TEXT_HTML*

Data : Tax ID,Amount,Valid From ,Valid To

Response : String status message

"Error occur during retrieving "

Resource : PaymentService API

Request : **GET** (Get all Payments)

Media *TEXT_HTML*

Data : Card Type, Card Number,

Name on Card, CVC, Expire Date, Status, Payment Date, Appointment ID

Response : String status message

`"Error occur during retrieving "`

Resource : PaymentService API

Request : **DELETE** (Delete Tax Entry if not used)

Media : *TEXT_ PLAIN*

Data : Tax Id

Response : String status message

`"Cannot delete tax entry already used "`

`"Tax entry deleted successfully"`

URL for API

Get: http://localhost:8010/Payment_Service/PaymentService/Payment/get

Push: http://localhost:8010/Payment_Service/PaymentService/Payment/add

Put: http://localhost:8010/Payment_Service/PaymentService/Payment/update/payment/6

Delete: http://localhost:8010/Payment_Service/PaymentService/Payment/delete/payment/1

Service Development and Testing

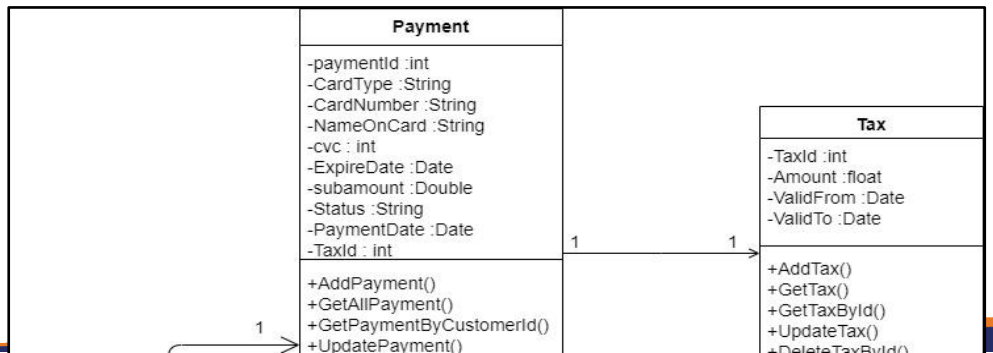
- Dependency management tools
  - IDE: Eclipse

  Database: MySQL

  Back End: Java
  - Maven
  - JAX-RS
- Testing Tools
  - Postman
- Code quality checking tools

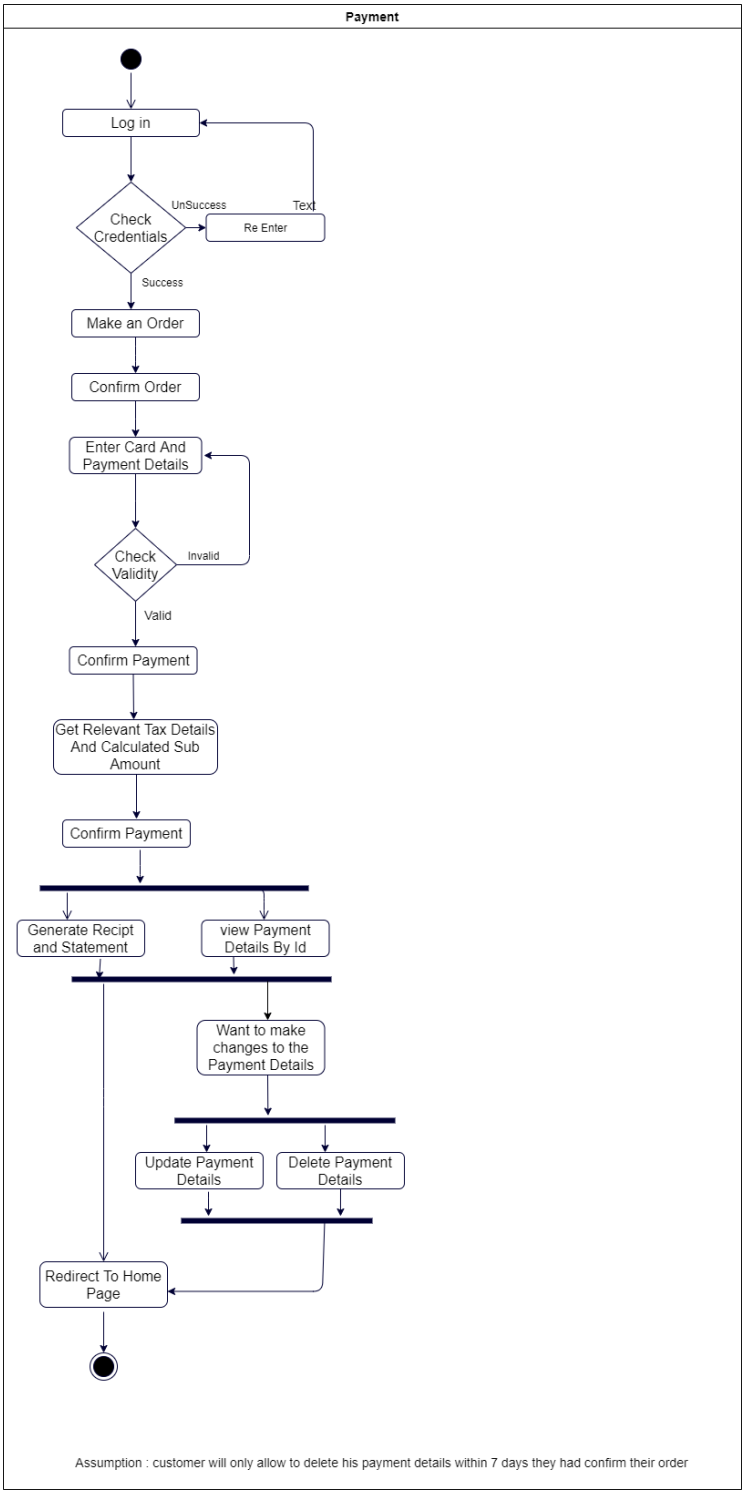
Internal logic design

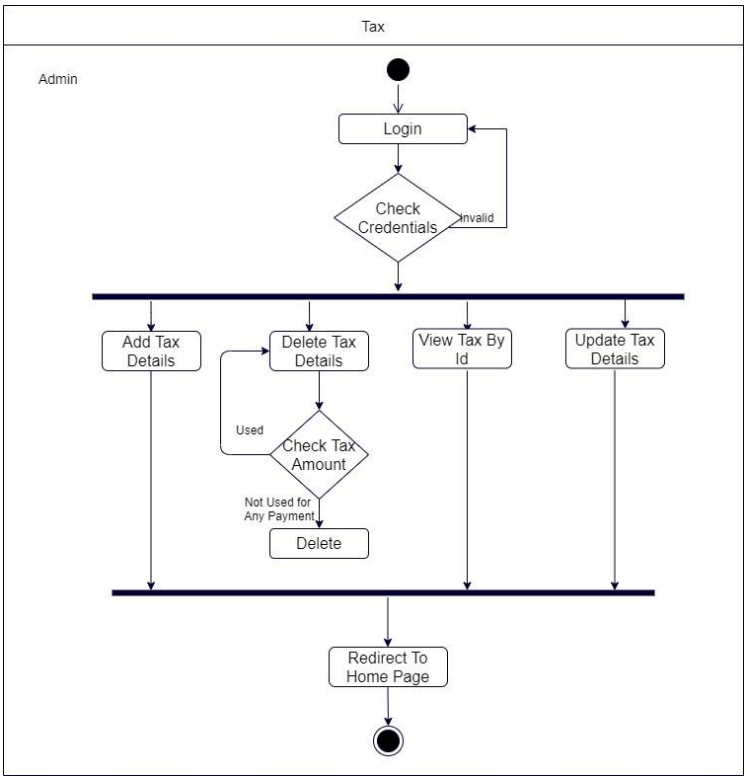
- **Class diagram**



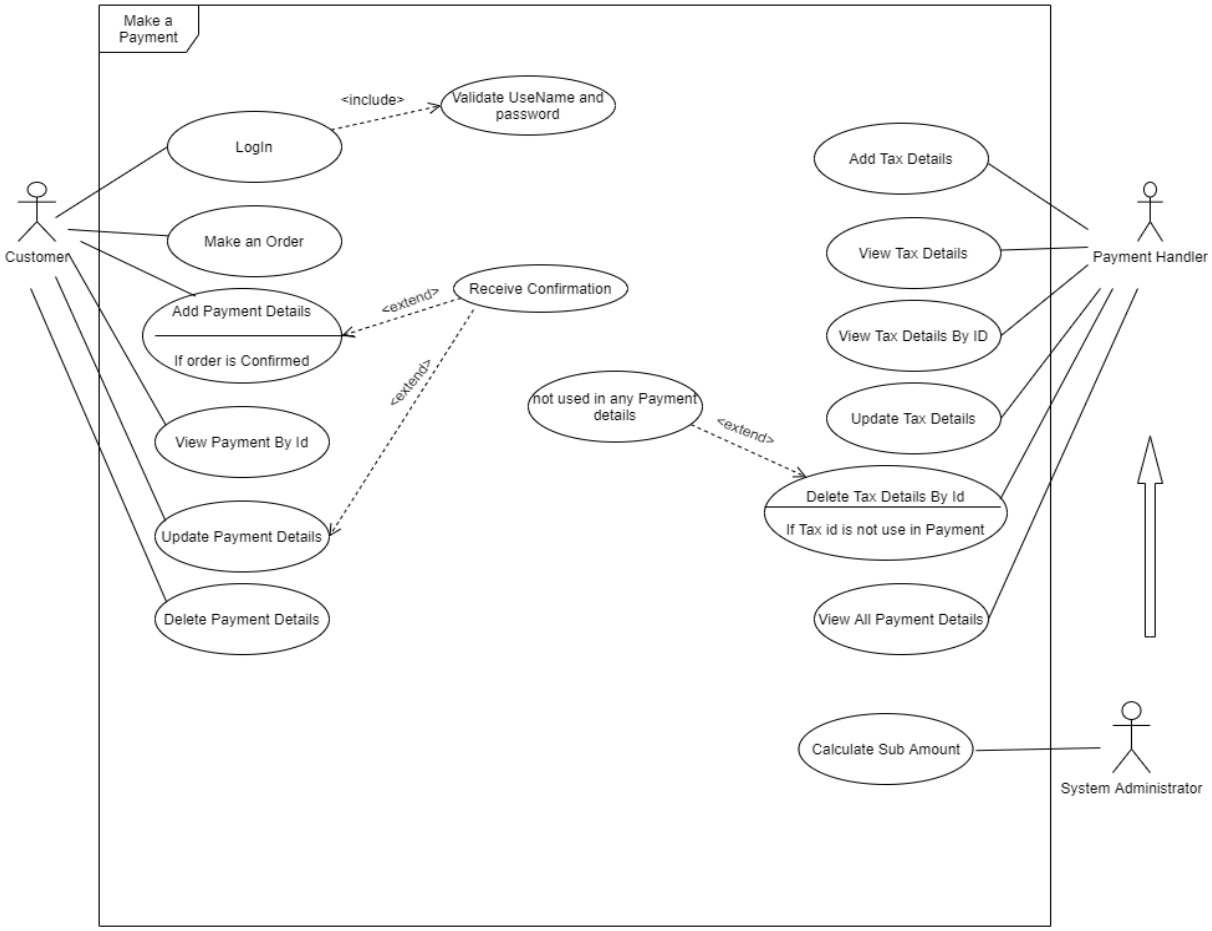
Activity Diagram

Activity Diagram





Use Case Diagram



Testing Methodology and Result

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 "Payment added successfully"	Saves into the database and get message "Payment added successfully"	pass
02)	Update payment details	Attributes to be updated along with the Payment ID	Display message as "Payment updated successfully"	Display message as "Payment updated successfully"	Pass
03)	View payment details by Customer id	URL for the API and Customer id	Display payment details of the Customer	Display payment details of the Customer	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
05)	Add Tax Entry	Attributes of Tax entry	Saves into the database and display message "Tax entry added successfully"	Saves into the database and display message "Tax entry added successfully"	Pass
06)	View all Tax entry	URL for the API Send Get request	Display Tax entry details	Display Tax entry details	Pass
07)	View all payment details	URL for the API Send Get request	Display payment details	Display payment details	Pass
08)	View all Tax entry by id	URL for the API Send Get request	Display Tax entry detail	Display Tax entry detail	Pass
09)	Update Tax entry by ID	Attributes to be updated along with the Tax ID	Display message as "Tax entry updated successfully"	Display message as "Tax entry updated successfully"	Pass
10)	Delete Tax Entry	Tax ID of the tax entry want to delete	Display message "Tax entry deleted successfully" if that tax entry was not used for a payment If used for a payment "Cannot delete tax entry already used "	Display message "Tax entry deleted successfully" if that tax entry was not used for a payment If used for a payment "Cannot delete tax entry already used "	Pass

Assumptions and Discussions

- Only admin can add Tax entry, update tax entry and Delete tax entry. But Tax entry can delete only if it isn't used for any payment.
- Tax amount varies with time period, system will calculate the tax amount for relevant time period
- Only admin can View all payment and Tax details.
- User can delete Payment details within 7 days after they confirmed their order

- Once user confirm the appointment it will calculate the charges using the details in the order table and tax table using a method written inside the payment class