Software Requirements Specification

for

< e Bug Tracker System >

Version 1.1 approved

<Faculty of Computer Science & Artificial intelligence, Helwan University>

<6 June 2021>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the e Bug Tracker System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system.

1.2 Document Conventions

This document follows MLA Format. Bold-faced text has been used to emphasize section and subsection headings. Italicized text is used to label and recognize diagrams

1.3 Intended Audience and Reading Suggestions

This document is to be read by the developer and the concerned staff. They might review the document to learn about the project and to understand the requirements.

Overall Description – Marketing staff must become accustomed to the various product features in order to effectively advertise the product.

System features – Testers need an understanding of the system features to develop meaningful test cases and give useful feedback to the developers.

External Interface Requirements – The hardware developers need to know the requirements of the device they need to build. The marketing staff also needs to understand the external interface requirements to sell the product.

Nonfunctional and Functional Requirements – The hardware developer

1.4 *Product Scope*

This system allows the customers to send bugs to the administrator Whenever a bug is raised from his software, see the bug case flow details and bug status along and receive solutions.

1.5 References

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

2. Overall Description

2.1 *Product Perspective*

The e Bug Tracker system should provide the facility of 'track' of bugs. This system is aimed towards customers whose software causes bugs. The proposed system is a solution to carry out sending, tracking, and solving bugs online.

2.2 Product Functions

- Access to the e Bug Tracker system for the administration, the customer, and staff are provided with an individual login.

- Administrators only can enter the details of staff, enter projects, view bugs sent from the customers, assign work to staff, view bug case flow status details, send messages to customers.

- the customer only can send the bug details to the administrator. Whenever a bug is raised from his software, see the bug case flow details and bug status along and receive solutions.

- staff only can view bugs assigned to them, give solution messages to customers, or can assign the bugs to other staff if the bug is related to them, and view bug case flow details.

The system enables customers to send bugs / receive the solution online.

2.3 User Classes and Characteristics

There are three types of users of the system.

1. Administrator

The following are the sub module in the administration module

- login
- Add Staff
- Add Project
- View customer's bugs
- Assign bug to staff
- View bug case flow
- Send solution messages

2. Customer/end user

The following are the sub module in the customer/end user module

- Register
- Login
- Send bugs
- View bug case flow
- View solution messages

3. Staff

The following are the sub module in the staff user module

- Login
- View assigned bugs
- Send solution messages
- Assign bug to staff
- Share bugs
- View bug case flow

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

For user documentation please refer to Section 3.

2.7 Assumptions and Dependencies

Following are the assumptions and dependencies which are related to this e Bug Tracker System:

- This project is a stand-alone project so it will not affect the system where it will be embedded.
- This project is a web-based project.
- This system will not depend on any other module. It will be web-based so everyone will independently contact it.
- It will not affect the environment at all.
- Roles and tasks are predefined.

3. External Interface Requirements

3.1 User Interfaces

Each part of the user interface intends to be as user-friendly as possible. The fonts and buttons used will be intended to be very fast and easy to load on web pages.

The pages will be kept light in space so that it won't take a long time for the page to load. The starting page will be the home page of the system.

All the operations in the user(customer/staff/admin) main page can be used if the user (customer/staff/admin) is logged in to his/her account.

There are pages that display the details accordingly.

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

4.0. System Features

4.1 *Add Employee*

Function	Admin can add the details of a new employee.
trigger	Select add employee from the admin main page.
Pre-condition	Admin has accessed the admin main page.
inputs	Name, username, password, category and email
Action	● Admin access the website ● Select login ● Entering username and password ● Enter submit ● Admin access the admin main page ● Select add employee ● Enter all data about employees

	● Enter submit ● The system automatically saves all data on the database ● The system send confirmation message
output	A message appears on the add employee page confirming the data has been successfully entered.
Post-condition	Admin should check if the confirmation message shown or not
Exception Paths	● The admin may abandon the operation at any time ● System didn't save the data on database

4.2 *Add project*

Function	Admin can add the details of a new project.
trigger	Select add project from the admin main page.
Pre-condition	Admin has accessed the admin main page.
inputs	Project name, details,category
Action	● Admin access the website ● Select login ● Entering username and password ● Enter submit ● Admin access the admin main page ● Select add project ● Enter all data about project ● Enter submit ● The system automatically saves all data on the database ● The system send confirmation message
output	A message appears on the add project page confirming the data has been successfully entered.
Post-condition	Admin should check if the confirmation message shown or not
Exception Paths	● The admin may abandon the operation at any time ● System didn't save the data on database

4.3 *View new bugs*

Function	Admin can view the new bug sended from the customer.

trigger	Select view bug from the admin main page.
Pre-condition	Admin has accessed the admin main page.
inputs	
Action	• Admin access the website • Select login • Entering username and password • Enter submit • Admin access the admin main page • Select view bug • The system brings data from the requested bug table on the database and displays it on a table on the view new bugs page
output	Table on view new bugs page consist of project name, error category, error details, print screen, due date
Post-condition	
Exception Paths	

4.4 Add bug

Function	Admin can add the details of a new bug.
trigger	Select add bug from the admin main page.
Pre-condition	Admin has accessed the admin main page.
inputs	ticket number, project name, error category, error details, print screen, due date
Action	• Admin access the website • Select login • Entering username and password • Enter submit • Admin access the admin main page • Select add bug • Enter all data about bug • Enter submit • The system automatically saves all data on bug table on the database • The system send confirmation message

output	A message appears on the add bug page confirming the data has been successfully entered.
Post-condition	Admin should check if the confirmation message shown or not
Exception Paths	● The admin may abandon the operation at any time ● System didn't save the data on database

4.5 *Assign Bug*

Function	● Admin can assign bugs to staff ● Staff can assign bugs to other staff
trigger	select assign bugs from admin/staff main page
Pre-condition	Admin/staff has accessed the admin/staff main page
Inputs	Ticket number, category name
Action	● Admin/staff access the website ● Select login ● Entering username and password ● Admin/staff access the admin /staff main page ● Select assign bug ● Enter ticket number ● Select Send ● The system record bug data on assigned bug table on database ● The system send confirmation message
output	A message appears on the assigned bug page confirming the data has been successfully entered.
Post-condition	Admin should check if the confirmation message shown or not
Exception Paths	● The admin may abandon the operation at any time ● System didn't save the data on database

4.6 *View Bug flow*

Function	Admin/staff/Customer can view bug flow

trigger	Select view bug flow from the admin/Staff/Customer main page.
Pre-condition	Admin/staff/Customer has accessed the admin/staff/customer main page
inputs	
Action	● Admin/staff/customer access the website ● Select login ● Entering username and password ● Enter submit ● Admin/staff/customer access the admin/staff/customer main page ● Select view bug flow ● The system brings data from assigned bug table, bug table, bug solution on the database and displays it on a table on the view new bugs flow page
output	**Table contains bug flow on view new bugs flow page for each user :** ● Admin: project name, error category, error details, print screen of the bug, due date to reply for the bug, which staffs are involved in solving the bug, and the status ● Staff:project name, error category, error details, print screen of the bug, due date to reply for the bug ● Customer:project name, error category, error details, print screen of the bug, due date to reply for the bug, status, and solution
Post-condition	
Exception Paths	

4.7 *Send solution message*

Function	Admin/staff can send solution message
trigger	select send message from admin/staff main page
Pre-condition	● Admin/staff has accessed the admin/staff main page ● Check if message sent before or not
Inputs	ticket number, solution
Action	● Admin/staff access the website ● Select login

	● Entering username and password ● Enter submit ● Admin/staff access the admin /staff main page ● Select send solution message ● Enter ticket number ● System send customer's email ● Select work gmail account ● Send message with solution
output	
Post-condition	Admin/staff should check if the message sent or not
Exception Paths	● The admin may abandon the operation at any time ● System didn't save the data on database

4.8 View assigned bug

Function	staff can view assigned bugs sended from the admin.
trigger	Select view assigned bugs from the staff main page.
Pre-condition	Staff has accessed the staff main page.
inputs	
Action	● Staff access the website ● Select login ● Entering username and password ● Staff access the staff main page ● Select view assigned bug ● The system brings data from assigned bug table on the database and displays it on a table on the view assigned bug page.
output	Table contains ticket numbers on view assigned bug page
Post-condition	
Exception Paths	

4.9 Send bug

Function	Customer can send bug

trigger	select send bug from customer main page
Pre-condition	Customer has accessed the customer main page
Inputs	project name, error category, error details, print screen of the bug, due date to reply for the bug
Action	• Customer access the website • Select login • Entering username and password • Enter submit • Customer access the Customer main page • Select send bug • Enter data of bug • System send data to requested bug table on database • System send confirmation message
output	A message appears on the send bug page confirming the data has been successfully sent.
Post-condition	customer should check if the message sent or not
Exception Paths	• The customer may abandon the operation at any time • System didn't save the data on database

4.10 Send message with ticket

Function	System send message with ticket
trigger	When a customer sends a bug, the system will generate the ticket and send a message with the ticket.
Pre-condition	Customer send a bug
Inputs	ticket number
Action	**when a customer send a bug:** • System generate a ticket number • System send customer's email • System work gmail account • System send a message with ticket number
output	
Post-condition	

Exception Paths	• System didn't save the data on database

4.11 Register

Function	New customer can register on the system
trigger	Select add register from the main page.
Pre-condition	• New customer has accessed the main page • Check if customer is related to project made by staffs or not
inputs	Name, username, password, email, project
Action	• New customer access the website • Select register • Entering all data about him • Enter submit • System check if a customer is related to a project made by staff or not from customer table on database • The system automatically saves all data on customer table on the database • The system send confirmation message
output	A message appears on the register page confirming the data has been successfully entered.
Post-condition	Customer check if he can login or not
Exception Paths	• The admin may abandon the operation at any time • System didn't save the data on database

4.12 Login

Function	Admin/Staff/Customer can log on the system
trigger	Select login from the main page.
Pre-condition	Admin/Staff/Customer has accessed the admin/staff/customer main page
inputs	Username, Password
Action	• Admin/Staff/Customer access the website • Select login • Entering username,password • Enter login

	• System check if username and password exist on customer/staff/admin tables on database or not • Admin/staff/customer can access admin/staff/customer main page
output	admin/staff/customer main page
Post-condition	Check if admin/staff/customer can access admin/staff/customer main page
Exception Paths	• The admin/staff/customer may abandon the operation at any time • System didn't manipulate with the data on database

4.13 logout

Function	Admin/Staff/Customer can logout the system
trigger	Select logout from the admin/staff/customer main page.
Pre-condition	Admin/Staff/Customer has accessed the admin/staff/customer main page
inputs	
Action	• Admin/Staff/Customer access the website • Select login • Carry out operations • Select logout
output	Exit from the system
Post-condition	
Exception Paths	• System didn't exit from website

4. Other Nonfunctional Requirements

4.1 Performance Requirements

Maximum possible quick response to the orders is required, also should provide fast updating of records. The changes if any made should be reflected automatically in the next screens. In order to

maintain an acceptable speed at a maximum number of uploads allowed from a particular customer as any number of users can access the system at any time. Also, the connections to the servers will be based on the attributes of the user like his location and server will be.

4.2 Safety Requirements: to prevent hacking

4.3 Security Requirements: to prevent unauthorized access.

4.4 Software Quality Attributes

The necessary qualities of software products are

1. Security:
   The application is password protected and also any process is done by only privileged users.

2. Maintainability:
   The application is to be designed so that it is easily maintained. Also, it should allow incorporating new requirements in any module of the system.

3. Reliability:
   The application will be able to handle two orders. When a user confirms his/her order the database will be updated immediately, and the next user will not face problems in order.

4. Availability Requirement
   The system is available 100% for the user and is used 24 hours a day and 365 days a year.The system shall be operational 24 hours a day and 7 days a week.

5. Efficiency Requirement
   Mean Time to Repair (MTTR) - Even if the system fails, the system will be recovered back up within an hour or less.

4.5 Business Rules

- only admin can add staff, add projects, assign bugs to staff.
- only the customer can send bugs, view his bug flow, view solutions of bugs.
- only staff can solve bugs, assign bugs to another staff, share bugs with the upper department.

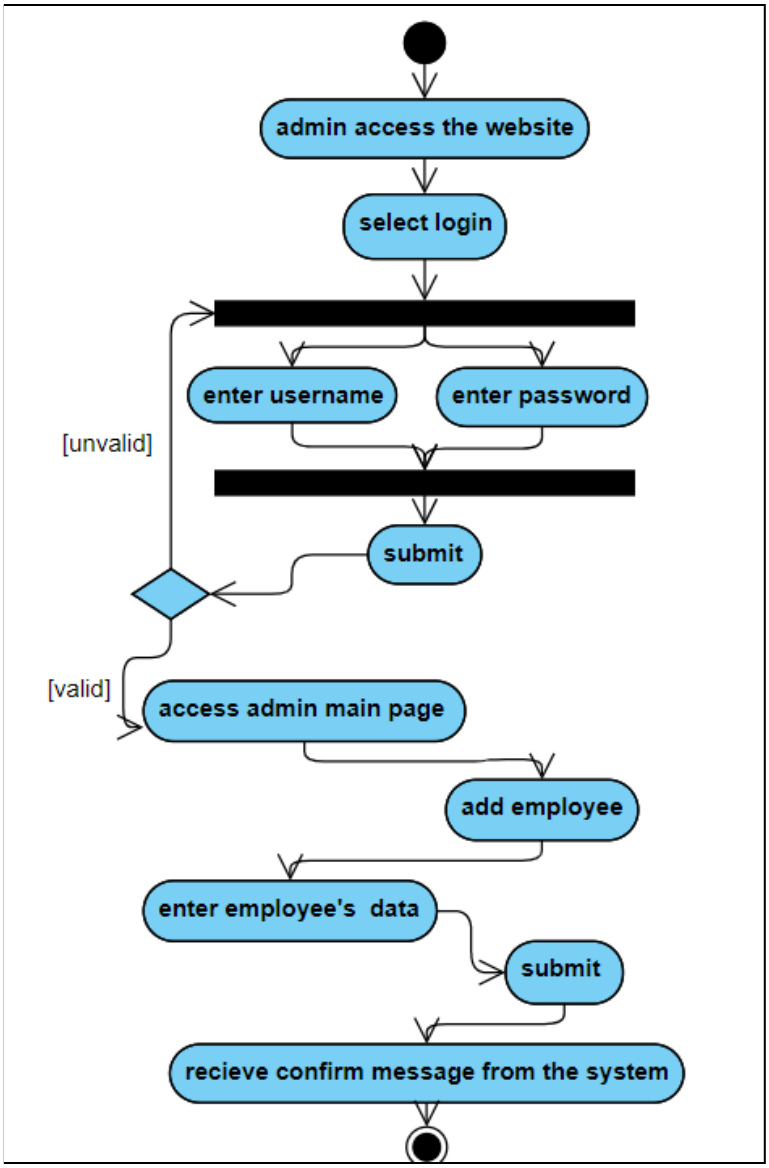
5. Other Requirement

Appendix A: Glossary

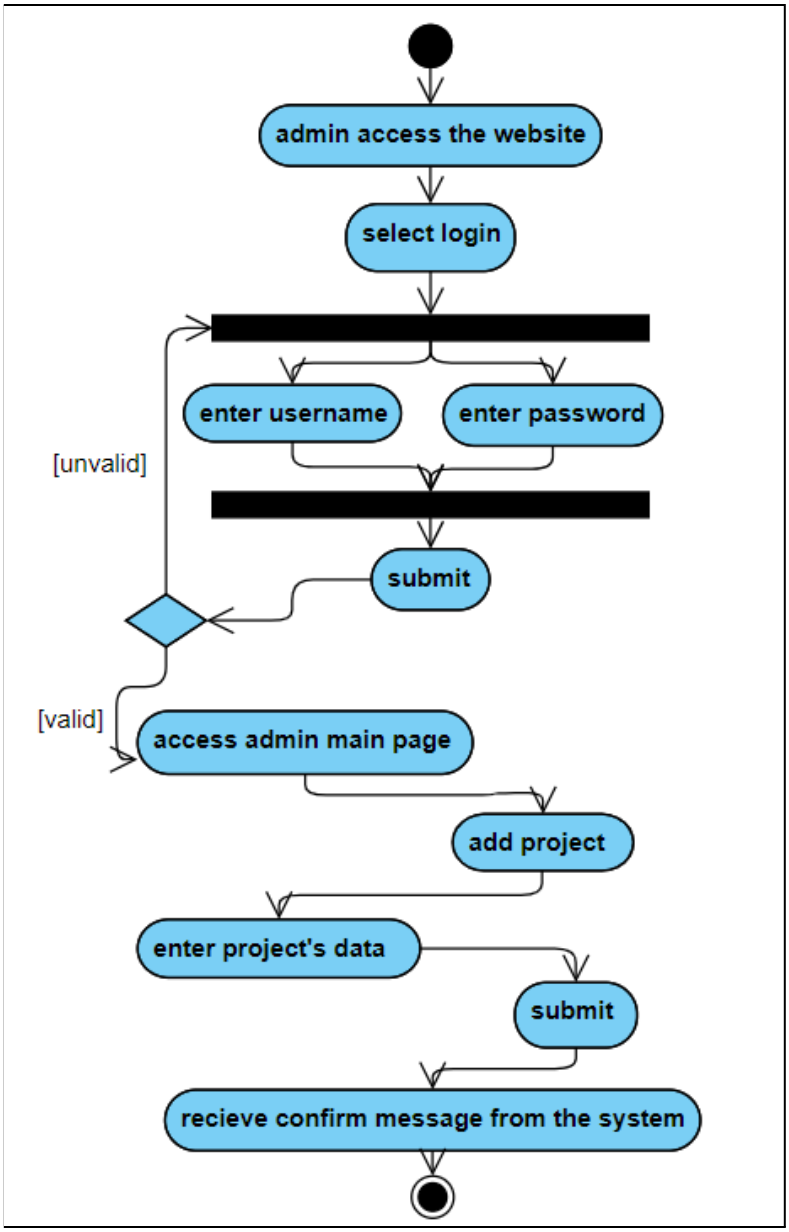
Term	Definition
Database	Collection of all the information monitored by this system.
Software Requirements Specification	A document that completely describes all the functions of a proposed system and the constraints under which it must operate. For example, this document
Stakeholder	Any person with an interest in the project who is not a developer
Administrator	person responsible for carrying out the administration of a business or organization.
Customer	A member listed in the database and purchased a system and the system resulted in an error.
Bug	A mistake, misconception, or misunderstanding on the part of a software developer.
staff	all the people employed by a particular organization.
User	Admin or Staff or Customer

Appendix B: Analysis Models

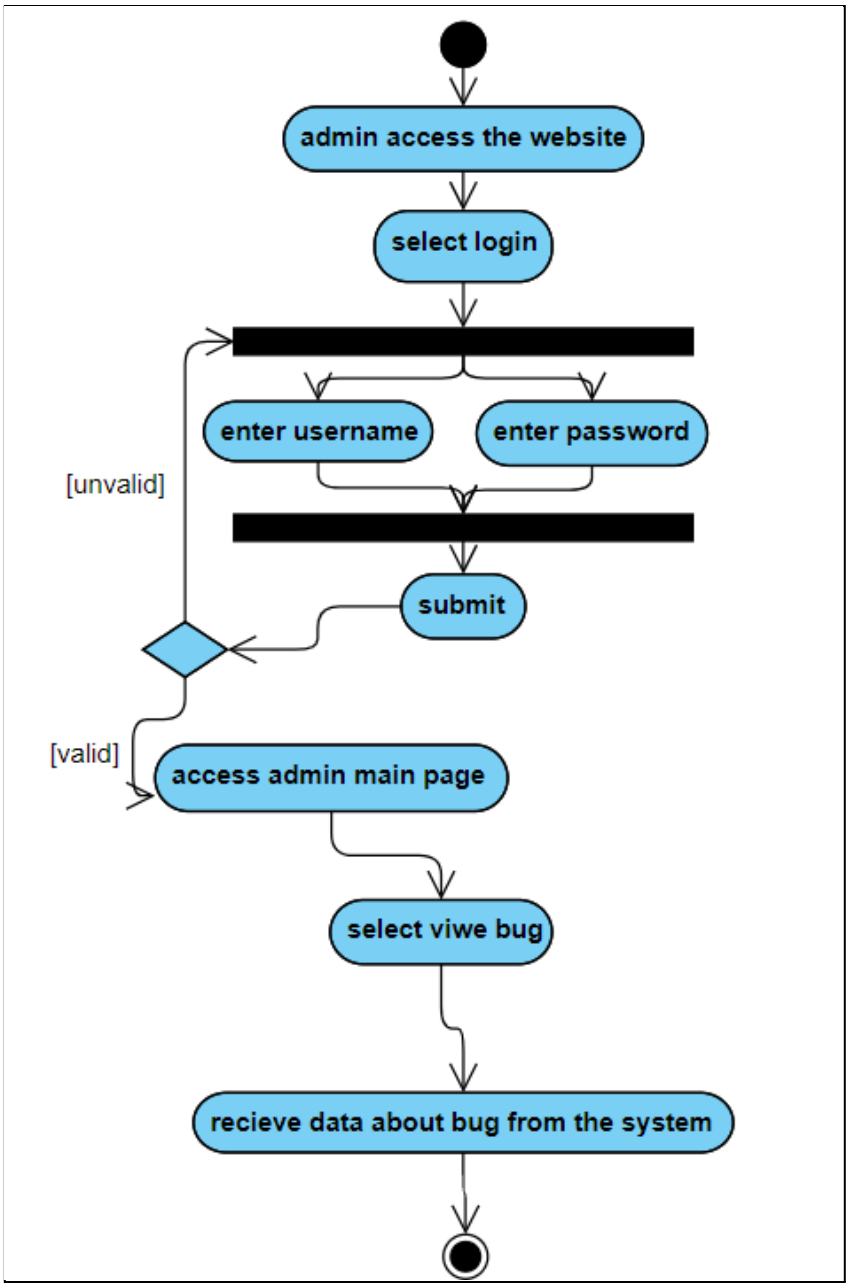
B.1 Activity Diagrams:



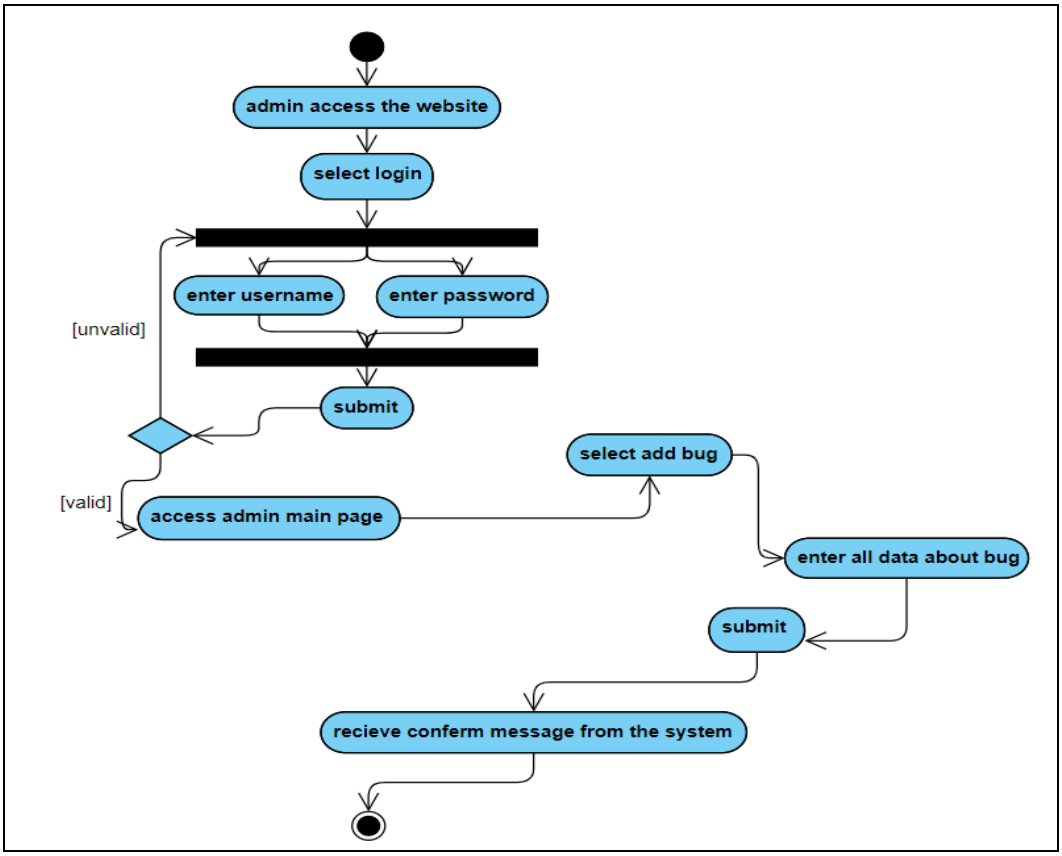
add employee


admin access the website
select login
enter username
enter password
[unvalid]
submit
[valid]
access admin main page
add project
enter project's data
submit
recieve confirm message from the system

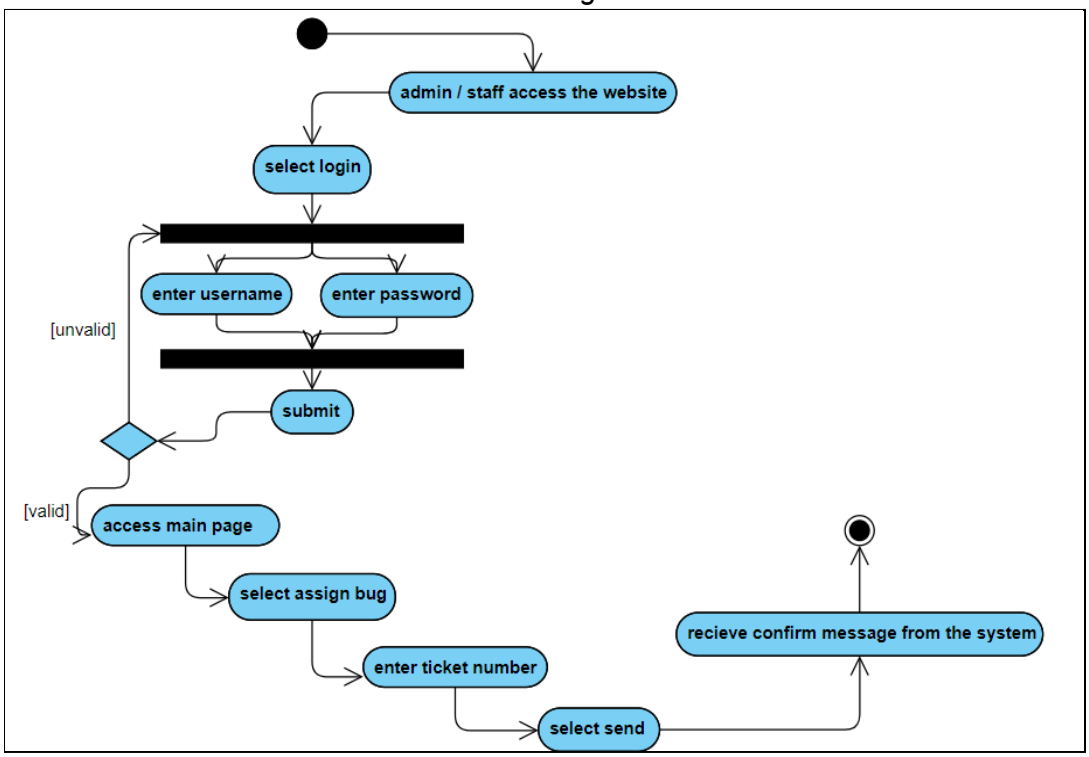
add project


admin access the website
select login
enter username
enter password
[unvalid]
submit
[valid]
access admin main page
select viwe bug
recieve data about bug from the system

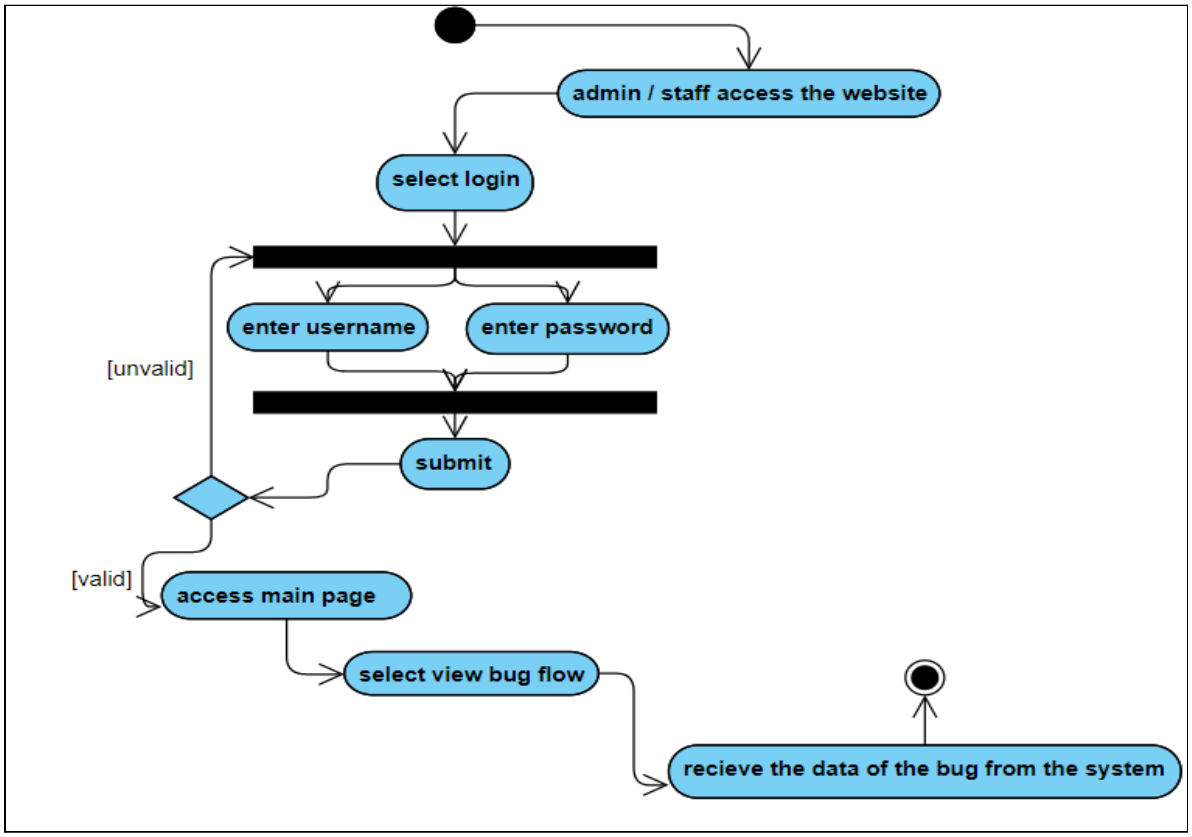
view new bug



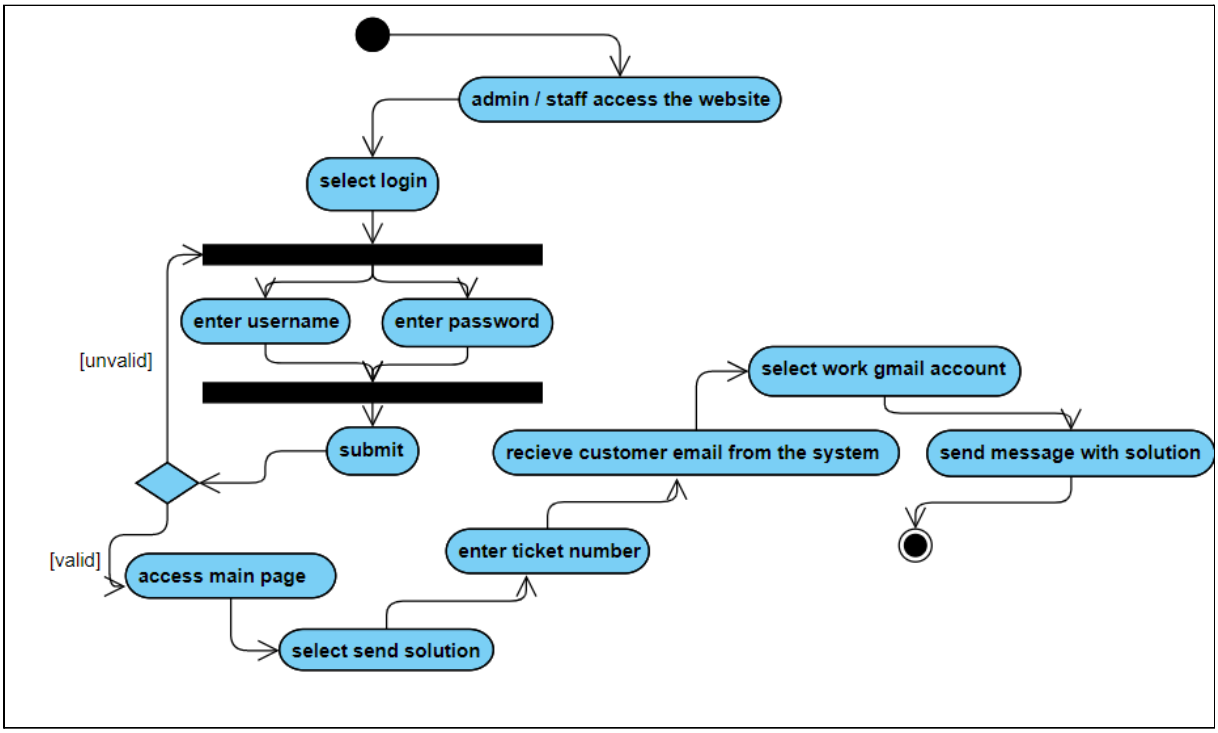
add bug



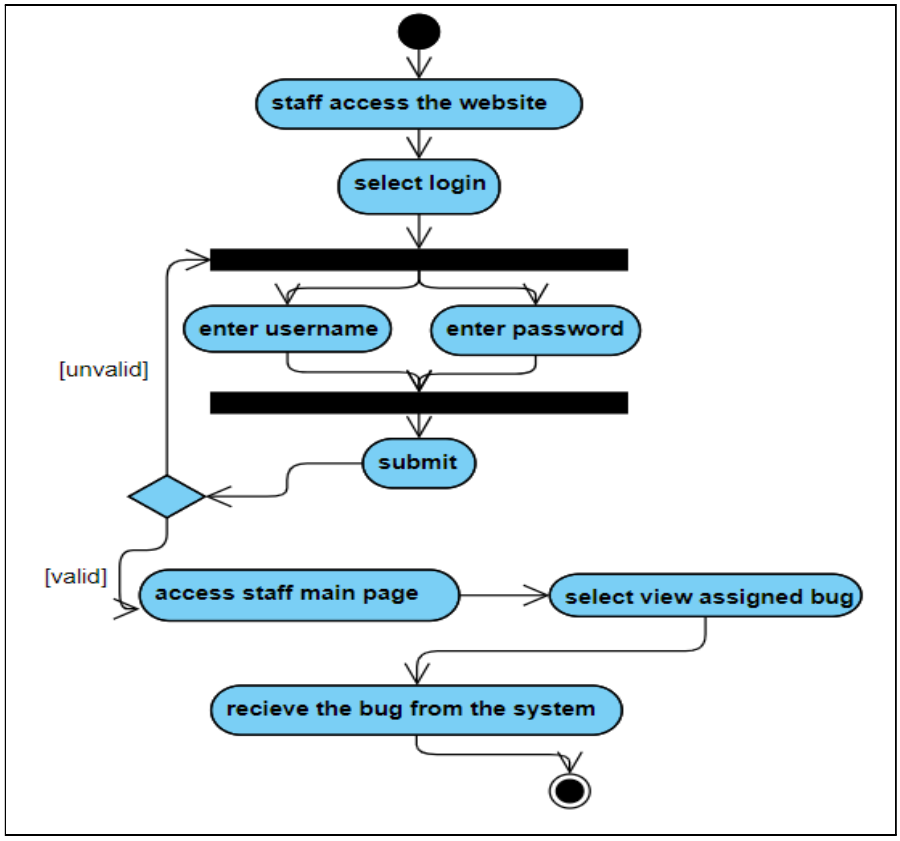
assign bug to staff



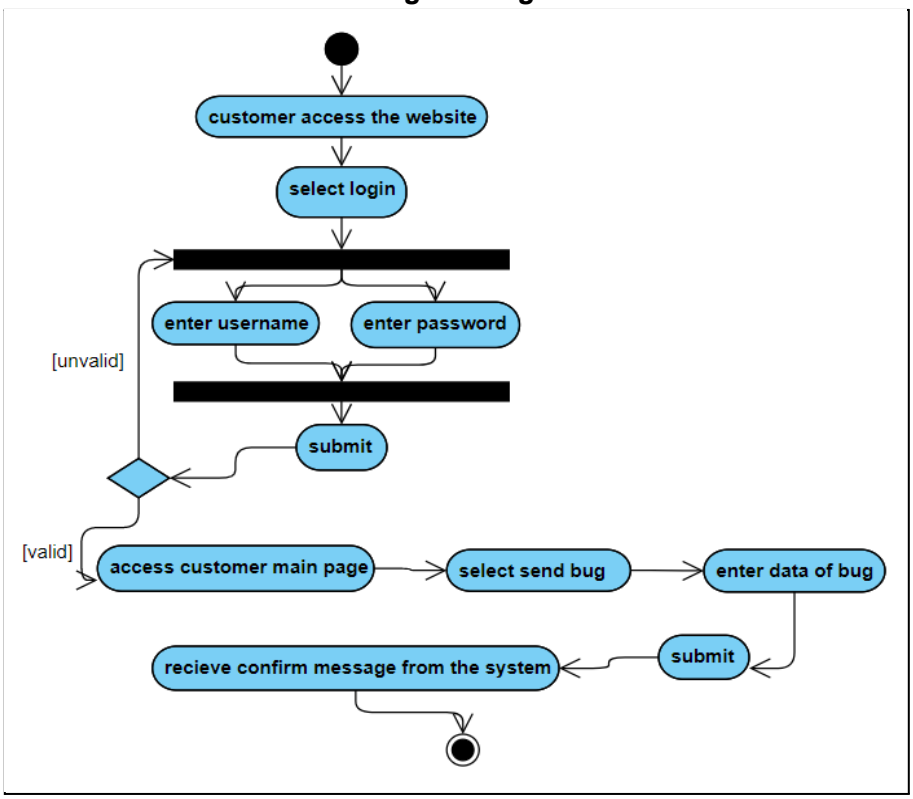
view bug flow



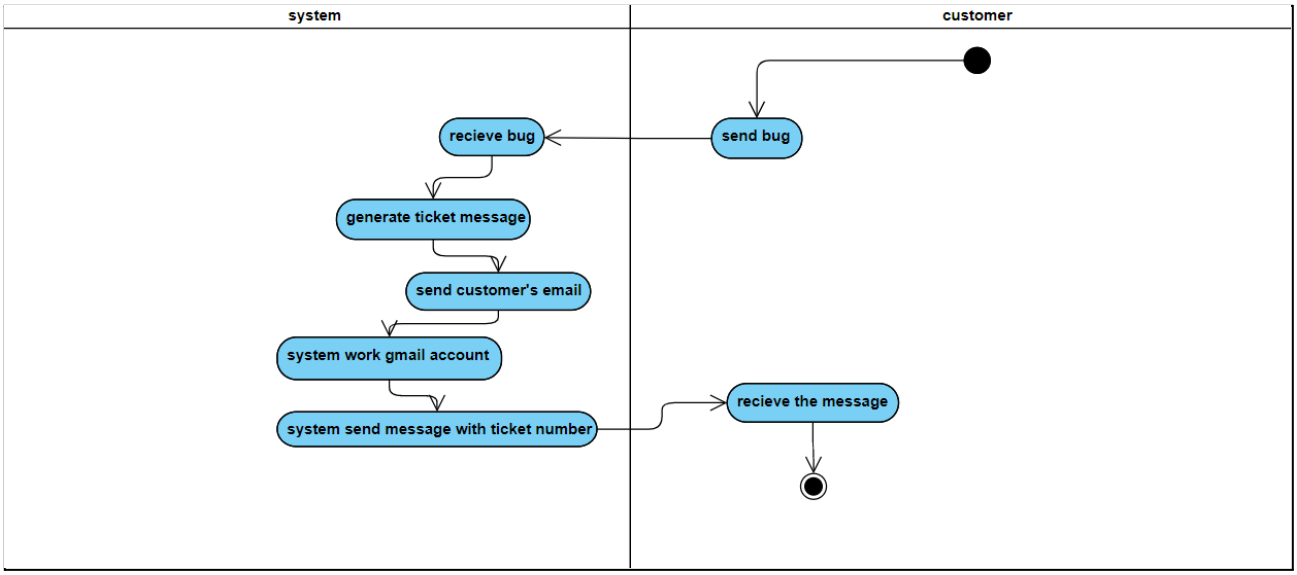
send solution message



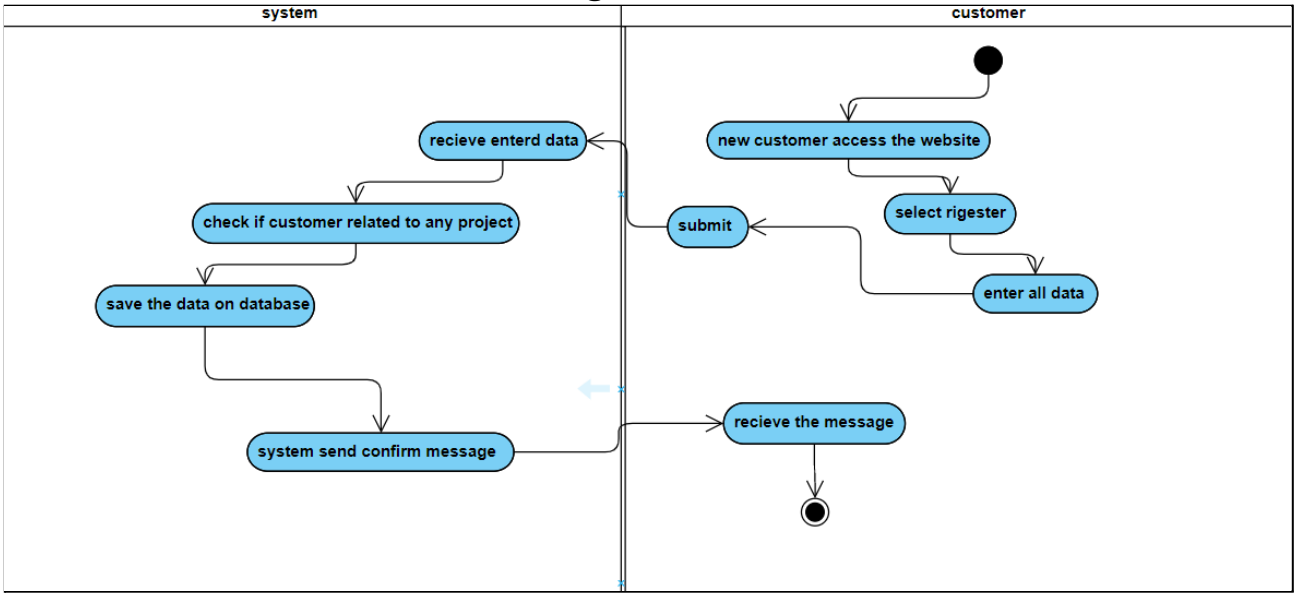
view assigned bug



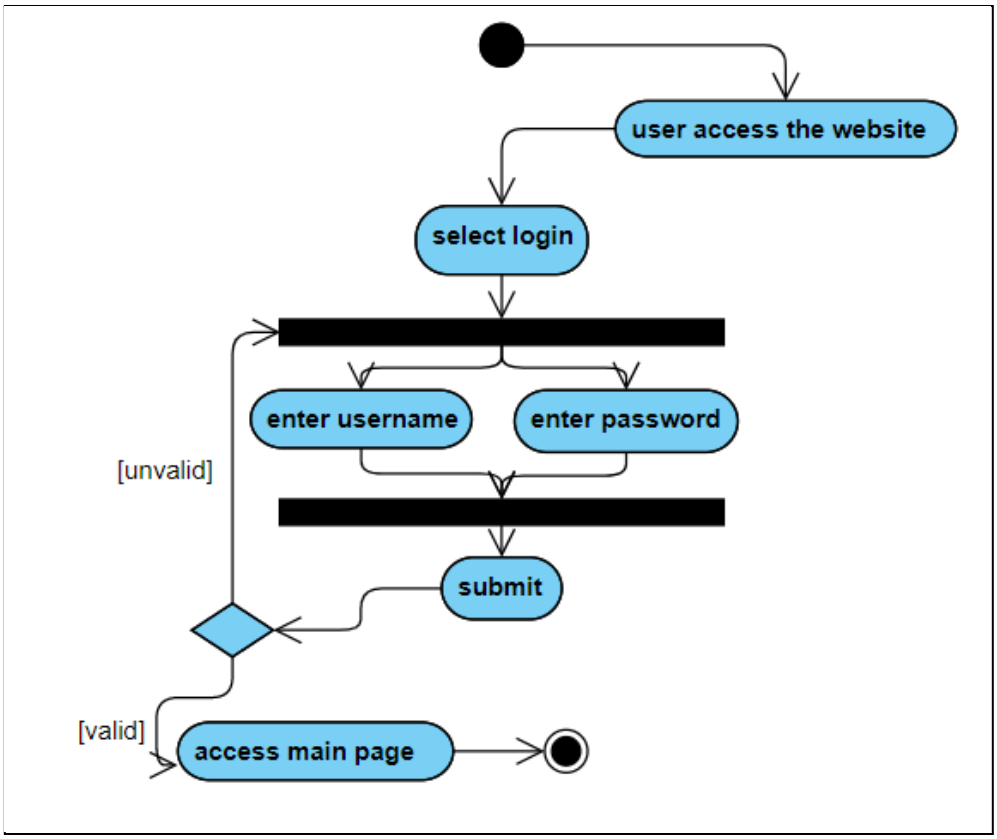
send bug



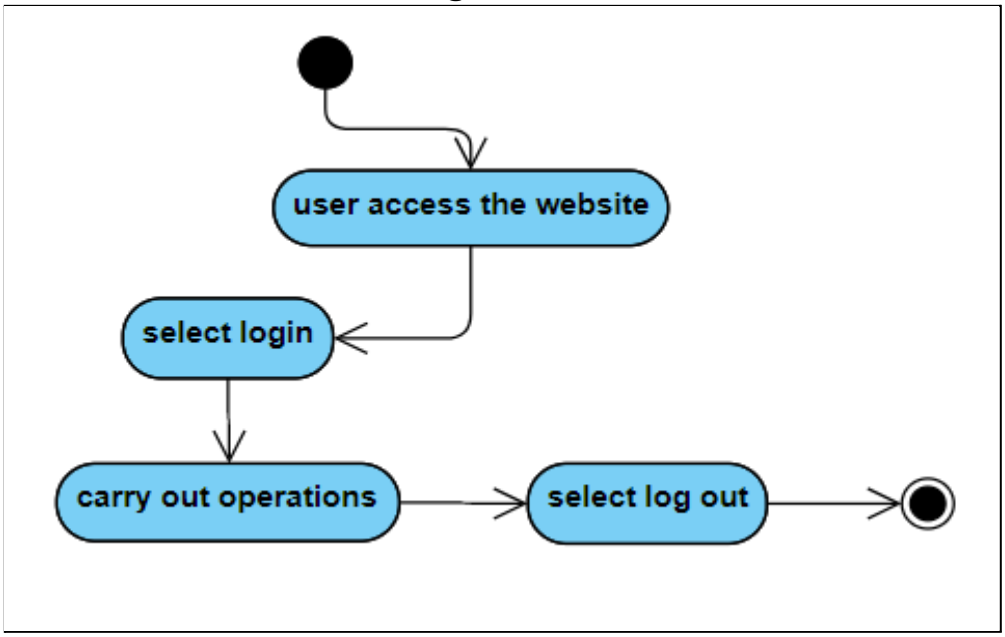
send message with ticket



register



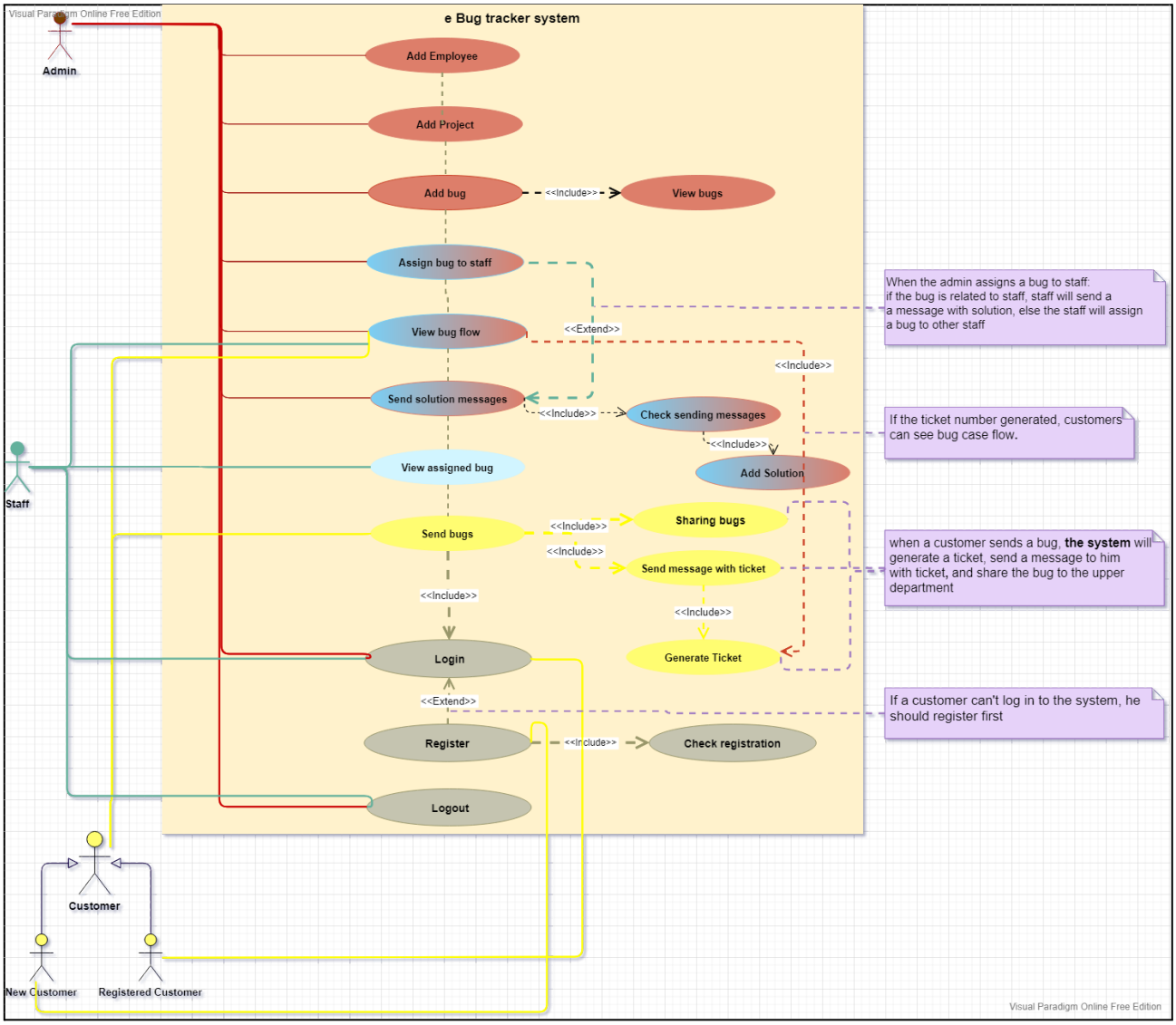
login



log out

B.2 Use Case Diagram:

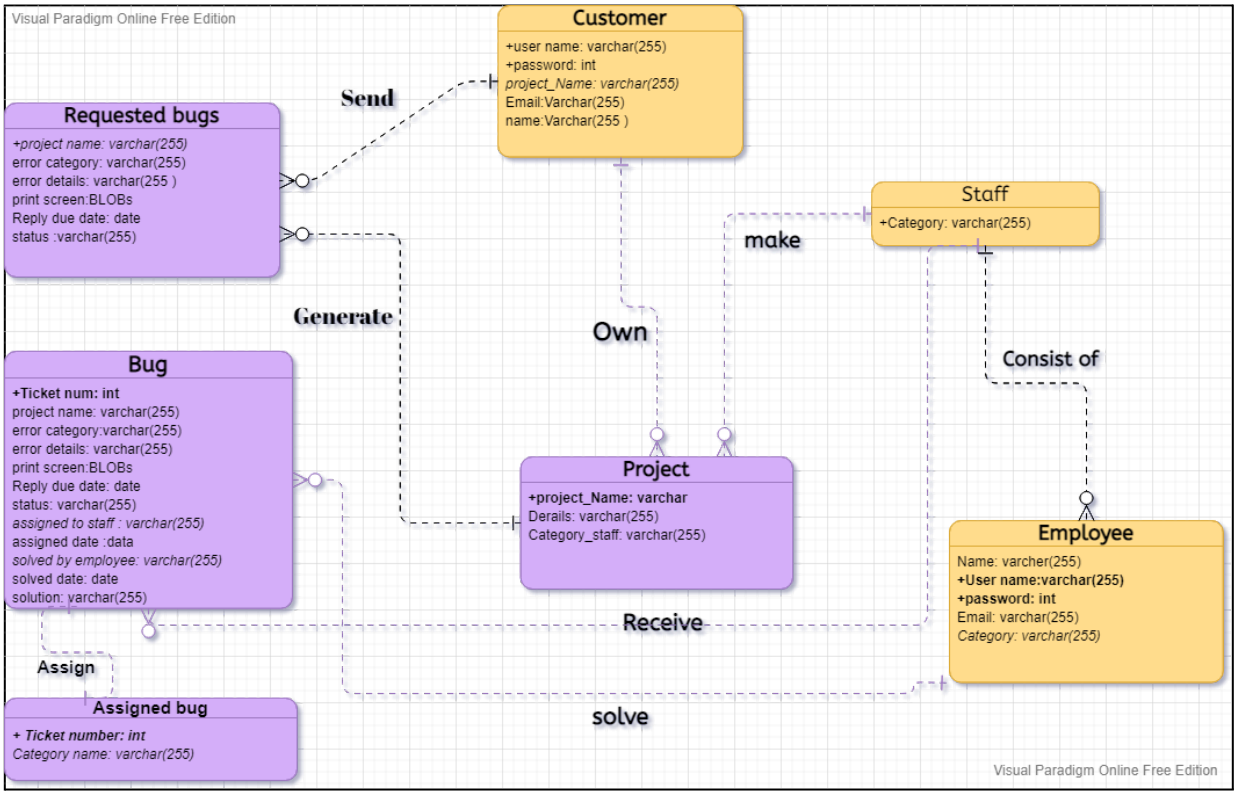
B.2.1 Use Case Diagram:



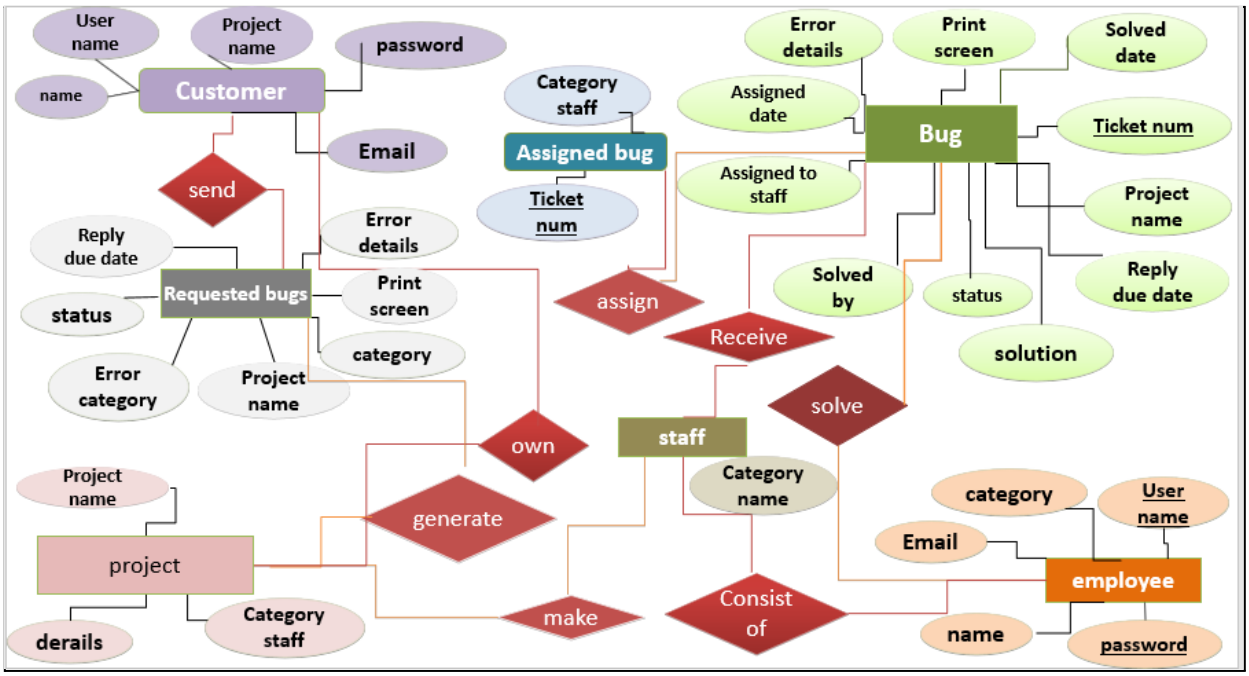
B.2.2 Use Case Description

Identifier	UC1 Bug tracking System.
Initiator	Customer / Admin / Staff.
Goal	Bugs can be solved easily.
Pre-condition	Register to the application, Log in using username and password.
Post-condition	Customers can send the detected bugs, Message sending incudes the bug solution.
Main success scenario	1) Customer must register to the application and log in using username and password. 2) Customer sends the bug details to the administrator. 3) Admin view the bug. 4) Admin can send bug solutions directly to the customer or assign the bug to the related staff. 5) Staff may view the bugs assigned to them, he can directly give a solution message to the customer or assign the bug to other staff if the bug is related to them. 6) Message send to the customer include the bug solution. 7) Admin can view case flow details include bug details, which staff are involved in solving the bug, and the status. 8) Staff may view bug case flow details with which he is involved. 9) Customer may see bug case flow details and bug status along with remedy by using a ticket number. **Extensions:** 1) Invalid username or password. 2) Customer enters incorrect details about the bug. 3) Customer submit incorrect ticket number.

B.3 Entity Relationship Diagram

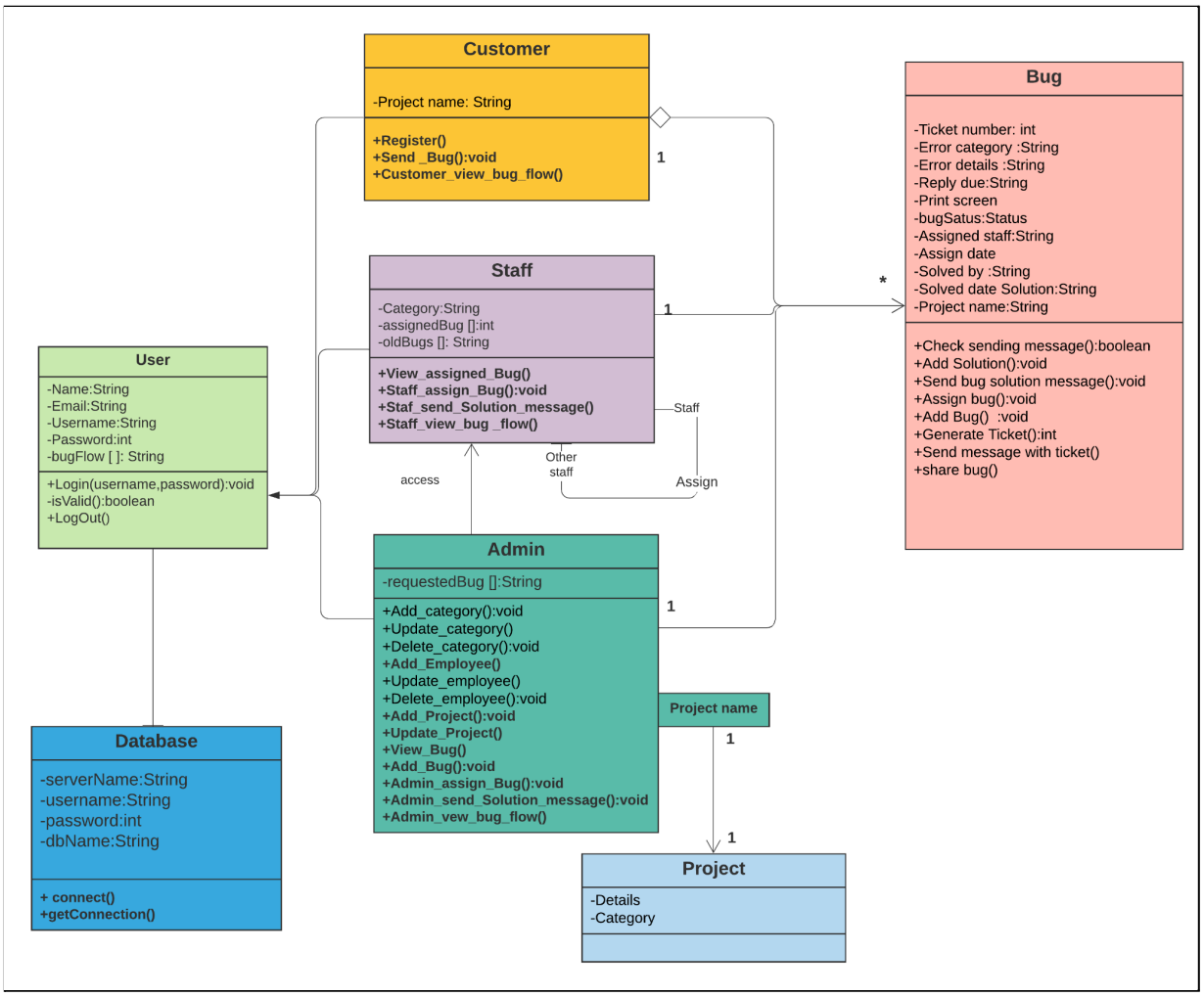


The bold-faced text has been used to emphasize primary keys.
Italicized text is used to emphasize foreign key

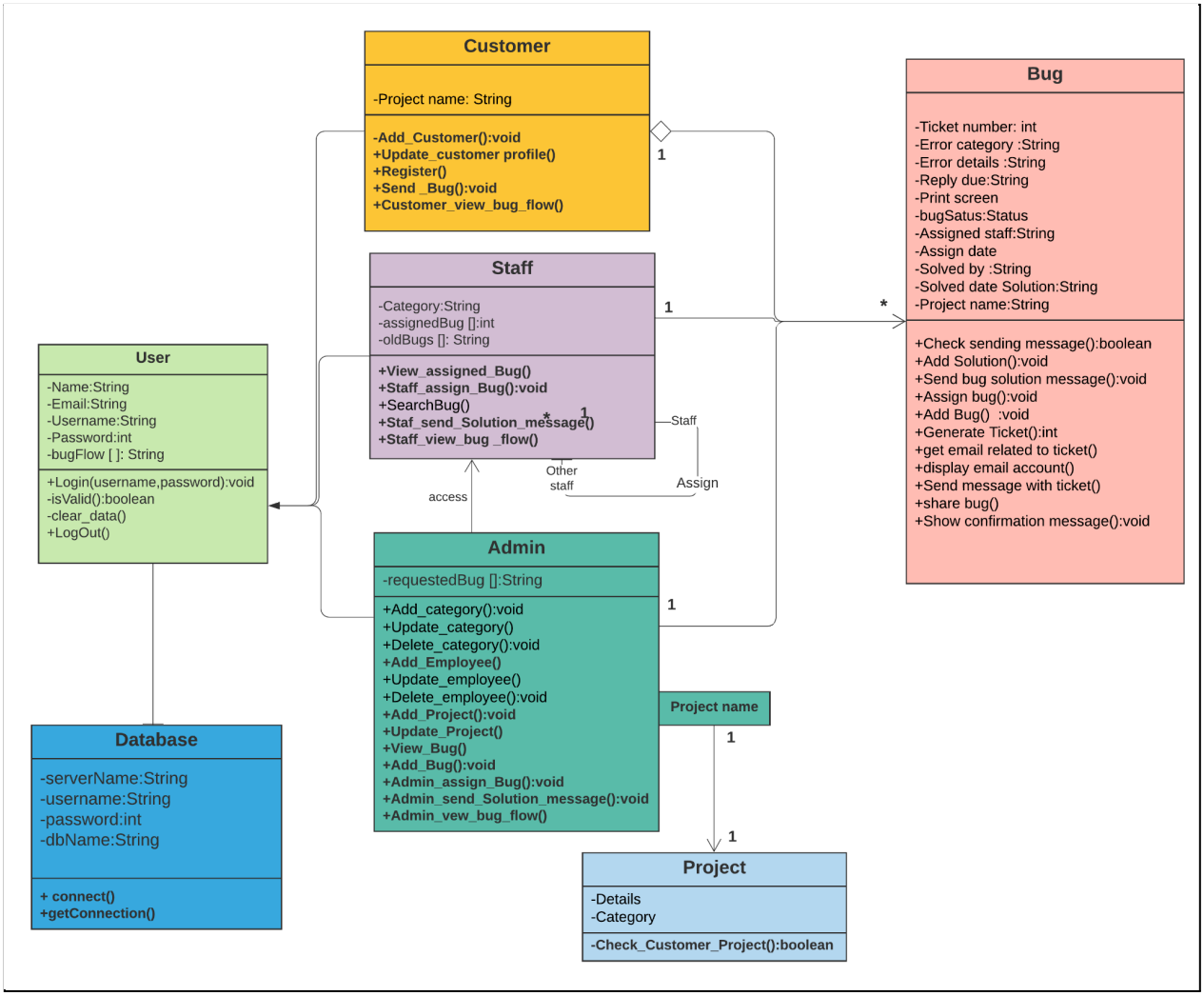


B.4 Class Diagram

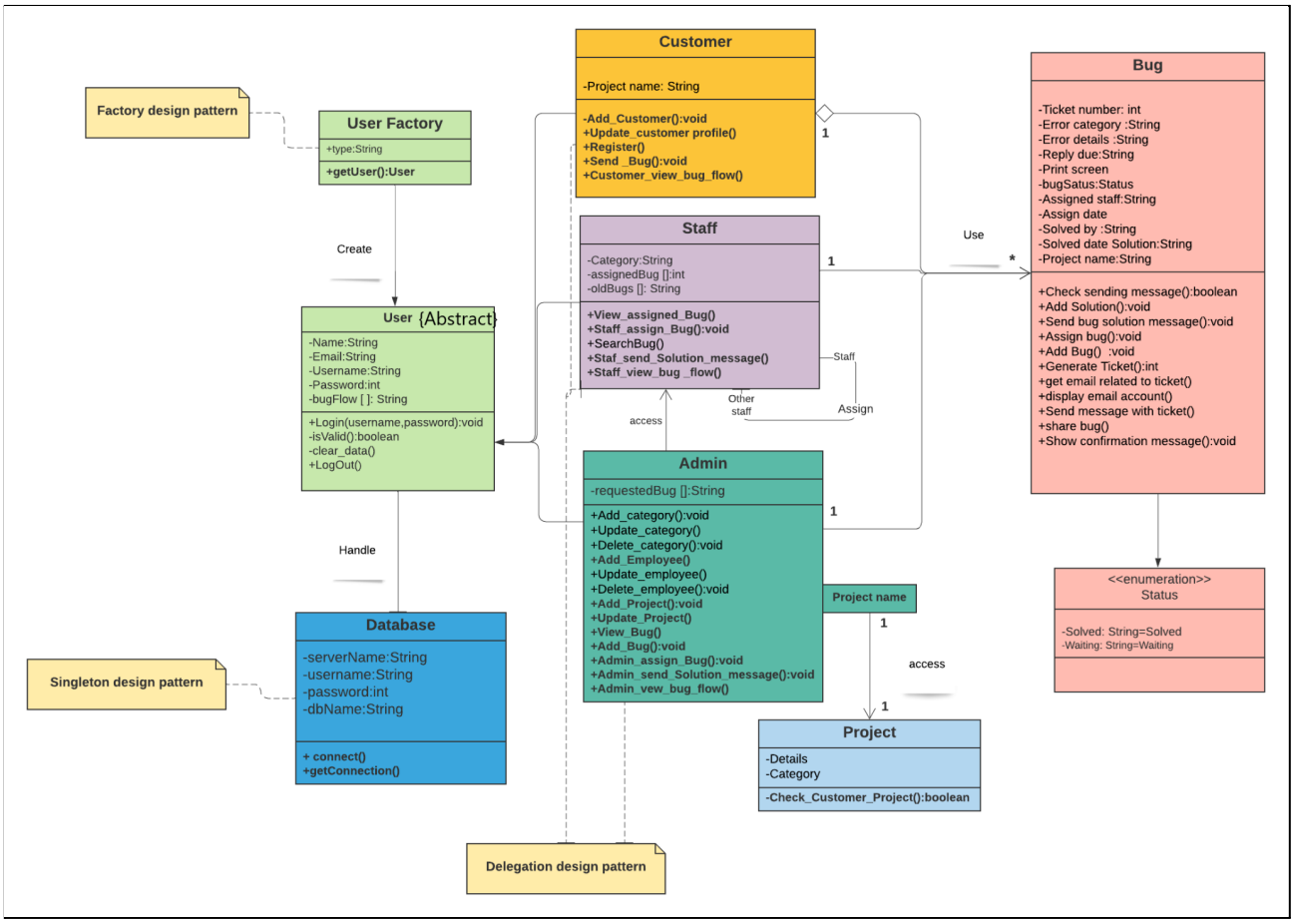
B.4.1 Version 1



B.4.2 Version 2

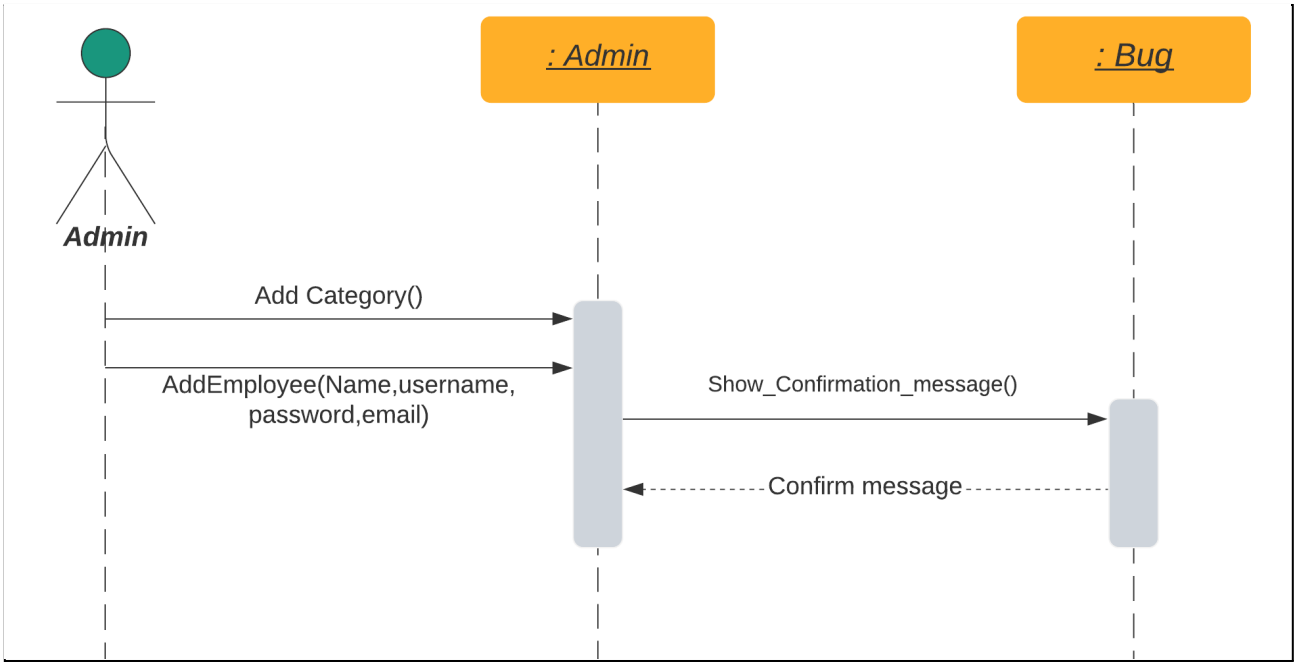


B.4.3 Version 3

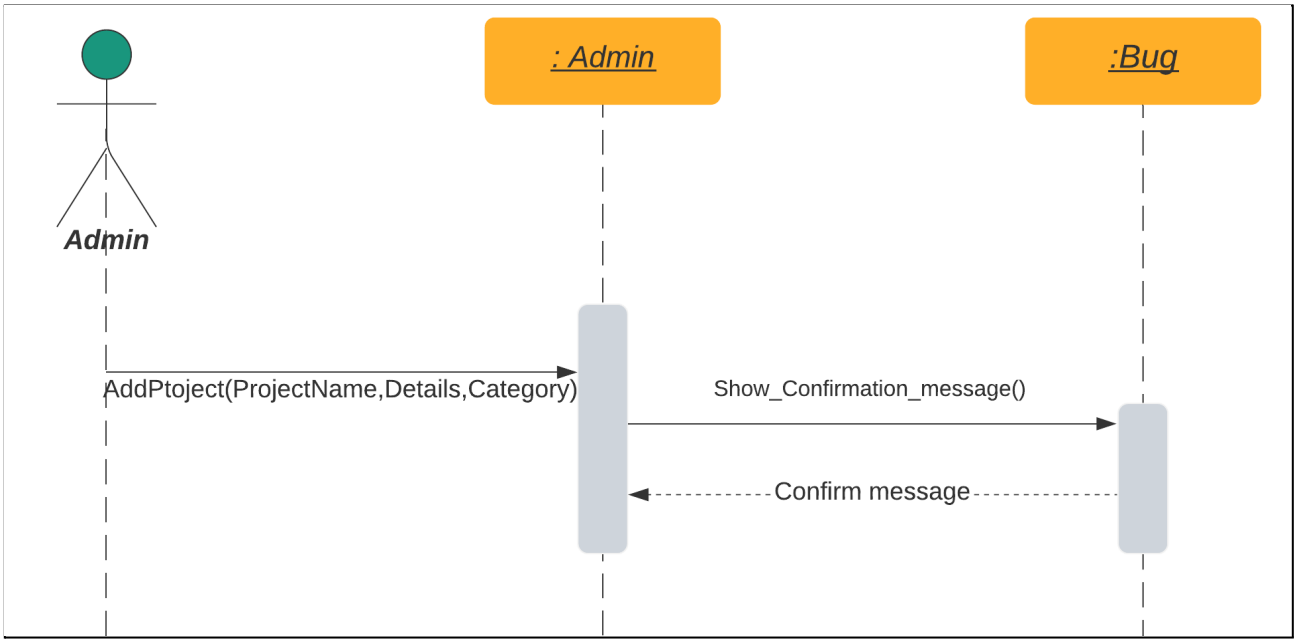


B.5 Sequence Diagram

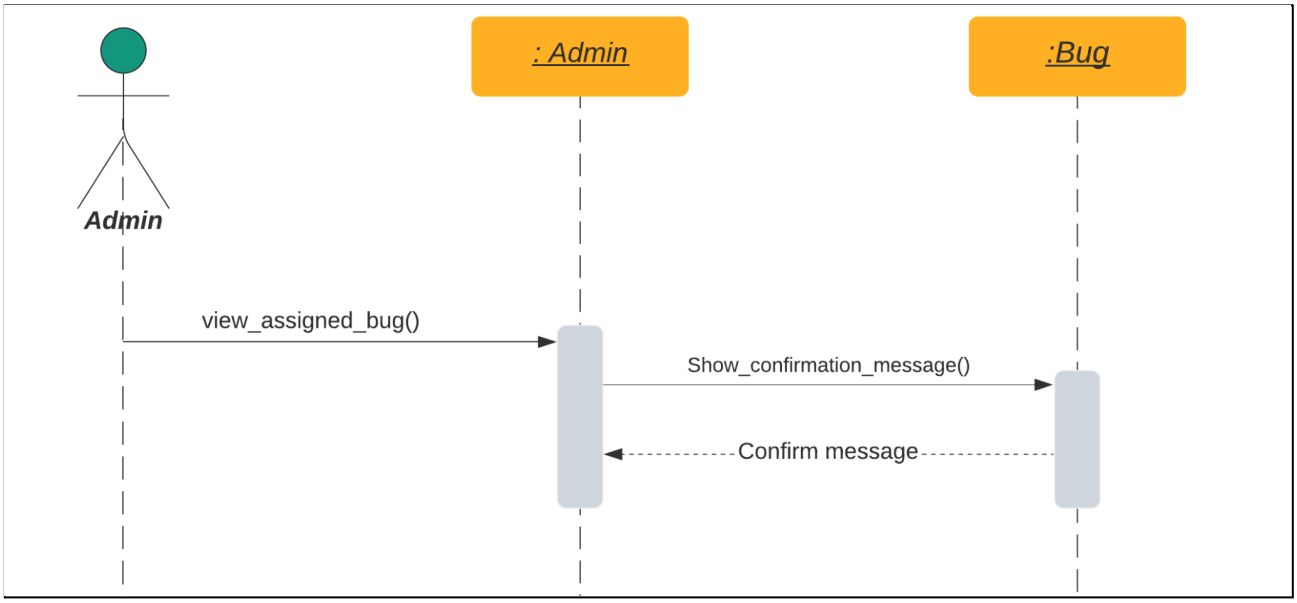
B.5.1.1 Add Employee



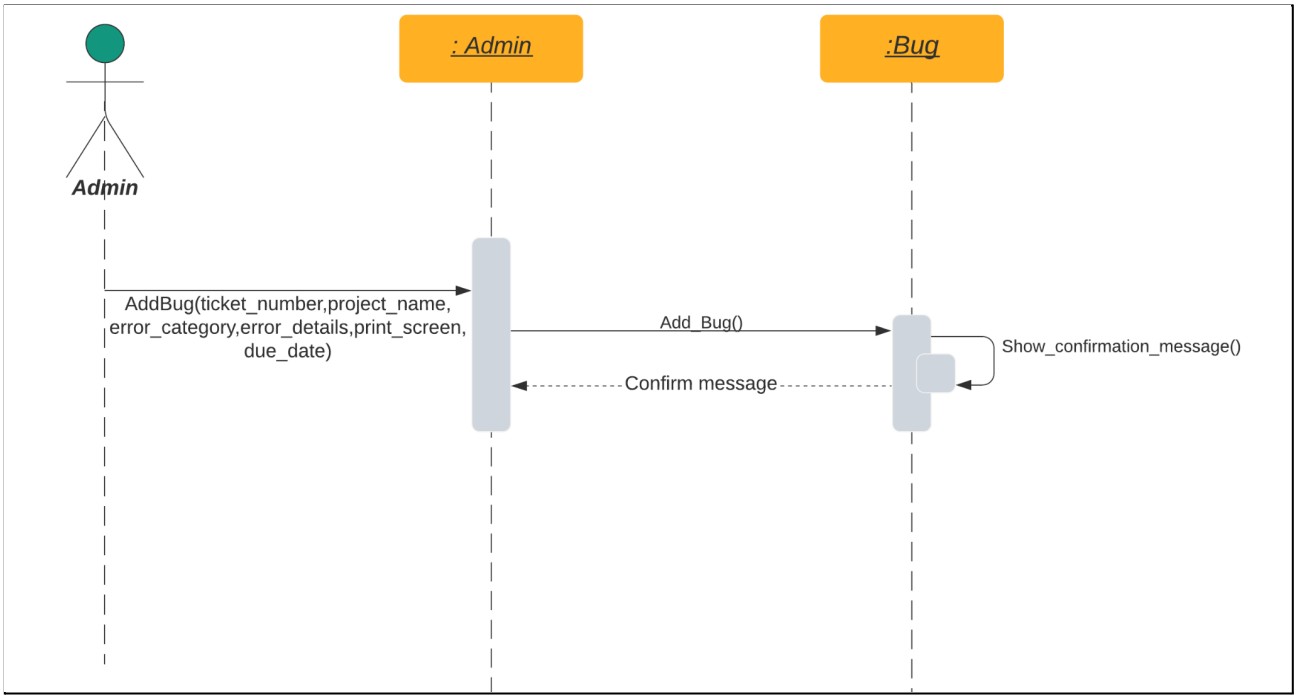
B.5.1.2 Add Project



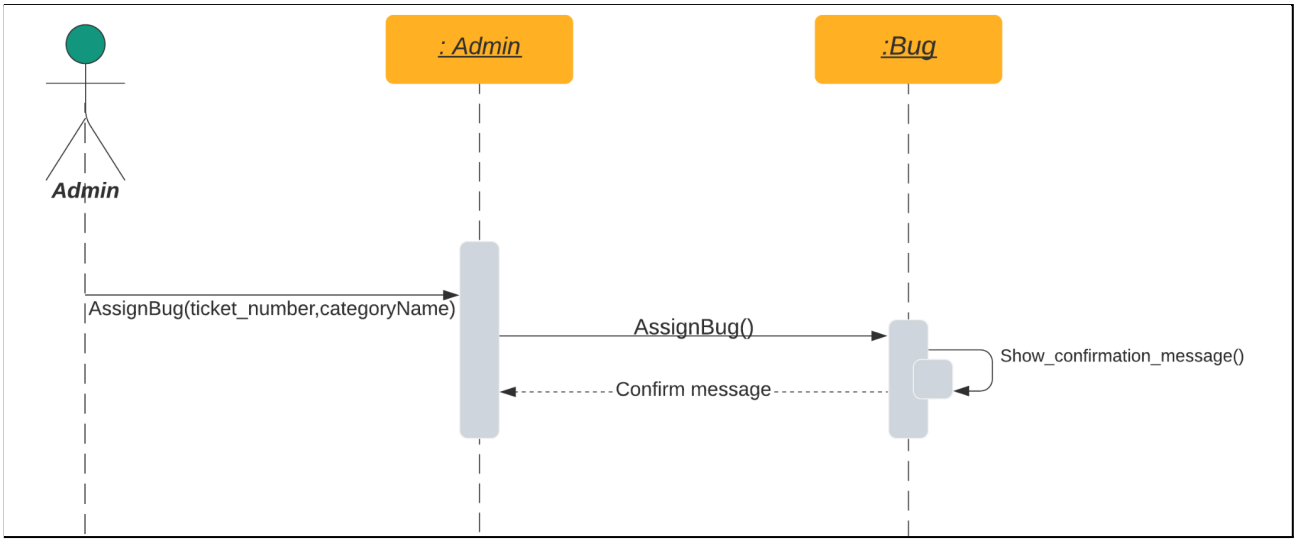
B.5.1.3 View new bugs



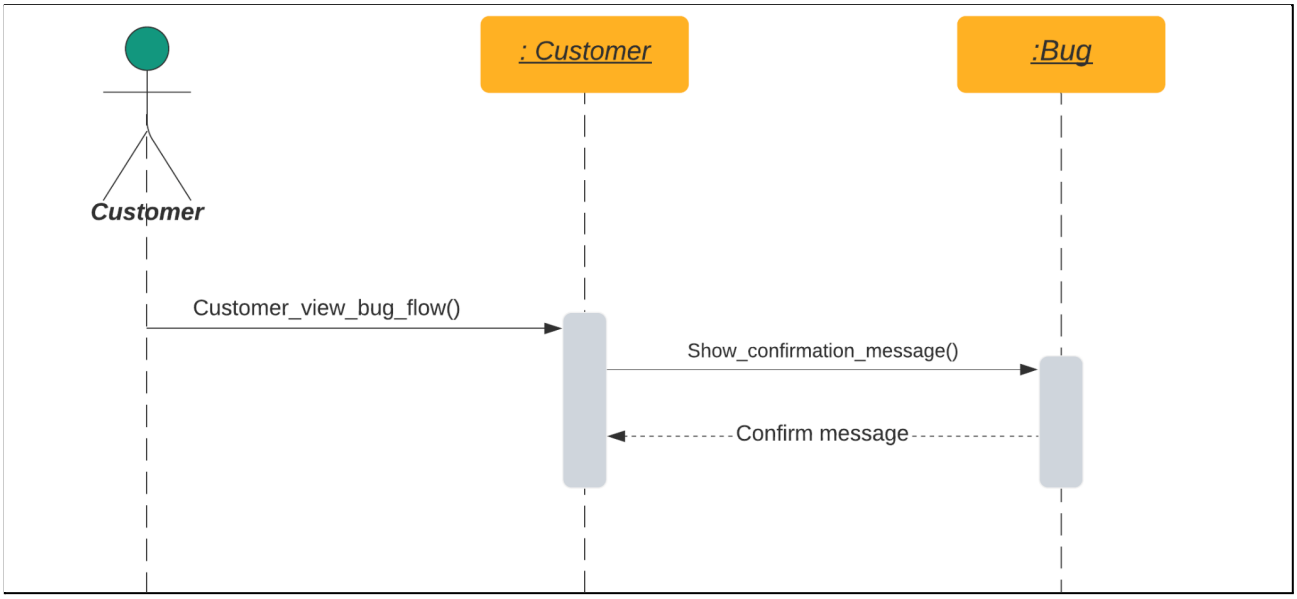
B.5.1.4 Add Bug



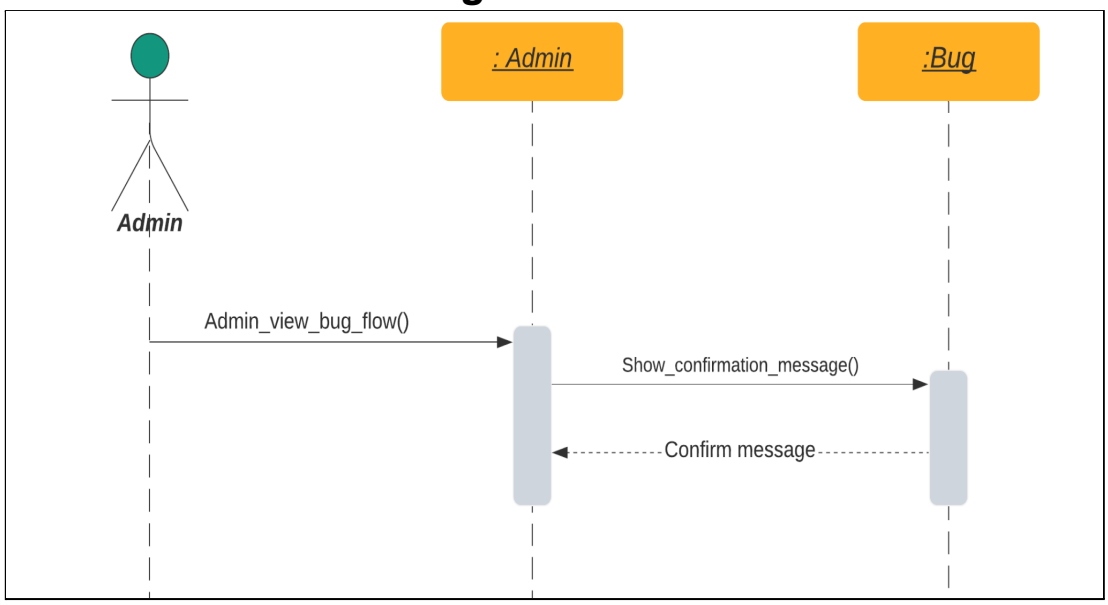
B.5.5 Assign bug



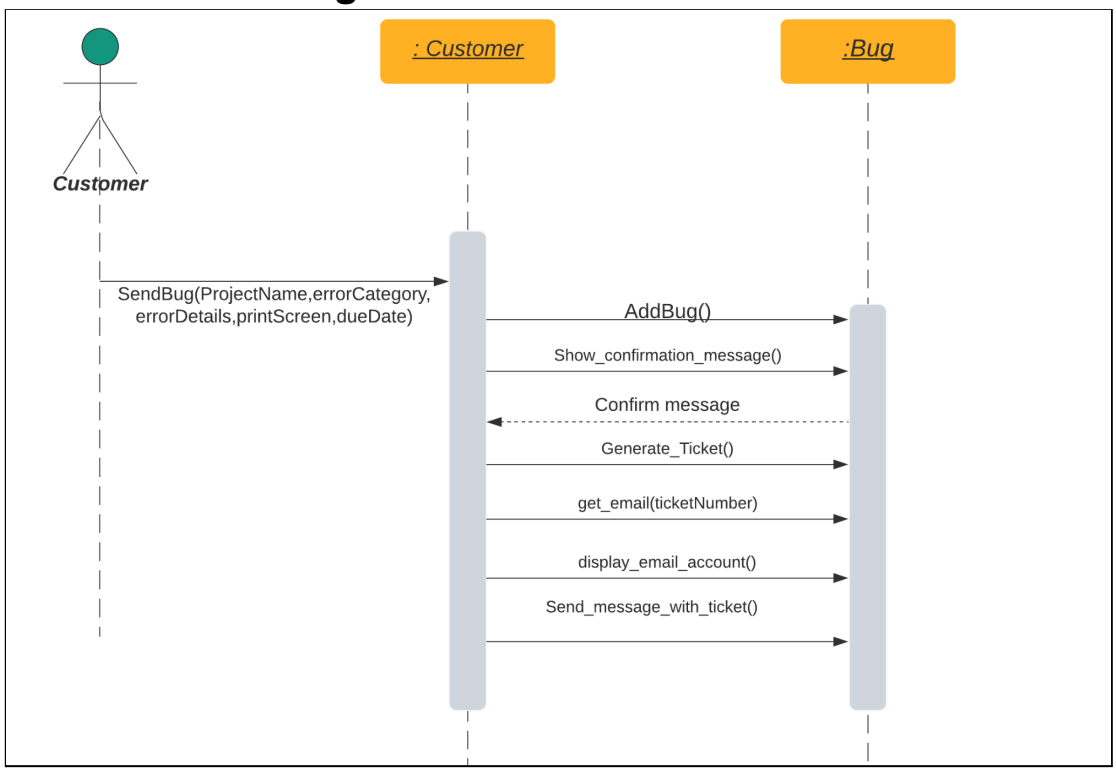
B.5.6 View assigned bug



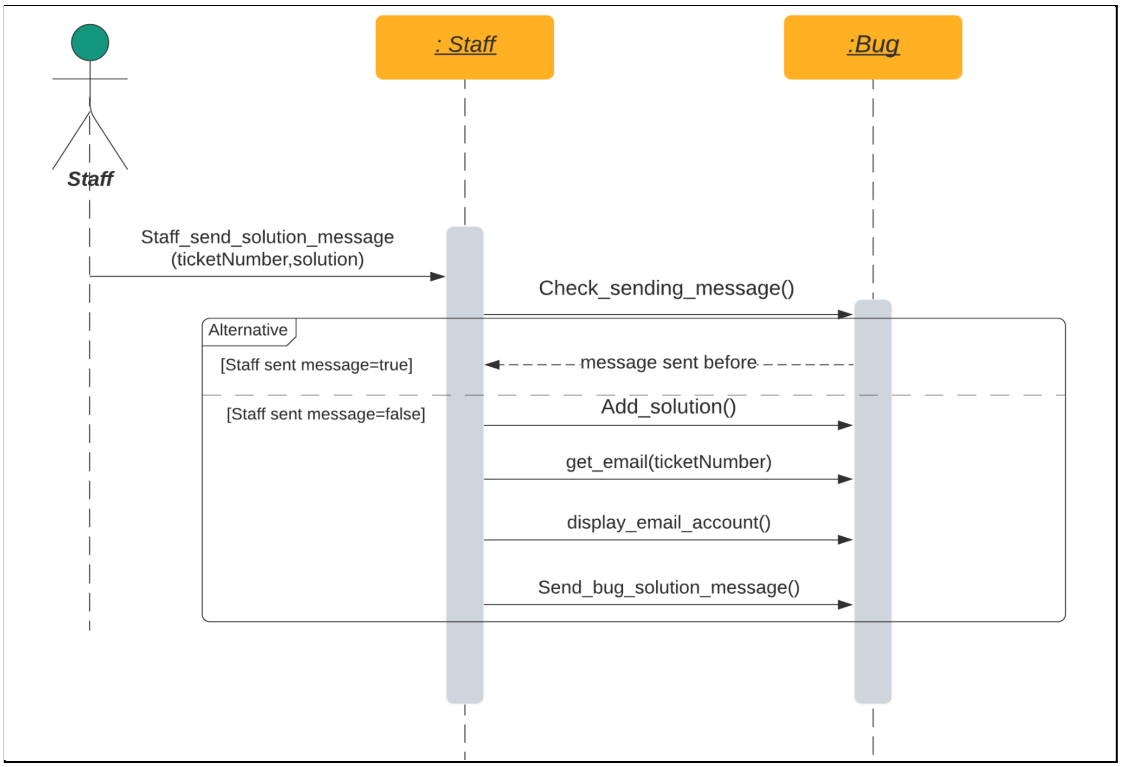
B.5.1.7 Admin view bug flow



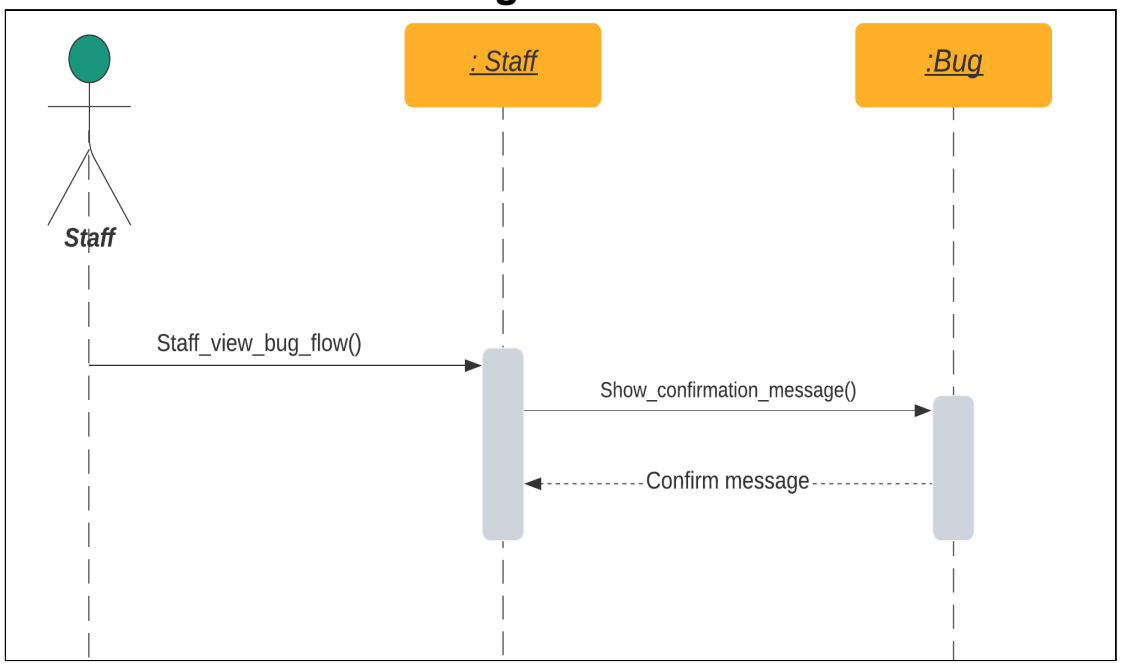
B.5.1.8 Send Bug



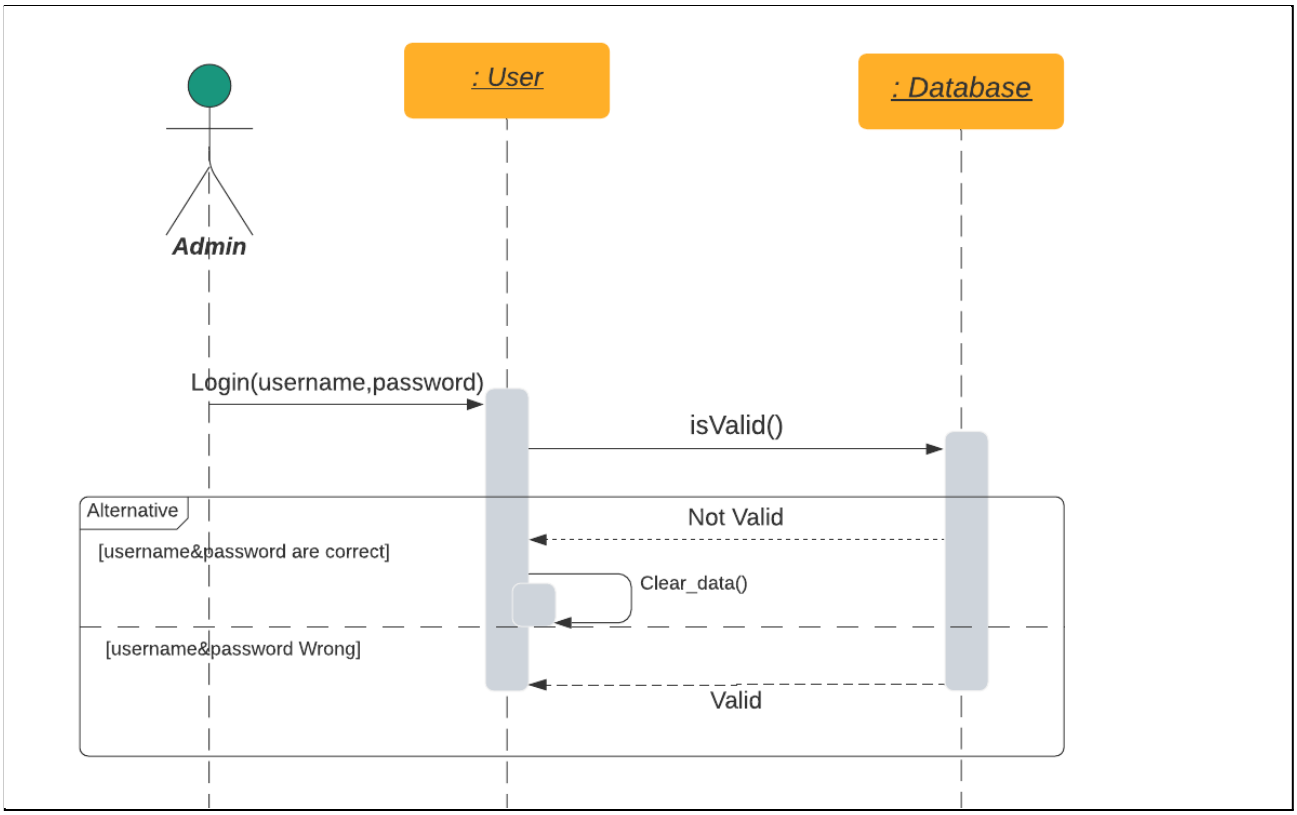
B.5.1.9 Send solution message



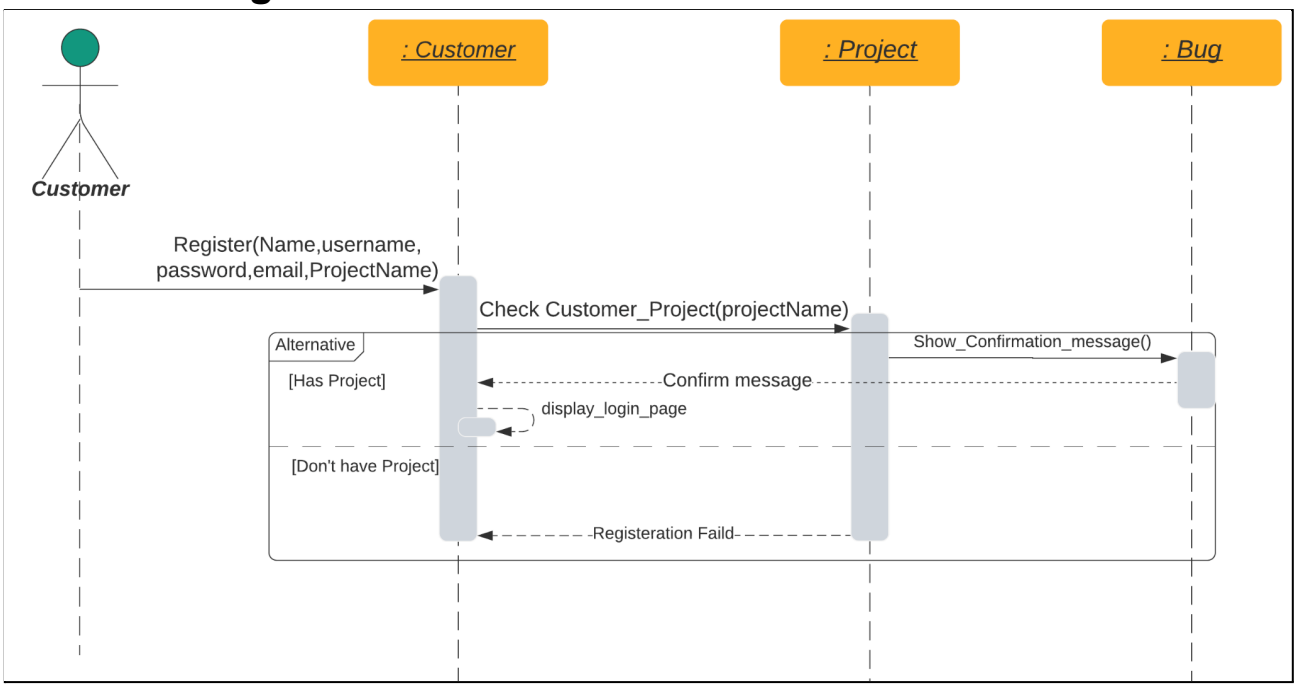
B.5.1.10 Staff view bug flow



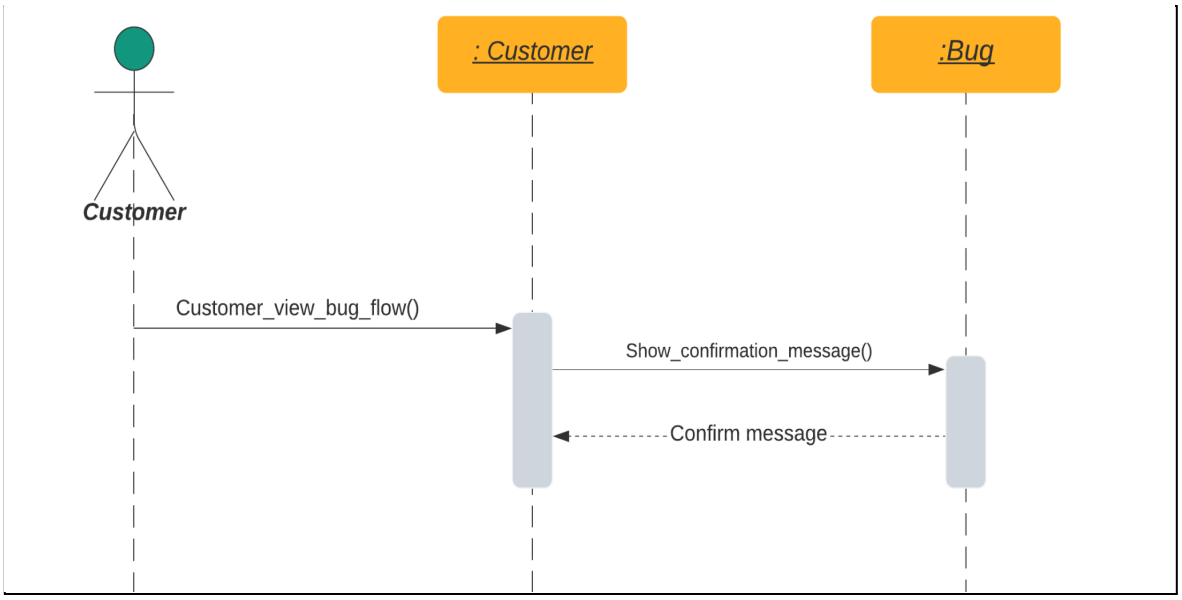
B.5.1.11 Login



B.5.1.12 Register

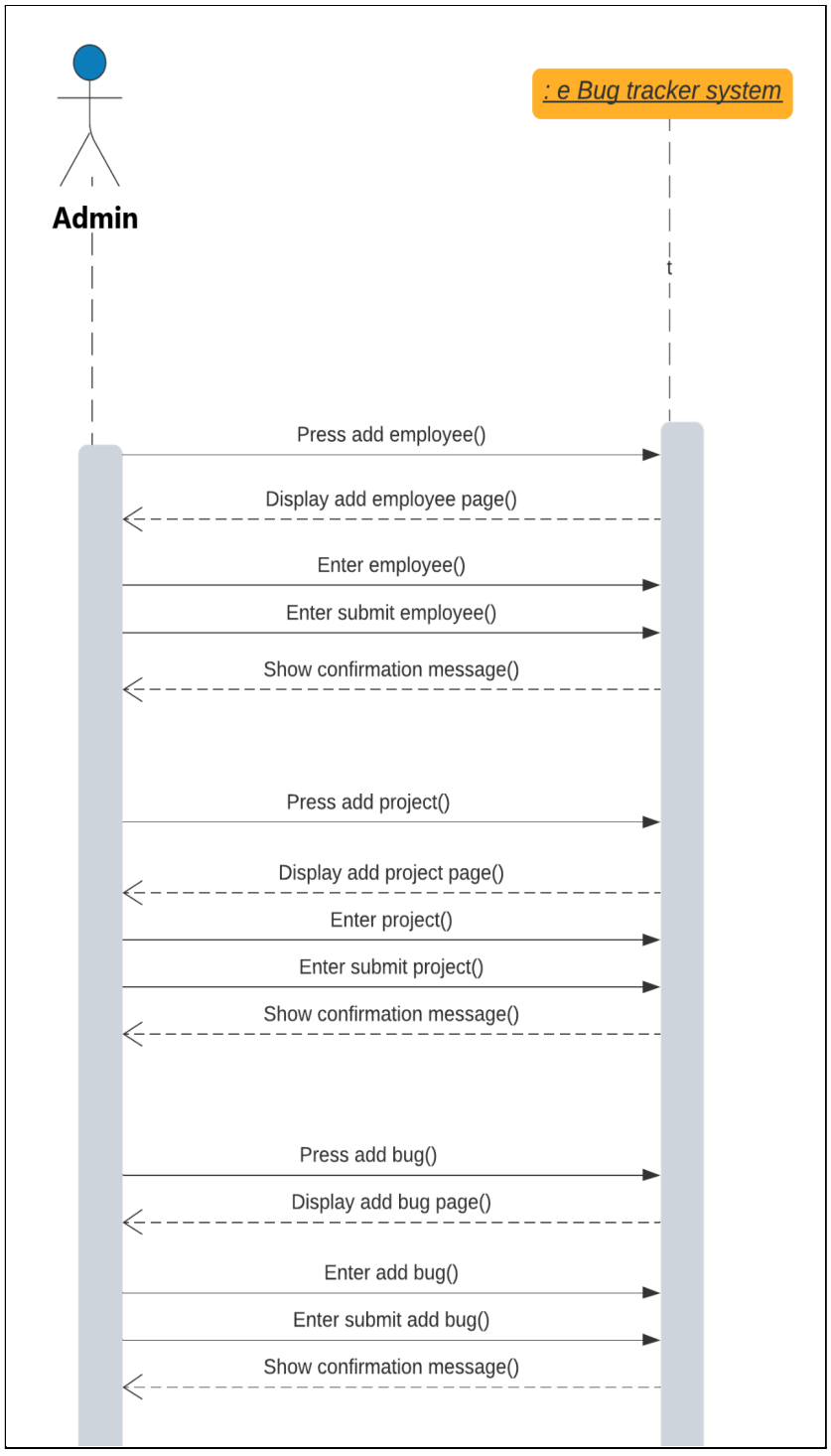


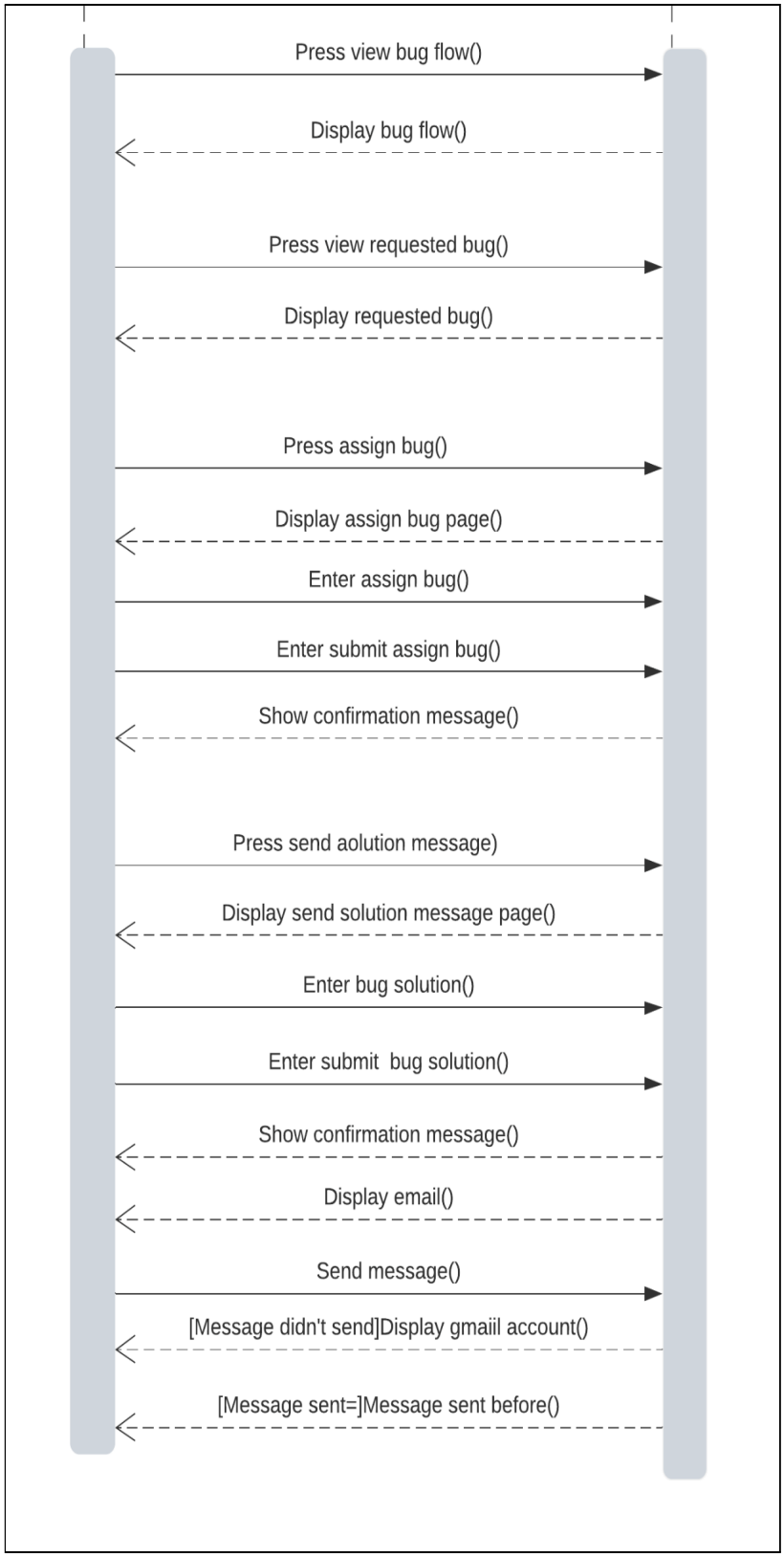
B.5.2.13 Customer view bug flow



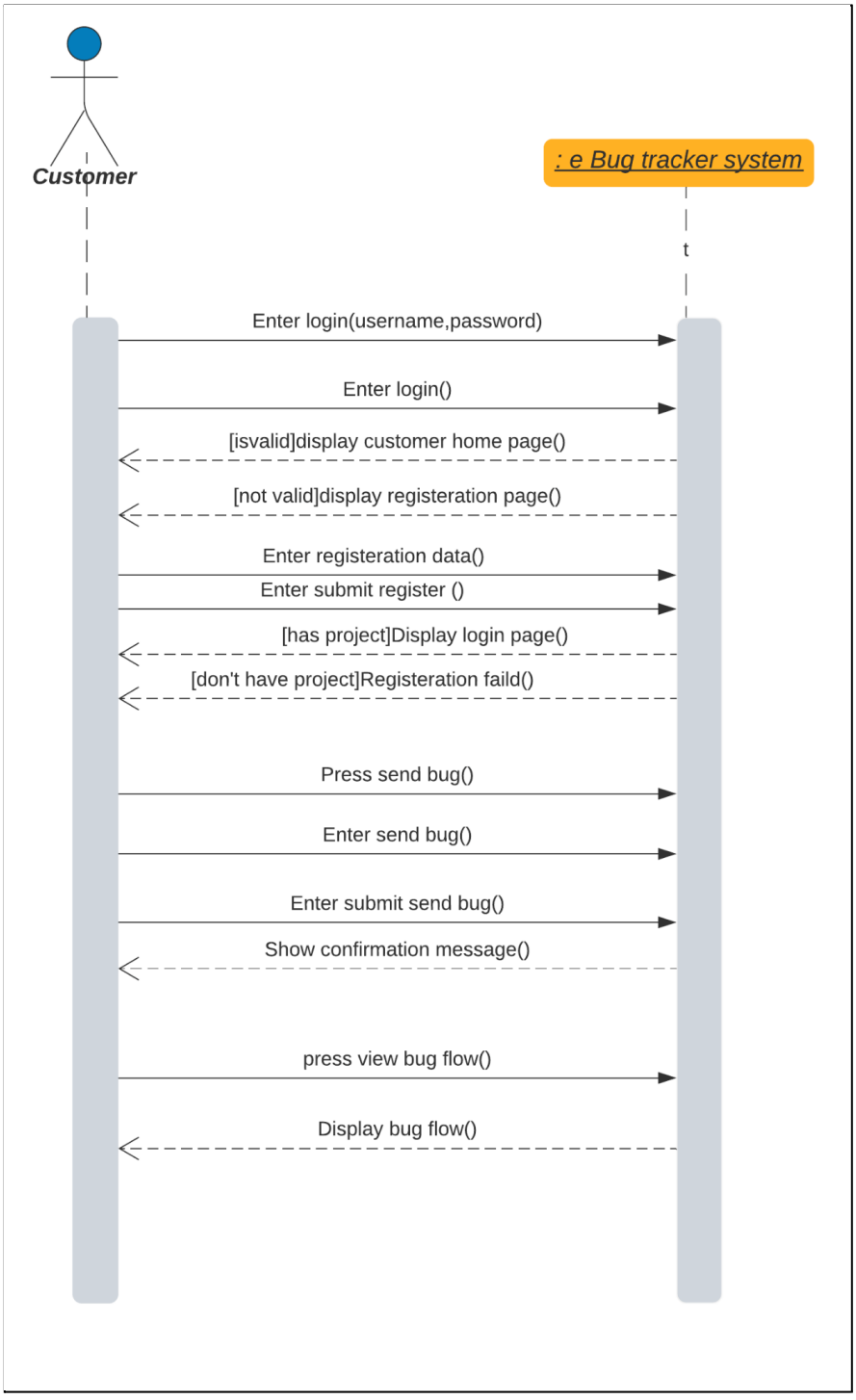
B.5.2 System Sequence

B.5.2.1 Admin

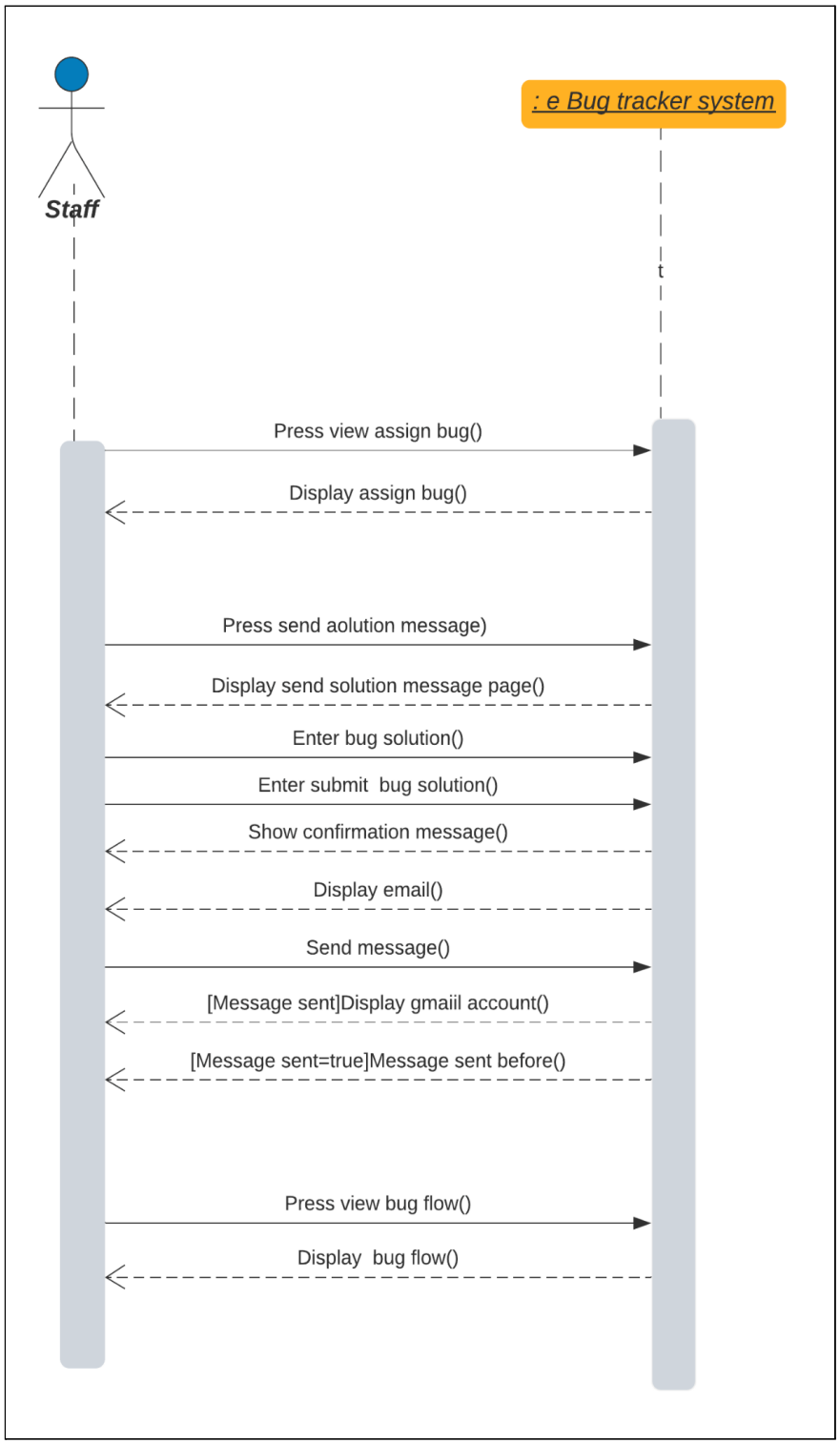

: e Bug tracker system
Admin
t
Press add employee()
Display add employee page()
Enter employee()
Enter submit employee()
Show confirmation message()
Press add project()
Display add project page()
Enter project()
Enter submit project()
Show confirmation message()
Press add bug()
Display add bug page()
Enter add bug()
Enter submit add bug()
Show confirmation message()


Press view bug flow()
Display bug flow()
Press view requested bug()
Display requested bug()
Press assign bug()
Display assign bug page()
Enter assign bug()
Enter submit assign bug()
Show confirmation message()
Press send aolution message)
Display send solution message page()
Enter bug solution()
Enter submit bug solution()
Show confirmation message()
Display email()
Send message()
[Message didn't send]Display gmaiil account()
[Message sent=]Message sent before()

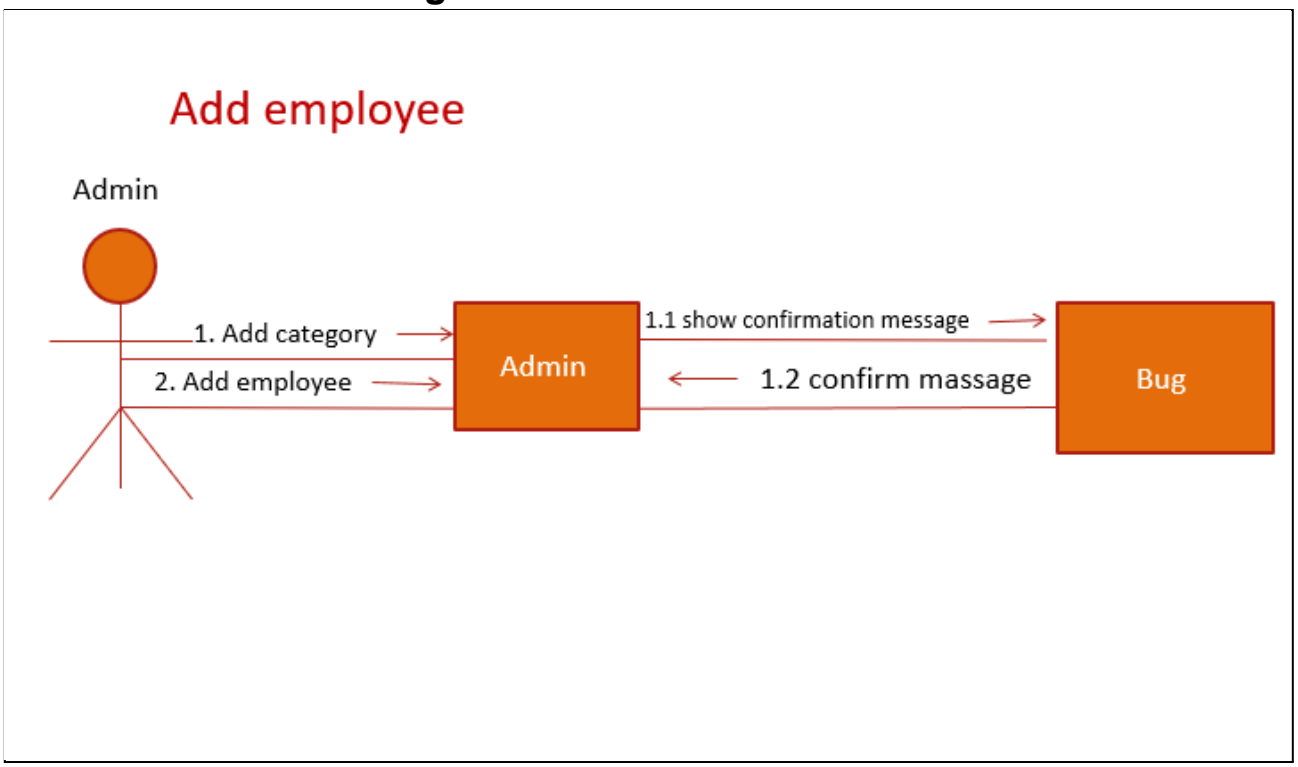
B.5.2.2 Customer

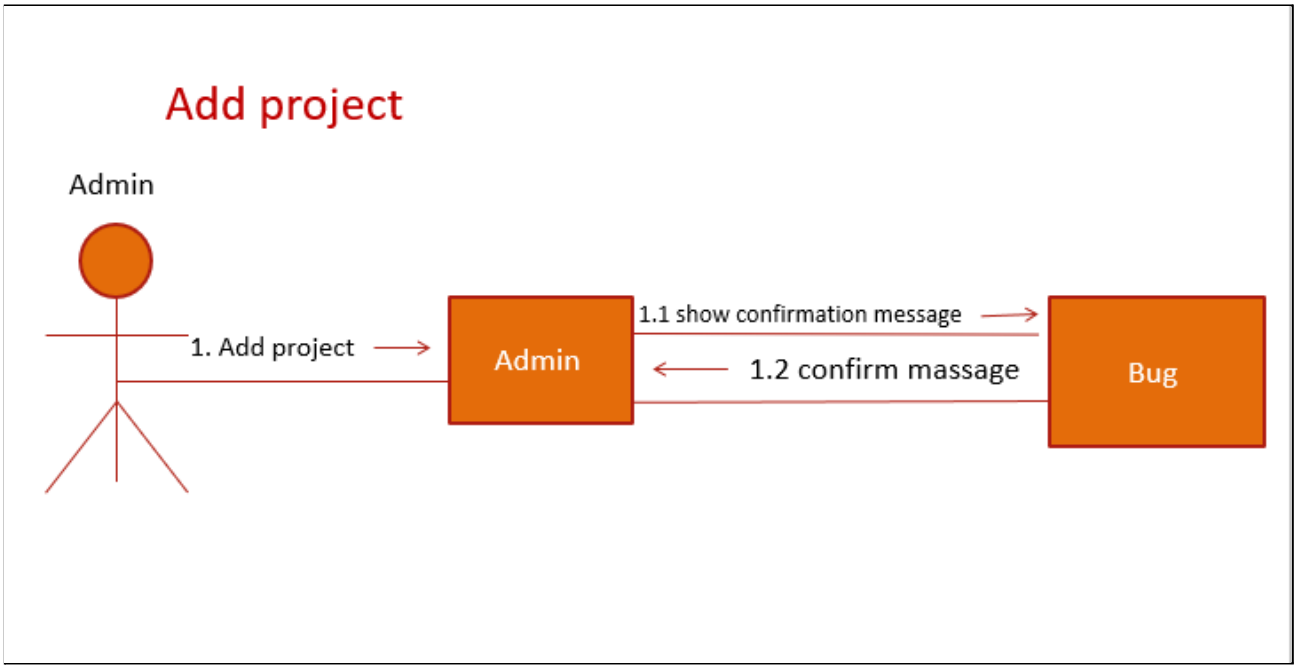


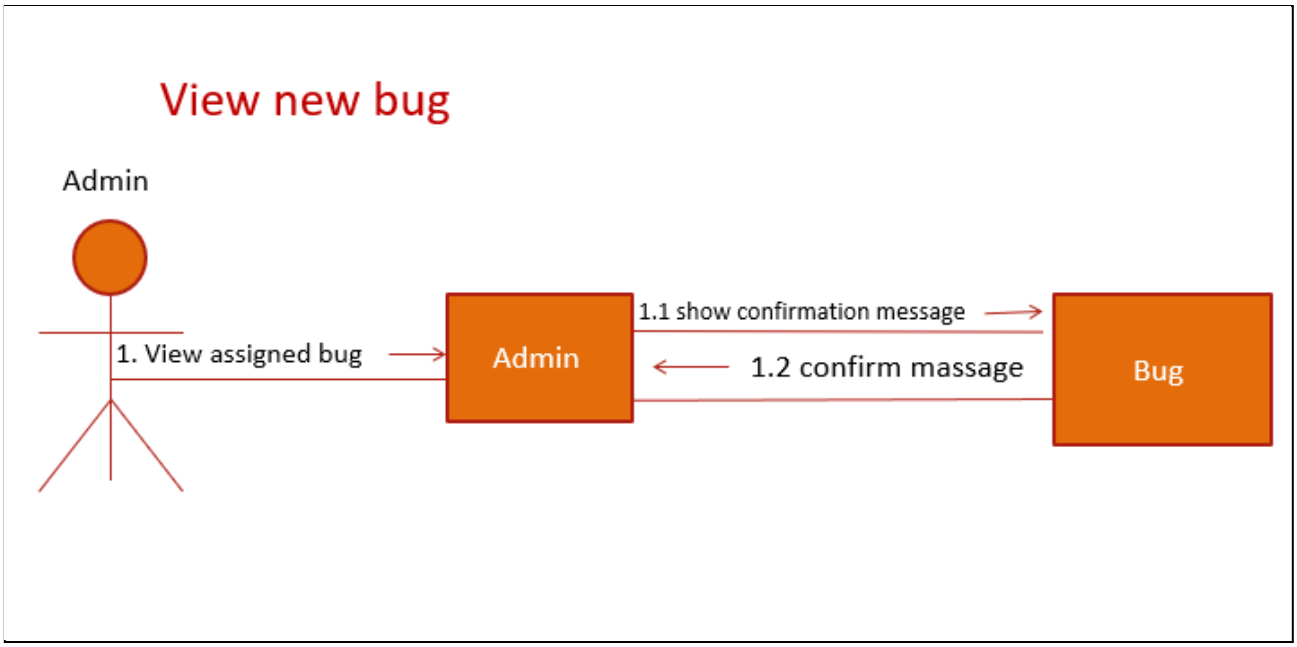
B.5.2.3 Staff

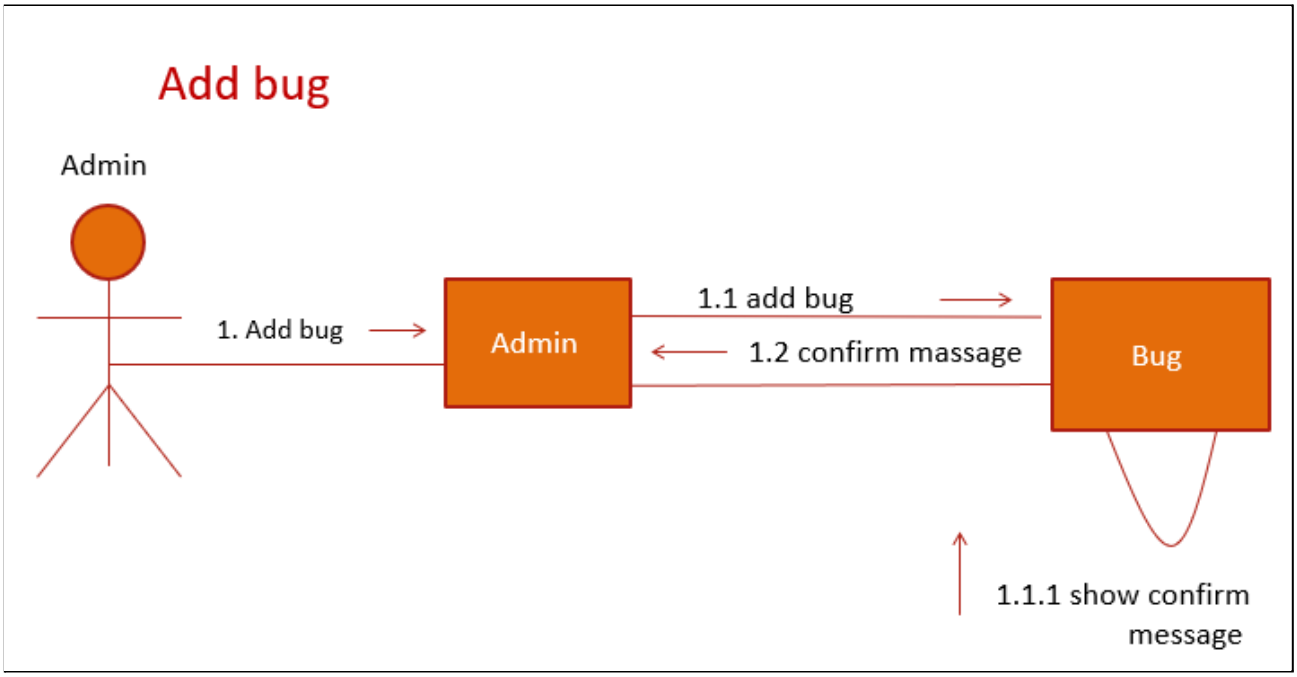

Staff
: e Bug tracker system
t
Press view assign bug()
Display assign bug()
Press send aolution message)
Display send solution message page()
Enter bug solution()
Enter submit bug solution()
Show confirmation message()
Display email()
Send message()
[Message sent]Display gmaiil account()
[Message sent=true]Message sent before()
Press view bug flow()
Display bug flow()

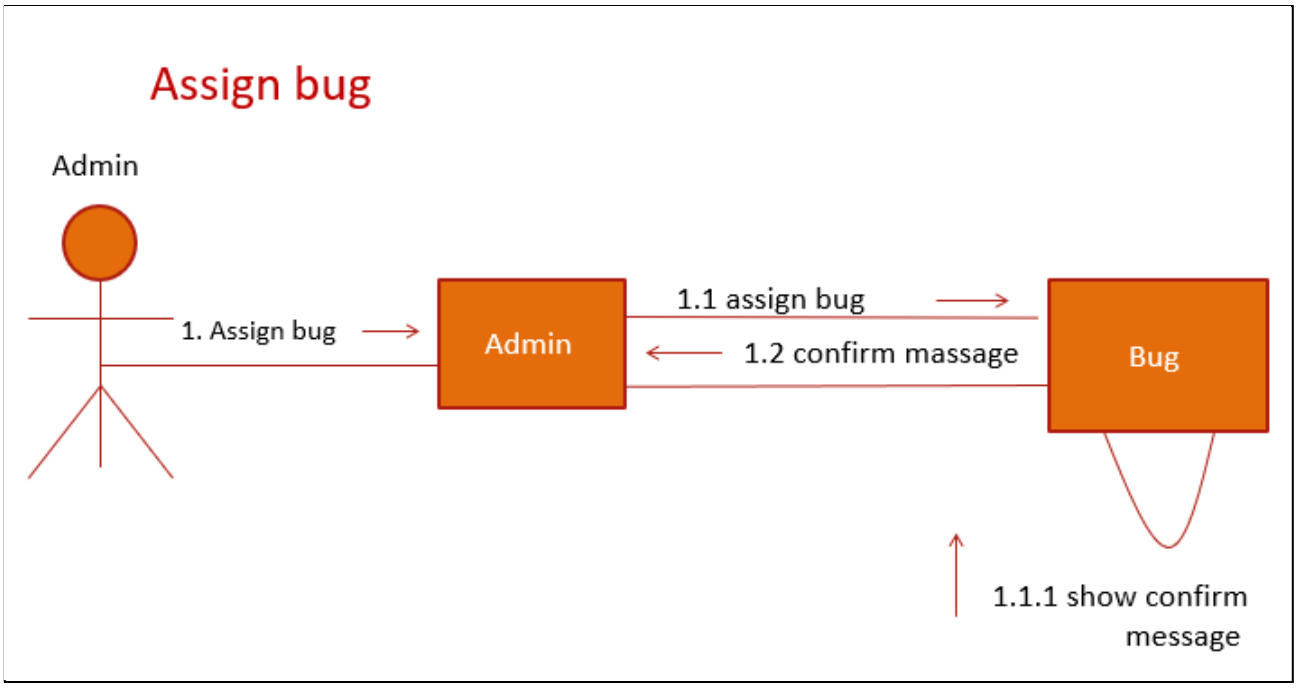
B.6 Collaboration Diagram

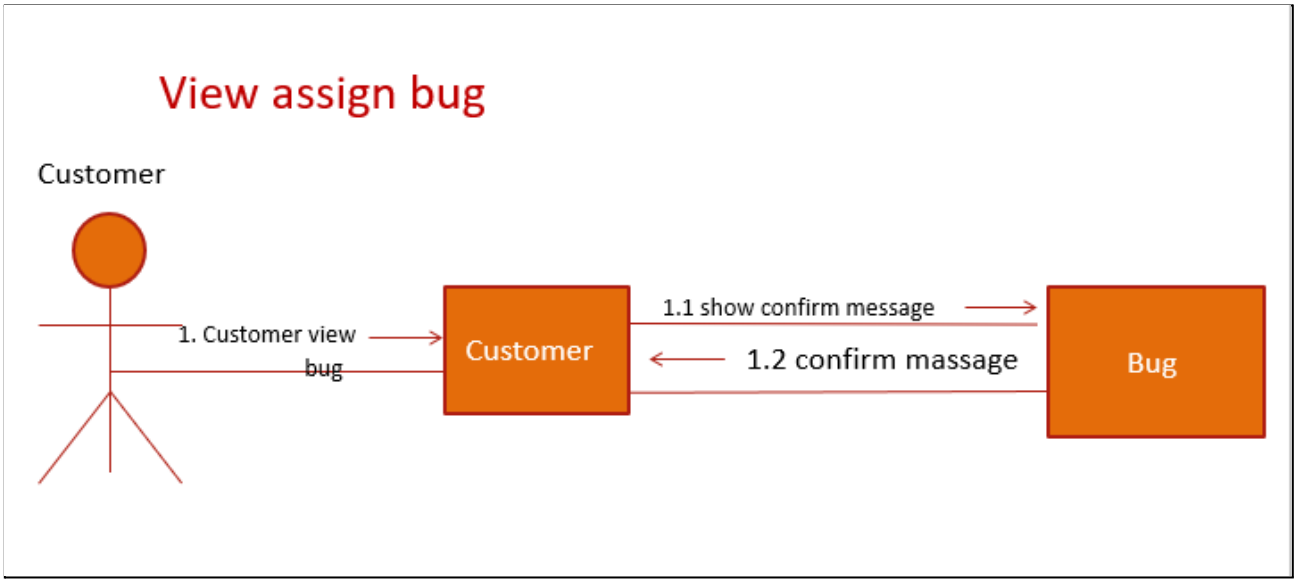


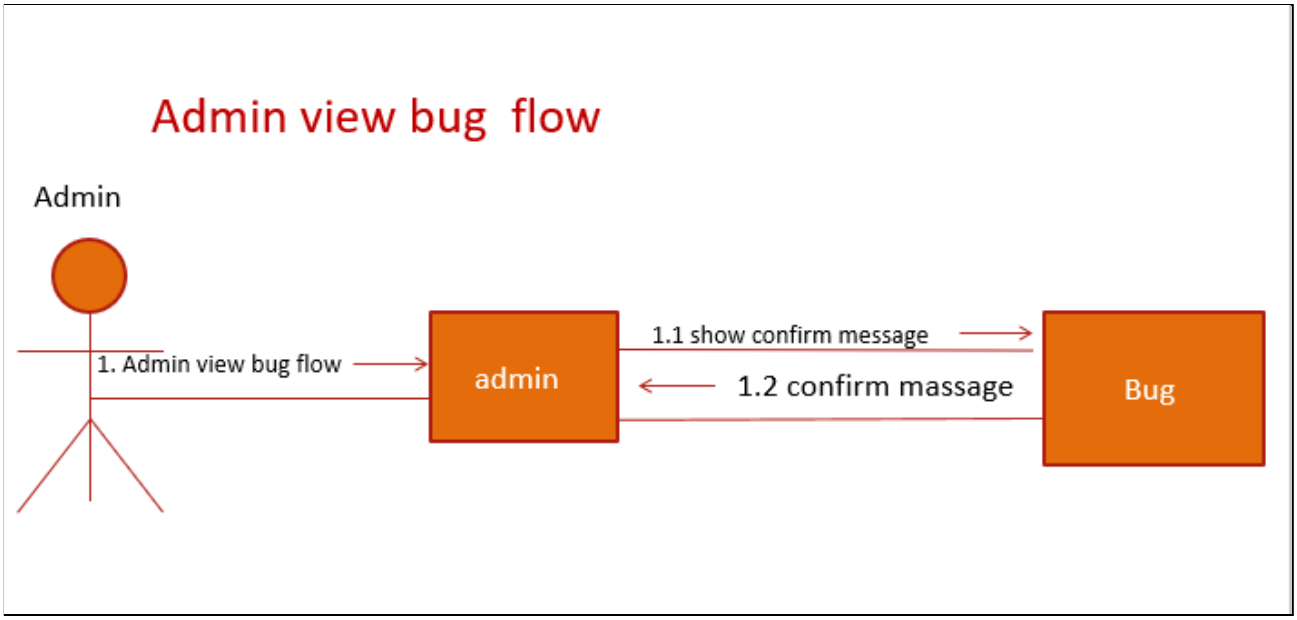


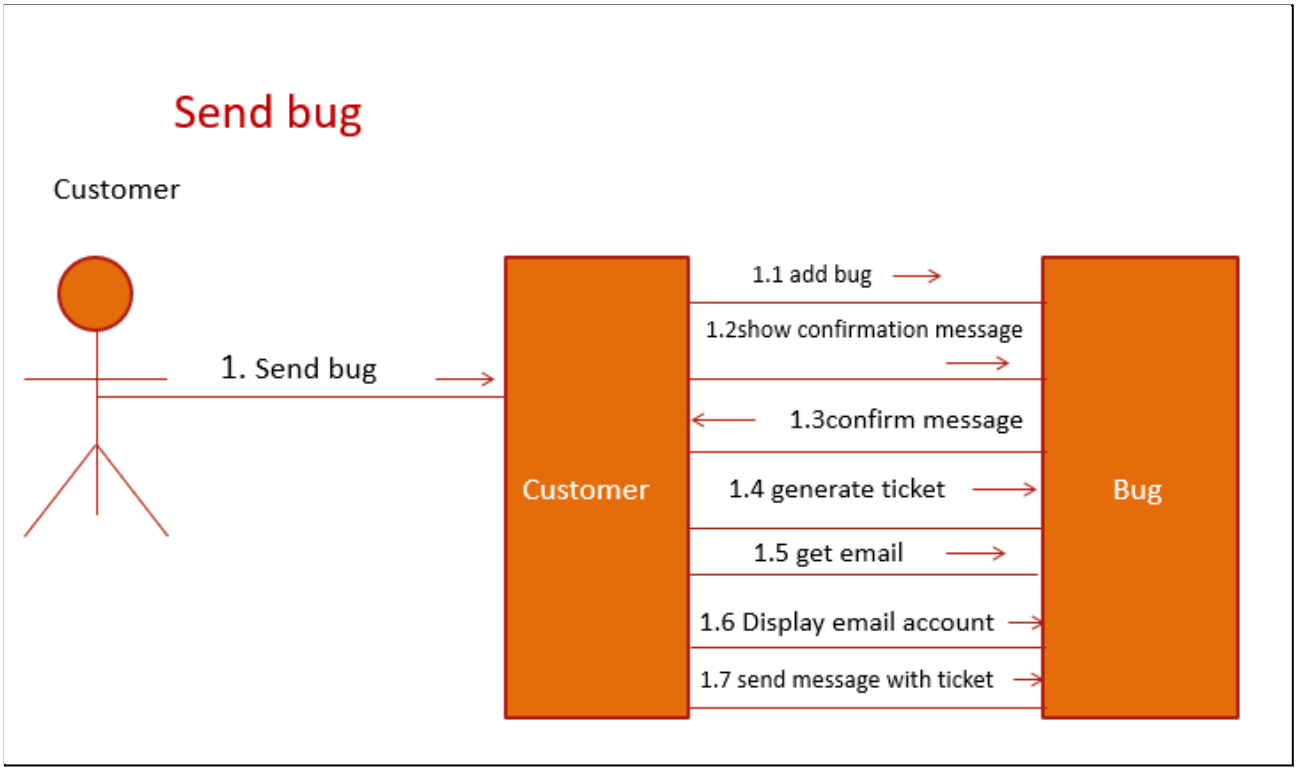

View new bug
Admin
1. View assigned bug
Admin
1.1 show confirmation message
1.2 confirm massage
Bug

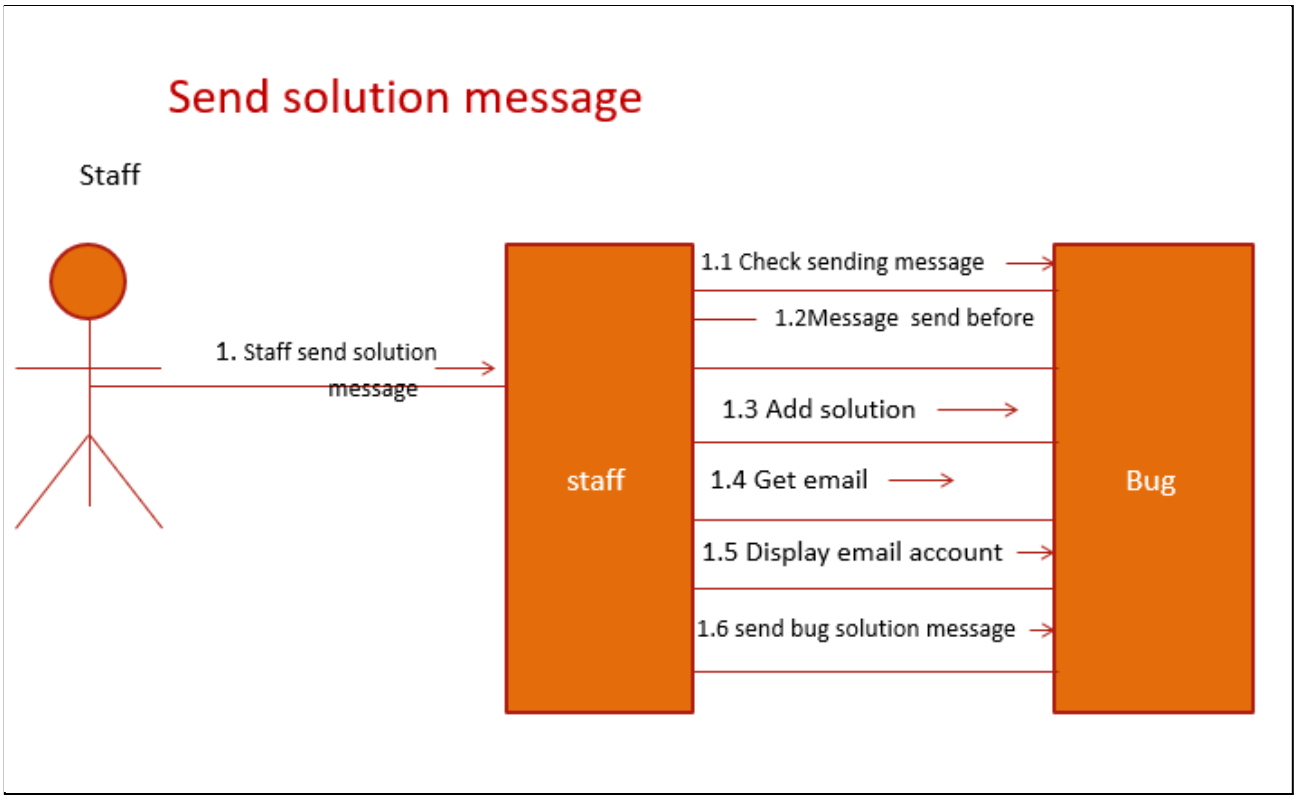

Add bug
Admin
1. Add bug
Admin
1.1 add bug
1.2 confirm massage
Bug
1.1.1 show confirm message

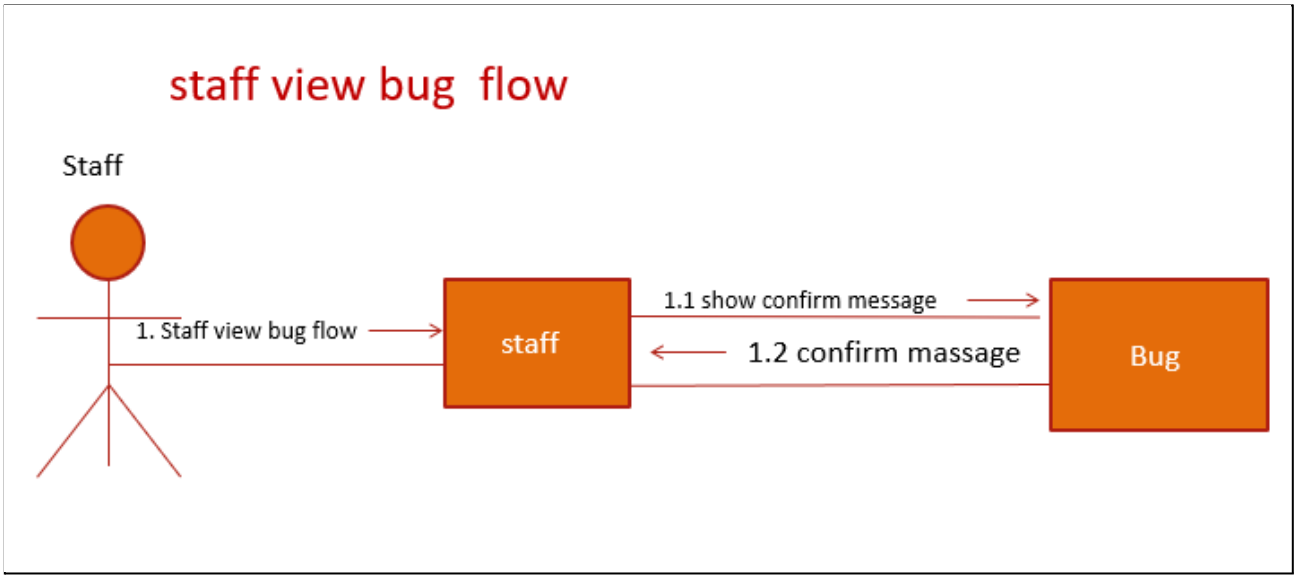

Assign bug
Admin
1. Assign bug
Admin
1.1 assign bug
1.2 confirm massage
Bug
1.1.1 show confirm message

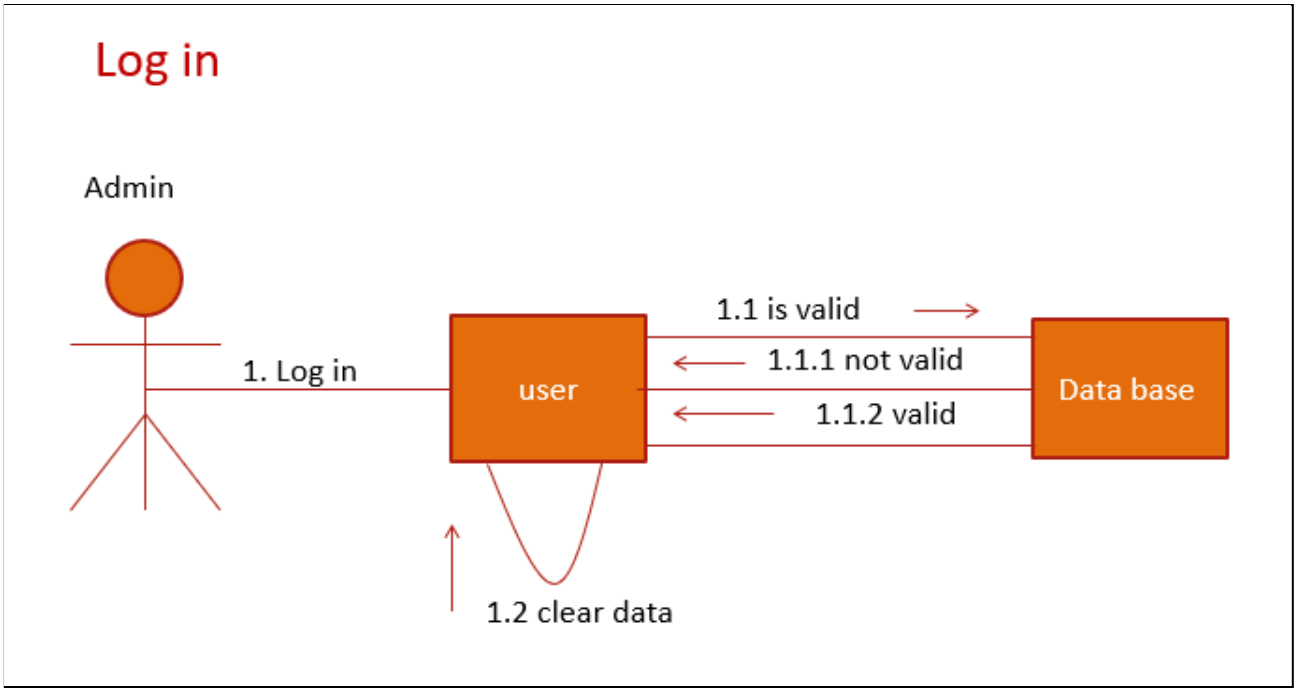

View assign bug
Customer
1. Customer view bug
Customer
1.1 show confirm message
1.2 confirm massage
Bug

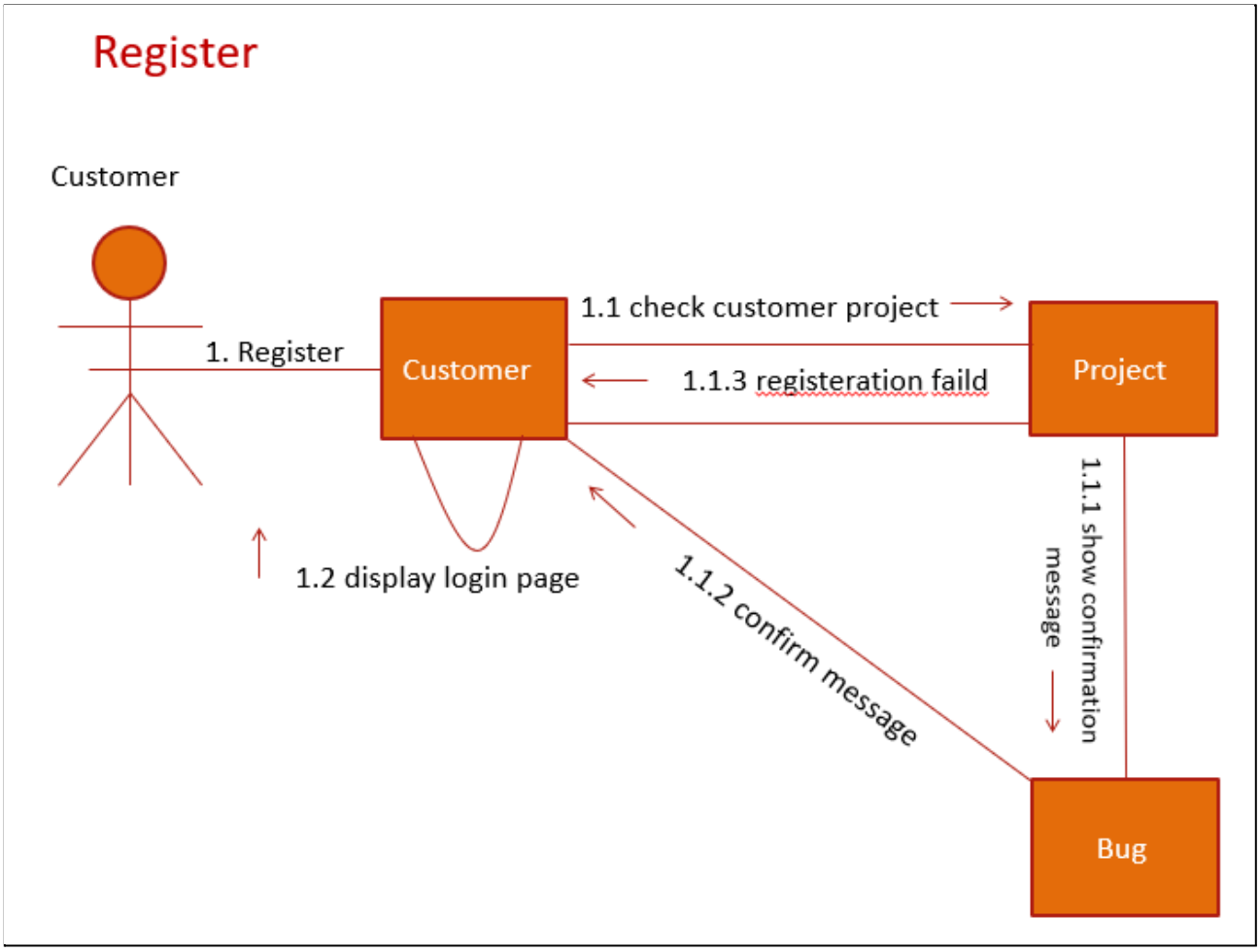

Admin view bug flow
Admin
1. Admin view bug flow
admin
1.1 show confirm message
1.2 confirm massage
Bug

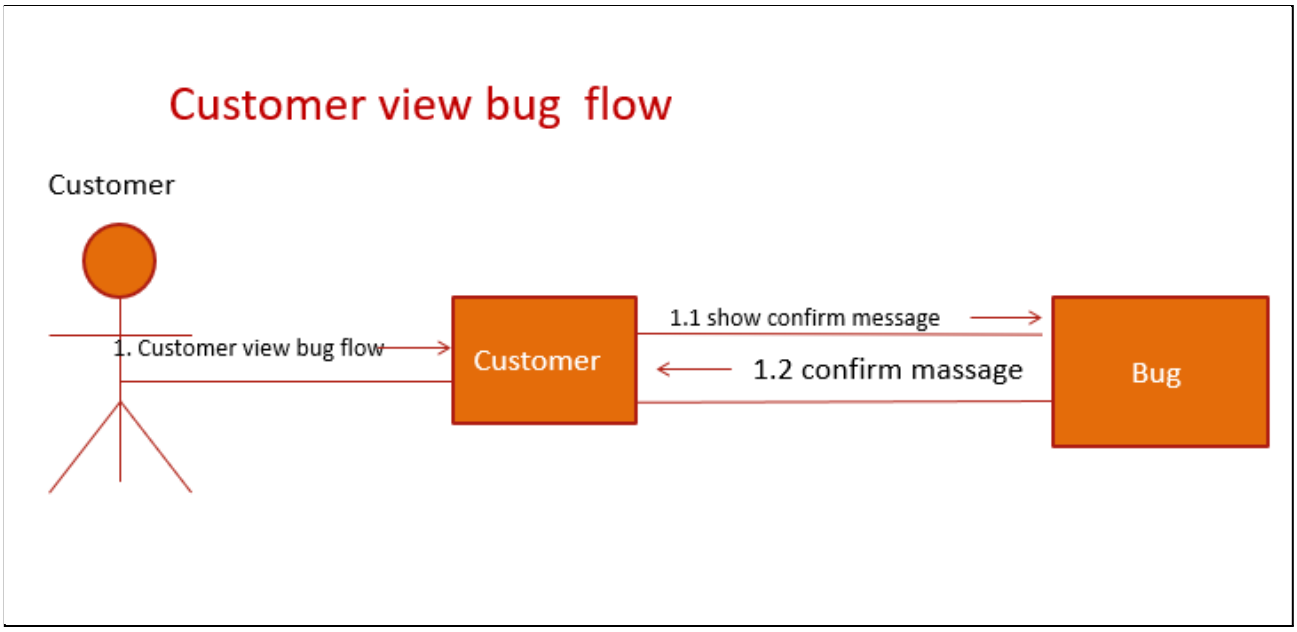

Send bug
Customer
1. Send bug
Customer
1.1 add bug
1.2show confirmation message
1.3confirm message
1.4 generate ticket
1.5 get email
1.6 Display email account
1.7 send message with ticket
Bug


Send solution message
Staff
1. Staff send solution message
staff
1.1 Check sending message
1.2Message send before
1.3 Add solution
1.4 Get email
1.5 Display email account
1.6 send bug solution message
Bug

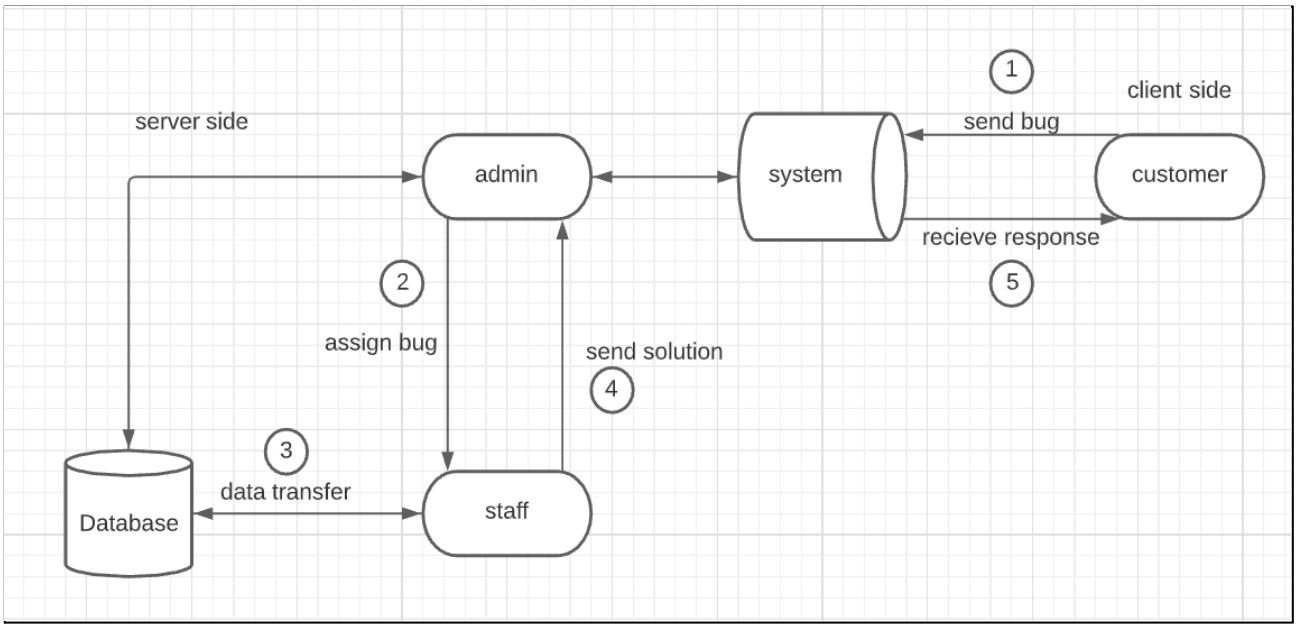

staff view bug flow
Staff
1. Staff view bug flow
staff
1.1 show confirm message
1.2 confirm massage
Bug


Log in
Admin
1. Log in
user
1.1 is valid
1.1.1 not valid
1.1.2 valid
Data base
1.2 clear data


Register
Customer
1. Register
Customer
1.1 check customer project
1.1.3 registeration faild
Project
1.2 display login page
1.1.2 confirm message
1.1.1 show confirmation message
Bug


Customer view bug flow
Customer
1. Customer view bug flow
Customer
1.1 show confirm message
1.2 confirm massage
Bug

B.7 System Architectural Design



overall architecture

B.8 Design Pattern

B.8.1 singleton:

- **Used on :** Database
- **Used for:**ensuring that it is never possible to create more than one instance of a Database class.
- **Problem:** in our system ,there should only exist one instance of the database class, having many instances of this class may lead to unwanted behaviours.
- **Benefit**: No more redundant instances and better security.
- **How to implement Database as singleton class:**The use of a public constructor cannot guarantee that no more than one instance will be created. The singleton instance must also be accessible to all classes that require it, therefore it must often be public

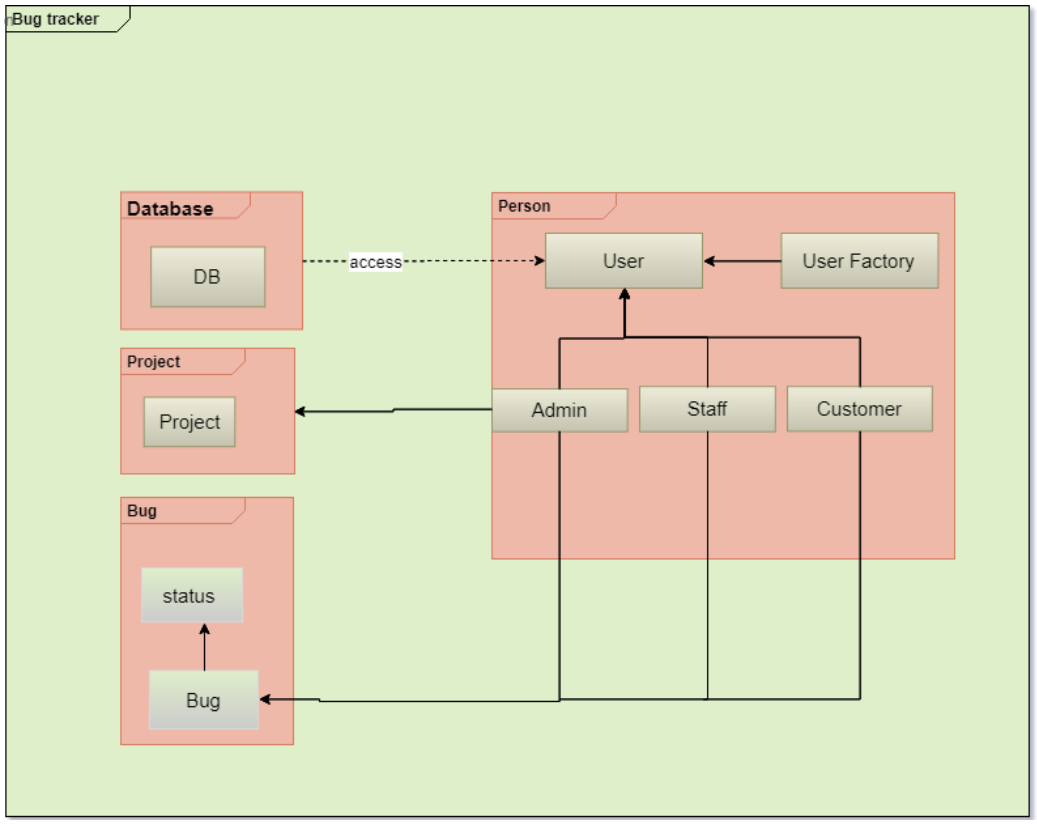
B.8.3 Delegation

- **Used on :**Admin,staff, and Customer
- **Used for**:minimize development cost by reusing methods
- **Problem:** Duplication of chunks of code.
- **Benefit:**making use of a method that already exists in the other class
- **How to implement Admin,staff, and Customer as delegator class:**
- **The delegating method** (Admin assign bug ,Staff assign bug)in **the delegator class**(Admin,Staff) calls a **method**(Assign bug) in **the delegate class**(Bug) to perform the required task(assign bug to staff to solve a bug).
- **The delegating method** (Send bug,Add bug)in **the delegator class**(Admin,Customer) calls a **method**(Add bug) in **the delegate class**(Bug) to perform the required task(add bug on database)
- **The delegating method(**Admin send solution message,Staff send solution message)in **the delegator class**(Admin,Customer) calls **methods**(Check sending message,Add solution,get email related to ticket,display email account,Show confirmation message) in **the delegate class**(Bug) to perform the required task(send a solution to customer via email account)
- An association must exist between the delegator and delegate classes.

B.8.2 Factory

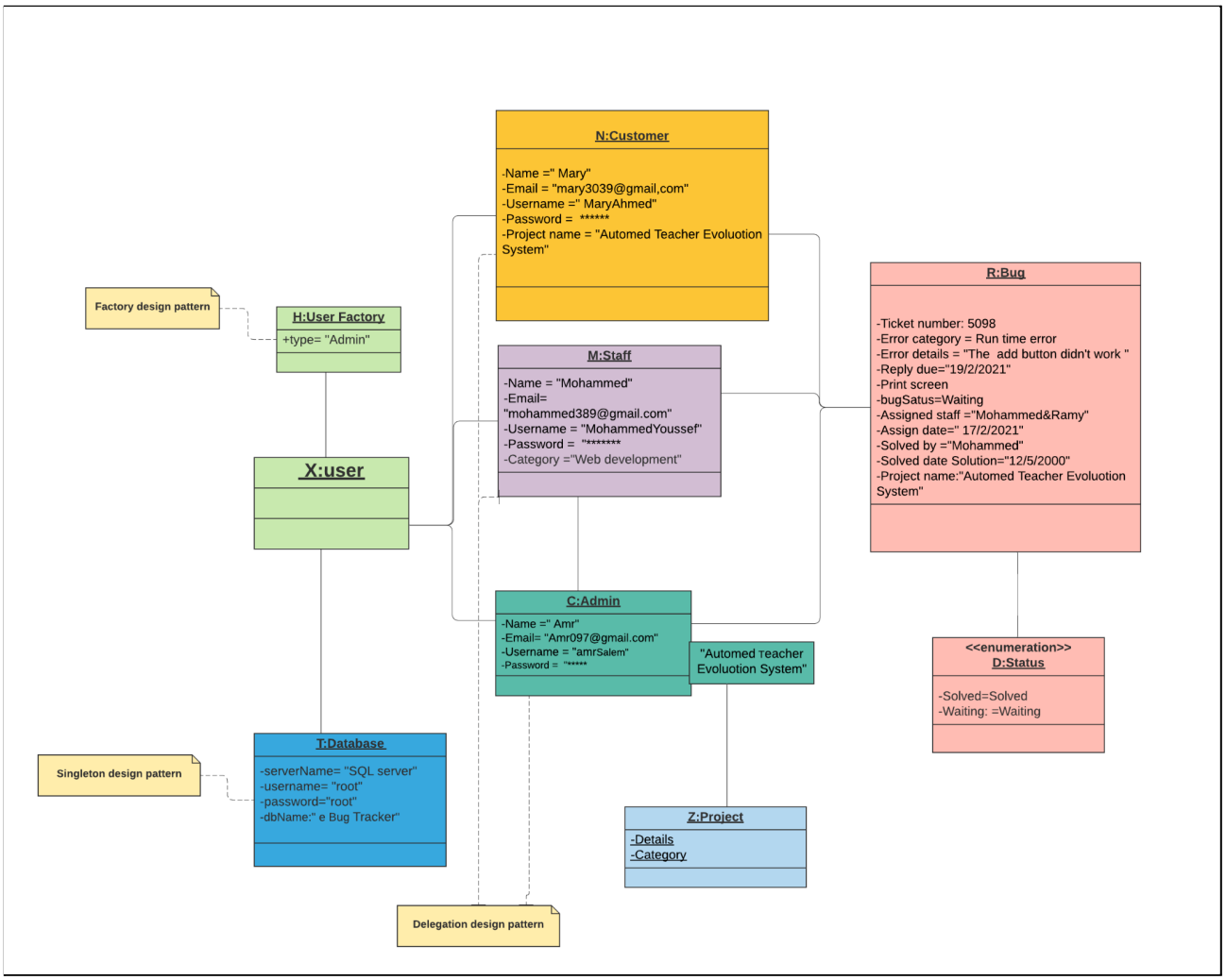
- **Used on :** User Factory
- **Used for:**creating an object without exposing the creation logic to the client and referring to a newly created object using a common interface.
- **Benefit:**The factory design pattern says that defines an abstract class and lets the subclasses decide which object to instantiate.
- **How to implement User Factory as Factory class:** Define a factory method inside an Abstract method.Let the subclass implement the above factory method and decide which object to create.

B.9 Package Diagram



B.10 Object Diagram

Preconditions and Postconditions are illustrated in Functional Requirements.



Appendix C: To Be Determined List

- None of the sections