Object Oriented Analysis and Design

Automated Bill Payment System

Project Deliverable 2

Submitted to:

Ma'am Amina mirza

Group Members:

Bsef19m013 - Osama Sultan

Bsef19m032 - Ameena Abdullah

Bsef19m029 - Iqra Hanif

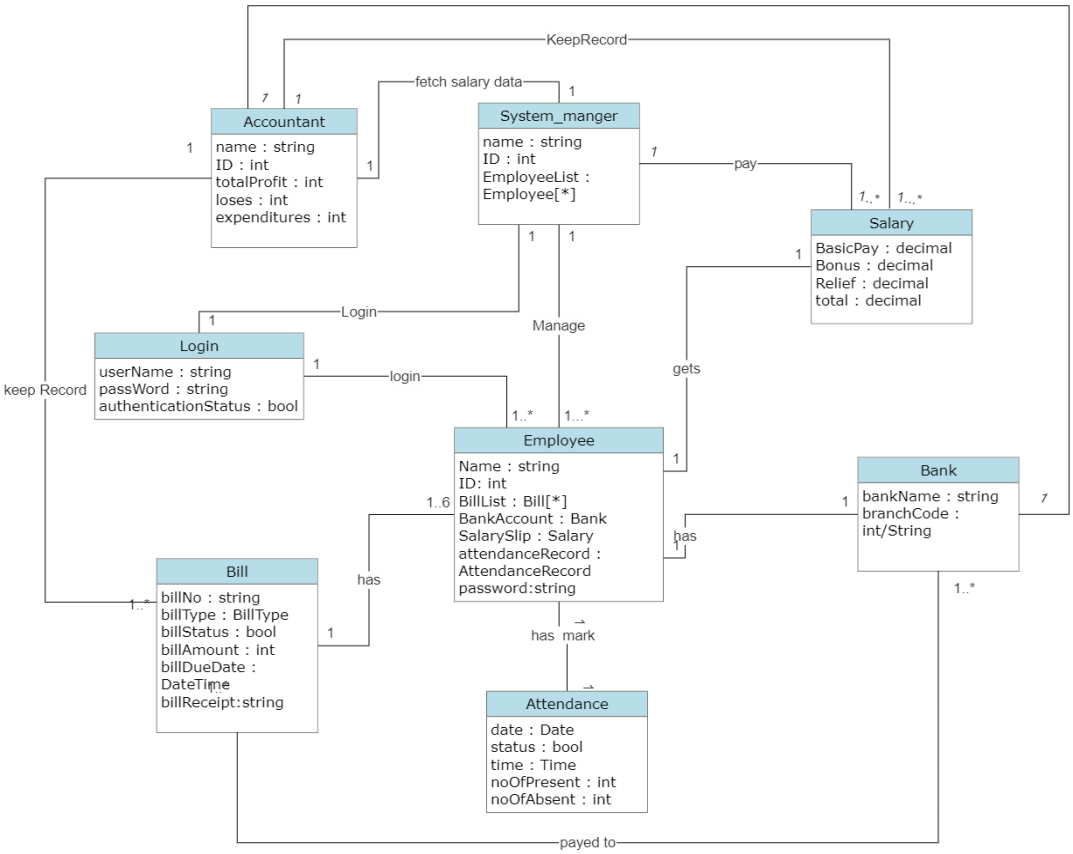
Automated Bill payments Application

Table of Contents

*All the diagrams in this document are created on smartdraw.com

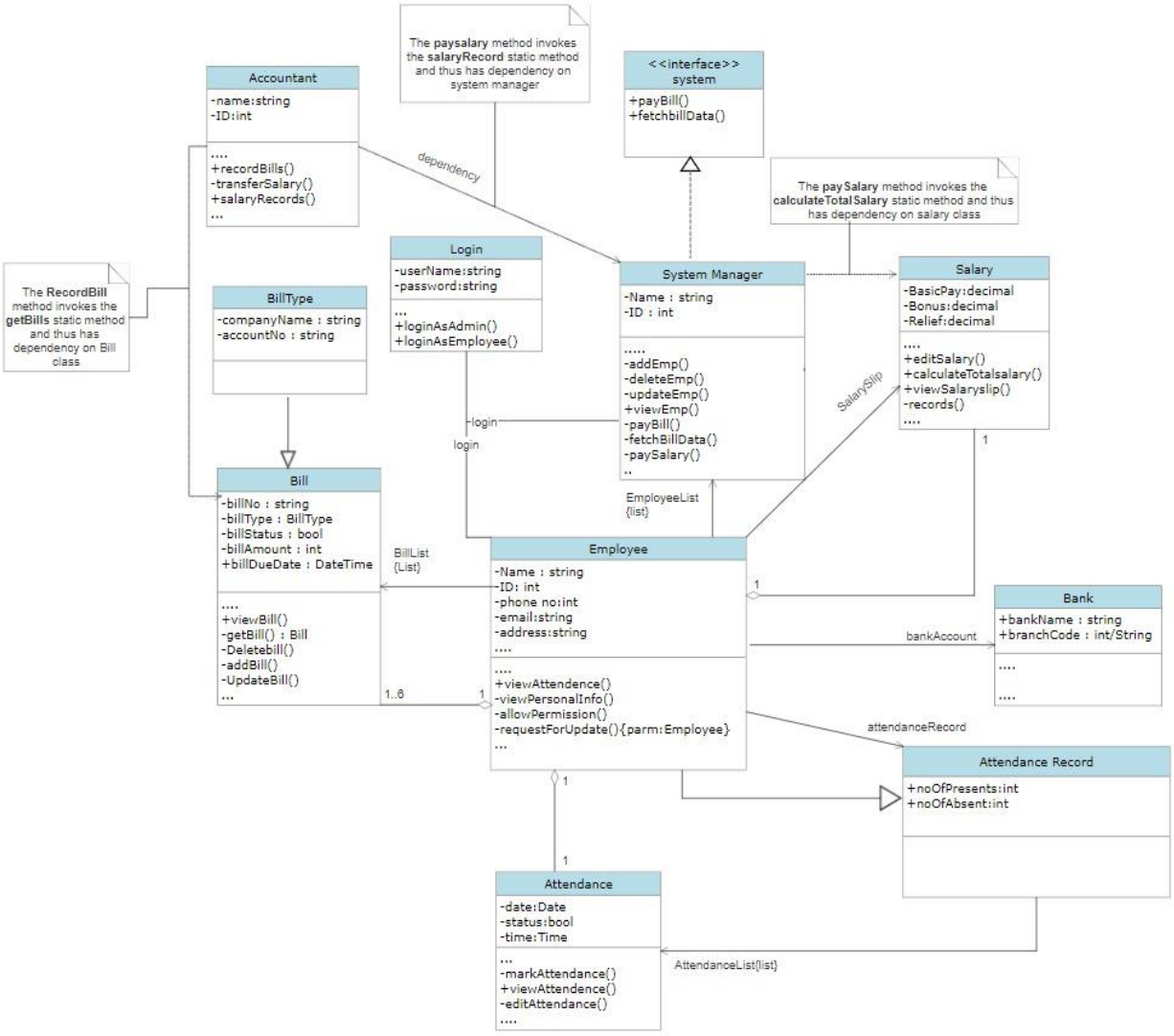
Domain Model

(Visual representation of conceptual classes of Automated Bill payment System)



Design Class Diagram

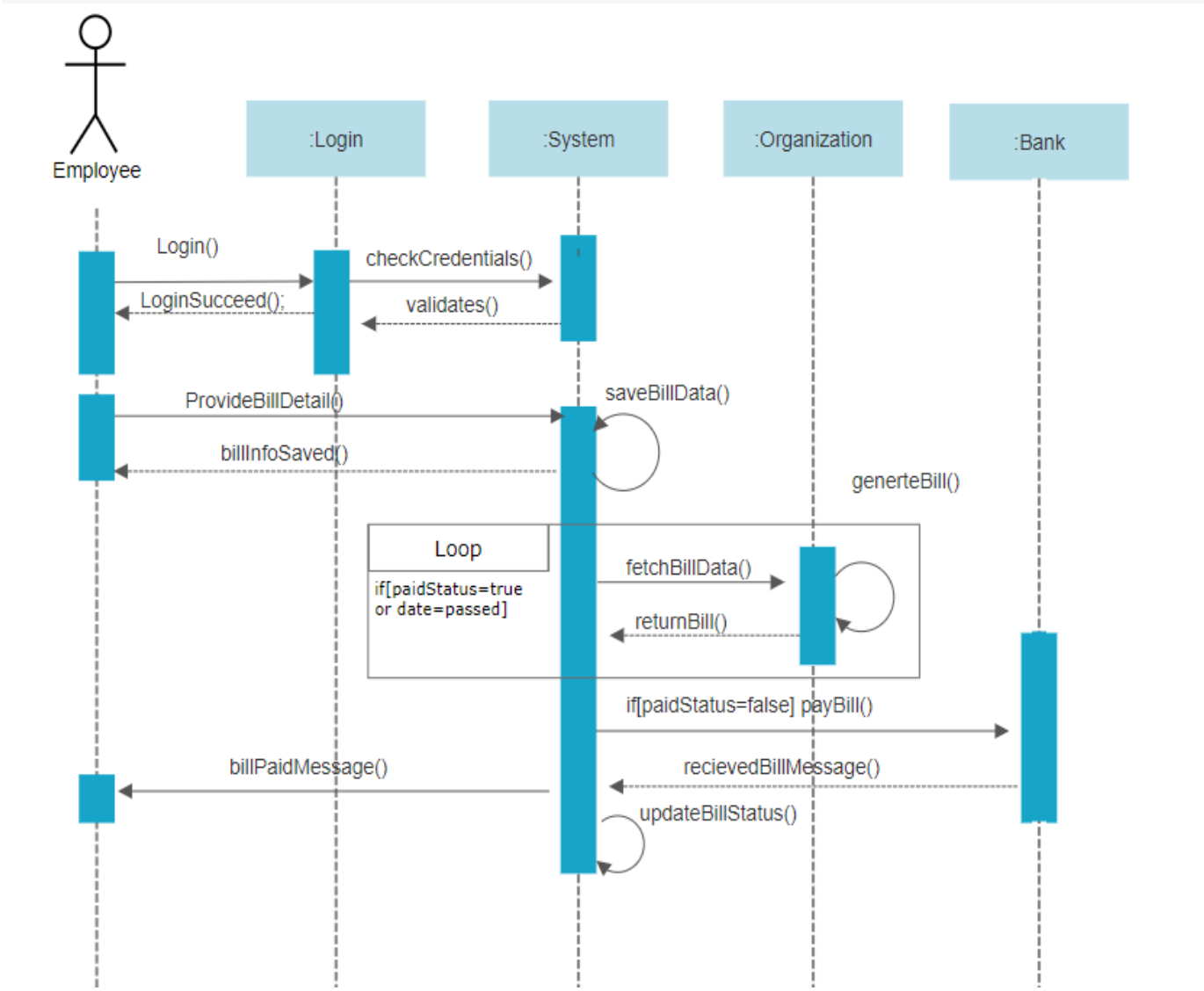
(Detailed Visual Representation of system class and their relationships)



Sequence Diagram

(Sequence flow of bill payment among different states)

- Employee with add bill information once and system will then automatically fetch data from organization site, check status and performs accordingly.

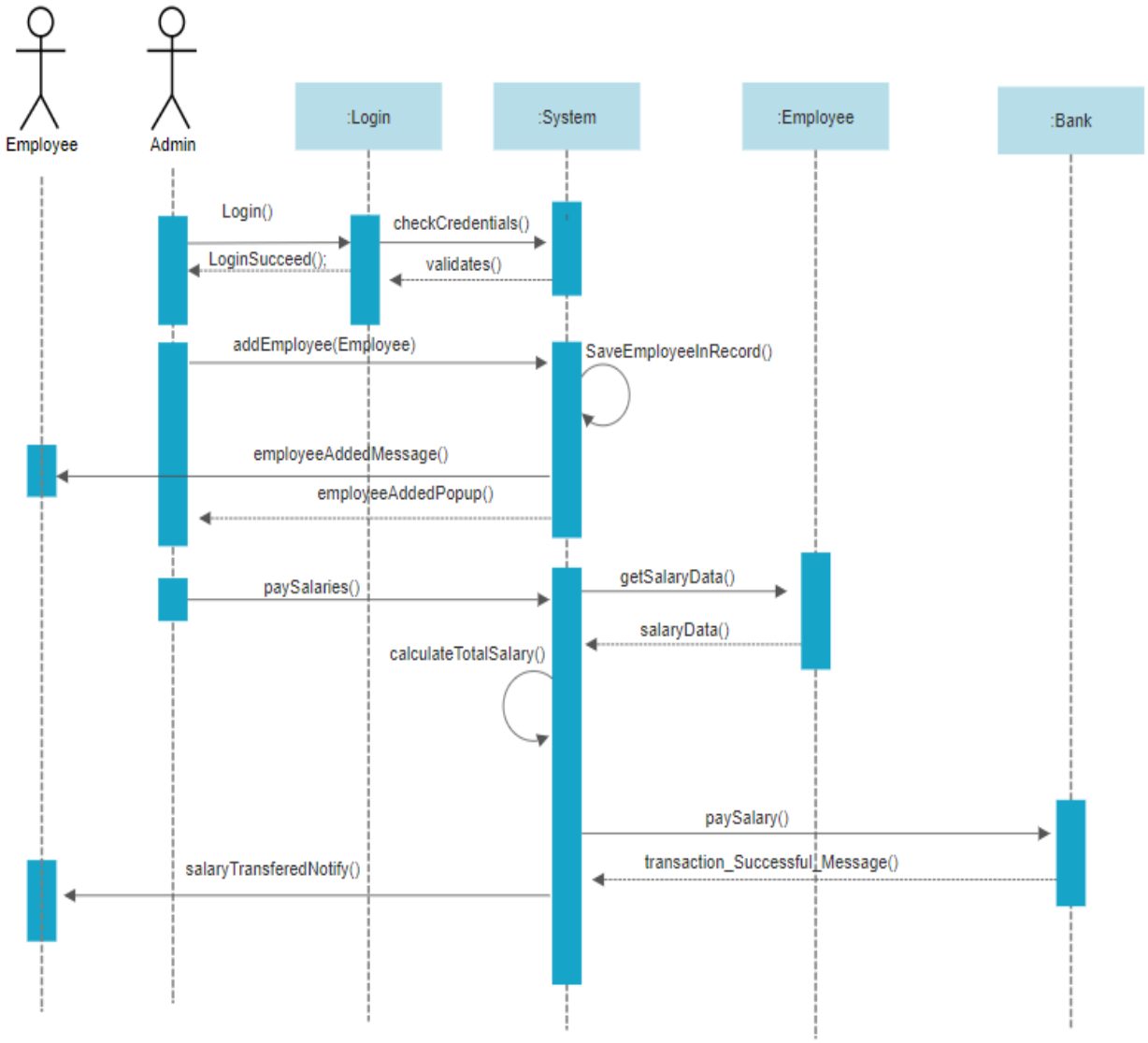


*We can have multiple actors in a Sequence Diagram

Source: https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/

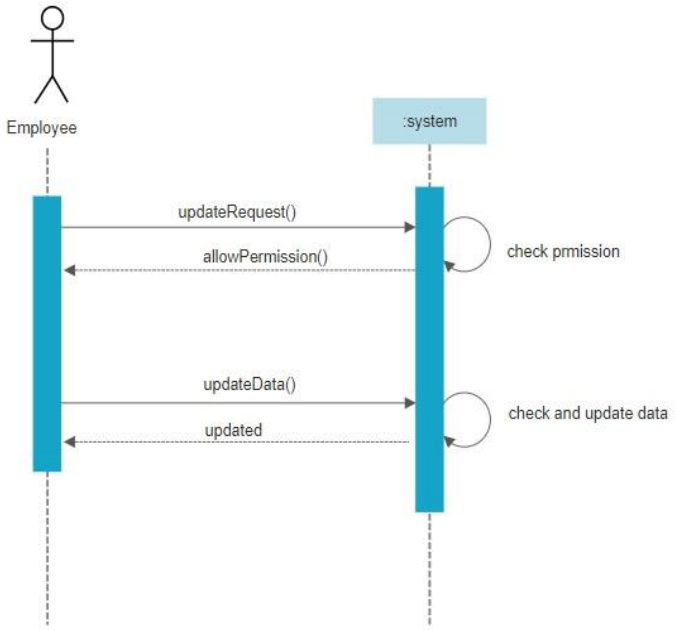
(Sequence flow of login to add employee and then paying salaries to employees among different states of admin)

- Admin will login and add employee to employee list then admin will pay salaries to all existing employee. System will calculate salary of each employee after checking employee salary data. Bills will be deducted from salary

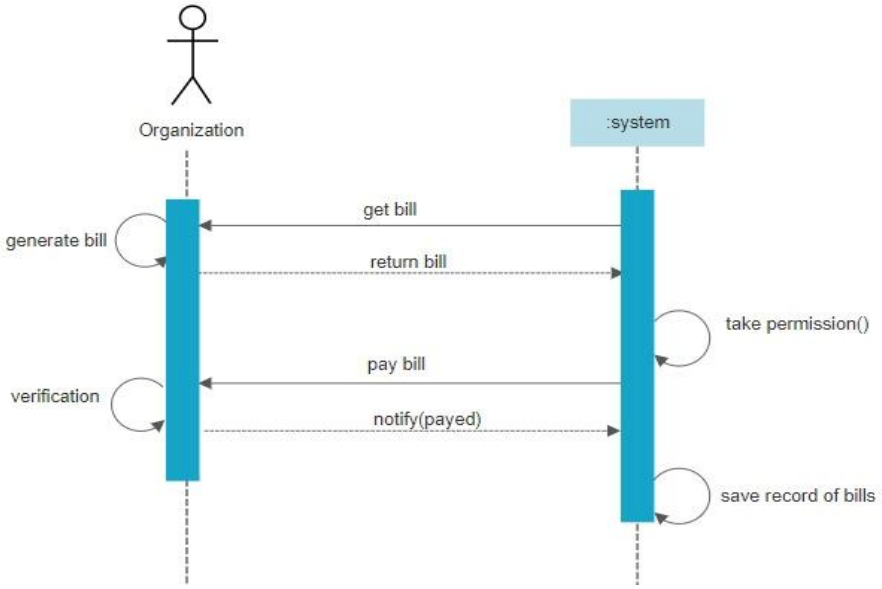


System Sequence Diagrams

(System Sequence Flow of Risk Management Use Case)



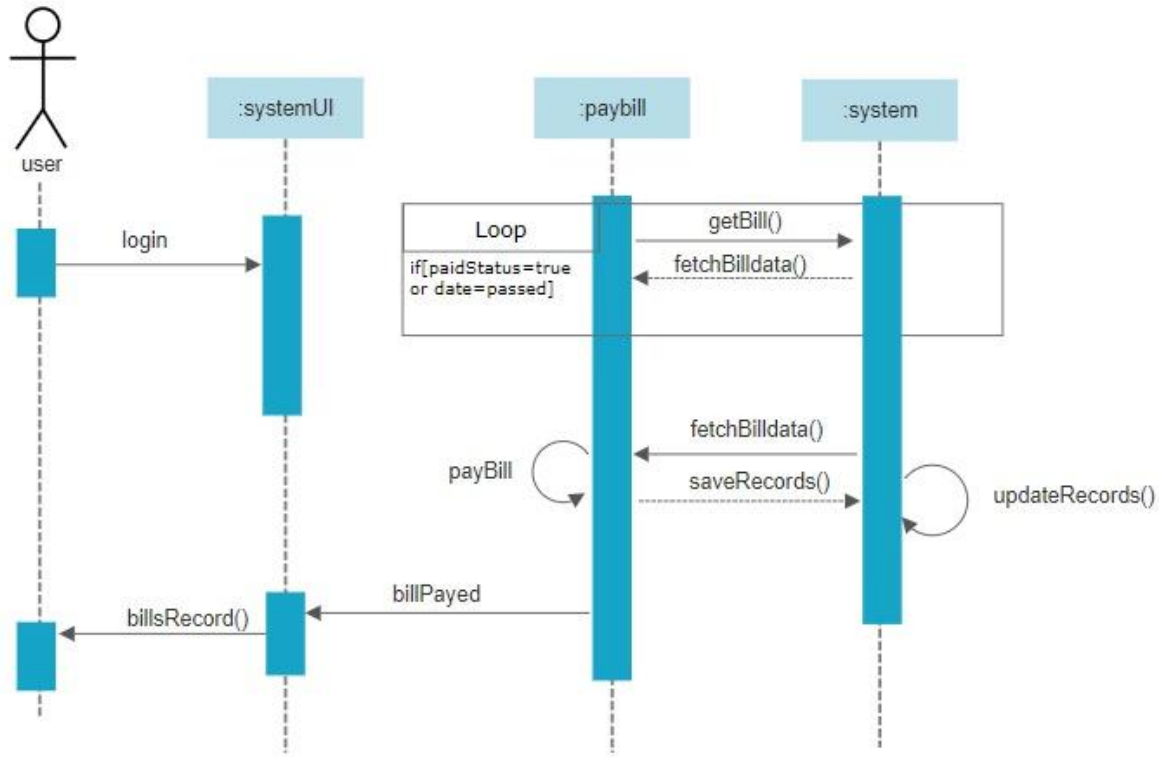
(System Sequence Flow of Business Management Use Case)



(System Sequence Diagram of Architecture)

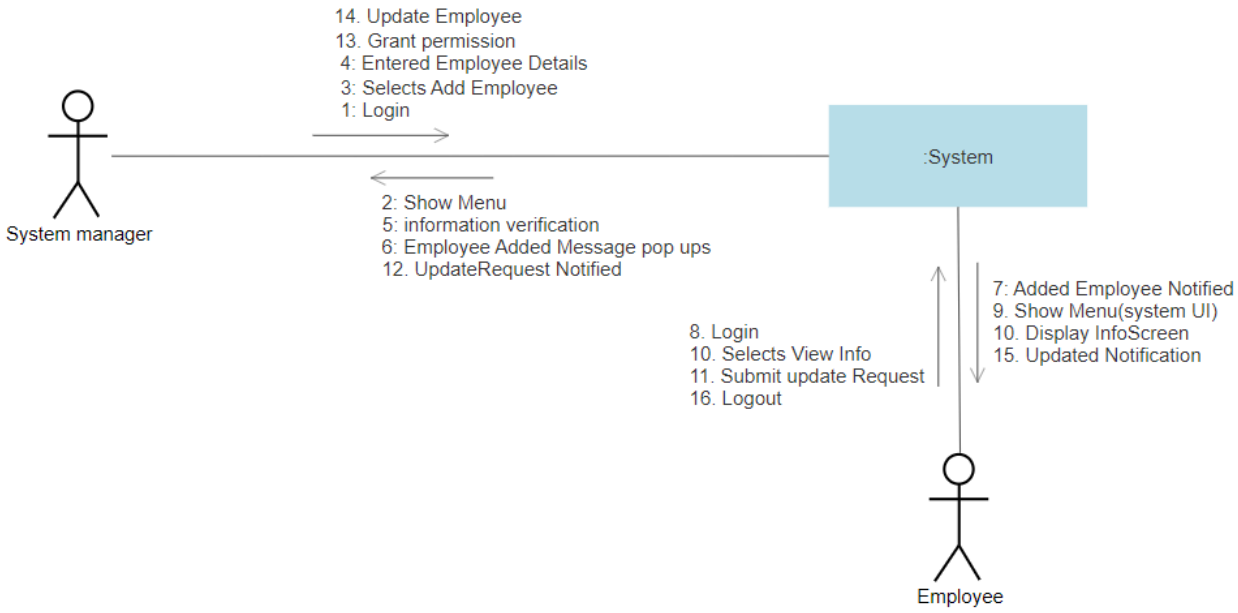
Here system is divided into three parts

- **Model**
- **View**
- **Controller**

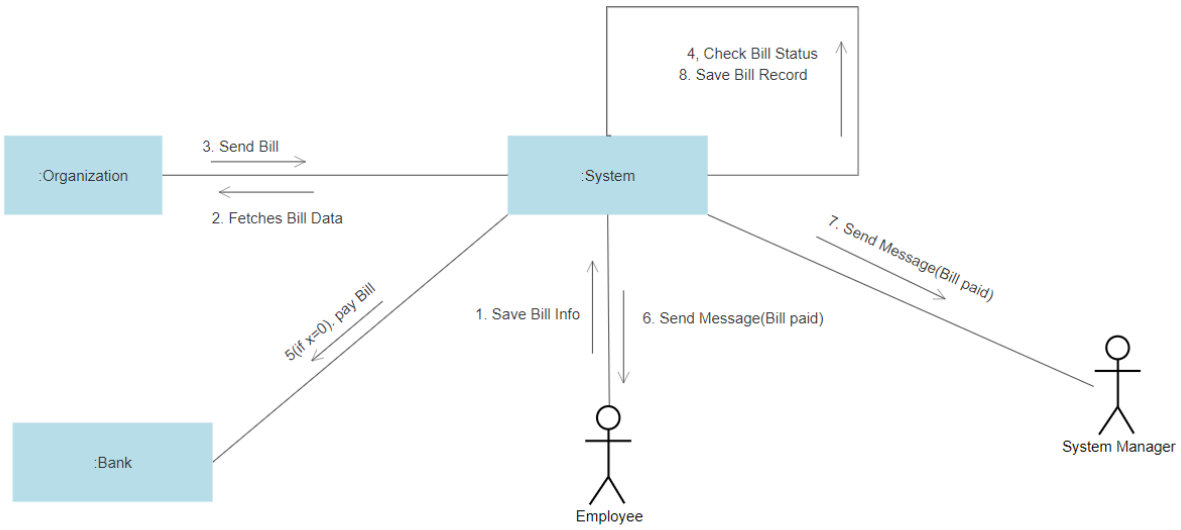


Collaboration Diagram

(Risk Use Case- Update Permission and Employee update Request)



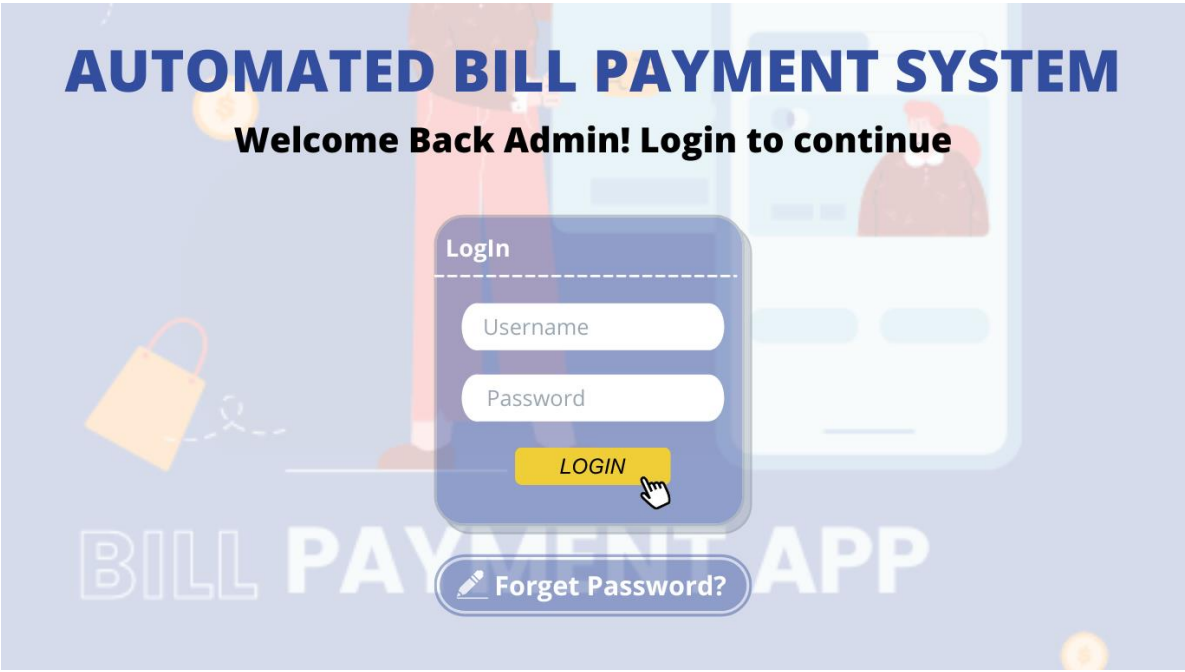
(Pay Bills and Save Record)

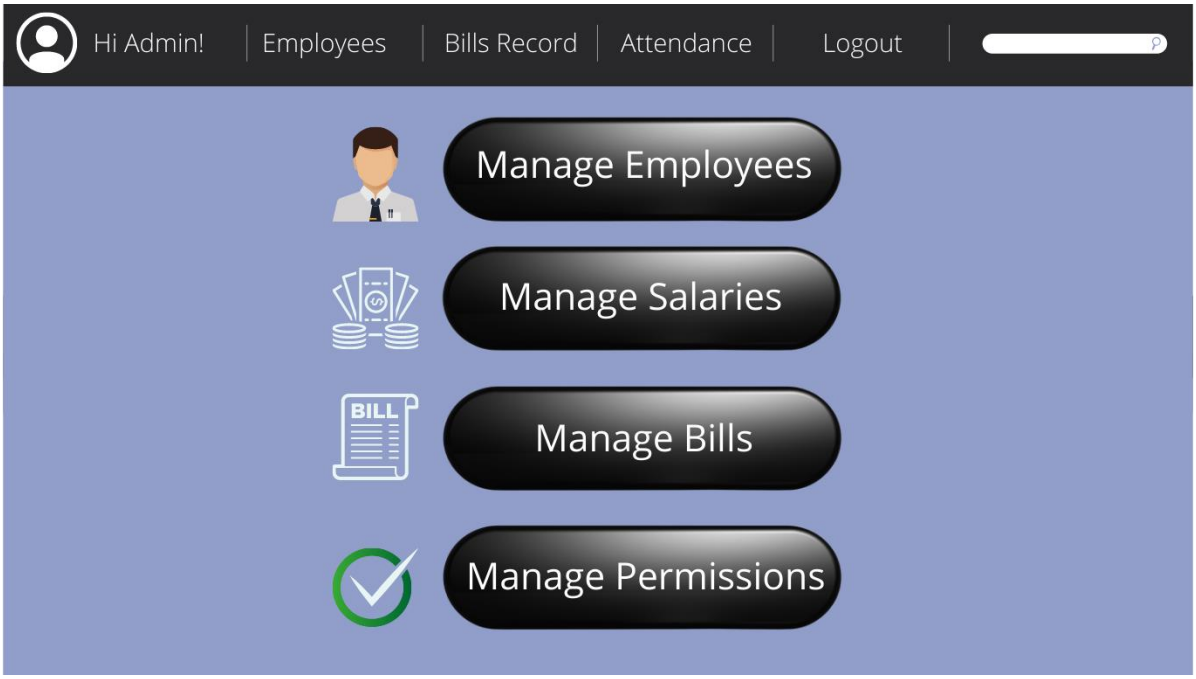


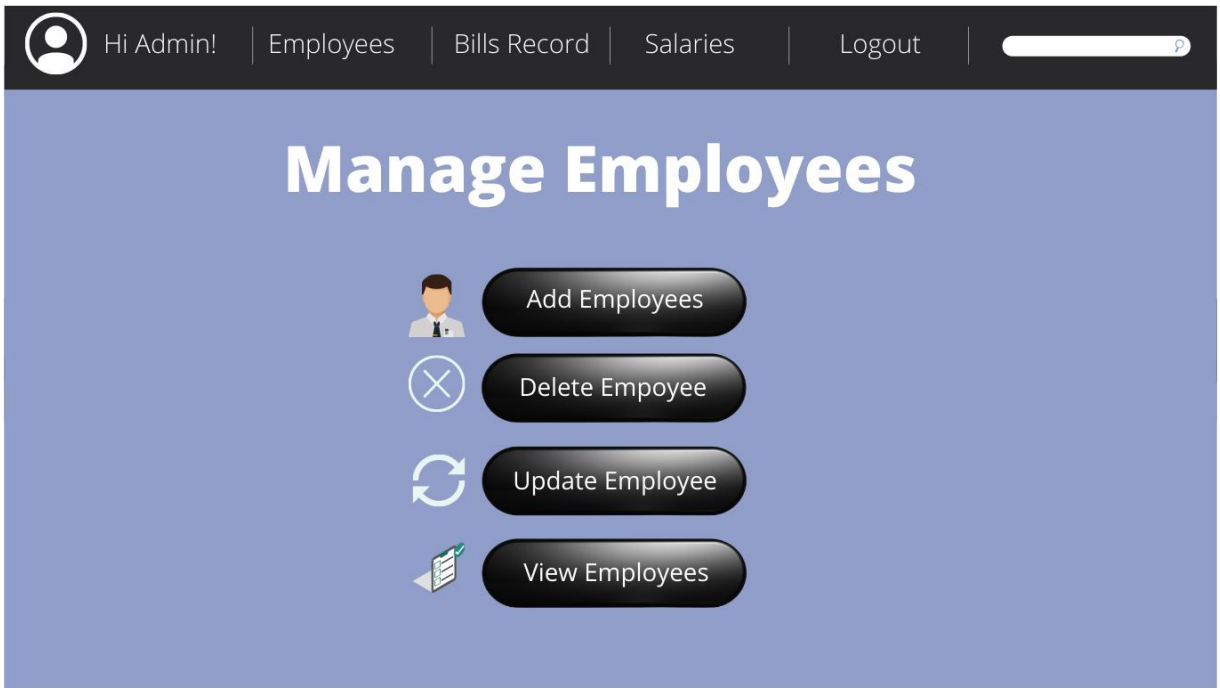
- In message 5 : x means Bill Status and Date and 0 means false

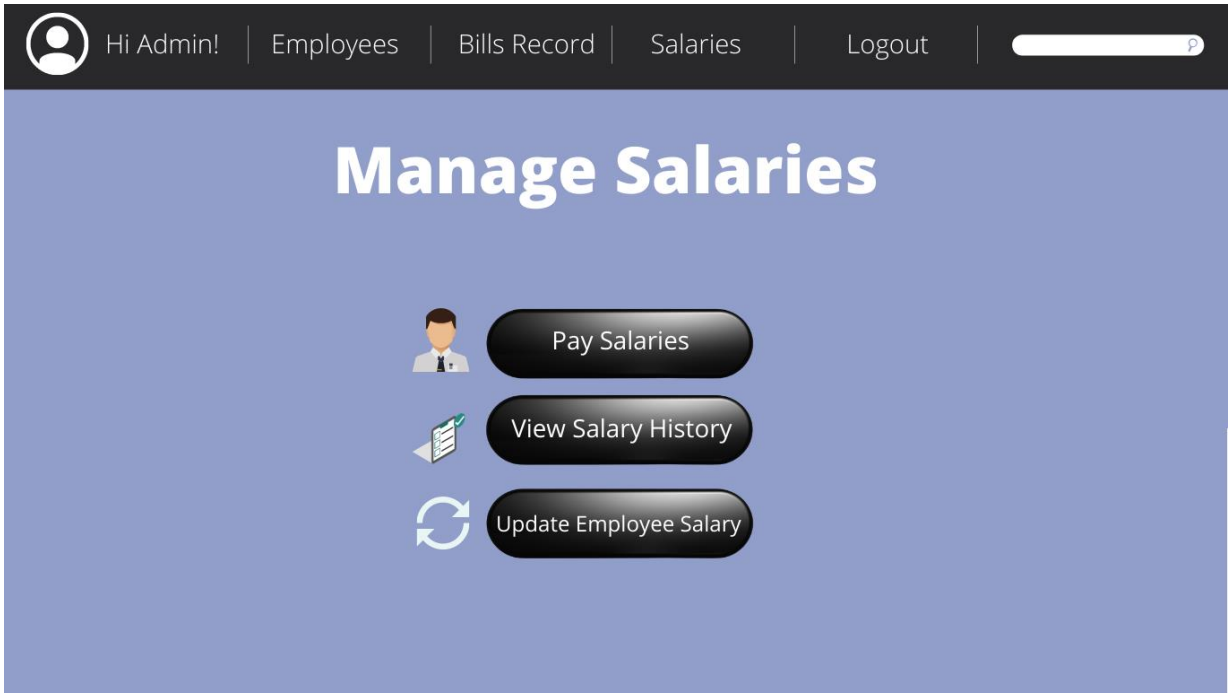
Prototype’s

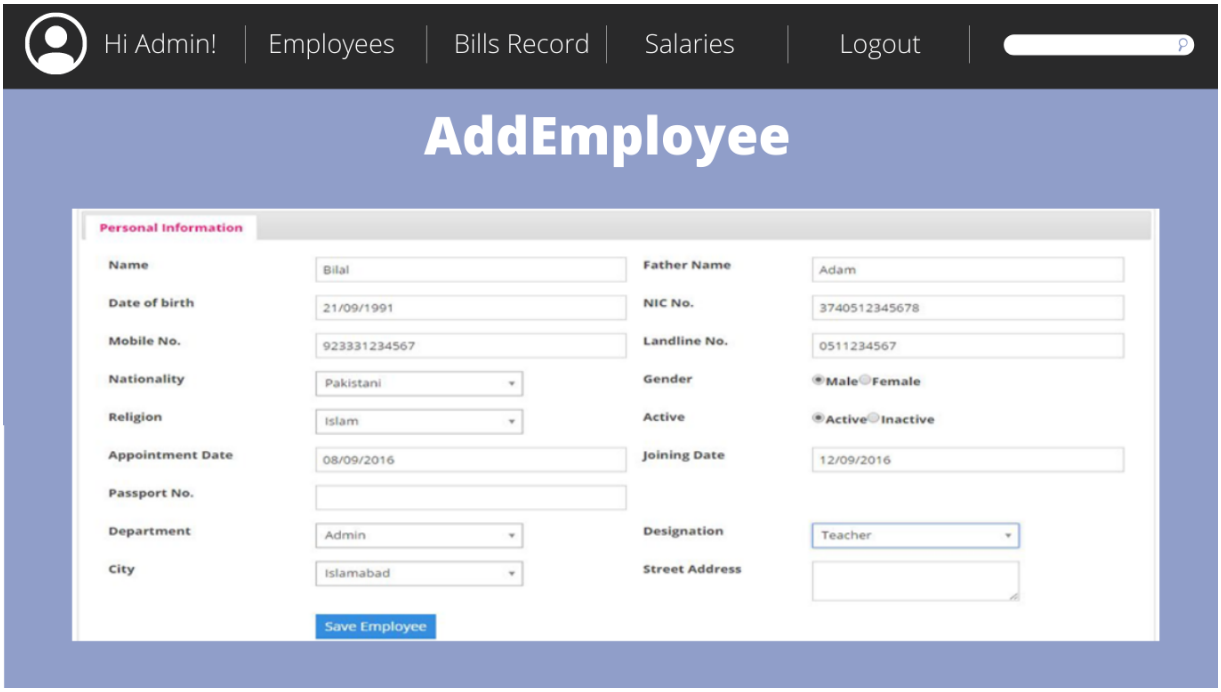
- **Admin End UI:**

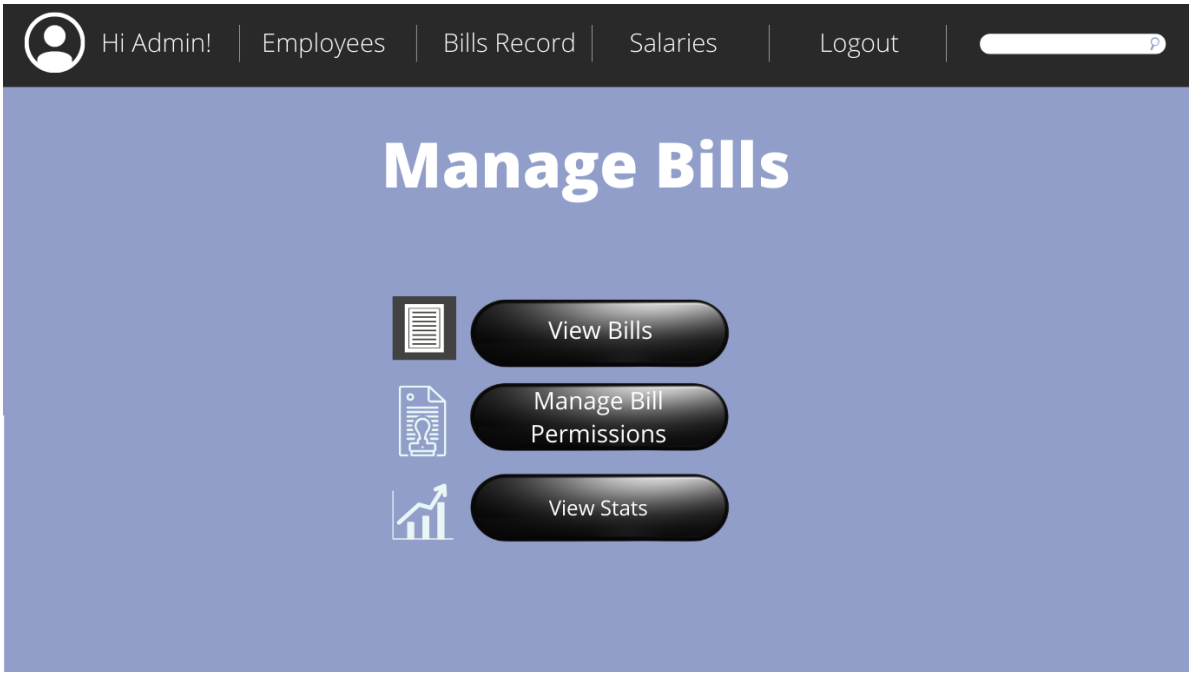


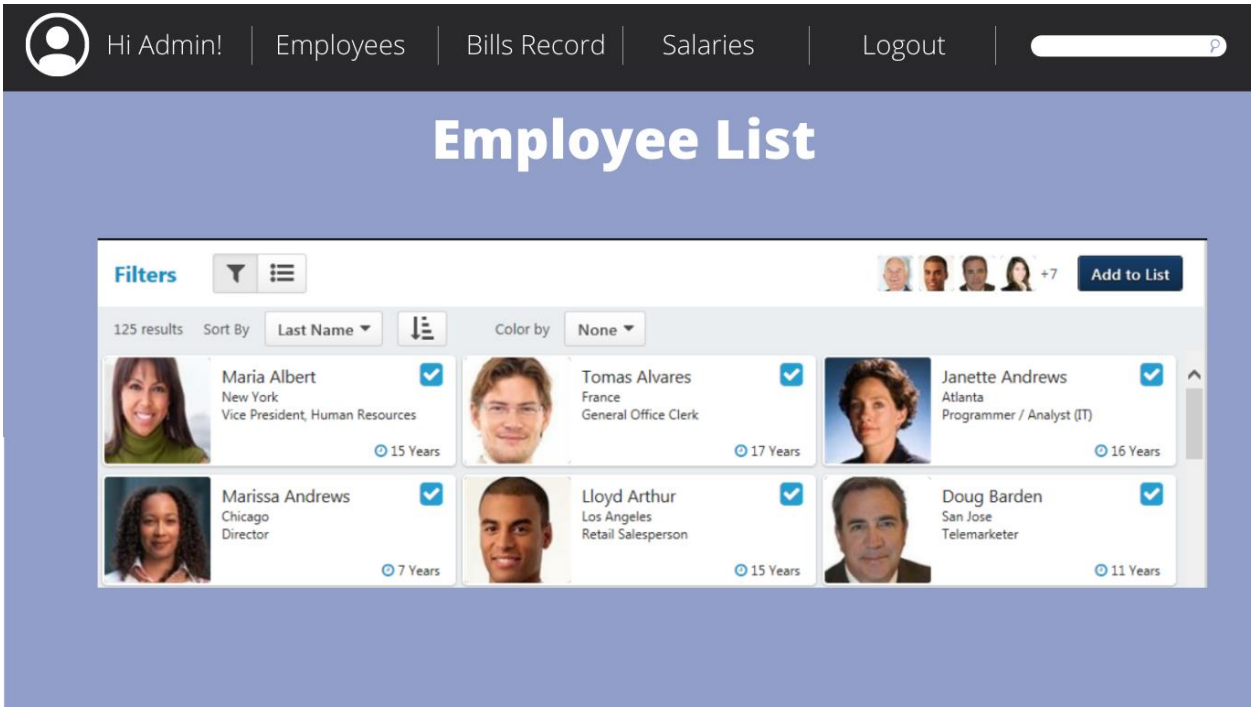



Hi Admin!
Employees
Bills Record
Salaries
Logout
Manage Employees
Add Employees
Delete Empoyee
Update Employee
View Employees

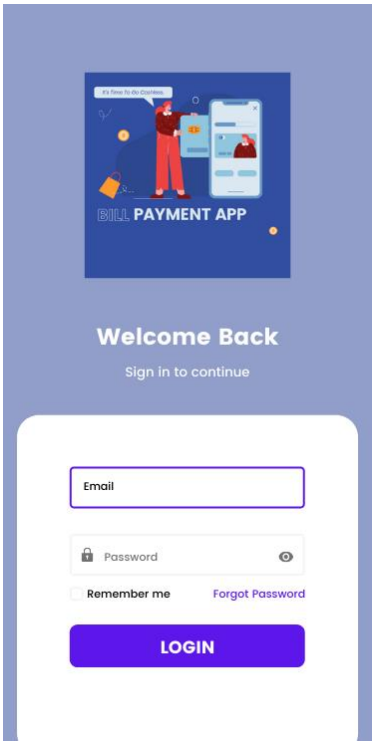

Hi Admin!
Employees
Bills Record
Salaries
Logout
Manage Salaries
Pay Salaries
View Salary History
Update Employee Salary

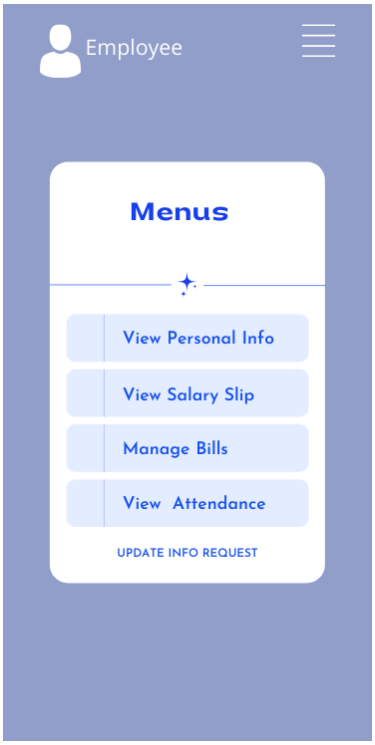

Hi Admin!
Employees
Bills Record
Salaries
Logout
AddEmployee
Personal Information
Name
Bilal
Father Name
Adam
Date of birth
21/09/1991
NIC No.
3740512345678
Mobile No.
923331234567
Landline No.
0511234567
Nationality
Pakistani
Gender
Male
Female
Religion
Islam
Active
Active
Inactive
Appointment Date
08/09/2016
Joining Date
12/09/2016
Passport No.
Department
Admin
Designation
Teacher
City
Islamabad
Street Address
Save Employee

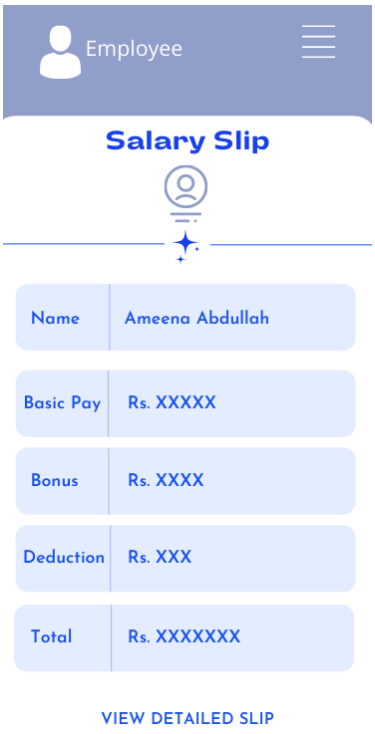

Hi Admin!
Employees
Bills Record
Salaries
Logout
Manage Bills
View Bills
Manage Bill Permissions
View Stats

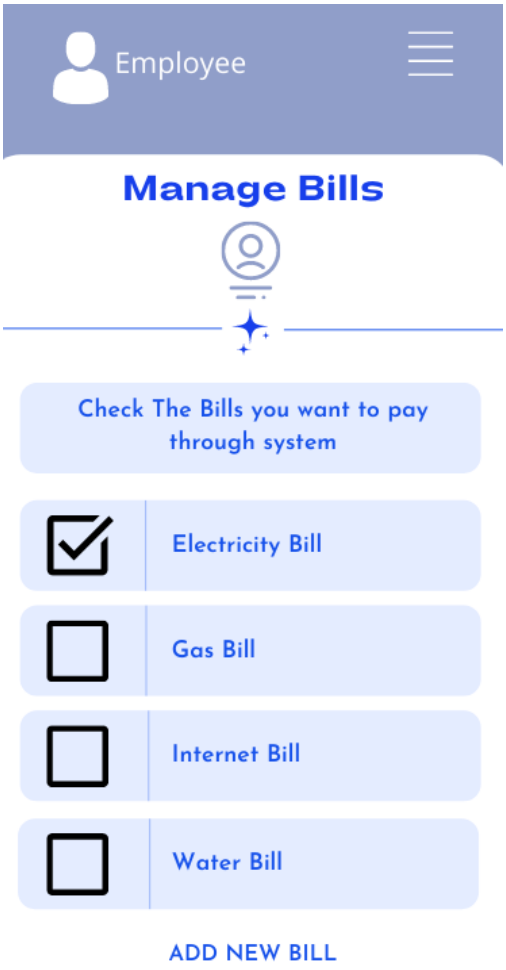

Hi Admin!
Employees
Bills Record
Salaries
Logout
Employee List
Filters
+7
Add to List
125 results
Sort By
Last Name
Color by
None
Maria Albert
New York
Vice President, Human Resources
15 Years
Tomas Alvares
France
General Office Clerk
17 Years
Janette Andrews
Atlanta
Programmer / Analyst (IT)
16 Years
Marissa Andrews
Chicago
Director
7 Years
Lloyd Arthur
Los Angeles
Retail Salesperson
15 Years
Doug Barden
San Jose
Telemarketer
11 Years

- Employee End UI:








Employee
Manage Bills
Check The Bills you want to pay through system
Electricity Bill
Gas Bill
Internet Bill
Water Bill
ADD NEW BILL

USE CASE CONTRACTS

1. **Contract #1:**

Operation:	PayBill(Bill_Id: Bill_Id, Amount: integer)
Cross References:	Use Case- Bill Payment
Pre-Condition:	The data is fetched from the billing service
Post Condition:	• PayBill instance p was created. • p is associated with BillRecord. • p.Bill_Id becomes Bill_Id. • Based on the Bill_Id, the bill is paid.

2. **Contract #2:**

Operation:	AllowPermission(GrantPermission:GrantPermission)
Cross References:	Use Case- Manage Employee Permission
Pre-Condition:	Employee has logged in and, on the panel, to either check mark or not.
Post Condition:	• An instance of Employee e is created. • Instance e is associated with System Manager. • If an employee marks yes which displays the information on UI.

3. **Contract #3:**

Operation:	LoginEmployee(Username:string, Password:string)
Cross References:	Use Case- Login
Pre-Condition:	Employee has to visit the app, and enter the login credentials.
Post Condition:	• An Employee e instance is created.

	• If the credentials match with the record, the user is logged in. • If credentials fail to match, an instance f is called that says forget password notification.

4. Contract #4

Operation:	ViewEmpInfo(DisplayInfo:DisplayInfo)
Cross References:	Use Case- View Employee Information
Pre-Condition:	User is logged as employee e into the app and able to click on the view information button.
Post Condition:	• e.DisplayInfo is called. • UI is displayed to the user with all his information.

5. Contract #5

Operation:	SendUpdateRequest(EmpInfo:Empinfo)
Cross References:	Use Case- Update Request
Pre-Condition:	Employee is logged in to the app and is on UI where he can view information.
Post Condition:	• An instance manager m is created. • Employee is associated with the system manager. • .EmployeeInfo is called. • If an employee's information needs updating, a request will be sent to the admin panel.

6. Contract #6

Operation:	UpdateInfo()
Cross References:	Update Employee Information

Pre-Condition:	Employee's information update requests have been received to the admin panel (one who has logged in as employee.
Post Condition:	• An instance employee e is created • Instance e.employeeInfo is associated with the manager. • If an update request is received, the admin updates the employees information.

7. **Contract #7**

Operation:	PaySalary(EmployeeSalary:EmployeeSalary, Amount: Integer)
Cross References:	Use Case- Salary Details
Pre-Condition:	Admin is logged in to the system to pay salary to the employees.
Post Condition:	• Instance salary p is already created. • A function that calculates the net salary is called. • p.EmployeeSalary becomes EmployeeSalary • Net salary is transferred to the employee's bank.

8. **Contract #8**

Operation:	ViewAttendance(Attendances:Attendances, Absents:Integer, NoOfDays: Integer)
Cross References:	Use Case- View Attendance
Pre-Condition:	Employee is logged in to the system and clicks on View Attendance button to see his attendance record.
Post Condition:	• An instance employee e is created. • e.Attendance is associate with Attendances

	• Attendance record is displayed.

9. Contract #9

Operation:	MarkLeave(Leave:Leave, No_ofAttendances:Integer)
Cross References:	Use Case- Attendance System
Pre-Condition:	Employee is on panel/page to mark leave.
Post Condition:	• An instance employee e is already created. • It is associated with Leave. • A request is sent to the admin to either mark leave or decline.

10. Contract #9

Operation:	BillRecords()
Cross References:	Use Case- update Bill Records
Pre-Condition:	Manager is logged in to the system and clicks on the update bill Records button to update records.
Post Condition:	• An instance employee e is created. • e.updatebill is associated with billRecord . • Updated record stored in system.

References

- https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/
- https://creately.com/blog/diagrams/sequence-diagram-tutorial/
- https://searchsoftwarequality.techtarget.com/definition/collaboration-diagram#:~:text=A%20collaboration%20diagram%2C%20also%20known,the%20role%20of%20each%20object.