Table of Contents

1. EXECUTIVE SUMMARY

‘PayIt - Payroll Management System’ is designed to make the existing manual system-automatic, by using a computerized and full-edged computer software, so that all the valuable data and information can be stored for a longer period with easy access and manipulation. This application can maintain and view computerized records without getting redundant entries. The project helps to manage employee data providing better services for the client.

It has been developed for organizations, keeping in view the requirements of companies to issue salaries. The application helps organizations to compute the salaries of their employees. The employee details and their salary are handled by the administrator whereas, the users can view their salary details as well as they are provided with the options to submit their feedbacks to the admin.

Users of the System

- ADMINISTRATOR
- USERS (EMPLOYEES)

Main Functions

- Admin can manage all the employees.
- Admin can manage employee’s salary bonus.
- Admin can manage employee’s attendance.
- Admin can manage employee’s salary and generate pay slips.
- Admin can view Feedbacks submitted by the employees.
- Admin can generate various reports based on different criteria.
- Admin can see an overview of employees at different positions and total salary and bonus paid.

- Users can view their monthly salary details.
- Users can send their suggestions or feedback to the admin.
- Users can view and edit their profile.
- Users can change their password.
- User can see an overview the total salary and bonus received.

2. BACKGROUND

2.1. Existing System

The organizations maintain their most of their day-to-day data the designated employees in the registers. The whole work is done manually. It is very time consuming and may not be error free. Payroll mistakes can happen faster than you think. For employees, their only source of income is their monthly salary. If the salary is not paid accurately or if there is a delay in releasing salary creates chaos. Such irregularities can take a toll on the morale of the employees and ultimately affect the business productivity.

If the data is not properly tracked or recorded, then there are chances of misunderstanding between the organization and the employees. To solve this problem, the payroll management system database project plays a major role.

2.2. Definition of Problem

1. Storing the data manually is difficult and time-consuming.
2. Manually stored data may contain errors or invalid data.
3. May affect the business productivity.

2.3. Proposed System

PayIt – The Payroll Management System is a simple application that aims to save time, make the system cost effective and management records efficiently. It can effectively maintain and view computerised records without getting redundant entries.

It provides admin to manage employees, issue monthly salary to the employees and generate payslips. It also helps to generate various employee reports on the basis of their salary, joined date details and many more. The user could view their issued salary and their profile. Users are also provided with a facility to send their valuable feedbacks or suggestions to the admins. The admins can also view all the feedbacks submitted by the employees and make necessary changes to make their workplace a better one.

Advantages of Proposed System:

- **It is user friendly: -** PayIt – The Payroll Management System is designed in such a way that every admin and employees can get access to the application and all the information can be stored and retrieved in an easy manner.
- **It provides more efficiency and flexibility: -** The application is user-friendly. So, all the admin and employees can login to the application.
- **24 hours accessibility:** - So, it can be accessed 24 hours anywhere around the globe.
- **Provides information on employee's profile :** It provides a detailed view of all the employees with salary, position details and many more.
- **Accurate salary processing is possible :** Calculations are automated so it is highly accurate. It also helps to maintain the monthly allowances, basic pay, gross salary, etc.
- **Ensures Security :** It also implements security measures and confidentiality as only the admins and registered employees are able to login with their unique username and password.
- **It is very interactive and saves time.**
- **Reduces paper works.**

3. PROJECT OVERVIEW

3.1. Objective of the project

The objective of the project was to computerize the payroll system which was earlier handled manually where every precaution has been taken in each process involved in the complex salary computation. By computerization, it implies all the processes are entirely handled by the computer. One of the most important facilities that the application provides is report generation and printing which provides users with hard copies of the data also. This framework enables its users to save time, make the system cost effective and management records efficiently. The employee details, their attendance and their salary are handled by the administrator whereas, the users can view their salary details as well as they are provided with the options to submit their valuable feedbacks or suggestions to the admin.

3.2. Stakeholders

- Admin

Admin is a responsible for operating the whole system. The Admin can log in using a valid username and password. Admin can add and manage the Employees, manage employee attendance, issue monthly salary, and generate pay slips for the employees, view submitted feedbacks.

- Users (Employees)

Employees are user who can log in to the application by using a valid username and password. They can view their monthly salary, view and edit their profile and submit feedbacks to the admin.

3.3. Scope of project

The scope of the project is identified at end of the initial investigation. The project deals in issuing the salary for all the employees. Data updates are also done by the admin.

3.4. Feasibility Analysis

3.4.1 Feasibility study

Every project is feasible for given unlimited resources and infinitive time. Feasibility study is an evaluation of the proposed system regarding its workability, impact on the organization, ability tomeet the user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development. Here the resources availability and requirements are said to feasible to create the proposed system.

3.4.2. Technical Feasibility

Technical feasibility assesses whether the current technical resources are sufficient for the new system. If they are not available, can they be upgraded to provide the level of technology necessary for the new system? It checks whether the proposed system can be implemented in the present system without supporting the existing hardware. Currently,

- Technology exists to develop a system.
- The proposed system can hold data to be used.
- The proposed system can provide adequate response.Hence, we can say that the proposed system is technically feasible.

3.4.3. Operational Feasibility

Operational feasibility determines if the human resources are available to operate the system onceit has been installed. The resources that are required to implement or install are already available with the Breakdown Assist. The persons of this Assist need no exposure to computer but have to be trained to use this particular software. The project is optimally feasible.

3.4.4. Schedule Feasibility

An evaluation of the time needed for the development of this project. The time schedule requiredfor the development of this project is very important, since more development time effects machinetime, costs and delays in the development of the other systems. So the project should be completewithin affixed schedule time as far as this is concerned.

Schedule feasibility study for the design is shown below

Problem identification	5
Requirement analysis	10
Overall design	20
Construction	22
Testing	15

3.4.5. Economic Feasibility

Economic feasibility determines whether the time and money are available to develop the system.It also includes the purchase of new equipment, hardware and software. Since software product must be cost effective in the development, on maintenance and in the use. It is affordable to allocatethe required resources.

4. OVERALL PROJECT PLANNING

4.1. Development Environment

Hardware Specifications

- Intel i3 or above
- Memory: at least 4GB
- Display: Color monitor
- Keyboard: Windows Compatible
- Mouse: Windows Compatible

Software Specifications

Technology used:

i. Server side

- Front end : C# .net
- IDE : Visual Studio 2022
- Back end : SQL server
- Operating System : Windows

4.2. Constraints

The set of constraints that we come across this system is as follows:

- User Interface is only in English i.e.no other language option is available.
- Admin can login with his assigned username and password i.e. no guest facility is available.

4.3. Deliverables

List of documents that shall be delivered are User Manual

- System maintenance documentation.
- Application archive with source code.
- Database backup and DDL script.
- Complete source code.

4.4. Assumptions and Dependencies

a) Assumptions

- All roles are created in the system already but further registration of users on given roles can be done.
- Roles and tasks are predefined and are made known to the administrator.
- The code should be free of compilation errors/syntax errors.
- The product must have an interface which is simple enough to understand.
- Roles and tasks are predefined and are made known to the administrator.
- End users should have basic knowledge of computer.

b) Dependencies

- All necessary hardware and software are available for implementing and use of the tool.
- All roles are created in the system already.
- The proposed system should be designed, developed and implemented based on the software requirements specifications document.

4.5. Risks

Some of the risks are follows

- Database crash will cause heavy data loss
- Wrong input will cause discrepancies in data
- Availability of the network.

4.6. Process Model

The process model for developing the project is waterfall model

The phases are: -

- Requirement analysis
- System study
- Designing
- Coding
- Testing
- Maintenances

4.7. Test Strategy

4.7.1. System Testing

When a system is developed, it is hoped that it performs properly. In practice however some errors always occur. The main purpose of testing and information system is to find the errors and correct them. A successful test is one which finds an error. The main objectives of system testing are:

- To ensure during operation the system will perform as per specifications.
- To make sure that the system meets the requirements during operation.
- To verify that the controls incorporated in the system function as intended.
- To see that when correct inputs are fed to the system the outputs are correct.
- To make sure that during operation incorrect input and output will be deleted.

The scope of a system test should include both manual operations and computerized. Operation system testing is a comprehensive evaluation of the programs, manual procedures, computer operations and controls. System testing is the process of checking if the developed system is working according to the original objectives and requirements. All testing needs to be conducted in accordance with the test conditions specified earlier.

4.7.2 Types of testing

Unit Testing

Unit Testing will be done to test field validations, navigation, functionality of the programs and its block. These tests are applied on various functions within each program and other critical program blocks.

Module Testing

Module Testing will be each program done to test the interaction between the various programs within one module. It checks the functionality of with relation to other programs within the same module. It then tests the overall functionality of each module.

Integration Testing

The major concerns of integration testing are developing an incremental strategy that will limit the complexity of entire actions among components as they are added to the system. Developing a component as they are added to the system, developing an implementation and integration schedules that will make the modules available when needed, and designing test cases that will demonstrate the viability of the evolving system. Though each program works individually they should work after linking them together. This is also referred to as interfacing. Data may be lost across interface and one module can have adverse effect on another. Subroutines after linking may not do the desired function expected by the main routine. Integration testing is a systematic technique for constructing program structure while at the same time conducting tests to uncover errors associated with the interface. In the testing, the programs are constructed and tested in small segments.

Validation Testing

This provides the final assurance that the software meets all the functional, behavioral and performance requirements. The software is completely assembled as a package. Validation succeeds when the software functions in a manner in which user wishes. Validation refers to the process of using software in live environment in order to find errors. During the course of validation, the system failure may occur and sometime the coding has to be hanged according to

the requirement. Thus the feedback from the validation phase generally produces changes in the software. Once the application was made of all logical and interface errors, inputting dummy data ensure that the software developed satisfied all the requirements of the user. The dummy data is known as test cases.

Output Testing

After performing the validation testing, the next step is output testing of the proposed system since no system could be useful if it does not produce the required output in the specific format. Asking the users about the format of output they required, tests the output generated in two ways. One is on screen and another is printed format. The output format on the screen found to be correct as the format was designed in the system design phase according to the user needs. For the hard copyalso, the output comes out as the specified requirement by the user. Hence output testing does notresult in any correction in the system.

Acceptance Testing

Acceptance testing (also known as user acceptance testing) is a type of testing carried out in order to verify if the product is developed as per the standards and specified criteria and meets all the requirements specified by customer. This type of testing is generally carried out by a user/customer where the product is developed externally by another party. Acceptance testing falls under black box testing methodology where the user is not very much interested in internal working/coding of the system, but evaluates the overall functioning of the system and compares it with the requirements specified by them. User acceptance testing is considered to be one of the most important testing by user before the system is finally delivered or handled over to the end user. Acceptance testing is also known as validation testing, final testing, QA testing, factory acceptance testing and application testing etc. And in software engineering, acceptance testing may be carried out at two different levels; one at the system provider level and another at the end user level (hence called user acceptance testing, field acceptance testing or end-user testing). Acceptance test refers to the acceptance of data into the system for processing. The acceptance test contributes to the consistency and smooth working of the system. The system under consideration is tested for users at a time for developing and making changes whenever required.

4.8. Testing environment and tools

The hardware specification used for testing:

Operating system	Windows 11
Memory	8 GB
Hard Disk	512 GB

The software specification used for testing:

Front End	C# .net
Back End	SQL server
Operating System	Windows 11

5. ITERATION PLANNING

5.1. Schedule

SERIAL NO.	TASK	DURATION
1	Problem identification	5 days
2	Requirement Specification	10 days
3	Database Design and Analysis	12 days
4	Design Analysis	9 days
5	Coding	24 days
6	Testing	10 days
	Total	70 days

5.2. Risk

- Wrong input
- Software installation issues.
- Database crash will cause heavy data loss

6. HIGH LEVEL SYSTEM ANALYSIS

6.1. User Characteristics

All users of the system are expected to have basic knowledge of using a computer and basic knowledge in English language.

Users of the system:

- Admin
- Employee

6.2. Summary of system features/Functional requirements

Manage Employees

Admin can add, update, view and delete all the employees and their details.

Manage Salary Bonus

Admin can add, update, view and delete all the employee's salary bonus.

Manage Attendance

Admin can add, update, view and delete all the employee's attendance.

Issue Salary and Generate Payslips

Admin can add, update, view and delete all the employee's salary and generate pay slips.

View Feedbacks

Admin can view Feedbacks submitted by the employees.

Generate Reports

Admin can generate various reports based on different criteria.

View Salary

Users can view their monthly salary details.

Edit Profile

Users can view and edit their profile, change password.

Give Feedback of the service

The user can give valuable suggestions or feedback to the admin based on the service provided.

6.3. Non Functional Requirements/Supplementary Specifications

The non-functional requirements which define the system performance are:

Accuracy:

The level of accuracy in the proposed system will be high. All operations would be done correctly and it ensures that whatever information that comes from the center is accurate.

Reliability:

The reliability of the proposed system will be high. The reason for the increased reliability of the system is that system uses correct formulas to calculate the results.

Immediate Response:

The system is highly responsive because it uses well accurate formulas to calculate required results provided the user should enter the valid input data.

Easy to Operate:

The system should be easy to operate and should be such that it can be developed within a short period of time and fit in the limited budget of the user.

The other non-functional requirements are:

- Security
- Maintainability
- Extensibility
- Reusability
- Resource utilizations

6.4. Glossary

Admin	Administrator
Employee	Appointed by Admin

6.5. Business Rules

The information should be correct and valid.

6.6. Use Case

Login Page

Admin and Employees can log in to the system using their unique username and password.

Admin Dashboard

It displays the count of employees at different positions(Manager, Senior, Junior), the total salary and total bonus paid to all the employees.

Employee Dashboard

It displays the total salary and total bonus paid to the employee.

Manage Employees

Admin can add, update, view and delete all the employees and their details.

Manage Salary Bonus

Admin can add, update, view and delete all the employee’s salary bonus.

Manage Attendance

Admin can add, update, view and delete all the employee’s attendance.

Issue Salary and Generate Payslips

Admin can add, update, view and delete all the employee’s salary and generate pay slips.

View Profile

Users can view and edit their profile. Users also have options to change their password.

View Salary

Users can view their monthly salary details.

Give Feedback of the service

The user can give valuable suggestions or feedback to the admin based on the service provided.

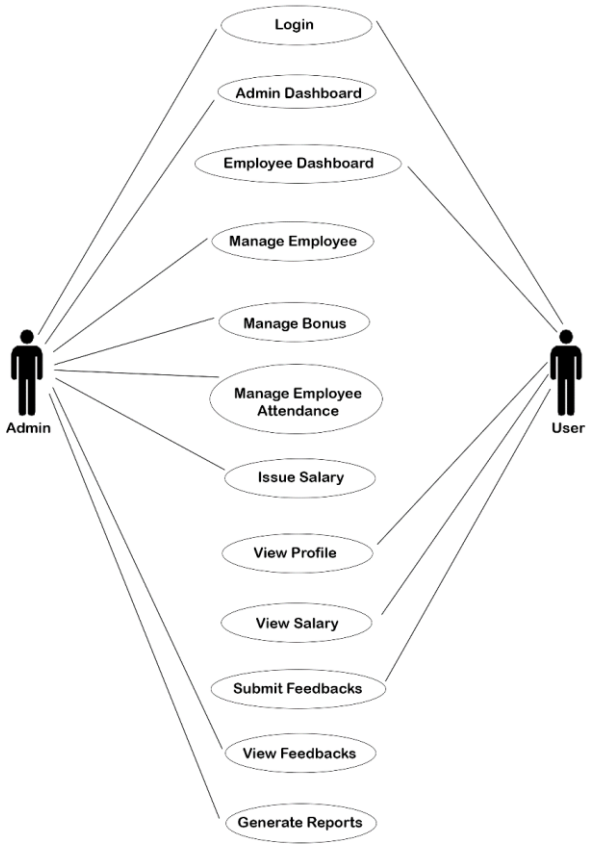
View Feedbacks

Admin can view Feedbacks submitted by the employees.

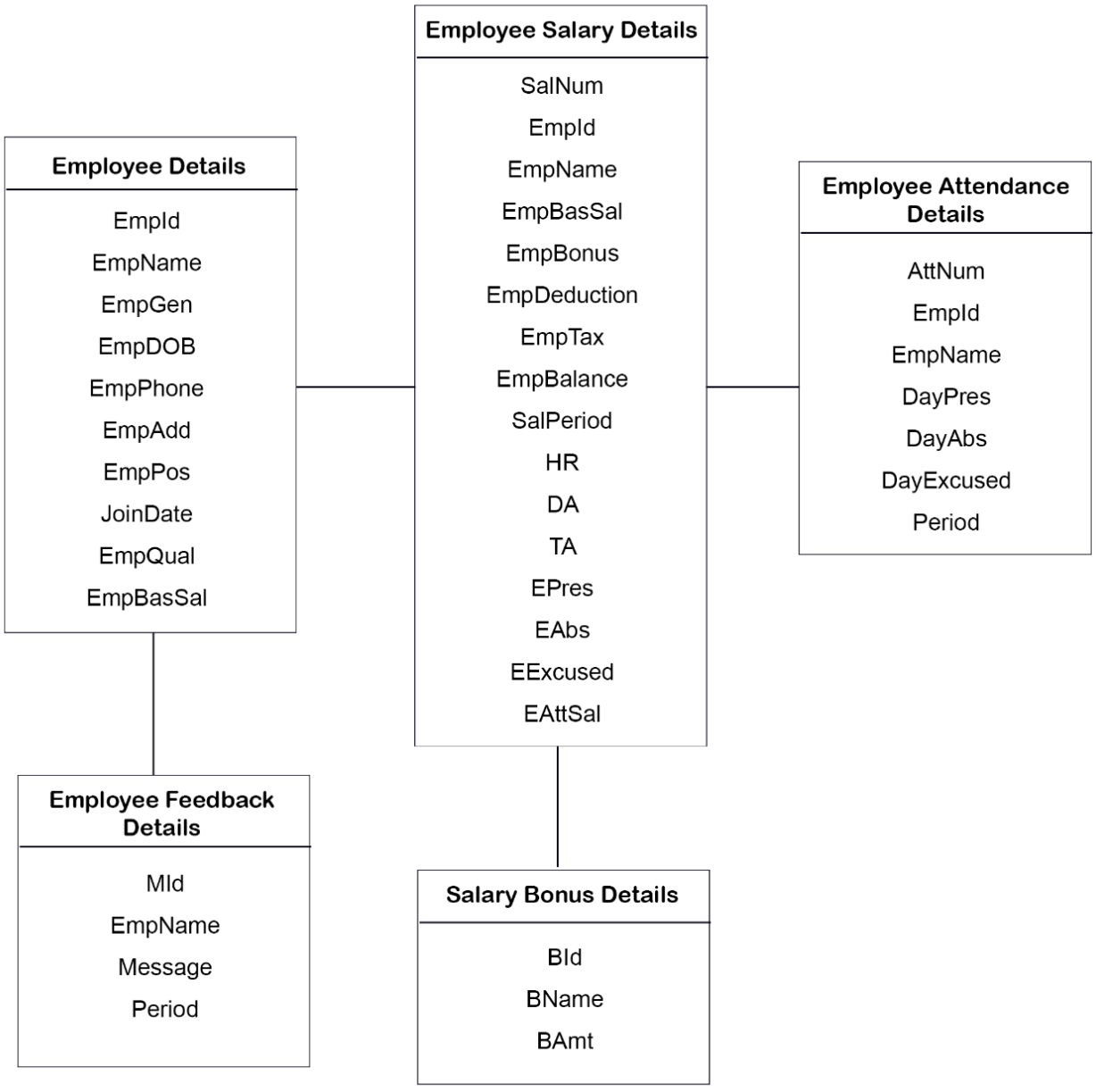
Generate Reports

Admin can generate various reports based on different criteria.

6.7. Use-case Diagram



7. DOMAIN MODEL



8. USE CASE MODEL

8.1 Use case text

Scope: Payroll Management

Primary Actor: Admin

Stakeholders and Interests:

- **Users**: Could Login and view their monthly salary, submit feedbacks to the admin and view and edit their profile.
- **Administrator:** Admin is a responsible for operating whole system. The admin gets logged in by valid username and password. Admin can add and manage Employees, manage Employee Attendance, manage Employee Salary and generate Payslips, View Feedbacks from employees and generate reports.

Preconditions:

No user can use the system without logging into the system.

Success Guarantee (Post conditions):

Really easy to use the system.

Main Success Scenario:

Use Case: Login

1. System prompts user for entering account information.
2. User enters account information.
3. System validates the account information based on the information stored in the database.
4. System redirects to user/admin account.

Use Case: Admin Dashboard

1. System redirects to admin's dashboard page.
2. Admin views details regarding the total bonus and total salary issued, the total number of employees at various positions.

Use Case: User/Employee Dashboard

1. System redirects to employee's dashboard page.
2. The user views details regarding the total bonus and total salary recieved.

Use Case: Admin - Manage Employee

1. System redirects to admin's employee page.
2. The admin views all the employees and their details.
3. System prompts admin for entering employee information.
4. Admin enters employee information and clicks save button.
    a. System validates new account information and if there's no issue, the entered data is stored in the database.
5. Admin clicks on the employee's data from the table and enters updated employee information and clicks update button.
    a. System validates the account information and if there's no issue, the entered data is updated in the database.
6. Admin clicks on employee's data from the table and clicks the delete button.
    a. System validates the account information and if there's no issue, the data is deleted from the database.

Use Case: Admin - Manage Employee Bonus

1. System redirects to admin's employee bonus page.
2. The admin views all the bonus and their details.
3. System prompts admin for entering bonus information.
4. Admin enters bonus information and clicks save button.
    a. System validates new bonus information and if there's no issue, the entered data is stored in the database.
5. Admin clicks on the bonus data from the table and enters updated bonus information and clicks update button.
    a. System validates the account information and if there's no issue, the entered data is updated in the database.

6. Admin clicks on bonus data from the table and clicks the delete button.
    a. System validates the bonus information and if there's no issue, the data is deleted from the database.

Use Case: Admin - Manage Employee Attendance

1. System redirects to admin's employee attendance page.
2. The admin views all the attendance and their details.
3. System prompts admin for entering attendance information.
4. Admin enters attendance information and clicks save button.
    a. System validates new attendance information and if there's no issue, the entered data is stored in the database.
5. Admin clicks on the attendance data from the table and enters updated attendance information and clicks update button.
    a. System validates the account information and if there's no issue, the entered data is updated in the database.
6. Admin clicks on attendance data from the table and clicks the delete button.
    a. System validates the attendance information and if there's no issue, the data is deleted from the database.

Use Case: Admin – Issue Salary

1. System redirects to admin's employee salary page.
2. The admin views all the employee's salary and their details.
3. System prompts admin for entering salary information.
4. Admin after selecting an employee from the dropdown list, enters salary information and clicks compute button.
    a. System validates new salary information and if there's no issue, the calculated data are displayed.
5. Admin after computing salary, the save button is clicked.
    a. System validates the salary information and if there's no issue, the entered data is stored in the database.

Use Case: User- View Profile

1. System redirects to user's profile page.
2. User views all their personal details.
3. For editing their personal information, the user can click the edit button.
    a. System redirects to user's profile editing page.
    b. System prompts the user for entering details to be updated.
    c. Users enters the new details and clicks save button.
        i. System validates the entered information and if there's no issue, the details are updated in the database.
    d. After the user clicks view profile page, the system redirects to the employee's profile page.
4. For changing their password, the user can click the change password button.
    a. System redirects to user's change password page.
    b. System prompts the user for entering new password.
    c. User enters the new password and clicks save button.
        i. System validates the entered information and if there's no issue, the password is updated in the database.

Use Case: User- View Salary

1. System redirects to employee's dashboard page.
2. The user views details regarding the salary received.

Use Case: User- Submit Feedback

1. System redirects to employee's feedback page.
2. System prompts the user for entering their valuable suggestions or feedback.
3. User enters feedback and clicks submit button.
    a. System validates the entered information and if there's no issue, the feedback is stored in the database.

Use Case: Admin- View Feedback

1. System redirects to admin's view feedback page.
2. The admin views feedback received from all the employees.

Use Case: Admin- Generate Reports

1. System redirects to admin's generate reports page.
2. The admin click on the button appropriate for generating the desired report.
3. Based on the selected button, the System redirects to the appropriate reports page.
4. Admin can generate various reports based on different criteria.

Extensions:

Create Account

3.The entered account information is wrong or invalid.

1. System tells the User that the entered information is invalid.
2. The use case continues at step 1.

Admin - Manage Employee

4. The entered employee information doesn't contain all the necessary details.

1. System tells the Admin that some information is missing.
2. The use case continues at step 3.

5. , 6. The admin doesn't click on the employee's data from the table.

1. System tells the Admin to select the employee to be deleted.
2. The use case continues at step 3.

Admin - Manage Bonus

4. The entered bonus information doesn't contain all the necessary details.

1. System tells the Admin that some information is missing.
2. The use case continues at step 3.

5. , 6. The admin doesn't click on the bonus data from the table.

1. System tells the Admin to select the bonus data to be deleted.
2. The use case continues at step 3.

Admin - Manage Bonus

4. The entered bonus information doesn't contain all the necessary details.

1. System tells the Admin that some information is missing.
2. The use case continues at step 3.

5. , 6. The admin doesn't click on the bonus data from the table.

1. System tells the Admin to select the bonus data to be deleted.
2. The use case continues at step 3.

Admin - Manage Attendance

4. The entered attendance information doesn't contain all the necessary details.

1. System tells the Admin that some information is missing.
2. The use case continues at step 3.

5. , 6. The admin doesn't click on the attendance data from the table.

1. System tells the Admin to select the attendance data to be deleted.
2. The use case continues at step 3.

Admin - Issue Salary

4. The admin doesn't select any employee and/or bonus from the dropdown list.

1. System tells the Admin to select the Employee details.
2. The use case continues at step 3.

5. The entered salary information doesn't contain all the necessary details.

1. System tells the Admin that some information is missing.
2. The use case continues at step 3.

User- View Profile

3.The entered account information deals with an exception.

1. System displays the exception message to the User.
2. The use case continues at step 3.a.

4.a. The entered password is blank or empty.

1. System displays a 'Password cannot be Empty' message to the User.
2. The use case continues at step 4.a.

4.b. The entered data in new password and confirm password field aren't the same.

1. System displays a 'Password not Matching' message to the User.
2. The use case continues at step 4.a.

User- Submit Feedback

3. The feedback is blank or empty.
    1. System displays a ‘Feedback cannot be Empty’ message to the User.
    2. The use case continues at step 2.

Special Requirements:

1. Text must be visible from 1 meter.
2. We want robust recovery when the system fails.
3. Language internationalization on the text displayed.

Frequency of Occurrence:

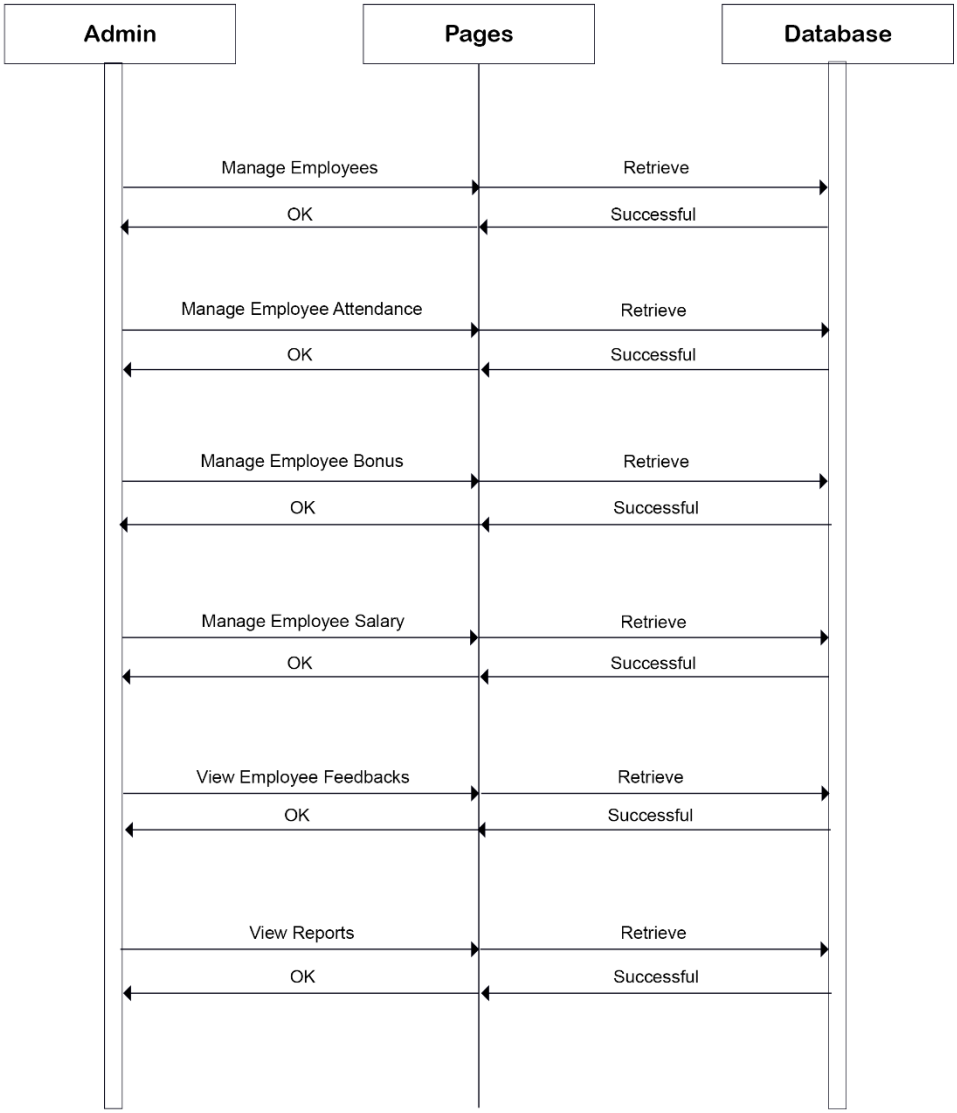
Could be nearly continuous.

Open Issues:

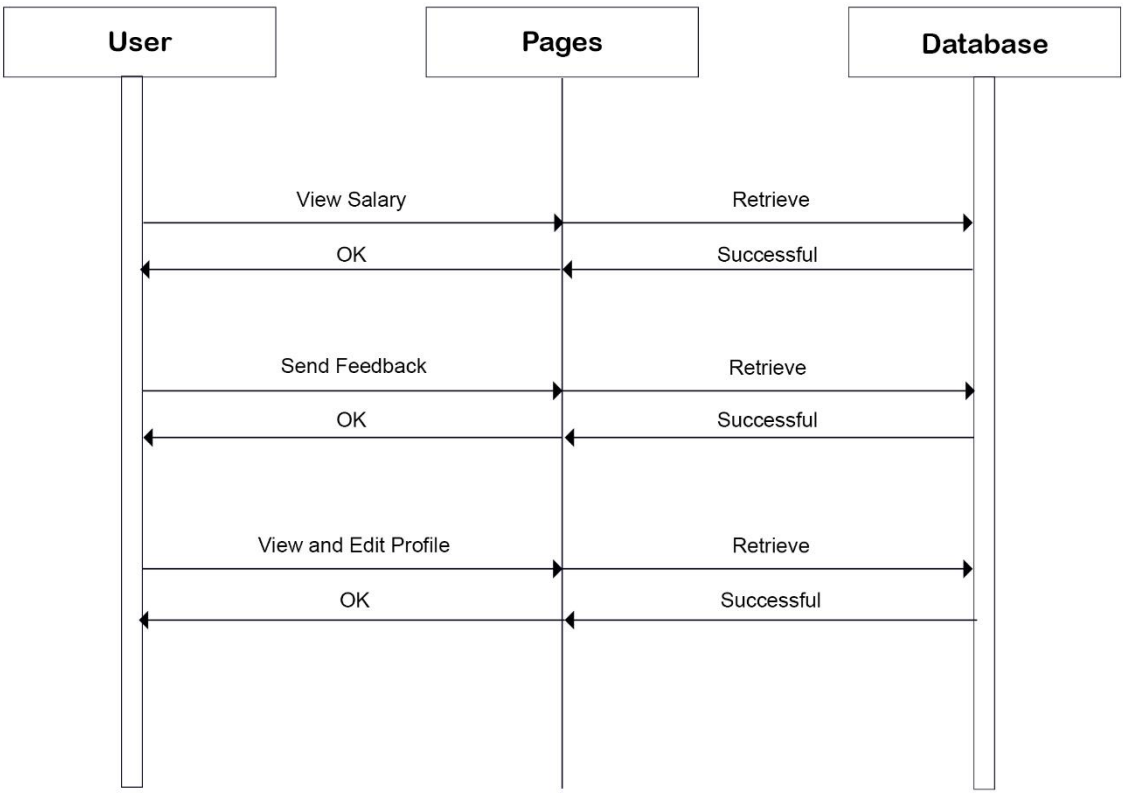
1. Explore the recovery issues.

8.2 System Sequence Diagram

Admin Side



User Side



8.3 Operation Contracts

Operation: Registered User (username: string, password: string)

Cross Reference: Use case: User login

Preconditions: Proper communication between Windows Forms

Conditions :

- A new login instance was created was created by the admin.
- Accepted username and password and stored it in respective attributes.
- After validation, the user gets logged in to the system.

Operation: Admin (username: string, password: string)

Cross Reference: Use case: Admin login

Preconditions: Proper communication between Windows Forms

Conditions :

- An admin login instance was defined.
- Accepted username and password and stored it in respective attributes.
- After validation, the admin gets logged in to the system.
-

8.4 Reports

Employee Gender Report: Admin can generate reports based on Employee Gender – Male or Female

Employee Position Report: Admin can generate eports based on employees at various positions like Manager, Senior or Junior.

Employee Attendance Report: Admin can generate reports based on employee's attendance i.e. the admin can view the total worked days and total non-worked days of each employee.

Employee Birthday Report: Admin can generate reports based on employee's birthday based on the selected month.

Employee Join Date Report: Admin can view data reports based on employee's join date between the selected year range.

Employee ID Card Report: Admin can generate ID cards for employees.

Monthly Salary Report: Admin can view data reports on employee's salary based on the selected month and year.

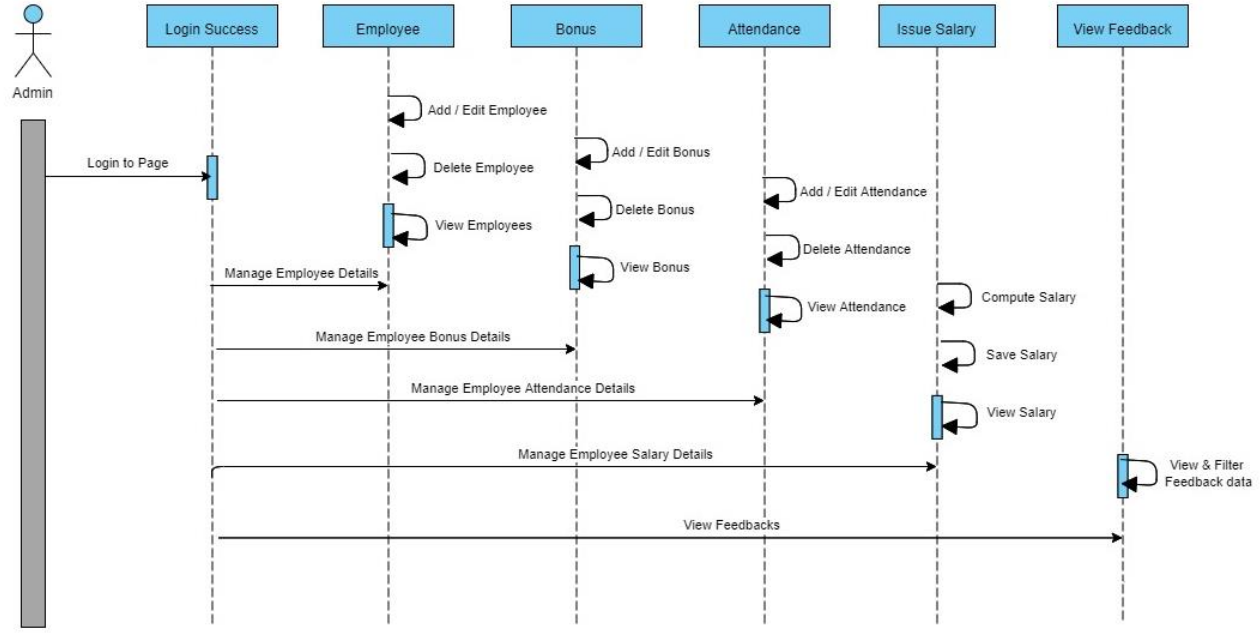
Overall Employee Report: Admin can view employee data based on the selected employee name.

Salary Overview: Admin can view data reports on total salary and bonus paid based on the selected month and year.

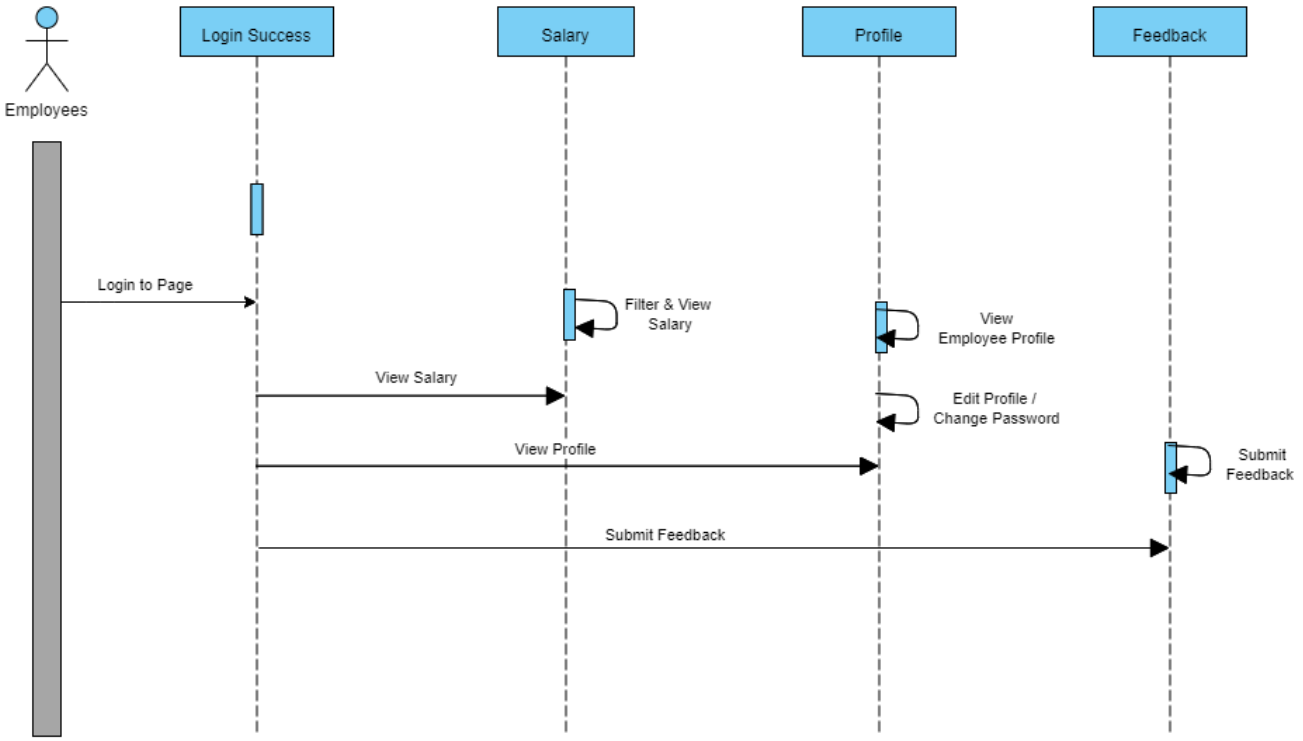
9. DESIGN MODEL

9.1 Sequence Diagram

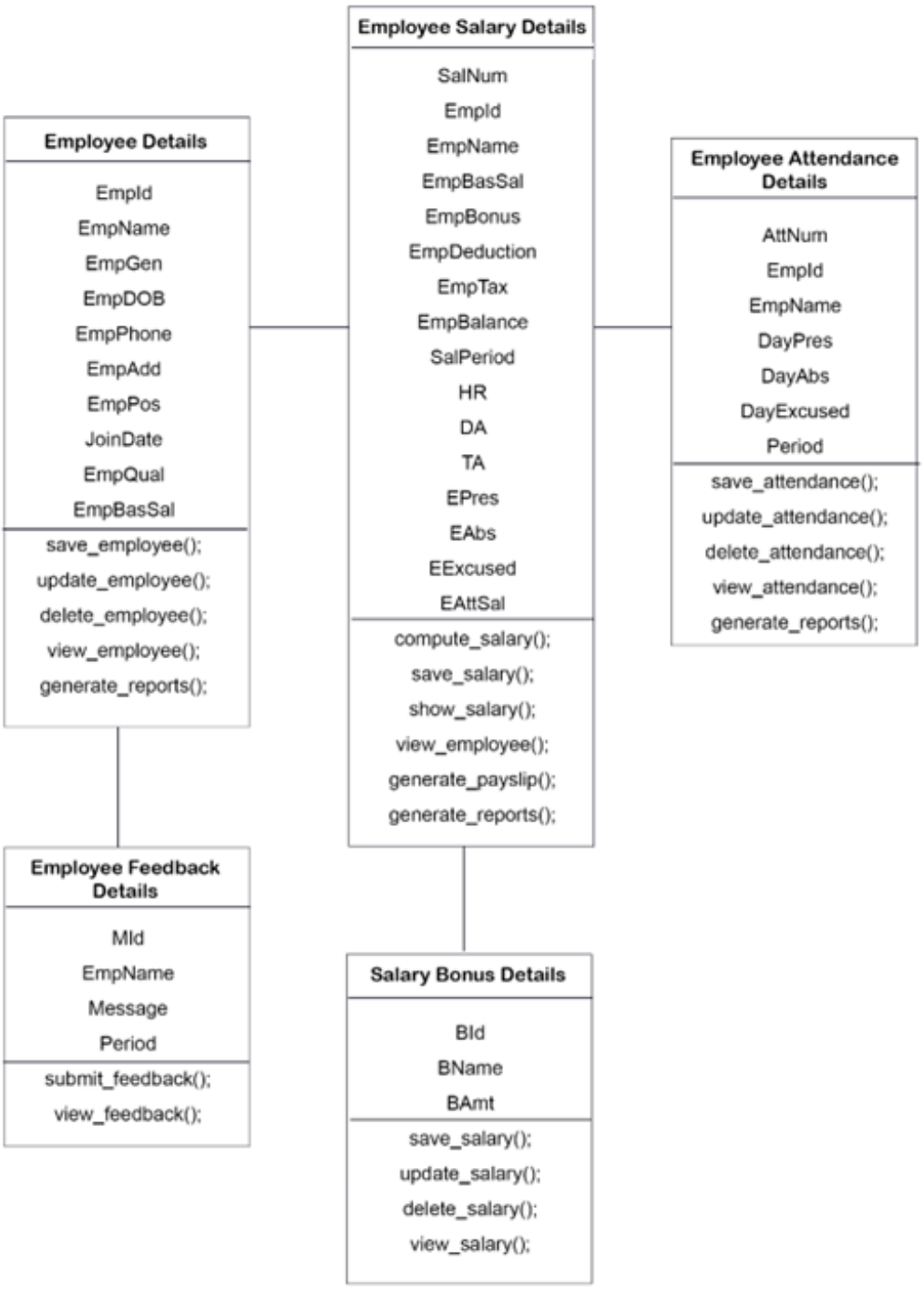
Admin



User



9.2 Class Diagram



9.3 UI Design

ADMIN:

Login Page: The admin must enter a username and a password to login to the application.

Home Page: Displays the total no of employees present at various positions , the total salary paid and total bonus paid.

Employee Page: Admin can view/add/update/delete employee details.

Bonus Page: Admin can view/add/update/delete salary bonus details.

Attendance Page: Admin can view/add/update/delete employee attendance details.

Salary Page: Admin can compute and save employee salary details and generate pay slips.

Reports Page: Admin can generate various work reports based on Employee Gender, Employee Position, Employee Attendance, Employee Birthday, Employee Join Date, Employee ID Card, Monthly Salary, Overall Employee Report and Salary Overview.

Feedback Page: Admins can view all the feedback submitted by the employees.

Logout Page: By clicking this button, the admin gets logged out of the application.

EMPLOYEE:

Login Page: The employee must enter a username and a password to login to the application.

Home Page: Displays the total salary and total bonus recieved.

Salary Page: The employee can view their salary details. They can even view the salary details based on the month and year selected.

Feedback Page: : The employee can submit their valuable feedbacks and suggestions to the admin.

Profile Page: The employee could view/update their personal details. They also have the option to change their password.

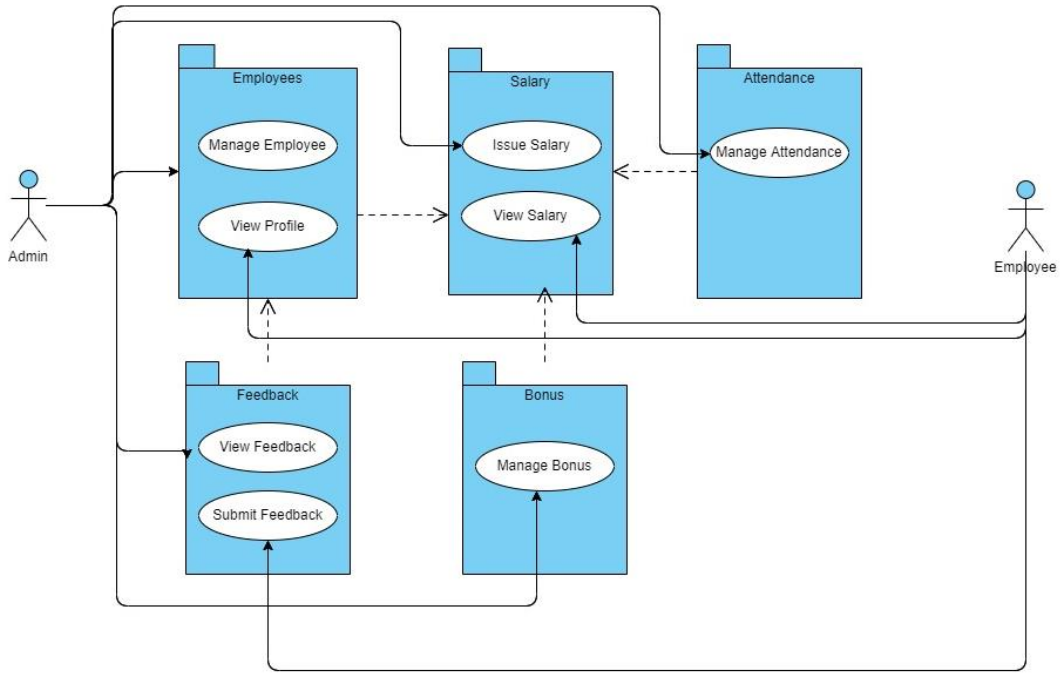
Logout Page: By clicking this button, the employee gets logged out of the application.

9.4 Theoretical Background

PayIt- The Payroll Management System is developed using one of the widely used front-end tool C# .net and at the back-end, we used SQL Server. The Visual Studio 2022 is used as the Integrated Development Environment(IDE). The Operating System used to develop this application is Windows 11.

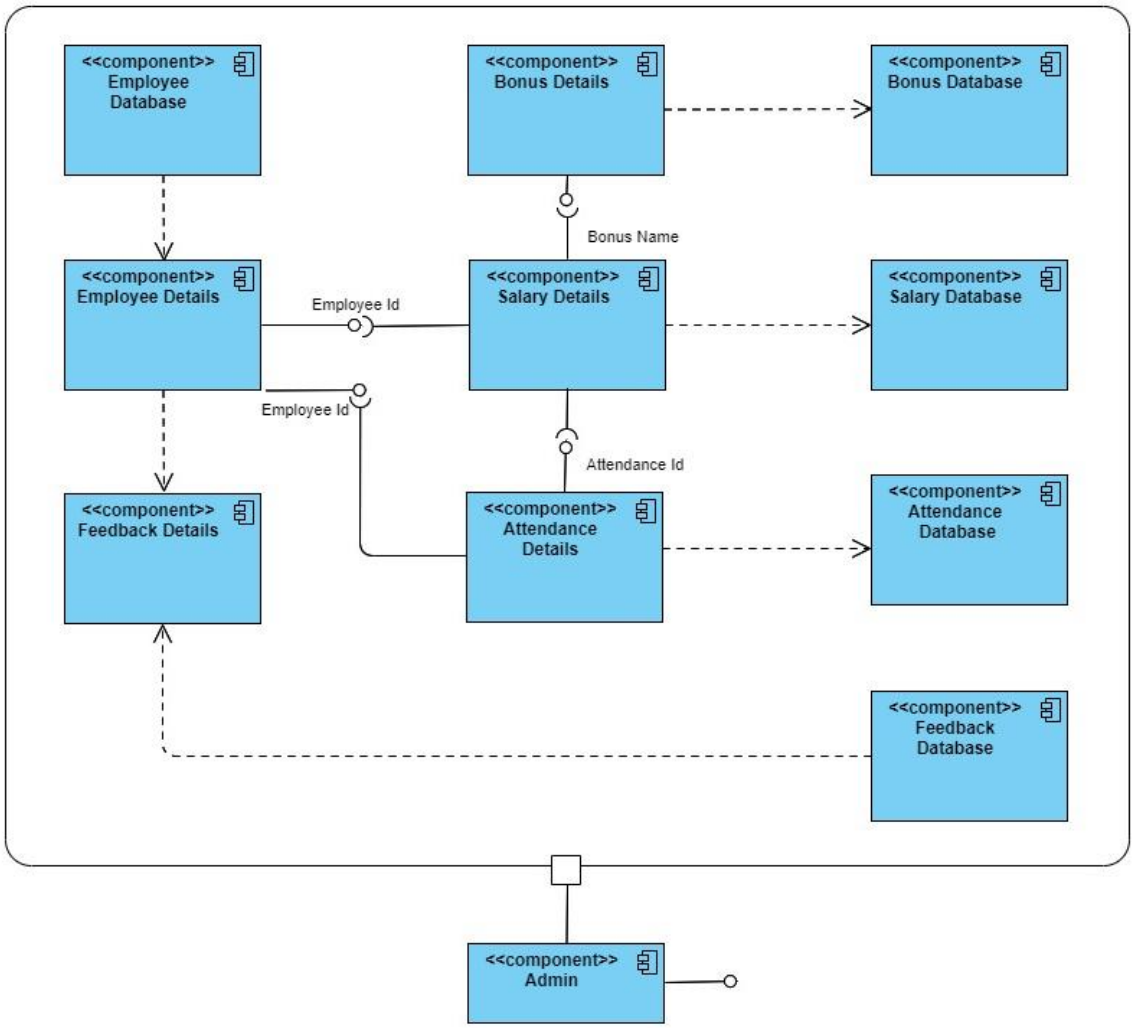
9.5 Architecture

9.5.1 Package diagram

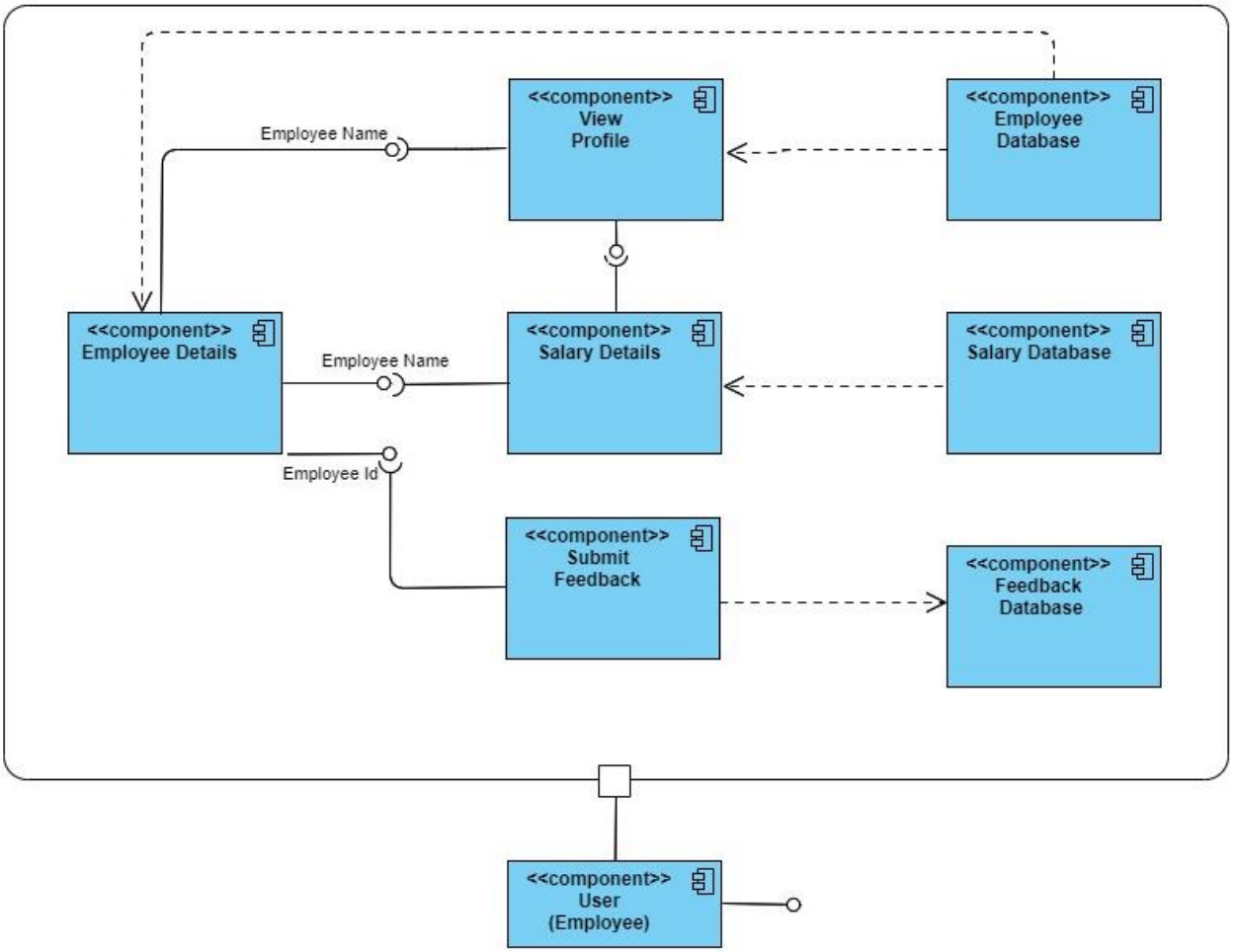


9.5.2 Component diagrams

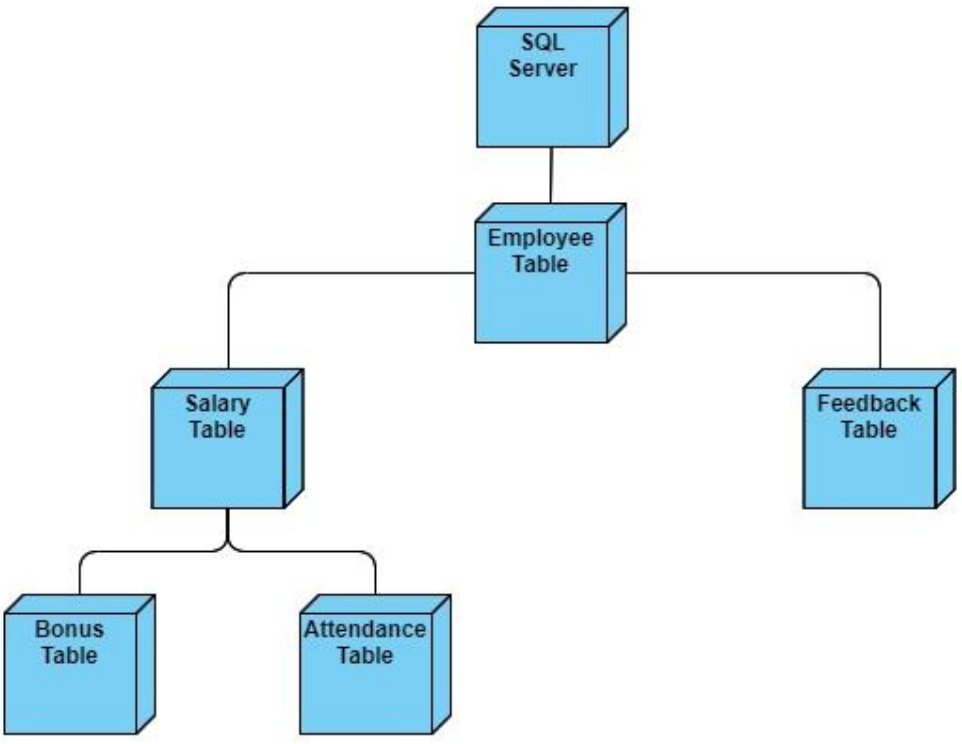
<u>Admin Side</u>



User Side



9.5.3 Deployment diagram



9.6 Database Design

Data Base Name : PayrollDB

EMPLOYEE INFO TABLE

S.No	Field name	Data Type	Description	Constraints
1	EmpId	int	ID of employee	Identity (1000, 1) Primary Key, Not Null
2	EmpName	varchar (50)	Name of employee	Not Null
3	EmpGen	varchar (10)	Gender of the employee	Not Null
4	EmpDOB	date	DOB of employee	Not Null
5	EmpPhone	varchar (50)	Phone number of employee	Not Null
6	EmpAdd	varchar (50)	Address of employee	Not Null
7	EmpPos	varchar (50)	Position of employee	Not Null
8	JoinDate	date	Date of joining of employee	Not Null
9	EmpQual	varchar (50)	Qualification of employee	Not Null
10	EmpBasSal	int	Basic Salary of employee	Not Null
11	EmpPass	varchar (50)	Password	Not Null

EMPLOYEE ATTENDANCE TABLE

S.No	Field name	Data Type	Description	Constraints
1	AttNum	int	ID of Employee Attendance	Identity (1 1) Not Null, Primary Key
2	EmpId	int	ID of employee	Not Null, Foreign Key
3	DayPres	int	No of days worked by employee	Not Null
4	DayAbs	int	No of days absent by employee	Not Null
5	DayExcused	int	No of days excused for employee	Not Null
6	Period	varchar (50)	Period of leave of employee	Not Null

EMPLOYEE SALARY TABLE

S.No	Field name	Data Type	Description	Constraints
1	SalNum	int	ID of salary	Identity (1,1) Not Null, Primary Key
2	EmpId	int	ID of employee	Not Null, Foreign Key
3	EmpBasSal	int	Basic Salary of employee	Not Null
4	EmpBonus	int	Bonus of employee	Not Null
5	EmpDeduction	varchar (50)	Amount deducted from Salary	Not Null
6	EmpTax	int	Tax of employee	Not Null
7	EmpBalance	int	Salary of employee	Not Null
8	SalPeriod	varchar (50)	Salary Period	Not Null
9	HR	float (53)	House Rent Allowance of employee	Default ((0)), Not Null
10	DA	float (53)	Dearness Allowance of employee	Default ((0)), Not Null
11	TA	float (53)	Travel Allowance of employee	Default ((0)), Not Null
12	EPres	int	No of days worked by employee	Default ((0)), Not Null
13	EAbs	int	No of days absent by employee	Default ((0)), Not Null
14	EExcused	int	No of days excused for employee	Default ((0)), Not Null
15	EAttSal	float (53)	Salary for attendance of employee	Default ((0)), Not Null

SALARY BONUS TABLE

S.No	Field name	Data Type	Description	Constraints
1	BId	int	ID of Bonus	Identity (500, 1) Not Null, Primary Key
2	BName	varchar (50)	Bonus Name	Not Null
3	BAmt	int	Bonus Amount	Not Null

EMPLOYEE FEEDBACK TABLE

S.No	Field name	Data Type	Description	Constraints
1	MId	int	Feedback ID	Identity (1, 1) Not Null, Primary Key
2	EmpId	int	ID of employee	Not Null, Foreign Key
3	Message	varchar (100)	Bonus Amount	Not Null
4	Period	date	Feedback Period	Not Null

10. TESTING

10.1 Test cases

Test Scenario: Checking Login Functionality

Test Case 1: Invalid User Login

An unregistered or unauthorized login attempt must be blocked.

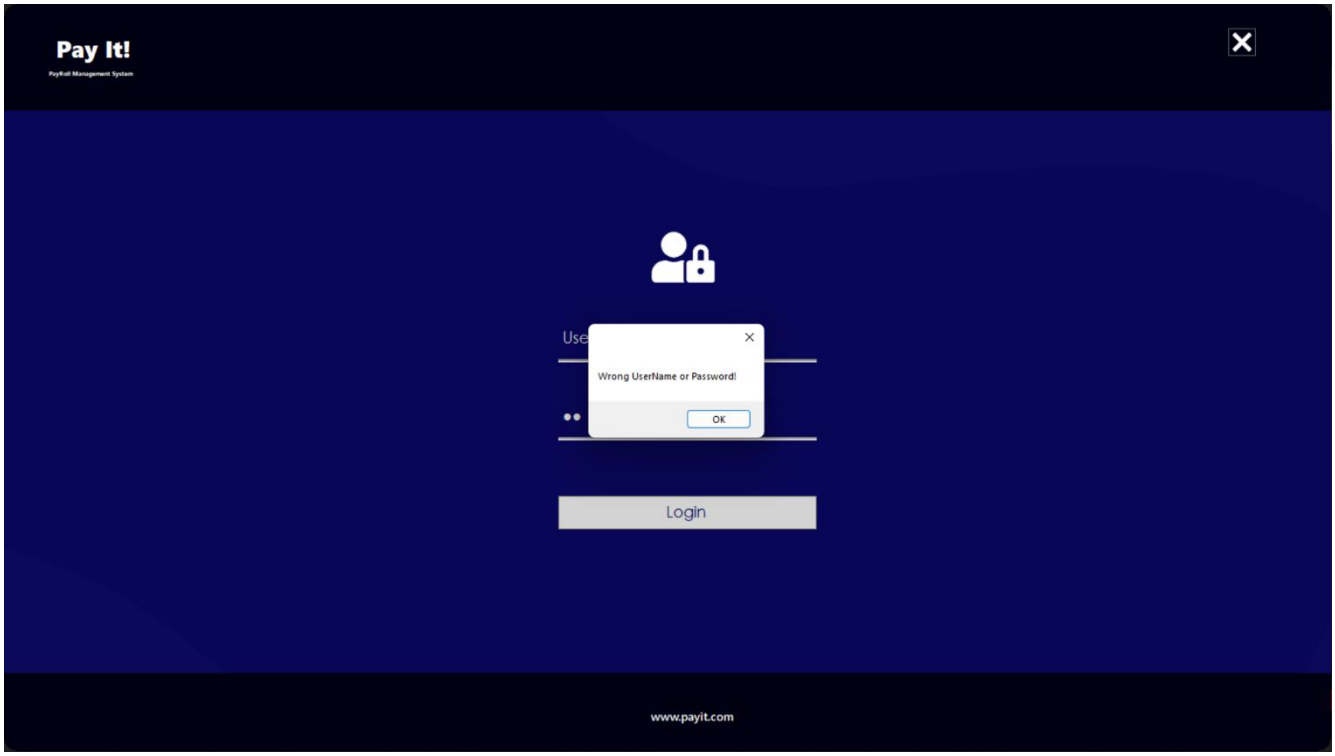
Precondition: Unauthorized users do not have valid credentials to log in.

Assumption: Only authorized users have access to valid credentials for logging in.

Test Steps:

1. Go to Login Page
2. Enter credentials
3. Submit the credentials

Expected Result: A login attempt with invalid or wrong input results in an unsuccessful login message.



Test Case 2: Check results on not storing all the required fields by Admin

All the attempts to store employee data must be blocked if all the required fields aren't filled by the admin.

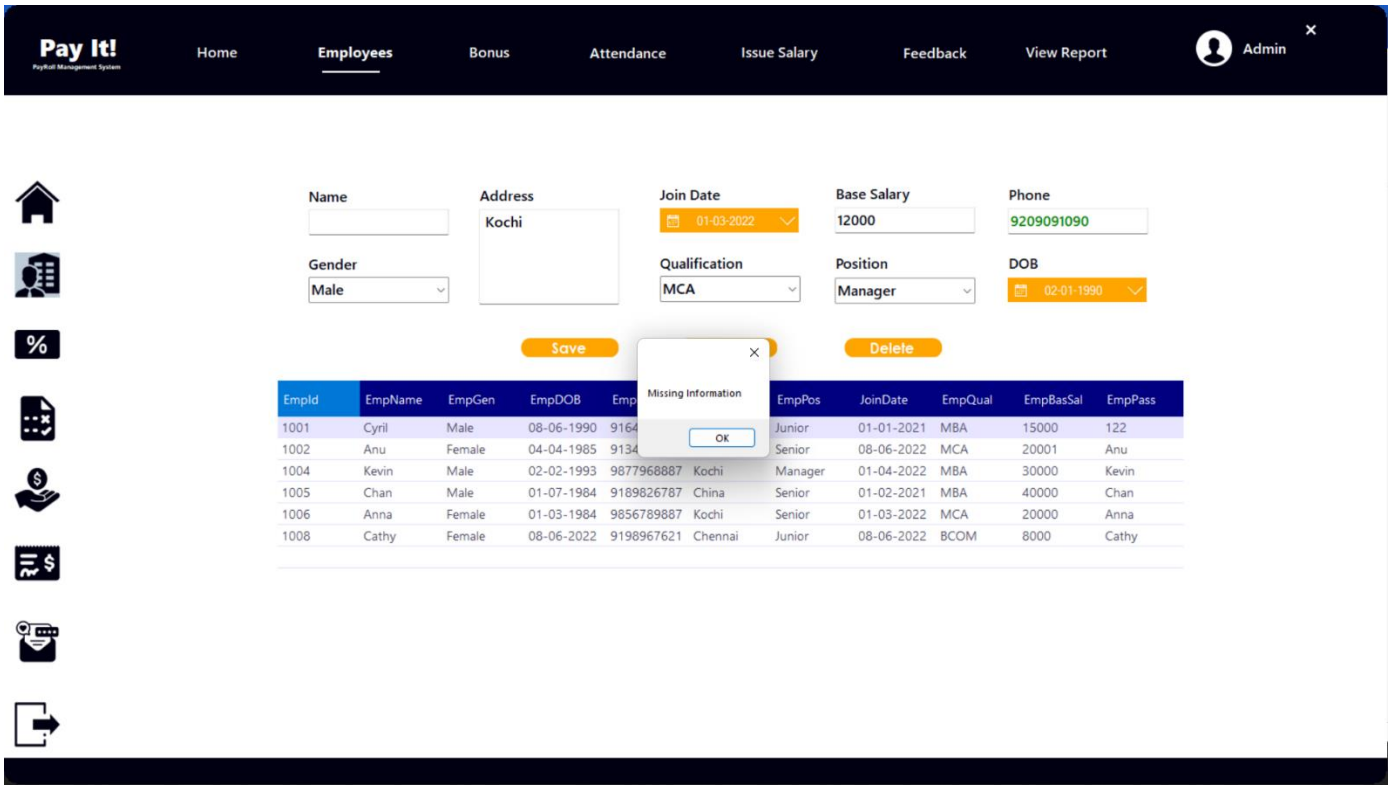
Precondition: Admin does not have all the necessary data to store an employee.

Assumption: Only authorized administrators have access to store data.

Test Steps:

1. Log in to the system as Admin using the correct admin's credentials.
2. Go to the employee page.
3. Submit the details of the employee without filling all or many of the required fields.

Expected Result: An attempt by the admin to submit all the details of the employee without filling all or many of the required fields, results in an unsuccessful error message.



10.2 Test Report

In both the test cases, as we got the expected result, we can easily come to the conclusion that the testing process was successful and hence proved that our application is efficient enough to store and process data. There is almost about 99.9% of passed test cases.

10.3Sample Code used for testing

Test Case 1: Invalid User Login

```
private void LoginBtnn_Click(object sender, EventArgs e)
    {
      if (User.Text == "" || Pass.Text == "")
        MessageBox.Show("Missing Information!!");

      else if (User.Text == "Admin" && Pass.Text == "123")
      {
        Users = User.Text;
        Home Obj = new Home();
        Obj.Show();
        this.Hide();
      }

      else
      {
        Con.Open();
        string Query = "select count(*) from EmployeeTbl where EmpName='" +
User.Text+ "' and EmpPass='" + Pass.Text + "' ";
        SqlDataAdapter sda = new SqlDataAdapter(Query, Con);
        DataTable dt = new DataTable();
        sda.Fill(dt);
        if (dt.Rows[0][0].ToString() == "1")
```

```
            {
                Users = User.Text;
                UserHome Obj = new UserHome();
                Obj.Show();
                this.Hide();
            }

            else
                MessageBox.Show("Wrong UserName or Password!");
            Con.Close();
        }

        }
```

Test Case 2: Check results on not storing all the required fields by Admin

```
    private void SaveBtn_Click(object sender, EventArgs e)
        {
            if (EmpNameTb.Text == "" || EmpPhoneTb.Text == "" ||EmpGenCb.SelectedIndex
    == -1 || EmpAddTb.Text == "" || EmpSalTb.Text == "" || EmpQualCb.SelectedIndex == -1)
            {
                MessageBox.Show("Missing Information");
            }

            else
            {
                try
                {
                    Con.Open();
                    SqlCommand cmd = new SqlCommand("insert into
    EmployeeTbl(EmpName,EmpGen,EmpDOB,EmpPhone,EmpAdd,EmpPos,JoinDate,Emp
```

```
Qual,EmpBasSal,EmpPass)
values(@EN,@EG,@ED,@EP,@EA,@EPOS,@JD,@EQ,@EBS,@EPa)", Con);
                cmd.Parameters.AddWithValue("@EN", EmpNameTb.Text);
                cmd.Parameters.AddWithValue("@EG",
EmpGenCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@ED", EmpDOB.Value.Date);
                cmd.Parameters.AddWithValue("@EP", EmpPhoneTb.Text);
                cmd.Parameters.AddWithValue("@EA", EmpAddTb.Text);
                cmd.Parameters.AddWithValue("@EPOS",
EmpPosCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@JD", JDate.Value.Date);
                cmd.Parameters.AddWithValue("@EQ",
EmpQualCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@EBS", EmpSalTb.Text);
                cmd.Parameters.AddWithValue("@EPa", EmpNameTb.Text);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Employee Saved!!");
                Con.Close();
                showEmployee();
                Clear();
            }

            catch(Exception Ex)
            {
                MessageBox.Show(Ex.Message);
            }

        }
    }
```

11. TRANSITION

11.1 System Implementation

Implementation is the process of having the system personal checks out and put equipment's to use, train the users to use the new system and construct any file that are needed to see it. The final and important phases in the system lifecycle are the implementation of new system. The file conversion is the most time consuming and expensive activity in the implementation stage. System implementation refers to the step necessary to install a new system to put into the operation. The implementation has different meaning, raining from the conversion of basic application to complete replacement of computer system. Implementation includes all these activities that take place to covert from old system to new system. The new system may be totally new replacing and existing manual or automated system or it may be major modification to an existing system. The method of implementation and time scale adopted is found out initially. The system is tested properly and at the same time the users are trained in new procedure. Proper implementation is essential to provide a reliable system to meet organization requirements. Successful implementation may not guarantee improvement in the organization using the new system, but it will prevent improper installation. The implementation involves the following things:-

1. Careful planning.
2. Investigation of the system and constrains
3. Design the methods to achieve the changeover.
4. Train the staff in the changed phase.
5. Evaluation of change over method.

After converting as a package, it has been delivered to the customers where it is implemented and tailored to meet the specific requirements.

11.2 System Maintenance

Like housework, dirty clothes and weeds, system work never seems to an end; users almost always want changes or encounter problems. Thus the system maintenance part of the system process deserves special attention. It is during system maintenance that the analyst:

1. Resolves necessary changes
2. Correct errors.
3. Enhance or modifies the system.
4. Assign staff to perform maintenance activities.
5. Provides for scheduled maintenance.

Most system spends the bulk of their time in the maintenance phase, with constant enhancements and repairs. Studies show that more money is spent in this forth phase than in all of the others combined. Writing system is that require as little maintenance as possible is one of the primary goal as well as one of the benefits of today's modern methodology of software development. Maintenance ids divided into three categories.

1. Corrective maintenance.
2. Adaptive maintenance.
3. Preventive maintenance.

11.2.1 Corrective maintenance

It has to do with the removal of residual errors present in the product when it is delivered as well as errors introduced into the software during its maintenance accounts for about 20% of the maintenance cost.

11.2.2 Adaptive maintenance

It involves adjusting the application to changes in the environment, that is a new release of the hardware or the operating system or anew database system. It also accounts for nearly 20% of the maintenance cost.

11.2.3 Preventive maintenance

It involves changing the software to improve some qualities. It accounts for over 50% of maintenance costs. Here changes are due to the functions offered by the application, and new functions, improve the performance of application etc.

Maintenance is not such a difficult task. The above three maintenance tasks can be easily carried out under this system.

12. ANNEXURE

12.1 References

Websites

[1] YouTube

[2] Random sites

https://W3schools.com/

http://www.stackoverflow.com/questions

http://tutorialspoint.com/

https://www.geekforgeeks.com

12.2 Annexure I: User Interview Questionnaires

1. How would you approach the system?

2. What about usability of this system?

3. What are normal project requirements?

12.3 CONCLUSION

An automated payroll system enables the admin to process its payroll through a computerized system. A manual payroll system requires that the payroll be processed by hand and is, therefore, a considerably slower procedure than an automated system. The former makes payroll processing simpler and reduces errors, which are more likely in the manual system. The admin uses an attendance-keeping system to track attendance and pay employees accordingly. Therefore, each employee's attendance, as well as salary, is computed accurately.

Payroll seems simple at its core but becomes complicated because of the various deductions that come into play. Employers must withhold taxes from each paycheck and make sure accurate funds are paid to the correct government agency. Payroll processing duties can create a huge burden and unwanted stress for small business owners. A missed deadline or incorrect filing of taxes can result in fines or jail time. To avoid these issues, small, middle-sized, and large businesses can all benefit from using payroll systems.

The payroll system software can mitigate errors in the payroll process and reduce the amount of effort involved in calculating employee hours, wages, and tax withholdings. Successfully, the system has been designed in response to the system analysis. All possible errors in the program have been eliminated. Necessary validation techniques have been used and normal, abnormal and extremely data were used to test the system.

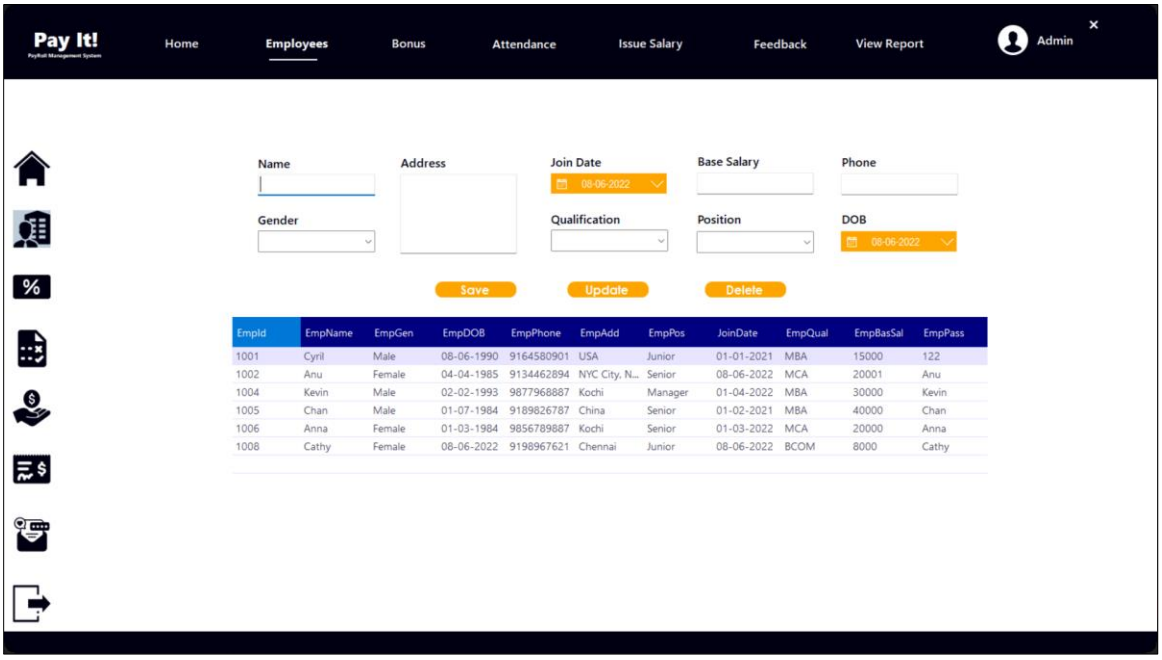
However, doing this project has been a good boost to our confidence as future IT members. We thank all people who help us to complete this project work successfully.

12.4 SAMPLE CODE

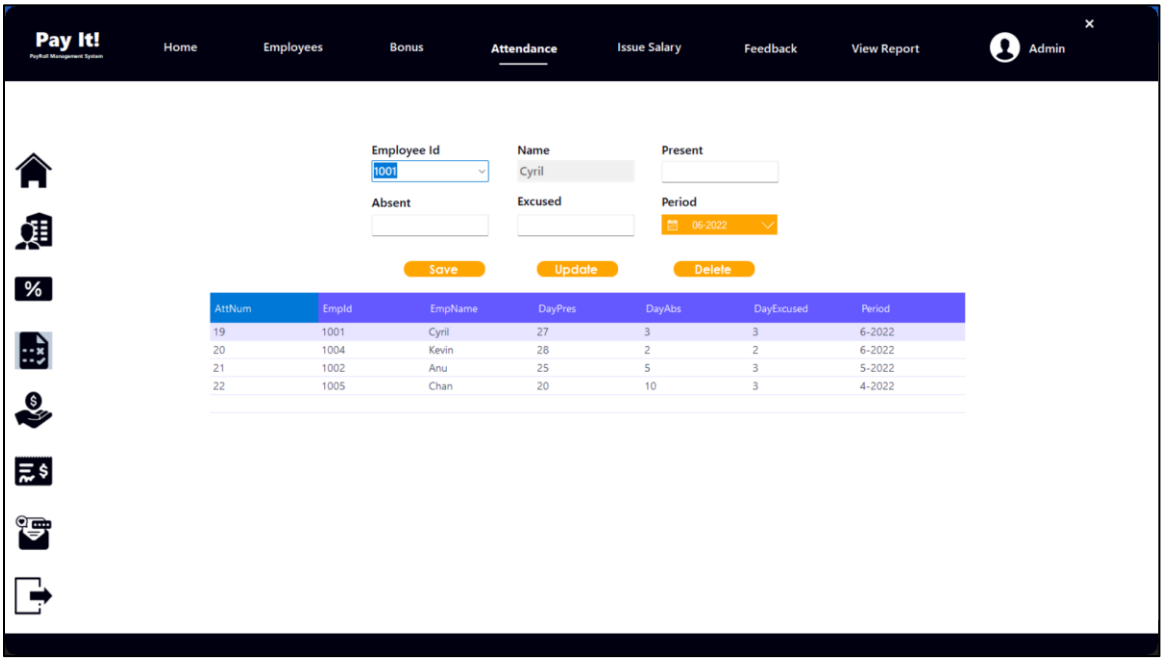
- Screenshots

12.4.1. Main

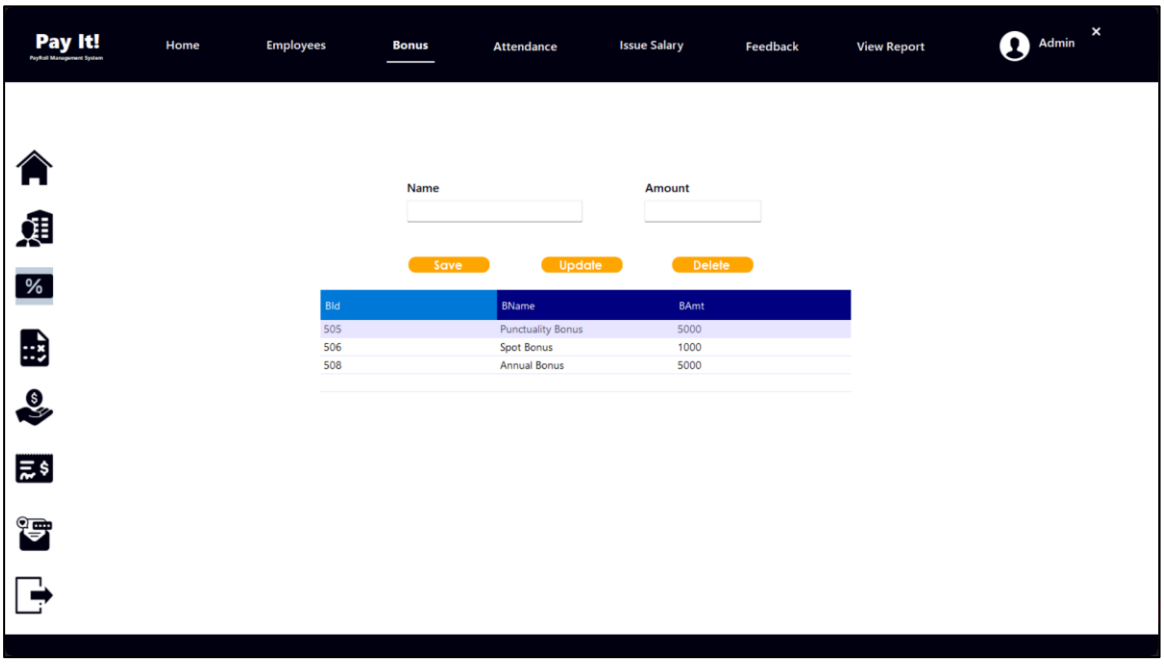
ADMIN EMPLOYEE PAGE



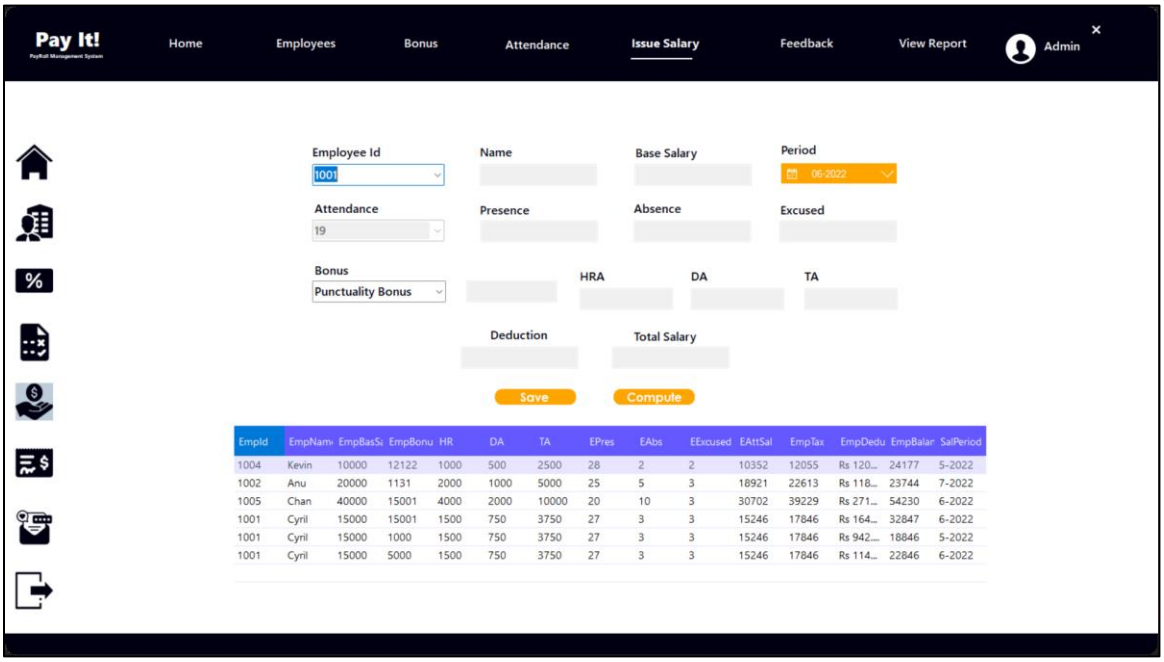
ADMIN EMPLOYEE ATTENDANCE PAGE



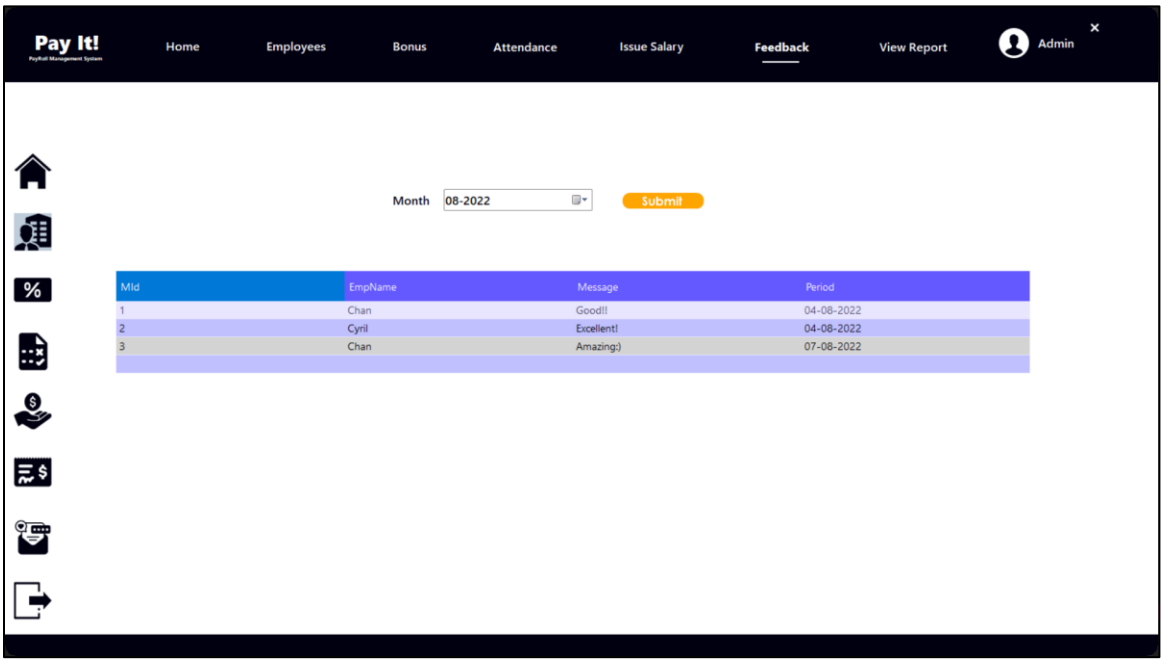
ADMIN EMPLOYEE BONUS PAGE



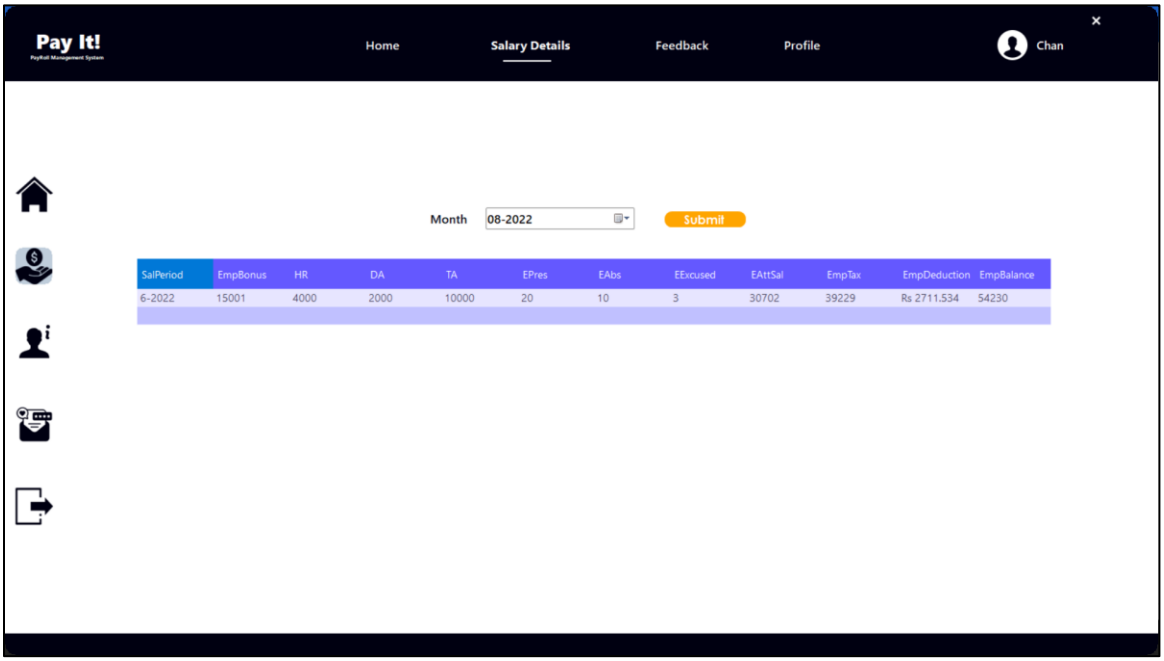
ADMIN ISSUE SALARY PAGE



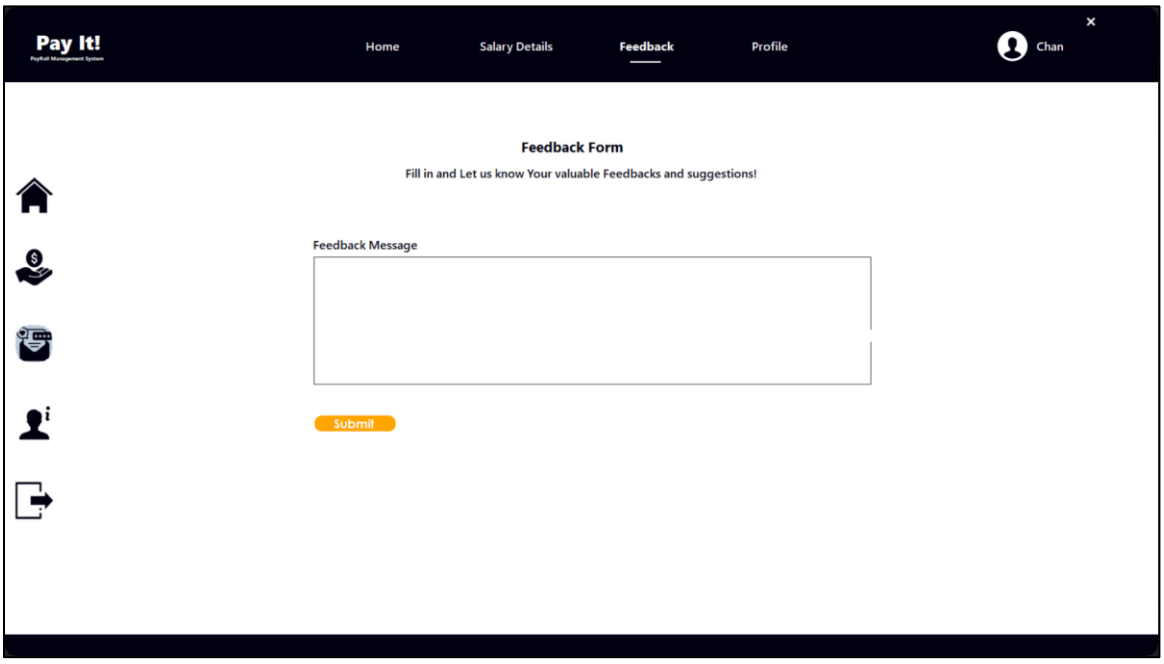
ADMIN VIEW FEEDBACK PAGE



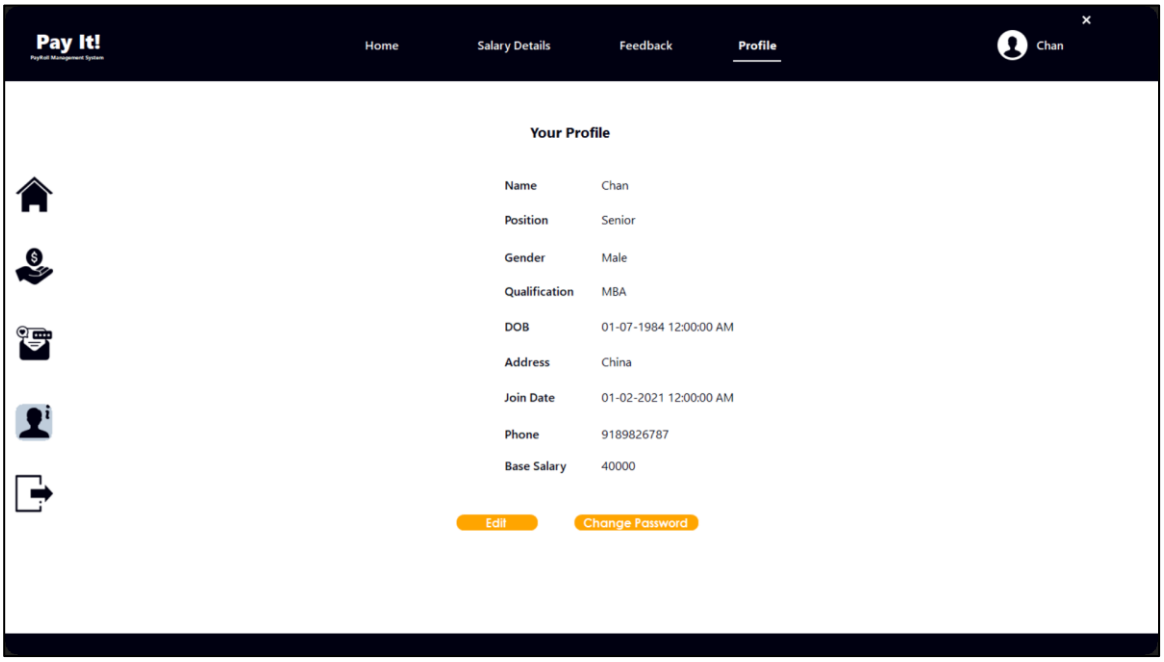
USER VIEW SALARY PAGE



USER FEEDBACK PAGE



USER PROFILE PAGE



- Sample Code

12.4.2. Main

Employee Page of Admin

Employees.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace PayrollSystem
{
    public partial class Employees : Form
    {
        public Employees()
        {
            InitializeComponent();
            showEmployee();
        }

        private void pictureBox9_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
```

```
        SqlConnection Con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\Trials\PayrollSystem\PayRollDB.mdf;Integrated Security=True;Connect Timeout=30");
        private void Clear()
        {
            EmpNameTb.Text = "";
            EmpPhoneTb.Text = "";
            EmpGenCb.SelectedIndex = -1;
            EmpAddTb.Text = "";
            EmpSalTb.Text = "";
            EmpQualCb.SelectedIndex = -1;
            EmpPosCb.SelectedIndex = -1;
            Key = 0;
        }
        private void showEmployee()
        {
            Con.Open();
            string Query = "select * from EmployeeTbl";
            SqlDataAdapter sda = new SqlDataAdapter(Query, Con);
            SqlCommandBuilder Builder = new SqlCommandBuilder(sda);
            var ds = new DataSet();
            sda.Fill(ds);
            EmployeeDGV.DataSource = ds.Tables[0];
            Con.Close();
        }
        private void SaveBtn_Click(object sender, EventArgs e)
        {
            if (EmpNameTb.Text == "" || EmpPhoneTb.Text == "" ||EmpGenCb.SelectedIndex == -1 || EmpAddTb.Text == "" || EmpSalTb.Text == "" || EmpQualCb.SelectedIndex == -1)
            {
                MessageBox.Show("Missing Information");
            }
```

```
        else
        {
            try
            {
                //string Period = JDate.Value.Day + "-" + JDate.Value.Month + "-" +
JDate.Value.Year;
                Con.Open();
                SqlCommand cmd = new SqlCommand("insert into
EmployeeTbl(EmpName,EmpGen,EmpDOB,EmpPhone,EmpAdd,EmpPos,JoinDate,Emp
Qual,EmpBasSal,EmpPass)
values(@EN,@EG,@ED,@EP,@EA,@EPOS,@JD,@EQ,@EBS,@EPa)", Con);
                cmd.Parameters.AddWithValue("@EN", EmpNameTb.Text);
                cmd.Parameters.AddWithValue("@EG",
EmpGenCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@ED", EmpDOB.Value.Date);
                cmd.Parameters.AddWithValue("@EP", EmpPhoneTb.Text);
                cmd.Parameters.AddWithValue("@EA", EmpAddTb.Text);
                cmd.Parameters.AddWithValue("@EPOS",
EmpPosCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@JD", JDate.Value.Date);
                cmd.Parameters.AddWithValue("@EQ",
EmpQualCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@EBS", EmpSalTb.Text);
                cmd.Parameters.AddWithValue("@EPa", EmpNameTb.Text);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Employee Saved!!");
                Con.Close();
                showEmployee();
                Clear();
            }
```

```
            catch(Exception Ex)
            {
                MessageBox.Show(Ex.Message);
            }
        }
    }

    int Key;

    private void EmployeeDGV_CellContentClick(object sender,
DataGridViewCellEventArgs e)
    {
        EmpNameTb.Text = EmployeeDGV.SelectedRows[0].Cells[1].Value.ToString();
        EmpGenCb.SelectedItem =
EmployeeDGV.SelectedRows[0].Cells[2].Value.ToString();
        EmpDOB.Value =
Convert.ToDateTime(EmployeeDGV.SelectedRows[0].Cells[3].Value);
        EmpPhoneTb.Text = EmployeeDGV.SelectedRows[0].Cells[4].Value.ToString();
        EmpAddTb.Text = EmployeeDGV.SelectedRows[0].Cells[5].Value.ToString();
        EmpPosCb.SelectedItem =
EmployeeDGV.SelectedRows[0].Cells[6].Value.ToString();
        JDate.Value =
Convert.ToDateTime(EmployeeDGV.SelectedRows[0].Cells[7].Value);
        EmpQualCb.SelectedItem =
EmployeeDGV.SelectedRows[0].Cells[8].Value.ToString();
        EmpSalTb.Text = EmployeeDGV.SelectedRows[0].Cells[9].Value.ToString();
        if (EmpNameTb.Text == "")
        {
            Key = 0;
        }
        else
        {
```

```
            Key = Convert.ToInt32(EmployeeDGV.SelectedRows[0].Cells[0].Value.ToString());
        }
    }

    private void EditBtn_Click(object sender, EventArgs e)
    {
        if (EmpNameTb.Text == "" || EmpPhoneTb.Text == "" || EmpGenCb.SelectedIndex == -1 || EmpAddTb.Text == "" || EmpSalTb.Text == "" || EmpQualCb.SelectedIndex == -1)
        {
            MessageBox.Show("Missing Information");
        }
        else
        {
            try
            {
                //string Period = JDate.Value.Day + "-" + JDate.Value.Month + "-" + JDate.Value.Year;
                Con.Open();
                SqlCommand cmd = new SqlCommand("update EmployeeTbl set EmpName=@EN,EmpGen=@EG,EmpDOB=@ED,EmpPhone=@EP,EmpAdd=@EA,EmpPos=@EPOS,JoinDate=@JD,EmpQual=@EQ ,EmpBasSal=@EBS where EmpId=@EmpKey", Con);
                cmd.Parameters.AddWithValue("@EN", EmpNameTb.Text);
                cmd.Parameters.AddWithValue("@EG", EmpGenCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@ED", EmpDOB.Value.Date);
                cmd.Parameters.AddWithValue("@EP", EmpPhoneTb.Text);
                cmd.Parameters.AddWithValue("@EA", EmpAddTb.Text);
                cmd.Parameters.AddWithValue("@EPOS", EmpPosCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@JD", JDate.Value.Date);
```

```
                cmd.Parameters.AddWithValue("@EQ",
EmpQualCb.SelectedItem.ToString());
                cmd.Parameters.AddWithValue("@EBS", EmpSalTb.Text);
                //cmd.Parameters.AddWithValue("@EPa", EmpNameTb.Text);
                cmd.Parameters.AddWithValue("@EmpKey", Key);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Employee Updated!!");
                Con.Close();
                showEmployee();
                Clear();
            }
            catch (Exception Ex)
            {
                MessageBox.Show(Ex.Message);
            }
        }
    }

    private void DeleteBtn_Click(object sender, EventArgs e)
    {
        if (Key == 0)
        {
            MessageBox.Show("Missing Information");
        }
        else
        {
            try
            {
                Con.Open();
                SqlCommand cmd = new SqlCommand("delete from EmployeeTbl where
EmpId=@EmpKey", Con);
                cmd.Parameters.AddWithValue("@EmpKey", Key);
```

```
            cmd.ExecuteNonQuery();
            MessageBox.Show("Employee Deleted!!");
            Con.Close();
            showEmployee();
        }
        catch (Exception Ex)
        {
            MessageBox.Show(Ex.Message);
        }
    }
}
private void pictureBox1_Click(object sender, EventArgs e)
{
    Home Objh = new Home();
    Objh.Show();
    this.Hide();
}
private void pictureBox8_Click(object sender, EventArgs e)
{
    Login Objl = new Login();
    Objl.Show();
    this.Hide();
}

private void EmpPhoneTb_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsDigit(e.KeyChar) && !char.IsControl(e.KeyChar))
    {
        e.Handled = true;
        MessageBox.Show("Error,Cannot Contain Letters ");
    }
}
```

```
private void EmpPhoneTb_TextChanged(object sender, EventArgs e)
{
   if (EmpPhoneTb.TextLength == 10)
      EmpPhoneTb.ForeColor = Color.Green;

   else
      EmpPhoneTb.ForeColor = Color.Red;
}

private void Home_Click(object sender, EventArgs e)
{
   Home Obj = new Home();
   Obj.Show();
   this.Hide();
}

private void label3_Click(object sender, EventArgs e)
{
   Bonus Obj = new Bonus();
   Obj.Show();
   this.Hide();
}

private void Attendance_Click(object sender, EventArgs e)
{
   Attendance Obj = new Attendance();
   Obj.Show();
   this.Hide();
}
```

```
private void Sal_Click(object sender, EventArgs e)
{
   Salary Obj = new Salary();
   Obj.Show();
   this.Hide();
}

private void Rep_Click(object sender, EventArgs e)
{
   Report Obj = new Report();
   Obj.Show();
   this.Hide();
}

private void homic_Click(object sender, EventArgs e)
{
   Home Obj = new Home();
   Obj.Show();
   this.Hide();
}

private void Empic_Click(object sender, EventArgs e)
{
   Employees Obj = new Employees();
   Obj.Show();
   this.Hide();
}

private void Bonic_Click(object sender, EventArgs e)
{
   Bonus Obj = new Bonus();
   Obj.Show();
```

```
    this.Hide();
}

private void Attic_Click(object sender, EventArgs e)
{
    Attendance Obj = new Attendance();
    Obj.Show();
    this.Hide();
}

private void Salic_Click(object sender, EventArgs e)
{
    Salary Obj = new Salary();
    Obj.Show();
    this.Hide();
}

private void repic_Click(object sender, EventArgs e)
{
    Report Obj = new Report();
    Obj.Show();
    this.Hide();
}

private void logic_Click(object sender, EventArgs e)
{
    Login Obj = new Login();
    Obj.Show();
    this.Hide();
}
```

```
        private void label4_Click(object sender, EventArgs e)
        {
            Home Obj = new Home();
            Obj.Show();
            this.Hide();
        }

        private void label2_Click(object sender, EventArgs e)
        {
            Feedback Obj = new Feedback();
            Obj.Show();
            this.Hide();
        }

        private void pictureBox3_Click(object sender, EventArgs e)
        {
            Feedback Obj = new Feedback();
            Obj.Show();
            this.Hide();
        }
    }
}
```

Admin Employee Attendance Page

Attendance.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace PayrollSystem
{
    public partial class Attendance : Form
    {
        SqlConnection Con = new SqlConnection(@"Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\Trials\PayrollSystem\PayRol
lDB.mdf;Integrated Security=True;Connect Timeout=30");
        public Attendance()
        {
            InitializeComponent();
            ShowAttendance();
            GetEmployees();
        }

        private void pictureBox9_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

         private void Clear()
        {

            EmpIdCb.SelectedIndex = -1;
            EmpNameTb.Text = "";
            PresentTb.Text = "";
            AbsTb.Text = "";
```

```
    ExcusedTb.Text = "";
}

private void ShowAttendance()
{
    Con.Open();
    string Query = "select * from AttendanceTbl";
    SqlDataAdapter sda = new SqlDataAdapter(Query, Con);
    SqlCommandBuilder Builder = new SqlCommandBuilder(sda);
    var ds = new DataSet();
    sda.Fill(ds);
    AttDGV.DataSource = ds.Tables[0];
    Con.Close();
}

private void GetEmployees()
{
    Con.Open();
    SqlCommand cmd = new SqlCommand("select * from EmployeeTbl",Con);
    SqlDataReader Rdr;
    Rdr = cmd.ExecuteReader();
    DataTable dt = new DataTable();
    dt.Columns.Add("EmpId",typeof(int));
    dt.Load(Rdr);
    EmpIdCb.ValueMember = "EmpId";
    EmpIdCb.DataSource = dt;
    Con.Close();
}
private void getEmployeeNamee()
{
    if (Con.State == ConnectionState.Closed)
    {
```

```
            Con.Open();
        }
        string Query = "select EmpName from EmployeeTbl where EmpId= " + EmpIdCb.SelectedValue.ToString() +" ";
        SqlCommand cmd = new SqlCommand(Query, Con);
        DataTable dt = new DataTable();
        SqlDataAdapter sda = new SqlDataAdapter(cmd);
        sda.Fill(dt);
        if (dt != null && dt.Rows.Count > 0)
        {
            foreach (DataRow dr in dt.Rows)
            {
                EmpNameTb.Text = dr["EmpName"].ToString();
            }
        }
        Con.Close();
    }
    private void SaveBtn_Click(object sender, EventArgs e)
    {
        if (EmpNameTb.Text == "" || PresentTb.Text == "" || AbsTb.Text == "" || ExcusedTb.Text == "" )
        {
            MessageBox.Show("Missing Information!");
        }
        else
        {
            try
            {
                string Periodd = AttDate.Value.Month + "-" + AttDate.Value.Year;
                Con.Open();
                SqlCommand cmd = new SqlCommand("insert into AttendanceTbl(EmpId,EmpName,DayPres,DayAbs,DayExcused,Period)
```

```
values(@EI,@EN,@DP,@DA,@DE,@PD)", Con);
                cmd.Parameters.AddWithValue("@EI", EmpIdCb.Text);
                cmd.Parameters.AddWithValue("@EN", EmpNameTb.Text);
                cmd.Parameters.AddWithValue("@DP", PresentTb.Text);
                cmd.Parameters.AddWithValue("@DA", AbsTb.Text);
                cmd.Parameters.AddWithValue("@DE", ExcusedTb.Text);
                cmd.Parameters.AddWithValue("@PD", Periodd);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Attendance Saved!!");
                Con.Close();
                ShowAttendance();
                Clear();
            }
            catch (Exception Ex)
            {
                MessageBox.Show(Ex.Message);
            }
        }
    }

    int Key;
    private void AttDGV_CellContentClick(object sender, DataGridViewCellEventArgs
e)
    {
        EmpIdCb.Text = AttDGV.SelectedRows[0].Cells[1].Value.ToString();
        EmpNameTb.Text = AttDGV.SelectedRows[0].Cells[2].Value.ToString();
        PresentTb.Text = AttDGV.SelectedRows[0].Cells[3].Value.ToString();
        AbsTb.Text = AttDGV.SelectedRows[0].Cells[4].Value.ToString();
        ExcusedTb.Text = AttDGV.SelectedRows[0].Cells[5].Value.ToString();
        AttDate.Value = Convert.ToDateTime(AttDGV.SelectedRows[0].Cells[6].Value);
```

```
            if (EmpNameTb.Text == "")
            {
                Key = 0;
            }

            else
            {
                Key = Convert.ToInt32(AttDGV.SelectedRows[0].Cells[0].Value.ToString());
            }
        }

        private void EmpIdCb_SelectedValueChanged(object sender, EventArgs e)
        {
            if(EmpIdCb.SelectedValue != null)
            getEmployeeNamee();
        }

        private void UpdateBtn_Click(object sender, EventArgs e)
        {
            if (EmpNameTb.Text == "" || PresentTb.Text == "" || AbsTb.Text == "" ||
ExcusedTb.Text == "")
            {
                MessageBox.Show("Missing Information!");
            }
            else
            {
                try
                {
                    string Periodd = AttDate.Value.Month + "-" + AttDate.Value.Year;
                    Con.Open();
                    SqlCommand cmd = new SqlCommand("update AttendanceTbl set
EmpId=@EI,EmpName=@EN,DayPres=@DP,DayAbs=@DA,DayExcused=@DE,Period
```

```
=@PD where AttNum=@AKey", Con);
                cmd.Parameters.AddWithValue("@EI", EmpIdCb.Text);
                cmd.Parameters.AddWithValue("@EN", EmpNameTb.Text);
                cmd.Parameters.AddWithValue("@DP", PresentTb.Text);
                cmd.Parameters.AddWithValue("@DA", AbsTb.Text);
                cmd.Parameters.AddWithValue("@DE", ExcusedTb.Text);
                cmd.Parameters.AddWithValue("@PD", Periodd);
                cmd.Parameters.AddWithValue("@AKey", Key);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Attendance Updated!!");
                Con.Close();
                ShowAttendance();
                Clear();
            }
            catch (Exception Ex)
            {
                MessageBox.Show(Ex.Message);
            }
        }
    }
    private void pictureBox1_Click(object sender, EventArgs e)
    {
        Home Objh = new Home();
        Objh.Show();
        this.Hide();
    }

    private void pictureBox8_Click(object sender, EventArgs e)
    {
        Login Objl = new Login();
        Objl.Show();
        this.Hide();        }
```

```
private void label1_Click_1(object sender, EventArgs e)
{
   Home Obj = new Home();
   Obj.Show();
   this.Hide();
}

private void label2_Click(object sender, EventArgs e)
{
   Employees Obj = new Employees();
   Obj.Show();
   this.Hide();
}

private void label3_Click(object sender, EventArgs e)
{
   Bonus Obj = new Bonus();
   Obj.Show();
   this.Hide();
}

private void Sal_Click(object sender, EventArgs e)
{
   Salary Obj = new Salary();
   Obj.Show();
   this.Hide();
}

private void Rep_Click(object sender, EventArgs e)
{
   Report Obj = new Report();
   Obj.Show();
```

```
            this.Hide();
        }
        private void pictureBox2_Click(object sender, EventArgs e)
        {
            Home Obj = new Home();
            Obj.Show();
            this.Hide();
        }

        private void label4_Click(object sender, EventArgs e)
        {
            Home Obj = new Home();
            Obj.Show();
            this.Hide();
        }

        private void DelBtn_Click(object sender, EventArgs e)
        {
            if (Key == 0)
            {
                MessageBox.Show("Missing Information");
            }
            else
            {
                try
                {
                    Con.Open();
                    SqlCommand cmd = new SqlCommand("delete from AttendanceTbl where AttNum=@AKey", Con);
                    cmd.Parameters.AddWithValue("@AKey", Key);
                    cmd.ExecuteNonQuery();
                    MessageBox.Show("Employee Deleted!!");
```

```
            Con.Close();
            ShowAttendance();
        }
        catch (Exception Ex)
        {
            MessageBox.Show(Ex.Message);
        }
    }
}

private void homic_Click(object sender, EventArgs e)
{
    Home Obj = new Home();
    Obj.Show();
    this.Hide();
}

private void Empic_Click(object sender, EventArgs e)
{
    Employees Obj = new Employees();
    Obj.Show();
    this.Hide();
}

private void Bonic_Click(object sender, EventArgs e)
{
    Bonus Obj = new Bonus();
    Obj.Show();
    this.Hide();
}
```

```
private void Attic_Click(object sender, EventArgs e)
{
   Attendance Obj = new Attendance();
   Obj.Show();
   this.Hide();
}

private void Salic_Click(object sender, EventArgs e)
{
   Salary Obj = new Salary();
   Obj.Show();
   this.Hide();
}

private void repic_Click(object sender, EventArgs e)
{
   Report Obj = new Report();
   Obj.Show();
   this.Hide();
}
private void logic_Click(object sender, EventArgs e)
{
   Login Obj = new Login();
   Obj.Show();
   this.Hide();
}

private void pictureBox3_Click(object sender, EventArgs e)
{
   Feedback Obj = new Feedback();
   Obj.Show();
   this.Hide();
```

```
        }
        private void label13_Click(object sender, EventArgs e)
        {
            Feedback Obj = new Feedback();
            Obj.Show();
            this.Hide();
        }
    }
}
```

Admin Employee Bonus Page

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace PayrollSystem
{
    public partial class Bonus : Form
    {
        public Bonus()
        {
            InitializeComponent();
            showBonus();
        }
```

```
    SqlConnection Con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\Trials\PayrollSystem\PayRollDB.mdf;Integrated Security=True;Connect Timeout=30");
    private void Clear()
    {
        BNameTb.Text = "";
        BAmtTb.Text = "";
        Key = 0;
    }
    private void showBonus()
    {
        Con.Open();
        string Query = "select * from BonusTbl";
        SqlDataAdapter sda = new SqlDataAdapter(Query, Con);
        SqlCommandBuilder Builder = new SqlCommandBuilder(sda);
        var ds = new DataSet();
        sda.Fill(ds);
        BonusDGV.DataSource = ds.Tables[0];
        Con.Close();
    }
    private void SaveBtn_Click(object sender, EventArgs e)
    {
        if (BNameTb.Text == "" || BAmtTb.Text == "")
        {
            MessageBox.Show("Missing Information");
        }
        else
        {
            try
            {
                Con.Open();
                SqlCommand cmd = new SqlCommand("insert into BonusTbl(BName,BAmt)
```

```
values(@BN,@BA)", Con);
                cmd.Parameters.AddWithValue("@BN", BNameTb.Text);
                cmd.Parameters.AddWithValue("@BA", BAmtTb.Text);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Bonus Saved!!");
                Con.Close();
                showBonus();
                Clear();
            }
            catch (Exception Ex)
            {
                MessageBox.Show(Ex.Message);
            }
        }
    }

    int Key;
     private void EditBtn_Click(object sender, EventArgs e)
    {
        if (BNameTb.Text == "" || BAmtTb.Text == "")
        {
            MessageBox.Show("Missing Information");
        }
        else
        {
            try
            {
                Con.Open();
                SqlCommand cmd = new SqlCommand("update BonusTbl set BName=@BN,
BAmt=@BA where BId=@BKey", Con);
                cmd.Parameters.AddWithValue("@BN", BNameTb.Text);
                cmd.Parameters.AddWithValue("@BA", BAmtTb.Text);
```

```
                cmd.Parameters.AddWithValue("@BKey", Key);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Bonus Updated!!");
                Con.Close();
                showBonus();
                Clear();
            }
            catch (Exception Ex)
            {
                MessageBox.Show(Ex.Message);
            }
        }
    }

    private void pictureBox2_Click(object sender, EventArgs e)
    {
        this.Hide();
    }

    private void DelBtn_Click(object sender, EventArgs e)
    {
        if (Key == 0)
        {
            MessageBox.Show("Select the Bonus to be deleted!");
        }
        else
        {
            try
            {
                Con.Open();
                SqlCommand cmd = new SqlCommand("delete from BonusTbl where BId=@BKey",
Con);
```

```
            cmd.Parameters.AddWithValue("@BKey", Key);
            cmd.ExecuteNonQuery();
            MessageBox.Show("Bonus Deleted!!");
            Con.Close();
            showBonus();
           }
        catch (Exception Ex)
        {
            MessageBox.Show(Ex.Message);
        }
    }
}

 private void BonusDGV_CellContentClick_1(object sender, DataGridViewCellEventArgs e)
 {
    BNameTb.Text = BonusDGV.SelectedRows[0].Cells[1].Value.ToString();
    BAmtTb.Text = BonusDGV.SelectedRows[0].Cells[2].Value.ToString();
    if (BNameTb.Text == "")
    {
        Key = 0;
    }
    else
    {
        Key = Convert.ToInt32(BonusDGV.SelectedRows[0].Cells[0].Value.ToString());
    }
 }

private void pictureBox1_Click_1(object sender, EventArgs e)
 {
    Home Obj = new Home();
    Obj.Show();
    this.Hide();        }
```

```
private void pictureBox8_Click(object sender, EventArgs e)
{
   Login Obj = new Login();
   Obj.Show();
   this.Hide();
}

private void Home_Click(object sender, EventArgs e)
{
   Home Obj = new Home();
   Obj.Show();
   this.Hide();
}

private void Emp_Click(object sender, EventArgs e)
{
   Employees Obj = new Employees();
   Obj.Show();
   this.Hide();
}

private void Attendance_Click(object sender, EventArgs e)
{
   Attendance Obj = new Attendance();
   Obj.Show();
   this.Hide();
}

private void Sal_Click(object sender, EventArgs e)
{
   Salary Obj = new Salary();
   Obj.Show();
```

```
    this.Hide();
}

private void Rep_Click(object sender, EventArgs e)
{
    Report Obj = new Report();
    Obj.Show();
    this.Hide();
}

private void homic_Click(object sender, EventArgs e)
{
    Home Obj = new Home();
    Obj.Show();
    this.Hide();
}

private void Empic_Click(object sender, EventArgs e)
{
    Employees Obj = new Employees();
    Obj.Show();
    this.Hide();
}

private void Bonic_Click(object sender, EventArgs e)
{
    Bonus Obj = new Bonus();
    Obj.Show();
    this.Hide();
}
```

```
private void Attic_Click(object sender, EventArgs e)
{
   Attendance Obj = new Attendance();
   Obj.Show();
   this.Hide();
}

private void Salic_Click(object sender, EventArgs e)
{
   Salary Obj = new Salary();
   Obj.Show();
   this.Hide();
}

private void repic_Click(object sender, EventArgs e)
{

   Report Obj = new Report();
   Obj.Show();
   this.Hide();
}

private void logic_Click(object sender, EventArgs e)
{
   Login Obj = new Login();
   Obj.Show();
   this.Hide();
}

private void label4_Click(object sender, EventArgs e)
{
   Home Obj = new Home();
```

```
            Obj.Show();
            this.Hide();
        }

        private void pictureBox3_Click(object sender, EventArgs e)
        {
            Feedback Obj = new Feedback();
            Obj.Show();
            this.Hide();
        }

        private void label1_Click(object sender, EventArgs e)
        {
            Feedback Obj = new Feedback();
            Obj.Show();
            this.Hide();
        }
    }
}
```

Admin Issue Salary Page

Salary.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
```

```
namespace PayrollSystem
{
    public partial class Salary : Form
    {
        public Salary()
        {
            InitializeComponent();
            GetEmployees();
            GetAttendance();
            GetBonus();
            ShowSalary();
        }
        SqlConnection Con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\Trials\PayrollSystem\PayRollDB.mdf;Integrated Security=True;Connect Timeout=30");

        private void pictureBox9_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void Clear()
        {

            EmpIdCb.SelectedIndex = -1;
            EmpNameTb.Text = "";
            PresTb.Text = "";
            AbsTb.Text = "";
            ExcTb.Text = "";
        }

        private void ShowSalary()
```

```
        {
            Con.Open();
            string Query = "select
EmpId,EmpName,EmpBasSal,EmpBonus,HR,DA,TA,EPres,EAbs,EExcused,EAttSal,EmpTax,
EmpDeduction,EmpBalance,SalPeriod from SalaryTbl";
            SqlDataAdapter sda = new SqlDataAdapter(Query, Con);
            SqlCommandBuilder Builder = new SqlCommandBuilder(sda);
            var ds = new DataSet();
            sda.Fill(ds);
            SalDGV.DataSource = ds.Tables[0];
            Con.Close();
        }

        private void GetEmployees()
        {
            //to fetch data
            Con.Open();
            SqlCommand cmd = new SqlCommand("select * from EmployeeTbl", Con);
            SqlDataReader Rdr;
            Rdr = cmd.ExecuteReader();
            DataTable dt = new DataTable();
            dt.Columns.Add("EmpId", typeof(int));
            dt.Load(Rdr);
            EmpIdCb.ValueMember = "EmpId";
            EmpIdCb.DataSource = dt;
            Con.Close();
        }
        private void GetBonus()
        {
            Con.Open();
            SqlCommand cmd = new SqlCommand("select * from BonusTbl", Con);
            SqlDataReader Rdr;
```

```
            Rdr = cmd.ExecuteReader();
            DataTable dt = new DataTable();
            dt.Columns.Add("BName", typeof(string));
            dt.Load(Rdr);
            BonusIdCb.ValueMember = "BName";
            BonusIdCb.DataSource = dt;
            Con.Close();
        }
        private void GetAttendance()
        {
            Con.Open();
            SqlCommand cmd = new SqlCommand("select * from AttendanceTbl where EmpId="+
EmpIdCb.SelectedValue.ToString()+ " ", Con);
            SqlDataReader Rdr;
            Rdr = cmd.ExecuteReader();
            DataTable dt = new DataTable();
            dt.Columns.Add("AttNum", typeof(int));
            dt.Load(Rdr);
            AttNumCb.ValueMember = "AttNum";
            AttNumCb.DataSource = dt;
            Con.Close();
        }

        private void GetAttendanceData()
        {
            //to auto-fill
            if (Con.State == ConnectionState.Closed)
            {
                Con.Open();
            }
            string Query = "select * from AttendanceTbl where AttNum= " +
AttNumCb.SelectedValue.ToString() + " ";
```

```
            SqlCommand cmd = new SqlCommand(Query, Con);
            DataTable dt = new DataTable();
            SqlDataAdapter sda = new SqlDataAdapter(cmd);
            sda.Fill(dt);
            if (dt != null && dt.Rows.Count > 0)
            {
                foreach (DataRow dr in dt.Rows)
                {
                    PresTb.Text = dr["DayPres"].ToString();
                    AbsTb.Text = dr["DayAbs"].ToString();
                    ExcTb.Text = dr["DayExcused"].ToString();
                }
            }
            Con.Close();
        }
        private void getEmployeeNamee()
        {
            //to auto-fill
            if (Con.State == ConnectionState.Closed)
            {
                Con.Open();
            }
            string Query = "select EmpName,EmpBasSal from EmployeeTbl where EmpId= " + EmpIdCb.SelectedValue.ToString() + " ";
            SqlCommand cmd = new SqlCommand(Query, Con);
            DataTable dt = new DataTable();
            SqlDataAdapter sda = new SqlDataAdapter(cmd);
            sda.Fill(dt);
            if (dt != null && dt.Rows.Count > 0)
            {
                foreach (DataRow dr in dt.Rows)
                {
```

```
                EmpNameTb.Text = dr["EmpName"].ToString();
                BaseSalTb.Text = dr["EmpBasSal"].ToString();
            }
        }
        Con.Close();
    }

    private void getBonusAmt()
    {
        Con.Open();
        string Query = "select * from BonusTbl where BName= '" + BonusIdCb.SelectedValue.ToString() + "' ";
        SqlCommand cmd = new SqlCommand(Query, Con);
        DataTable dt = new DataTable();
        SqlDataAdapter sda = new SqlDataAdapter(cmd);
        sda.Fill(dt);
        if (dt != null && dt.Rows.Count > 0)
        {
            foreach (DataRow dr in dt.Rows)
            {
                BonusTb.Text = dr["BAmt"].ToString();
                //BonusTb.Text = dr["BAmt"].ToString();
            }
        }
        Con.Close();
    }
    private void SaveBtn_Click(object sender, EventArgs e)
    {
        if (EmpNameTb.Text == "" || PresTb.Text == "" || AbsTb.Text == "" || ExcTb.Text == "")
        {
            MessageBox.Show("Missing Information!");
        }
```

```
        else
        {
            try
            {
                string Period = SalDate.Value.Month + "-" + SalDate.Value.Year;
                Con.Open();
                SqlCommand cmd = new SqlCommand("insert into
SalaryTbl(EmpId,EmpName,EmpBasSal,EmpBonus,EmpDeduction,EmpTax,EmpBalance,SalPe
riod,HR,DA,TA,EPres,EAbs,EExcused,EAttSal)
values(@EI,@EN,@EBS,@EBon,@EA,@ET,@EBal,@SP,@HR,@DA,@TA,@EP,@EAbs,@
EEx,@EAttSal)", Con);
                cmd.Parameters.AddWithValue("@EI", EmpIdCb.Text);
                cmd.Parameters.AddWithValue("@EN", EmpNameTb.Text);
                cmd.Parameters.AddWithValue("@EBS", BaseSalTb.Text);
                cmd.Parameters.AddWithValue("@EBon", BonusTb.Text);
                cmd.Parameters.AddWithValue("@EA", DedTb.Text);
                cmd.Parameters.AddWithValue("@ET", TotTax);
                cmd.Parameters.AddWithValue("@EBal", GrdTot);
                cmd.Parameters.AddWithValue("@SP", Period);
                cmd.Parameters.AddWithValue("@HR", hr);
                cmd.Parameters.AddWithValue("@DA", da);
                cmd.Parameters.AddWithValue("@TA", ta);
                cmd.Parameters.AddWithValue("@EP", PresTb.Text);
                cmd.Parameters.AddWithValue("@EAbs", AbsTb.Text);
                cmd.Parameters.AddWithValue("@EEx", ExcTb.Text);
                cmd.Parameters.AddWithValue("@EAttSal", AttSal);
                cmd.ExecuteNonQuery();
                MessageBox.Show("Salary Saved!!");
                Con.Close();
                ShowSalary();
                Clear();
            }
```

```
            catch (Exception Ex)
            {
                MessageBox.Show(Ex.Message);
            }
        }
    }

    private void EmpIdCb_SelectionChangeCommitted(object sender, EventArgs e)
    { //when option is chnged,fetch data
        getEmployeeNamee();
        GetAttendance();
        GetAttendanceData();
    }

    private void BonusIdCb_SelectionChangeCommitted(object sender, EventArgs e)
    {
        getBonusAmt();
    }

    private void AttNumCb_SelectionChangeCommitted(object sender, EventArgs e)
    {
        GetAttendanceData();
    }

    int DailyBase = 0, Pres = 0, Abs = 0, Exc = 0;
    double hr = 0, ta = 0, da = 0,AttSal=0,Total=0,DGrdTot=0;

    private void printDocument1_PrintPage(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
    {
        e.Graphics.DrawString("PayIt Ltd", new Font("Arial", 12, FontStyle.Bold),
Brushes.BlueViolet, new Point(200, 65));
```

```
e.Graphics.DrawString("Employee PaySlip", new Font("Arial", 10, FontStyle.Bold), Brushes.Blue, new Point(180, 85));
//string SNum = SalDGV.SelectedRows[0].Cells[0].Value.ToString();
string EId = SalDGV.SelectedRows[0].Cells[0].Value.ToString();
string Name = SalDGV.SelectedRows[0].Cells[1].Value.ToString();
string BasSal = SalDGV.SelectedRows[0].Cells[2].Value.ToString();
string Bonus = SalDGV.SelectedRows[0].Cells[3].Value.ToString();
string Deduction = SalDGV.SelectedRows[0].Cells[12].Value.ToString();
string Tax = SalDGV.SelectedRows[0].Cells[11].Value.ToString();
string Balance = SalDGV.SelectedRows[0].Cells[13].Value.ToString();
string Period = SalDGV.SelectedRows[0].Cells[14].Value.ToString();
string hr = SalDGV.SelectedRows[0].Cells[4].Value.ToString();
string da = SalDGV.SelectedRows[0].Cells[5].Value.ToString();
string ta = SalDGV.SelectedRows[0].Cells[6].Value.ToString();
string Pres = SalDGV.SelectedRows[0].Cells[7].Value.ToString();
string Abs = SalDGV.SelectedRows[0].Cells[8].Value.ToString();
string Exc = SalDGV.SelectedRows[0].Cells[9].Value.ToString();
string AttSalary = SalDGV.SelectedRows[0].Cells[10].Value.ToString();
e.Graphics.DrawString("Employee ID: " + EId, new Font("Arial", 10, FontStyle.Bold), Brushes.Black, new Point(50, 150));
e.Graphics.DrawString("Employee Name: " + Name, new Font("Arial", 10, FontStyle.Bold), Brushes.Black, new Point(250, 150));
e.Graphics.DrawString("_____", new Font("Arial", 10, FontStyle.Bold), Brushes.Black, new Point(55, 170));
e.Graphics.DrawString("No of days worked: " + Pres + "    No of applied leaves: " + Abs + "    No of Paid leaves: " + Exc, new Font("Arial", 8, FontStyle.Bold), Brushes.Black, new Point(40, 205));
e.Graphics.DrawString("_____", new Font("Arial", 10, FontStyle.Bold), Brushes.Black, new Point(55, 220));
e.Graphics.DrawString("Base Salary: Rs " + BasSal, new Font("Arial", 8, FontStyle.Bold), Brushes.Black, new Point(50, 250));
e.Graphics.DrawString("HR: Rs " + hr, new Font("Arial", 8, FontStyle.Bold),
```

```
Brushes.Black, new Point(50, 270));
        e.Graphics.DrawString("DA: Rs " + da, new Font("Arial", 8, FontStyle.Bold),
Brushes.Black, new Point(50, 290));
        e.Graphics.DrawString("TA: Rs " + ta, new Font("Arial", 8, FontStyle.Bold),
Brushes.Black, new Point(50, 310));
        e.Graphics.DrawString("Attendance Salary: Rs " + AttSalary, new Font("Arial", 8,
FontStyle.Bold), Brushes.Black, new Point(50, 330));
        e.Graphics.DrawString("Bonus: Rs " + Bonus, new Font("Arial", 8, FontStyle.Bold),
Brushes.Black, new Point(50, 350));
        e.Graphics.DrawString("_____", new
Font("Arial", 10, FontStyle.Bold), Brushes.Black, new Point(55, 370));

        e.Graphics.DrawString("Tax: Rs " + Tax, new Font("Arial", 8, FontStyle.Bold),
Brushes.Black, new Point(50, 390));
        e.Graphics.DrawString("Total Deduction: " + string.Format("{0:F3}", Deduction), new
Font("Arial", 9, FontStyle.Bold), Brushes.Black, new Point(50, 410));
        e.Graphics.DrawString("_____", new
Font("Arial", 10, FontStyle.Bold), Brushes.Black, new Point(55, 430));
        e.Graphics.DrawString("Total: Rs " + Balance, new Font("Arial", 10, FontStyle.Bold),
Brushes.Blue, new Point(50, 470));
        e.Graphics.DrawString("_____", new
Font("Arial", 10, FontStyle.Bold), Brushes.Black, new Point(55, 490));
        e.Graphics.DrawString("Date: " + Period, new Font("Arial", 8, FontStyle.Bold),
Brushes.Black, new Point(50, 520));
    }
    private void pictureBox1_Click(object sender, EventArgs e)
    {
        Home Objh = new Home();
        Objh.Show();
        this.Hide();
    }
```

```
        private void Emp_Click(object sender, EventArgs e)
        {
            Employees Obj = new Employees();
            Obj.Show();
            this.Hide();
        }

        private void Bonus_Click(object sender, EventArgs e)
        {
            Bonus Obj = new Bonus();
            Obj.Show();
            this.Hide();
        }

        private void Attendance_Click(object sender, EventArgs e)
        {
            Attendance Obj = new Attendance();
            Obj.Show();
            this.Hide();
        }

        private void Rep_Click(object sender, EventArgs e)
        {
            Report Obj = new Report();
            Obj.Show();
            this.Hide();
        }
    private void pictureBox2_Click(object sender, EventArgs e)
        {
            Home Obj = new Home();
            Obj.Show();
            this.Hide();        }
```

```
private void Empic_Click(object sender, EventArgs e)
{
   Employees Obj = new Employees();
   Obj.Show();
   this.Hide();
}

private void Bonic_Click(object sender, EventArgs e)
{
   Bonus Obj = new Bonus();
   Obj.Show();
   this.Hide();
}

private void Attic_Click(object sender, EventArgs e)
{
   Attendance Obj = new Attendance();
   Obj.Show();
   this.Hide();
}

private void Salic_Click(object sender, EventArgs e)
{
   Salary Obj = new Salary();
   Obj.Show();
   this.Hide();
}

private void repic_Click(object sender, EventArgs e)
{
   Report Obj = new Report();
   Obj.Show();
```

```
            this.Hide();
        }
    private void label25_Click(object sender, EventArgs e)
        {
            Feedback Obj = new Feedback();
            Obj.Show();
            this.Hide();
        }

        private void pictureBox3_Click(object sender, EventArgs e)
        {
            Feedback Obj = new Feedback();
            Obj.Show();
            this.Hide();
        }

        private void pictureBox8_Click_1(object sender, EventArgs e)
        {
            Login Objl = new Login();
            Objl.Show();
            this.Hide();
        }

        private void Home_Click(object sender, EventArgs e)
        {
            Home Obj = new Home();
            Obj.Show();
            this.Hide();
        }
    private void pictureBox8_Click(object sender, EventArgs e)
        {
            Login Objl = new Login();
```

```
            Objl.Show();
            this.Hide();
        }

private void SalDGV_CellContentClick(object sender, DataGridViewCellEventArgs e)
        {
            printDocument1.DefaultPageSettings.PaperSize=new
System.Drawing.Printing.PaperSize("pprnm",450,600);
            if (printPreviewDialog1.ShowDialog() == DialogResult.OK)
            {
                printDocument1.Print();
            }
        }

        double GrdTot = 0, TotTax = 0,Deduct=0;
        private void ComputeBtn_Click(object sender, EventArgs e)
        {
            if (BaseSalTb.Text == "" || BonusTb.Text == "")
            {
                MessageBox.Show("Select the Employee");
            }
            else
            {
                hr = Convert.ToInt32(BaseSalTb.Text) * 0.10;
                ta = Convert.ToInt32(BaseSalTb.Text) * 0.25;
                da = Convert.ToInt32(BaseSalTb.Text) * 0.05;
                Pres = Convert.ToInt32(PresTb.Text);
                Abs = Convert.ToInt32(AbsTb.Text);
                Exc = Convert.ToInt32(ExcTb.Text);
                DailyBase = Convert.ToInt32(BaseSalTb.Text)/28;
                AttSal = ((DailyBase) * Pres) + ((DailyBase / 2) * Exc);
                Total = AttSal + hr + da + ta;
```

```
        double Tax = Total * 0.16;
        TotTax = Total - Tax;
        GrdTot = TotTax + Convert.ToInt32(BonusTb.Text);

        //deducting
        if (GrdTot > 50000 && GrdTot < 40000)
           DGrdTot = GrdTot - (GrdTot * 0.30);
        else if (GrdTot > 40000 && GrdTot < 20000)
           DGrdTot = GrdTot - (GrdTot * 0.20);
        else if (GrdTot > 20000 && GrdTot < 10000)
           DGrdTot = GrdTot - (GrdTot * 0.10);
        else
           DGrdTot = GrdTot - (GrdTot * 0.05);

        Deduct = GrdTot - DGrdTot;

        BalanceTb.Text = "Rs " + DGrdTot;
        DedTb.Text = "Rs " + Deduct;
        hra.Text = "Rs " + hr;
        textBox2.Text = "Rs " + da;
        textBox3.Text = "Rs " + ta;
    } }    }  }
```

Admin View Feedback Page

Feedback.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
```

```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PayrollSystem
{
    public partial class Feedback : Form
    {
        public Feedback()
        {
            InitializeComponent();
            ShowData();
        }

        SqlConnection Con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\Trials\PayrollSystem\PayRollDB.mdf;Integrated Security=True;Connect Timeout=30");

        private void ShowData()
        {
            Con.Open();
            SqlDataAdapter sda = new SqlDataAdapter("select * from FeedbackTbl", Con);
            SqlCommandBuilder Builder = new SqlCommandBuilder(sda);
            var ds = new DataSet();
            sda.Fill(ds);
            FeedbackDGV.DataSource = ds.Tables[0];
            Con.Close();
        }
        private void SubmitBtn_Click(object sender, EventArgs e)
        {
            Con.Open();
            //string Periodd = Date.Value.Month + "-" + Date.Value.Year;
```

```
        //SqlDataAdapter sda = new SqlDataAdapter("select * from FeedbackTbl where Period ='"
+ Periodd + "' ", Con);
        SqlDataAdapter sda = new SqlDataAdapter("select * from FeedbackTbl where (SELECT
YEAR(Period) AS year) = '" + Date.Value.Year + "' and (SELECT MONTH(Period) AS month)
= '" + Date.Value.Month + "' ", Con);
        SqlCommandBuilder Builder = new SqlCommandBuilder(sda);
        var ds = new DataSet();
        sda.Fill(ds);
        FeedbackDGV.DataSource = ds.Tables[0];
        Con.Close();
    }

    private void homic_Click(object sender, EventArgs e)
    {
        Home Obj = new Home();
        Obj.Show();
        this.Hide();
    }

    private void Empic_Click(object sender, EventArgs e)
    {
        Employees Obj = new Employees();
        Obj.Show();
        this.Hide();
    }

    private void Bonic_Click(object sender, EventArgs e)
    {
        Bonus Obj = new Bonus();
        Obj.Show();
        this.Hide();
    }
```

```
private void Attic_Click(object sender, EventArgs e)
{
   Attendance Obj = new Attendance();
   Obj.Show();
   this.Hide();
}

private void Salic_Click(object sender, EventArgs e)
{
   Salary Obj = new Salary();
   Obj.Show();
   this.Hide();
}

private void repic_Click(object sender, EventArgs e)
{
   Report Obj = new Report();
   Obj.Show();
   this.Hide();
}

private void logic_Click(object sender, EventArgs e)
{
   Login Obj = new Login();
   Obj.Show();
   this.Hide();
}

private void Home_Click(object sender, EventArgs e)
{
   Home Obj = new Home();
   Obj.Show();
```

```
    this.Hide();
}

private void Emp_Click(object sender, EventArgs e)
{
    Employees Obj = new Employees();
    Obj.Show();
    this.Hide();
}

private void Bonus_Click(object sender, EventArgs e)
{
    Bonus Obj = new Bonus();
    Obj.Show();
    this.Hide();
}

private void Attendance_Click(object sender, EventArgs e)
{
    Attendance Obj = new Attendance();
    Obj.Show();
    this.Hide();
}

private void Rep_Click(object sender, EventArgs e)
{
    Report Obj = new Report();
    Obj.Show();
    this.Hide();
}
```

```
        private void Sal_Click(object sender, EventArgs e)
        {
            Salary Obj = new Salary();
            Obj.Show();
            this.Hide();
        }
    }
}
```

User View Salary Page

UserSal.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PayrollSystem
{
    public partial class UserSal : Form
    {
        public UserSal()
        {
            InitializeComponent();
            UserNamee.Text = Login.Users;
            ShowYearly();
        }
```

```
SqlConnection Con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\Trials\PayrollSystem\PayRollDB.mdf;Integrated Security=True;Connect Timeout=30");
    private void pictureBox8_Click(object sender, EventArgs e)
    {
        Login Objl = new Login();
        Objl.Show();
        this.Hide();
    }

    private void pictureBox2_Click(object sender, EventArgs e)
    {
        UserSal Obj = new UserSal();
        Obj.Show();
        this.Hide();       }

    private void ShowYearly()
    {
        Con.Open();
        string Query = "select SalPeriod,EmpBonus,HR,DA,TA,EPres,EAbs,EExcused,EAttSal,EmpTax,EmpDeduction,EmpBalance from SalaryTbl where EmpName='" + UserNamee.Text + "' ";
        SqlDataAdapter sda = new SqlDataAdapter(Query, Con);
        SqlCommandBuilder Builder = new SqlCommandBuilder(sda);
        var ds = new DataSet();
        sda.Fill(ds);
        YearlyDGV.DataSource = ds.Tables[0];
        Con.Close();
    }

    private void SubmitBtn_Click(object sender, EventArgs e)
    {
```

```
            Con.Open();
            string Periodd = SalDate.Value.Month + "-" + SalDate.Value.Year;
            SqlDataAdapter sda = new SqlDataAdapter("select SalPeriod,EmpBonus,HR,DA,TA,EPres,EAbs,EExcused,EAttSal,EmpTax,EmpDeduction,EmpBalance from SalaryTbl where SalPeriod ='" + Periodd + "' and  EmpName='" + UserNamee.Text + "' ", Con);
            SqlCommandBuilder Builder = new SqlCommandBuilder(sda);
            var ds = new DataSet();
            sda.Fill(ds);
            YearlyDGV.DataSource = ds.Tables[0];
            Con.Close();
        }

        private void Sal_Click(object sender, EventArgs e)
        {
            UserSal Obj = new UserSal();
            Obj.Show();
            this.Hide();
        }

        private void HomeLbl_Click(object sender, EventArgs e)
        {
            UserHome Obj = new UserHome();
            Obj.Show();
            this.Hide();
        }

        private void Profile_Click(object sender, EventArgs e)
        {
            UserProfile Obj = new UserProfile();
            Obj.Show();
```

```
            this.Hide();
        }

        private void repic_Click(object sender, EventArgs e)
        {
            UserProfile Obj = new UserProfile();
            Obj.Show();
            this.Hide();
        }

        private void pictureBox9_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void label4_Click(object sender, EventArgs e)
        {
            UserFeedback Obj = new UserFeedback();
            Obj.Show();
            this.Hide();
        }

        private void pictureBox3_Click(object sender, EventArgs e)
        {
            UserFeedback Obj = new UserFeedback();
            Obj.Show();
            this.Hide();
        }
    }
}
```

User Feedback Page

UserFeedback.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PayrollSystem
{
   public partial class UserFeedback : Form
   {
      public UserFeedback()
      {
         InitializeComponent();
         UserNamee.Text = Login.Users;
      }

      SqlConnection Con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\Trials\PayrollSystem\PayRollDB.mdf;Integrated Security=True;Connect Timeout=30");

      private void Clear()
      {
         FeedbkTb.Text = "";
      }
```

```
private void SubmitBtn_Click(object sender, EventArgs e)
{
   if (FeedbkTb.Text == "")
   {
      MessageBox.Show("Missing Information");
   }
   else
   {
      try
      {
         Con.Open();
         SqlCommand cmd = new SqlCommand("insert into
FeedbackTbl(EmpName,Message,Period) values(@EN,@M,@P)", Con);
         cmd.Parameters.AddWithValue("@EN", UserNamee.Text);
         cmd.Parameters.AddWithValue("@M", FeedbkTb.Text);
         cmd.Parameters.AddWithValue("@P", DateTime.Now);
         cmd.ExecuteNonQuery();
         MessageBox.Show("Your Feedback has been Submitted!!");
         Con.Close();
         Clear();
      }
      catch (Exception Ex)
      {
         MessageBox.Show(Ex.Message);
      }
   }
}

private void pictureBox9_Click(object sender, EventArgs e)
{
   Application.Exit();
}
```

```
private void pictureBox2_Click(object sender, EventArgs e)
{
   UserHome Obj = new UserHome();
   Obj.Show();
   this.Hide();
}

private void Salic_Click(object sender, EventArgs e)
{
   UserSal Obj = new UserSal();
   Obj.Show();
   this.Hide();
}

private void pictureBox3_Click(object sender, EventArgs e)
{
   UserProfile Obj = new UserProfile();
   Obj.Show();
   this.Hide();
}

private void pictureBox8_Click(object sender, EventArgs e)
{
   Login Obj = new Login();
   Obj.Show();
   this.Hide();
}

private void HomeLbl_Click(object sender, EventArgs e)
{
   UserHome Obj = new UserHome();
```

```
            Obj.Show();
            this.Hide();
        }

        private void Sal_Click(object sender, EventArgs e)
        {
            UserSal Obj = new UserSal();
            Obj.Show();
            this.Hide();
        }

        private void Profile_Click(object sender, EventArgs e)
        {
            UserProfile Obj = new UserProfile();
            Obj.Show();
            this.Hide();
        }
    }}
```

User Profile Page

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PayrollSystem
```

```
{
    public partial class UserProfile : Form
    {
        public UserProfile()
        {
            InitializeComponent();
            UserNamee.Text = Login.Users;
            getEmployee();

        }
        SqlConnection Con = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=D:\Trials\PayrollSystem\PayRollDB.mdf;Integrated Security=True;Connect Timeout=30");

        private void HomeLbl_Click(object sender, EventArgs e)
        {
            UserHome Obj = new UserHome();
            Obj.Show();
            this.Hide();
        }

        private void pictureBox2_Click(object sender, EventArgs e)
        {
            UserHome Obj = new UserHome();
            Obj.Show();
            this.Hide();
        }

        private void Sal_Click(object sender, EventArgs e)
        {
            UserSal Obj = new UserSal();
            Obj.Show();
```

```
    this.Hide();
}

private void Salic_Click(object sender, EventArgs e)
{
    UserSal Obj = new UserSal();
    Obj.Show();
    this.Hide();
}

private void pictureBox8_Click(object sender, EventArgs e)
{
    Login Objl = new Login();
    Objl.Show();
    this.Hide();
}

private void pictureBox3_Click(object sender, EventArgs e)
{
    Application.Exit();
}

private void getEmployee()
{
    //to auto-fill
    if (Con.State == ConnectionState.Closed)
    {
        Con.Open();
    }
    string Query = "select * from EmployeeTbl where EmpName='" + UserNamee.Text + "' ";
    SqlCommand cmd = new SqlCommand(Query, Con);
    DataTable dt = new DataTable();
```

```
    SqlDataAdapter sda = new SqlDataAdapter(cmd);
    sda.Fill(dt);
    if (dt != null && dt.Rows.Count > 0)
    {
        foreach (DataRow dr in dt.Rows)
        {
            name.Text = dr["EmpName"].ToString();
            gen.Text = dr["EmpGen"].ToString();
            dob.Text = dr["EmpDOB"].ToString();
            ph.Text = dr["EmpPhone"].ToString();
            add.Text = dr["EmpAdd"].ToString();
            pos.Text = dr["EmpPos"].ToString();
            jd.Text = dr["JoinDate"].ToString();
            qua.Text = dr["EmpQual"].ToString();
            EmpBasSalary.Text = dr["EmpBasSal"].ToString();
        }
    }
    Con.Close();
}

private void Btn_Click(object sender, EventArgs e)
{
    UserProfileEdit Obj = new UserProfileEdit();
    Obj.Show();
    this.Hide();
}

private void label1_Click(object sender, EventArgs e)
{
    UserFeedback Obj = new UserFeedback();
    Obj.Show();
    this.Hide();        }
```

```
        private void pictureBox4_Click(object sender, EventArgs e)
        {
            UserFeedback Obj = new UserFeedback();
            Obj.Show();
            this.Hide();
        }

        private void bunifuThinButton21_Click(object sender, EventArgs e)
        {
            UserChngePass Obj = new UserChngePass();
            Obj.Show();
            this.Hide();
        }
    }
}
```