

PAYROLL MANAGEMENT SYSTEM

A Project Report

Submitted in partial fulfilment of the

Requirements for the award of the Degree of

BACHELOR OF SCIENCE (COMPUTER SCIENCE)

By

ATIF PARVEZ KHAN

Seat Number:

Under the esteemed guidance of

Prof. Javed Pathan

Assistant Professor



DEPARTMENT OF COMPUTER SCIENCE

RIZVI COLLEGE OF ARTS SCIENCE AND COMMERCE

Affiliated to University of Mumbai

MUMBAI PIN- 400050

MAHARASHTRA

2019-2020

RIZVI COLLEGE OF ARTS SCIENCE AND COMMERCE

(Affiliated to University of Mumbai)

MUMBAI-MAHARASHTRA-50

DEPARTMENT OF COMPUTER SCIENCE



CERTIFICATE

This is to certify that the project entitled, "**Payroll Management System**", is benefited work of **Atif Parvez Khan** bearing Seat Number: () submitted in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE** in **COMPUTER SCIENCE** from University of Mumbai.

Internal Guide

HOD

External Examiner

Date:

College Seal

ABSTRACT

Payroll system is the heart of any Human Resource System of an Organization. The solution has to take care of the calculation of salary as per rules of the company, income tax calculation and various deductions to be done from the salary including statutory deductions like less attendance and provident fund deductions.

It has to generate pay-slip. It is understood that we are tired of managing thousands of odd papers, pay slips, payroll reports, and salary details and so on. Imagine that we have a payroll processing system which will generate our pay slips and payroll reports within seconds. We can help others automated your payroll system by developing a customized payroll application that suits your specific requirements.

ACKNOWLEDGEMENT

I am pleased to present “Payroll Management System” project and take this opportunity to express my profound gratitude to all those people who helped me in completion of this project. I would like to thank my college for providing me with excellent facilities that helped us to complete and present this project.

I express my deepest gratitude towards my project guide **Prof. Javed Pathan** for his valuable and timely advice during the various phases in our project. I would also like to thank him for providing me with all proper facilities and support as the project co-coordinator. I would like to thank him for his support, patience and faith in our capabilities and for giving us flexibility in terms of working and reporting schedules.

I would also like to thank **Prof. Arif Patel** our Head of department. He was always there with his support and those wonderful insights whenever eagerly needed. I am immensely grateful for to Principal **Mrs. Anjum Ara Ahmad** for her constant support and consideration.

Finally, I would like to thank my **parents** especially my **Late Father** who was there with me with his help and support throughout my project and my friends who supported me in this project.

DECLARATION

I hereby declare that the project entitled, “Payroll Management System” is done at Home, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university, The project is done in partial fulfillment of the requirements for the award of degree Of **BACHELOR OF SCIENCE (COMPUTER SCIENCE)** to be submitted as 5th semester project as part of our curriculum

ATIF PARVEZ KHAN

TABLE OF CONTENTS

CHAPTER1. INTRODUCTION.....	1
Objective.....	1
Purpose of the Project.....	1
Scope of the project.....	1
Project Schedule.....	3
CHAPTER2. SYSTEM ANALYSIS.....	4
2.1 Existing System.....	4
2.2 Proposed System.....	4
2.3 Software and Hardware Requirement Specification.....	4
2.4 Justification of Technology Used.....	5
CHAPTER3. SYSTEM DESIGN.....	10
3.1 Methodology.....	10
3.2 UML Diagrams.....	11
3.2.1Use Case Diagram.....	11
3.2.2 Sequence Diagram.....	13
3.2.3 Activity Diagram.....	14
3.3 Data flow Diagram.....	15
3.4 E-R Diagram.....	16
CHAPTER 4. IMPLEMENTATION & CODING.....	18
4.1 Coding.....	18
4.2 Screenshots.....	44
4.3 Testing Approach.....	51
4.4 Test Case.....	53
CHAPTER5. RESULTS & DISCUSSION.....	56
CHAPTER6. CONCLUSION & FUTURE ENHANCEMENT.....	59
6.1 Conclusion.....	59
6.2 Future Enhancement.....	59
CHAPTER7.REFERENCE.....	60

CHAPTER 1. INTRODUCTION

Payroll system is the heart of any Human Resource System of an organization. The solution has to take care of the calculation of salary as per rules of the company, income tax calculation and various deductions to be done from the salary including statutory deductions like leave and provident fund deductions. It has to generate pay-slip. It is understood that we are tired of managing thousands of odd papers, pay slips, payroll reports, and salary details and so on. Imagine that we have a payroll processing system which will generate our pay slips and payroll reports within seconds. We can help others automated your payroll system by developing a customized payroll application that suits your specific requirements.

1.1Objective:

An attempt to make a system application which helps in generating pay slips, without the use of paper and the pay slips can be generated within seconds. It can perform function like adding employee data, managing leaves generating salary etc. Which in turn makes operation for the user easy as no paper work is done and record can be maintained in it with the help of database.

1.2 Purpose of the Project

Main aim of developing Payroll Management System is to provide an easy way not only to automate all functionalities involved managing leaves and Payroll for the employees of Company, but also to provide full functional reports to management of Company with the details about usage of leave facility. We are committed to bring the best way of management in the various forms of EPS. We understand that EPS is not a product to be sold, it is a tool to manage the inner operation of Company related to employee leave and Payroll.

1.3 Scope of the project

This Application works in Multiple PC's installed on multiple Computers but sharing same database by which users of different department can use it sitting at different locations simultaneously. But in future we can make the Application where the database will be hosted in order to manage the all departments which will be located in different places and by keeping domain of Application as Online.

Modules and their Description

This system comprises having 2 entities Modules:

1. Admin

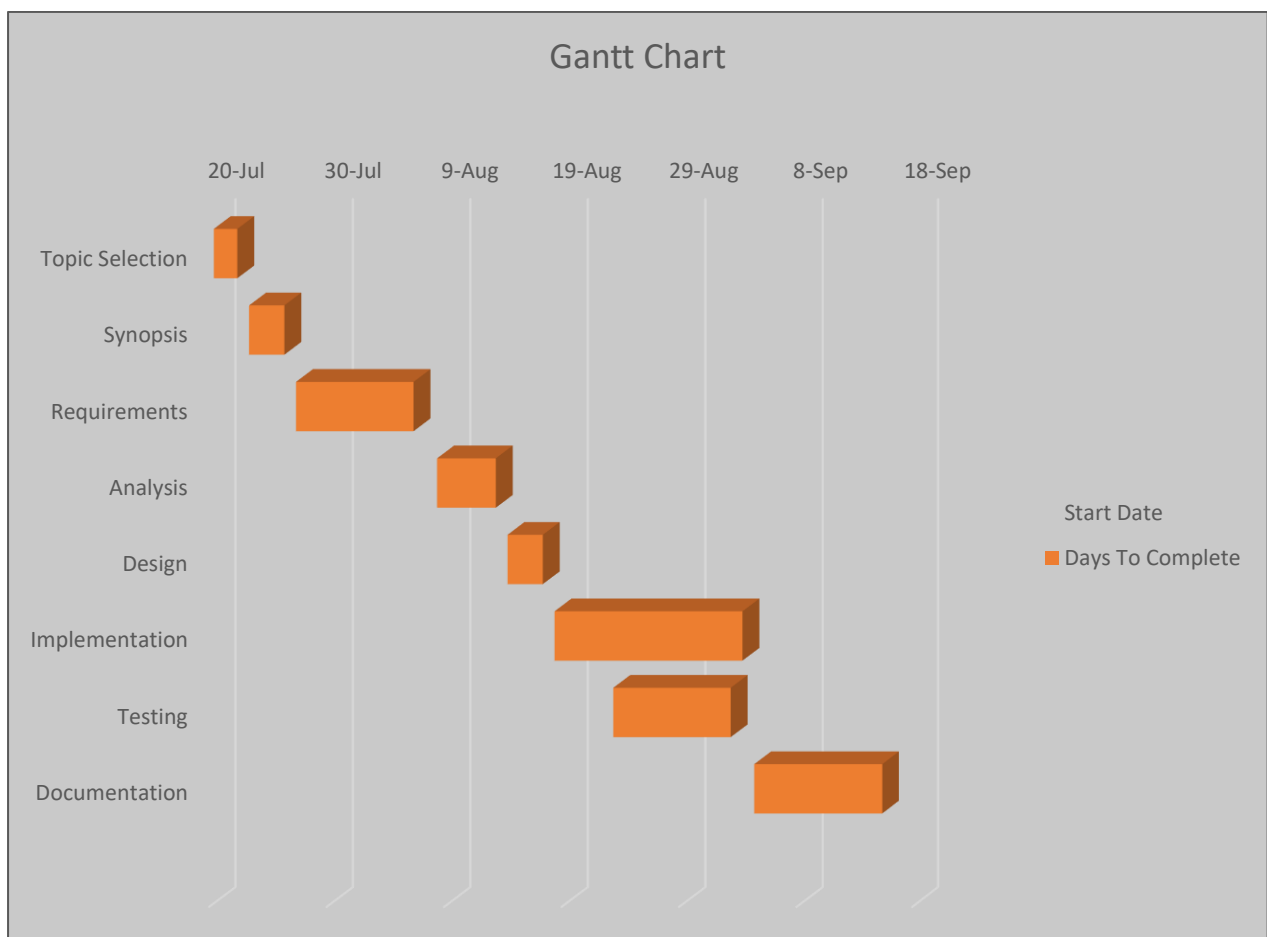
- **Login:** Admin can login using valid credentials
- **Manage Employee & Salary Details:** New employee details can be added/edit/deleted.
Login credentials are created for every employee
- **Approve/Decline Leaves:** Admin has authority to approve or decline employee leaves
- **Generate Monthly Salary:** Admin can generate employee's monthly salary.

2. Employee

- **Login:** Employee can login using valid login credentials.
- **Apply Leave & Check Status:** Employee can apply for a leave with a valid reason.
- **View & Print Salary Slip:** Employee can view monthly salary slip and print.

1.4 Project Schedule

Gantt chart is type of BAR CHART, which depicts the chronological development of various phases in project. It sometime also refers as the schedule chart because of which a team can keep track of the project in terms of time scheduling. A typical Gantt chart gives the start and finish dates of the various phases of development and displays it in the graphical form of bars.



CHAPTER 2. SYSTEM ANALYSIS

2.1 Existing System

The existing system only provides text-based interface, which is not as user-friendly as Graphical user Interface. Since the system is implemented in Manual, so the response is very slow. The transactions are executed in off-line mode, hence on-line data capture and modification is not possible. The Payroll Management System eliminates most of the limitations of the existing software.

2.2 Proposed System

Payroll Management System is a system application where admin maintains employee payroll details. Admin adds basic as well as salary details of the employee who is newly joined the firm and updates the data of the employee if there is any change.

Admin need to create valid login credentials for each employee. Admin can also delete employee details if the employee leaves the firm. Admin can calculate the salary. Admin can approve & and decline the leave request of an employee, Employee can login into the system and apply for a leave. Finally, salary slip of the employee will be view and employee can print the salary slip.

2.3 Software and Hardware Requirement Specification

2.3.1 Hardware Requirement:

1. i3Processor Based Computer or higher
2. Memory: 1 GB
3. Hard Drive: 50 GB

2.3.2 Software Requirement:

- 1.Windows 7 or higher
- 2.SQL Server
- 3.Visual studio
- 4.Programming Language: C#.Net

2.4 Justification of Technology Used:

Front End Technology

1. Visual Studio:

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

2. Microsoft .NET Framework

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet. The .NET Framework is designed to fulfil the following objectives:

To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely. To provide a code-execution environment that minimizes software deployment and versioning conflicts to provide a code-execution environment that guarantees safe execution of code, including code created by an unknown or semi-trusted third party. To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.

To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.

To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code, The .NET Framework has two main components: the common language runtime and the .NET Framework class library.

3. Common Language Runtime

The common language runtime manages memory, thread execution, code execution, code safety verification, compilation, and other system services. These features are intrinsic to the managed code that runs on the common language runtime.

4. Net Framework Class Library

The .NET Framework class library is a collection of reusable types that tightly integrate with the common language runtime. The class library is object oriented, providing types from which your own managed code can derive functionality. This not only makes the .NET Framework types easy to use, but also reduces the time associated with learning new features of the .NET Framework. In addition, third-party components can integrate seamlessly with classes in the .NET Framework the class library includes types that support a variety of specialized development scenarios. For example, you can use the .NET Framework to develop the following types of applications and services:

Console applications.

Scripted or hosted applications.

Windows GUI applications (Windows Forms).

ASP.NET applications.

XML Web services.

Windows services.

Back End Technology

1. Microsoft SQL Server

Microsoft SQL Server is a Structured Query Language (SQL) based, client/server relational database. Each of these terms describes a fundamental part of the architecture of SQL Server.

2. Database

A database is similar to a data file in that it is a storage place for data. Like a data file, a database does not present information directly to a user; the user runs an application that accesses data from the database and presents it to the user in an understandable format.

A database typically has two components: the files holding the physical database and the database management system (DBMS) software that applications use to access data. The DBMS is responsible for enforcing the database structure, including: Maintaining the relationships between data in the database. Ensuring that data is stored correctly and that the rules defining data relationships are not violated. Recovering all data to a point of known consistency in case of system failures.

3. Relational Database

There are different ways to organize data in a database but relational databases are one of the most effective. Relational database systems are an application of mathematical set theory to the problem of effectively organizing data. In a relational database, data is collected into tables (called relations in relational theory).

When organizing data into tables, you can usually find many different ways to define tables. Relational database theory defines a process, normalization, which ensures that the set of tables you define will organize your data effectively.

4. Client/Server:

In a client/server system, the server is a relatively large computer in a central location that manages a resource used by many people. When individuals need to use the resource, they connect over the network from their computers, or clients, to the server.

Examples of servers are: In a client/server database architecture, the database files and DBMS software reside on a server. A communications component is provided so applications can run on separate clients and communicate to the database server over a network. The SQL Server communication component also allows communication between an application running on the server and SQL Server. Server applications are usually capable of working with several clients at the same time. SQL Server can work with thousands of client applications simultaneously.

The server has features to prevent the logical problems that occur if a user tries to read or modify data currently being used by others. While SQL Server is designed to work as a server in a client/server network, it is also capable of working as a stand-alone database directly on the client. The scalability and ease-of-use features of SQL Server allow it to work efficiently on a client without consuming too many resources.

5. Structured Query Language (SQL)

To work with data in a database, you must use a set of commands and statements (language) defined by the DBMS software. There are several different languages that can be used with relational databases; the most common is SQL. Both the American National Standards Institute (ANSI) and the International Standards Organization (ISO) have defined standards for SQL. Most modern DBMS products support the Entry Level of SQL-92, the latest SQL standard (published in 1992).

6. SQL Server Features

Microsoft SQL Server supports a set of features that result in the following benefits: Ease of installation, deployment, and use SQL Server includes a set of administrative and development tools that improve your ability to install, deploy, manage, and use SQL Server across several sites.

7. Scalability

The same database engine can be used across platforms ranging from laptop computers running Microsoft Windows® 95/98 to large, multiprocessor servers running Microsoft Windows NT®, Enterprise Edition.

8. Data warehousing

SQL Server includes tools for extracting and analysing summary data for online analytical processing (OLAP). SQL Server also includes tools for visually designing databases and analysing data using English-based questions. System integration with other server software SQL Server integrates with e-mail, the Internet, and Windows.

9. Databases

A database in Microsoft SQL Server consists of a collection of tables that contain data, and other objects, such as views, indexes, stored procedures, and triggers, defined to support activities performed with the data. The data stored in a database is usually related to a particular subject or process, such as inventory information for a manufacturing warehouse.

SQL Server can support many databases, and each database can store either interrelated data or data unrelated to that in the other databases. For example, a server can have one database that stores personnel data and another that stores product-related data. Alternatively, one database can store current customer order data, and another; related database can store historical customer orders that are used for yearly reporting. Before you create a database, it is important to understand the parts of a database and how to design these parts to ensure that the database performs well after it is implemented.

10. Normalization Theory:

Relations are to be normalized to avoid anomalies. In insert, update and delete operations. Normalization theory is built around the concept of normal forms. A relation is said to be in a particular form if it satisfies a certain specified set of constraints. To decide a suitable logical structure for given database design the concept of normalization, which are briefly described below.

1st Normal Form (1 N.F): A relation is said to be in 1 NF is and only if all unaligned domains contain values only. That is the fields of an n-set should have no group items and no repeating groups.

2nd Normal Form (2 N.F): A relation is said to be in 2 NF is and only if it is in 1 NF and every non key attribute is fully dependent on primary key. This normal takes care of functional dependencies on non-key attributes.

3rd Normal Form (3 N.F): A relation is said to be in 3 NF is and only if it is in 2 NF and every non key attribute is non transitively dependent on the primary key. This normal form avoids the transitive dependencies on the primary key.

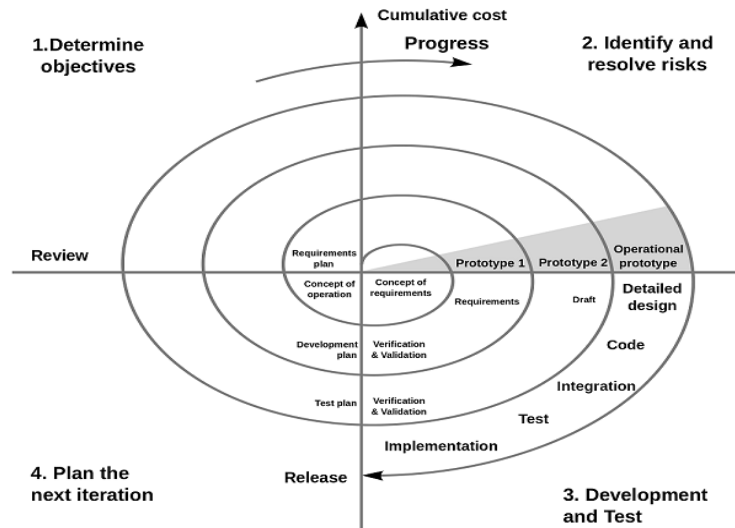
4th Normal Form (4 NF): A relation is said to be in 4 NF if and only if whenever there exists a multi valued dependency in a relation say A >> B then all of the

CHAPTER 3. SYSTEM DESIGN

3.1 Methodology

Software Development Model:

- A software development model serves the developer as a guiding tool during and throughout the development of the software, it facilitates to implement various steps /stages during the development of the overall project. The spiral model is a risk-driven software development process model. Based on the unique risk patterns of a given project, the spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall or evolutionary prototyping. Spiral model is one of the most important Software Development Life Cycle models, which provides support for Risk Handling. In its diagrammatic representation, it looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called a Phase of the software development process.



SPIRAL MODEL APPROACH

1.Planning

It includes estimating the cost, schedule and resources for the iteration. It also involves understanding the system requirements for continuous communication between the system analyst and the customer.

2.Risk Analysis

Identification of potential risk is done while risk mitigation strategy is planned and finalized.

3. Engineering

It includes testing, coding and deploying software at the customer site

4.Evaluation

Evaluation of software by the customer. Also, includes identifying and monitoring risks such as schedule slippage and cost overrun

5.Advantages

Additional functionality or changes can be done at a later stage. Risk of not meeting the schedule or budget.

6.Disadvantages

Cost estimation becomes easy as the prototype building is done in small fragments. It works best for large projects only also demands risk assessment Expertise.

3.2 UML DIAGRAMS

3.2.1 Use Case Diagram

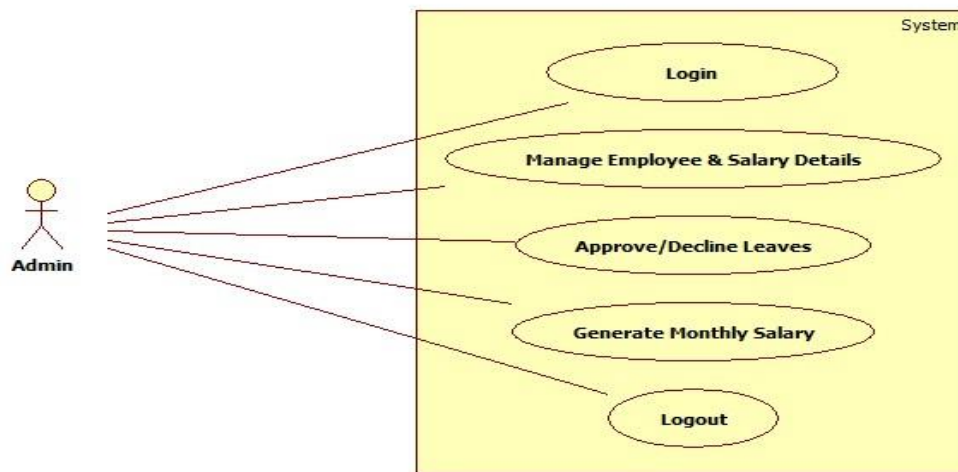


Fig. Use Case Diagram of Admin

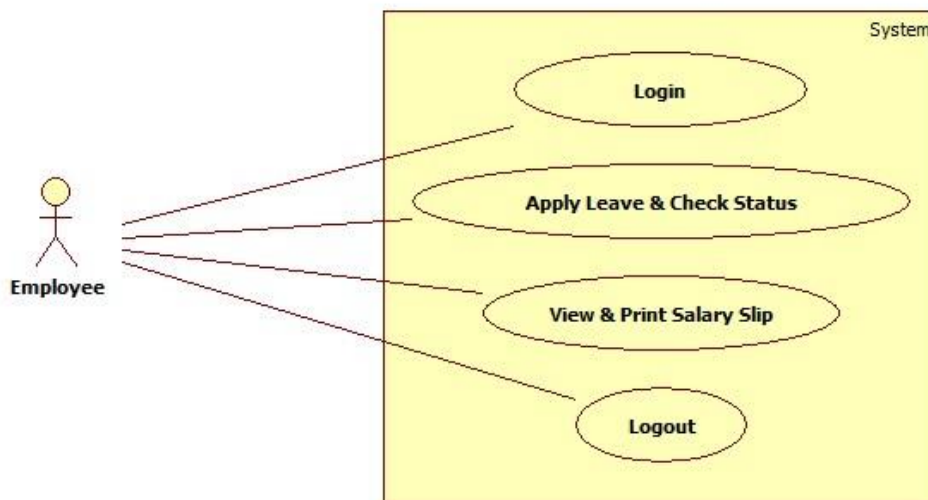


Fig. Use Case Diagram of Employee

The purpose of use case diagram is to capture the dynamic aspect of a system. A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses. It also includes actors which shows the interaction of the actors with the system. Actors can be human users or some internal application or external

application. The above diagram shows the interaction of admin and employee with the system i.e. Admin can add update delete the employee details, approve and decline leaves generate the salary manage employee etc. Same is the case with employee where employee can apply for leave view and print salary slip.

3.2.2 Sequence Diagram

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

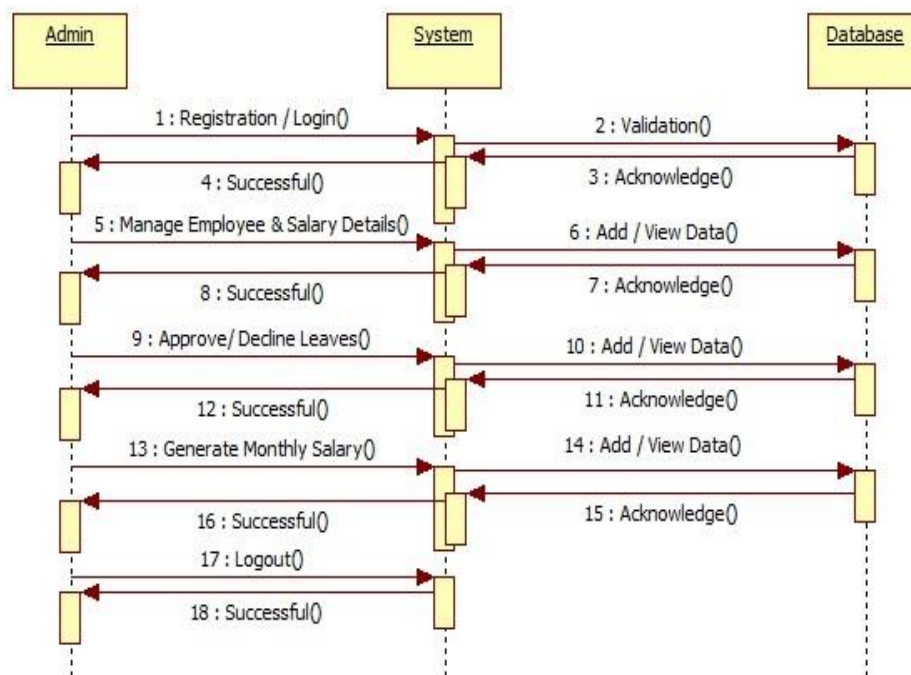


Fig. Sequence Diagram of Admin

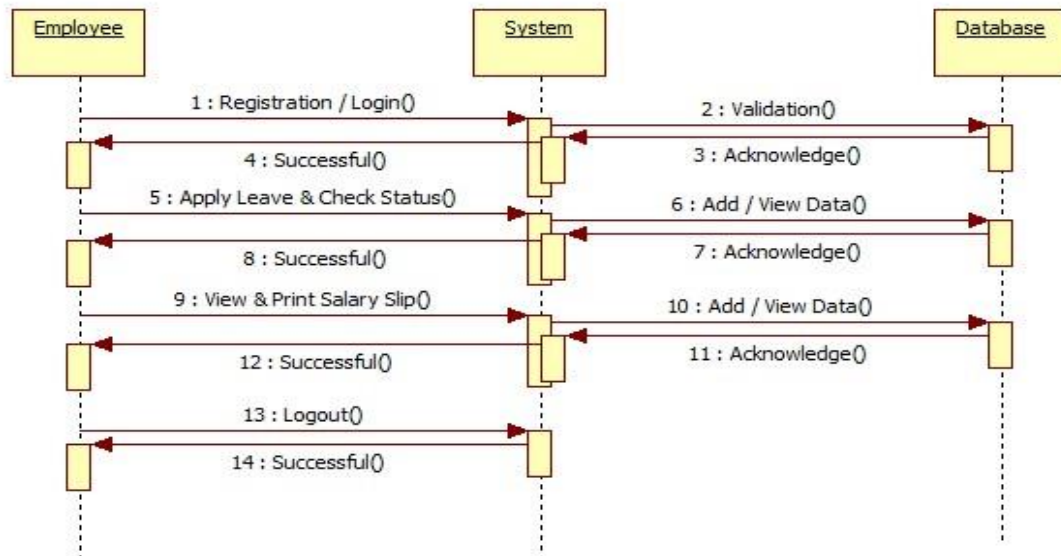


Fig. Sequence Diagram of Employee

3.2.3 Activity Diagram

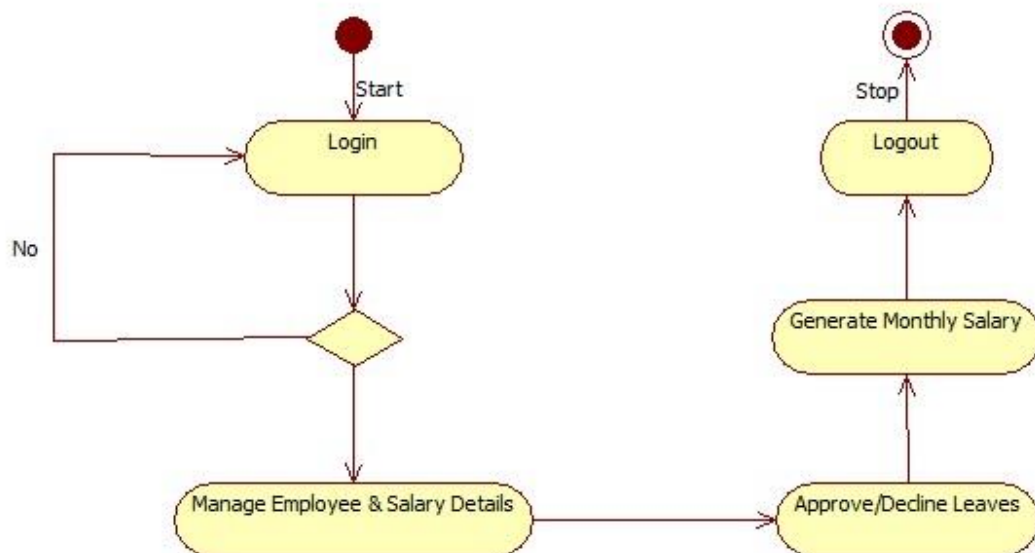


Fig. Activity Diagram of Admin

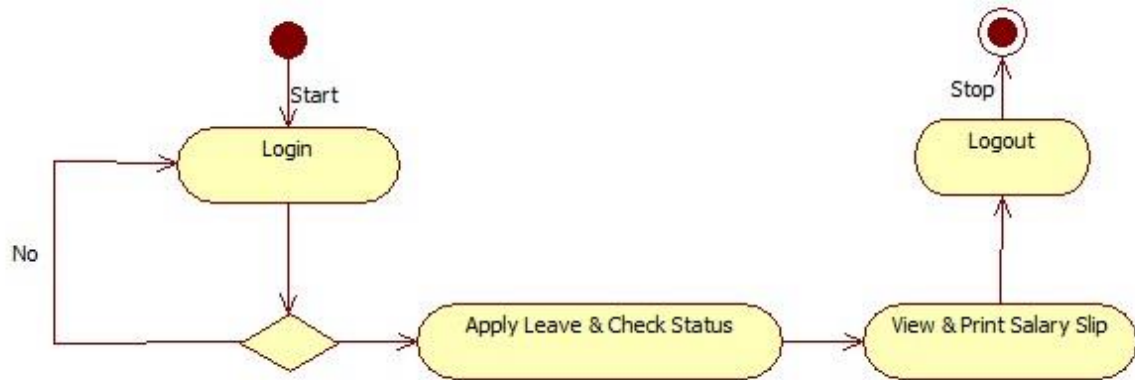


Fig. Activity Diagram of Employee

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams are intended to model both computational and organizational processes (i.e., workflows), as well as the data flows intersecting with the related activities. Although activity diagrams primarily show the overall flow of control, they can also include elements showing the flow of data between activities through one or more data stores.

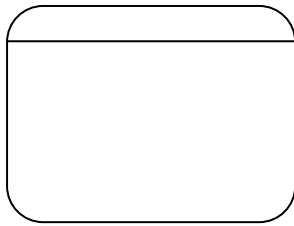
3.3 Data flow Diagram

A data flow diagram is graphical tool used to describe and analyse movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system.

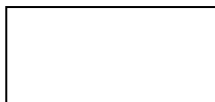
3.3.1 DFD SYMBOLS:

In the DFD, there are four symbols

1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data



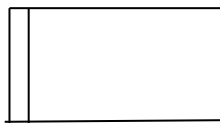
Process that transforms data flow.



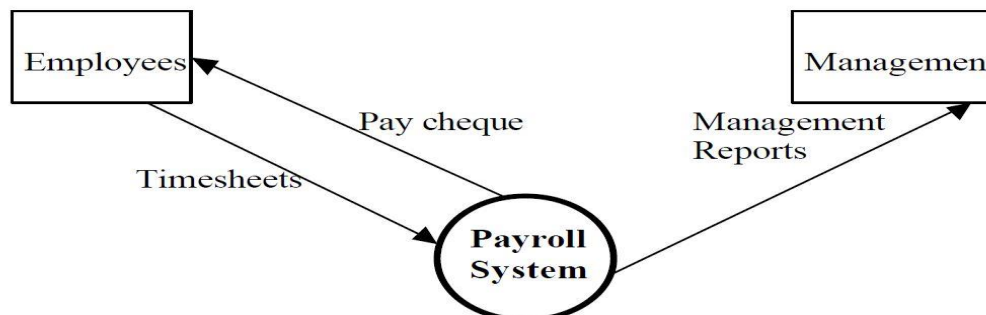
Source or Destination of data



Data flow



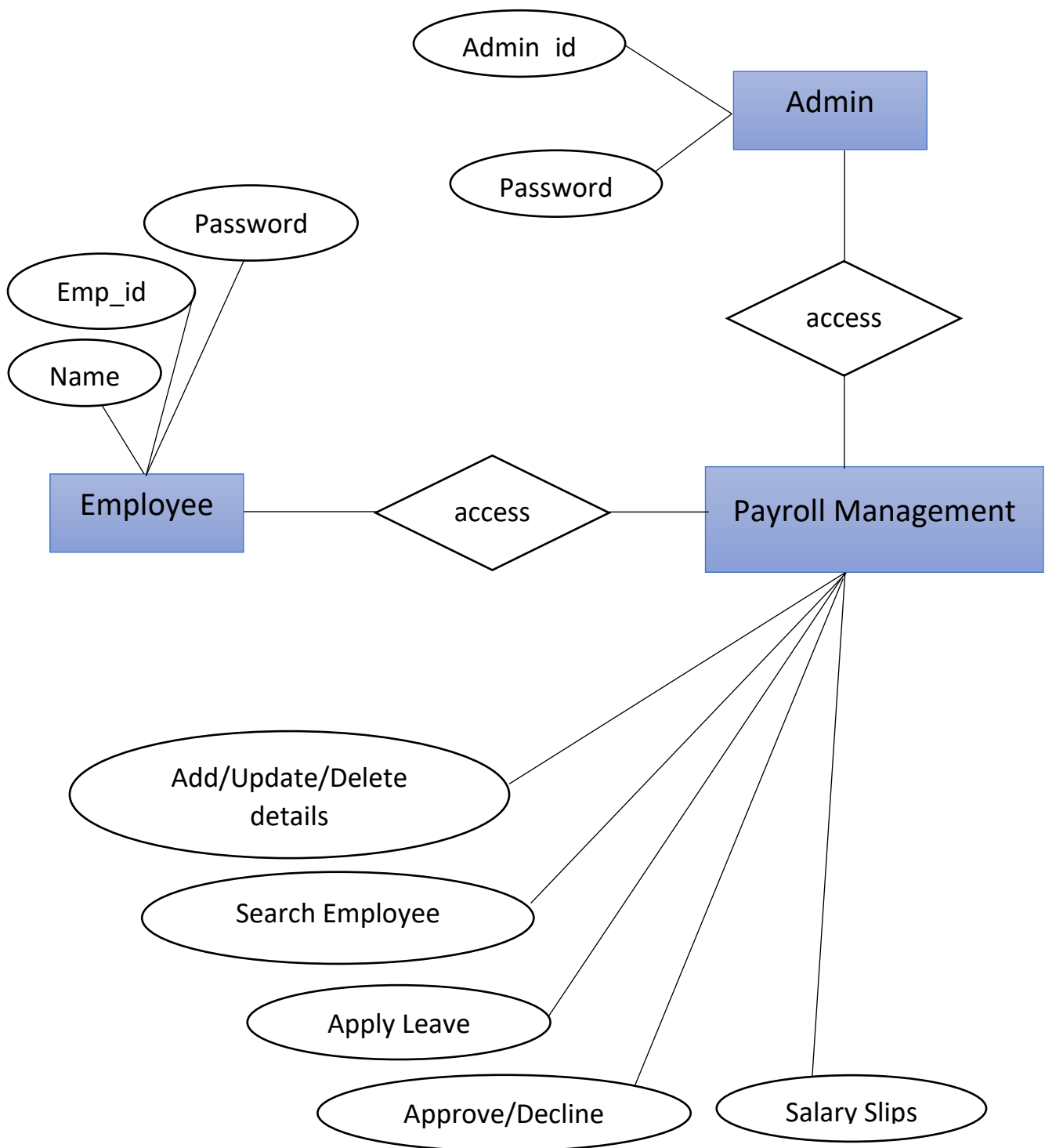
Data Store



3.4 E-R Diagram

ER Model is represented by means of an ER diagram. Any object, for example, entities, attributes of an entity, relationship sets, and attributes of relationship sets, can be represented with the help of an ER diagram. Entities are represented by means of rectangles. Rectangles are named with the entity set they represent. Attributes are the properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle). Multivalued attributes are depicted by double ellipse.

Derived attributes are depicted by dashed ellipse. Relationships are represented by diamond-shaped box. Name of the relationship is written inside the diamond-box. All the entities (rectangles) participating in a relationship, are connected to it by a line.



CHAPTER 4. IMPLEMENTATION & CODING

Project implementation (or project execution) is the phase where visions and plans become reality. This is the logical conclusion, after evaluating, deciding, visioning, planning, applying for funds and finding the financial resources of a project Implementation simply means carrying out the activities described in your work plan.

4.1 Coding

Admin login .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.Diagnostics;
using System.Configuration;
using System.Data.SqlClient;
using System.IO;

namespace WindowsFormsApp1
{
    public partial class Adminlogin : Form
    {
        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);

        public Adminlogin()
        {
            InitializeComponent();
        }

        private void Adminlogin_Load(object sender, EventArgs e)
        {
        }

        private void Button1_Click(object sender, EventArgs e)
        {
            try
            {
```



```

string id = textBox1.Text.Trim();
string password = textBox2.Text.Trim();
if(id != "" && password != "")
{
    SqlCommand cmd = new SqlCommand("select * from admin where
    admin_id = '"+id+"' and admin_pass = '"+password+ "'", con);
    con.Open();
    SqlDataReader sr = cmd.ExecuteReader();
    if(sr.HasRows)
    {

        Masterpage obj = new Masterpage();

        obj.Show();
        this.Hide();

    }
    else
    {
        MessageBox.Show("You are not Authorized..!!", "Error",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    con.Close();
}
else
{
    MessageBox.Show("Kindly enter details..!!", "Warning",
    MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
}
catch (Exception ex)
{
    throw (ex);
}
}

private void back(object sender, LinkLabelLinkClickedEventArgs e)
{
    Home obj = new Home();
    obj.Show();
    this.Hide();
}

private void textBox1_TextChanged(object sender, EventArgs e)
{
}
}
}

```

ManageDepartment.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Configuration;
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 WindowsFormsApp1
{
    public partial class ManageDepartment : Form
    {
        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);
        public ManageDepartment()
        {
            InitializeComponent();
            gridview();
        }

        private void Button4_Click(object sender, EventArgs e)
        {
            try
            {
                string depname = name.Text.Trim();
                if (depname != "")
                {
                    SqlCommand cmd = new SqlCommand("DepartmentDb", con);
                    cmd.CommandType = CommandType.StoredProcedure;
                    SqlParameter name = cmd.Parameters.Add("@name", SqlDbType.VarChar, 50);
                    name.Value = depname.ToUpper();
                    SqlParameter querytype = cmd.Parameters.Add("@querytype", SqlDbType.Int,
50);
                    querytype.Value = 0;
                    SqlParameter op = cmd.Parameters.Add("@op", SqlDbType.Int, 50);
                    op.Direction = ParameterDirection.Output;
                    con.Open();
                    cmd.ExecuteNonQuery();
                    con.Close();
                    string data = op.Value.ToString();
                    if (data != "0")
                    {
                        MessageBox.Show("This Name is Already exist..!!", "Warning",
                        MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                    }
                }
            }
            catch { }
        }
    }
}
```

```

    }
    else
    {
        MessageBox.Show("Successfully Added ..!!", "Success",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        cleardata();
        gridview();
    }
}
else
{
    MessageBox.Show("Please Provide Details..!!", "Warning",
    MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
}
catch(Exception ex)
{
    throw (ex);
}
}

private void Update_Click(object sender, EventArgs e)
{
    try
    {
        string depname = name.Text.Trim();
        string depid = id.Text.Trim();
        if (depname != "" && depid != "")
        {
            SqlCommand cmd = new SqlCommand("DepartmentDb", con);
            cmd.CommandType = CommandType.StoredProcedure;
            SqlParameter name = cmd.Parameters.Add("@name", SqlDbType.VarChar, 50);
            name.Value = depname.ToUpper();
            SqlParameter querytype = cmd.Parameters.Add("@querytype", SqlDbType.Int,
50);
            querytype.Value = depid;
            SqlParameter op = cmd.Parameters.Add("@op", SqlDbType.Int, 50);
            op.Direction = ParameterDirection.Output;
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
            string data = op.Value.ToString();
            if (data != "0")
            {
                MessageBox.Show("This Name is Already exist..!!", "Warning",
                MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
            else
            {

```

```

        MessageBox.Show("Successfully Updated ..!!", "Success",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        cleardata();
        gridview();
    }
    cleardata();
}
else
{
    MessageBox.Show("Please Select Record To Update..!!", "Warning",
    MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
}
catch (Exception ex)
{
    throw (ex);
}
}

private void Delete_Click(object sender, EventArgs e)
{
    try
    {
        string depname = name.Text.Trim();
        string depid = id.Text.Trim();
        if (depname != "" && depid != "")
        {
            SqlCommand cmd = new SqlCommand("delete from Department where dep_id
= "" + depid + """, con);
            con.Open();
            int count = cmd.ExecuteNonQuery();
            con.Close();
            if (count > 0)
            {
                MessageBox.Show("Successfully Delete Data..!!", "Success",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
                gridview();
            }
            else
            {
                MessageBox.Show("Some Issues are Here..!!", "Error",
                MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            cleardata();
        }
        else
        {
            MessageBox.Show("Please Select Record To Delete..!!", "Warning",
            MessageBoxButtons.OK, MessageBoxIcon.Exclamation

```

Manage Employee. cs

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

namespace WindowsFormsApp1
{
    public partial class ManageEmployee : Form
    {
        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);
        public ManageEmployee()
        {
            InitializeComponent();
            SqlCommand das = new SqlCommand("select top 1 cast(eid as int) as id from
            Employee order by eid desc", con);
            con.Open();
            SqlDataReader dr = das.ExecuteReader();
            if (dr.HasRows)
            {
                dr.Read();
                int id = Convert.ToInt32(dr["id"].ToString()) + 1;
                textBox3.Text = id.ToString();
            }
            else
            {
                textBox3.Text = "1";
            }
            con.Close();
            SqlDataAdapter da = new SqlDataAdapter("select * from Department", con);
            con.Open();
            DataTable dt = new DataTable();
            da.Fill(dt);
            int count = dt.Rows.Count;
            if(count> 0)
            {
                //Insert the Default Item to DataTable.
                DataRow row = dt.NewRow();
                row[0] = 0;
                row[1] = "Please select";
            }
        }
    }
}
```

```

        dt.Rows.InsertAt(row, 0);
        department.DataSource = dt;
        department.DisplayMember = "dep_name";
        department.ValueMember = "dep_id";
        con.Close();
        gridview();
    }
    else
    {
        Empty.Visible = true;
        Empty.Text = "Kindly Add Department..!!";
        panel1.Visible = false;
    }
}

private void Contact_KeyPress(object sender, KeyPressEventArgs e)
{
    if ((e.KeyChar != (char)8) && (!Char.IsDigit(e.KeyChar)))
    {
        MessageBox.Show("Digits expected", "Error", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
        e.Handled = true;
    }
}

private void Salary_KeyPress(object sender, KeyPressEventArgs e)
{
    if ((e.KeyChar != (char)8) && (!Char.IsDigit(e.KeyChar)))
    {
        MessageBox.Show("Digits expected", "Error", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
        e.Handled = true;
    }
}

private void Insert_Click(object sender, EventArgs e)
{
    try
    {
        string empname = name.Text.Trim();
        string empcontact = contact.Text.Trim();
        string empemail = email.Text.Trim();
        string empgender = "";
        if (radioButton1.Checked)
        {
            empgender = radioButton1.Text;
        }
        else if (radioButton2.Checked)
        {

```

```

        empgender = radioButton2.Text;
    }
    string empaddress = address.Text.Trim();
    string emppassword = password.Text.Trim();
    string empsalary = salary.Text.Trim();
    string empdepartment = department.SelectedValue.ToString();
    string empdesignation = designation.Text.Trim();

    if (empname != "" && empcontact != "" && empemail != "" && empgender != ""
    && empaddress != "" && emppassword != "" && empsalary != "" && empdepartment !=
    "0" && empdesignation != "")
    {
        System.Text.RegularExpressions.Regex rEMail = new
        System.Text.RegularExpressions.Regex(@"^[a-zA-Z][\w\.-]{2,28}[a-zA-Z0-9]@[a-zA-Z0-9]
        9][\w\.-]*[a-zA-Z0-9]\.[a-zA-Z][a-zA-Z\.-]*[a-zA-Z]$");
        if (!rEMail.IsMatch(empemail))
        {
            MessageBox.Show("E-Mail expected", "Error", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
            email.SelectAll();
        }
        else
        {
            Regex validator = new Regex("(3|4|5|6|7|8|9){1}[0-9]{9}");
            string match = validator.Match(empcontact).Value.ToString();
            if (match.Length != 10)
            {
                MessageBox.Show("10 digits are expected..!!", "Error",
                MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                SqlCommand cmd = new SqlCommand("EmployeeDb", con);
                cmd.CommandType = CommandType.StoredProcedure;
                SqlParameter name = cmd.Parameters.Add("@name", SqlDbType.VarChar,
                50);
                name.Value = empname;
                SqlParameter contact = cmd.Parameters.Add("@contact",
                SqlDbType.VarChar, 50);
                contact.Value = empcontact;
                SqlParameter email = cmd.Parameters.Add("@email",
                SqlDbType.VarChar, 50);
                email.Value = empemail.ToUpper();
                SqlParameter gender = cmd.Parameters.Add("@gender",
                SqlDbType.VarChar, 50);
                gender.Value = empgender;
                SqlParameter address = cmd.Parameters.Add("@address",
                SqlDbType.VarChar, 50);
                address.Value = empaddress;
            }
        }
    }
}

```

Manage Employee

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

namespace WindowsFormsApp1
{
    public partial class ManageEmployee : Form
    {
        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);
        public ManageEmployee()
        {
            InitializeComponent();
            SqlCommand das = new SqlCommand("select top 1 cast(eid as int) as id from
            Employee order by eid desc", con);
            con.Open();
            SqlDataReader dr = das.ExecuteReader();
            if (dr.HasRows)
            {
                dr.Read();
                int id = Convert.ToInt32(dr["id"].ToString()) + 1;
                textBox3.Text = id.ToString();
            }
            else
            {
                textBox3.Text = "1";
            }
            con.Close();
            SqlDataAdapter da = new SqlDataAdapter("select * from Department", con);
            con.Open();
            DataTable dt = new DataTable();
            da.Fill(dt);
            int count = dt.Rows.Count;
            if(count> 0)
            {
                //Insert the Default Item to DataTable.
                DataRow row = dt.NewRow();
                row[0] = 0;
                row[1] = "Please select";
                dt.Rows.InsertAt(row, 0);
            }
        }
    }
}
```



```

        department.DataSource = dt;
        department.DisplayMember = "dep_name";
        department.ValueMember = "dep_id";
        con.Close();
        gridview();
    }
    else
    {
        Empty.Visible = true;
        Empty.Text = "Kindly Add Department..!!";
        panel1.Visible = false;
    }
}

private void Contact_KeyPress(object sender, KeyPressEventArgs e)
{
    if ((e.KeyChar != (char)8) && (!Char.IsDigit(e.KeyChar)))
    {
        MessageBox.Show("Digits expected", "Error", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
        e.Handled = true;
    }
}

private void Salary_KeyPress(object sender, KeyPressEventArgs e)
{
    if ((e.KeyChar != (char)8) && (!Char.IsDigit(e.KeyChar)))
    {
        MessageBox.Show("Digits expected", "Error", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
        e.Handled = true;
    }
}

private void Insert_Click(object sender, EventArgs e)
{
    try
    {
        string empname = name.Text.Trim();
        string empcontact = contact.Text.Trim();
        string empemail = email.Text.Trim();
        string empgender = "";
        if (radioButton1.Checked)
        {
            empgender = radioButton1.Text;
        }
        else if (radioButton2.Checked)
        {
            empgender = radioButton2.Text;

```

```

    }
    string empaddress = address.Text.Trim();
    string emppassword = password.Text.Trim();
    string empsalary = salary.Text.Trim();
    string empdepartment = department.SelectedValue.ToString();
    string empdesignation = designation.Text.Trim();

    if (empname != "" && empcontact != "" && empemail != "" && empgender != ""
    && empaddress != "" && emppassword != "" && empsalary != "" && empdepartment !=
    "0" && empdesignation != "")
    {
        System.Text.RegularExpressions.Regex rEMail = new
        System.Text.RegularExpressions.Regex(@"^[a-zA-Z][\w\.-]{2,28}[a-zA-Z0-9]@[a-zA-Z0-
        9][\w\.-]*[a-zA-Z0-9]\.[a-zA-Z][a-zA-Z\.-]*[a-zA-Z]$");
        if (!rEMail.IsMatch(empemail))
        {
            MessageBox.Show("E-Mail expected", "Error", MessageBoxButtons.OK,
            MessageBoxIcon.Error);
            email.SelectAll();
        }
        else
        {
            Regex validator = new Regex("(3|4|5|6|7|8|9){1}[0-9]{9}");
            string match = validator.Match(empcontact).Value.ToString();
            if (match.Length != 10)
            {
                MessageBox.Show("10 digits are expected..!!", "Error",
                MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            else
            {
                SqlCommand cmd = new SqlCommand("EmployeeDb", con);
                cmd.CommandType = CommandType.StoredProcedure;
                SqlParameter name = cmd.Parameters.Add("@name", SqlDbType.VarChar,
                50);
                name.Value = empname;
                SqlParameter contact = cmd.Parameters.Add("@contact",
                SqlDbType.VarChar, 50);
                contact.Value = empcontact;
                SqlParameter email = cmd.Parameters.Add("@email",
                SqlDbType.VarChar, 50);
                email.Value = empemail.ToUpper();
                SqlParameter gender = cmd.Parameters.Add("@gender",
                SqlDbType.VarChar, 50);
                gender.Value = empgender;
                SqlParameter address = cmd.Parameters.Add("@address",
                SqlDbType.VarChar, 50);

```

Approve Leave.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.Diagnostics;
using System.Configuration;
using System.Data.SqlClient;
using System.IO;

namespace WindowsFormsApp1
{
    public partial class ApproveLeave : Form
    {
        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);
        public ApproveLeave()
        {
            InitializeComponent();
            gridview();
        }

        private void LeaveGridview_RowHeaderMouseClick(object sender,
        DataGridViewCellMouseEventArgs e)
        {
            status.Text =
            LeaveGridview.Rows[e.RowIndex].Cells[6].Value.ToString().ToUpper();
            id.Text = LeaveGridview.Rows[e.RowIndex].Cells[7].Value.ToString();
        }

        private void LeaveGridview_DataBindingComplete(object sender,
        DataGridViewBindingCompleteEventArgs e)
        {
            LeaveGridview.ClearSelection();
            foreach (DataGridViewBand band in LeaveGridview.Columns)
            {
                band.ReadOnly = true;
            }
            LeaveGridview.Columns[7].Visible = false;
        }

        private void Approve_Click(object sender, EventArgs e)
        {
            string empstatus = status.Text;
            string empid = id.Text;
```

```

        if(empstatus != "")
        {
            if (empstatus == "REQUESTED")
            {
                SqlCommand cmd = new SqlCommand("update Leave set status = 'approve'
where leave_id = '" + empid + "'", con);
                con.Open();
                int count = cmd.ExecuteNonQuery();
                con.Close();
                if (count > 0)
                {
                    MessageBox.Show("Leave is approved..!!", "Success",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                    gridview();
                }
                else
                {
                    MessageBox.Show("Some Issues are Here..!!", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            else
            {
                if (empstatus == "APPROVE")
                {
                    MessageBox.Show("You have already approve this leave..!!", "Warning",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
                }
                else if (empstatus == "DECLINE")
                {
                    MessageBox.Show("You have already Decline this leave..!!", "Warning",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
                }
                else
                {
                    MessageBox.Show("Some Issues are Here..!!", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
            }
            cleardata();
        }
        else
        {
            MessageBox.Show("Please Select Record To Approve..!!", "Warning",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
        }
    }

    private void Decline_Click(object sender, EventArgs e)

```

```

{
    string empstatus = status.Text;
    string empid = id.Text;
    if (empstatus != "")
    {
        if (empstatus == "REQUESTED")
        {
            SqlCommand cmd = new SqlCommand("update Leave set status = 'decline'
where leave_id = " + empid + "", con);
            con.Open();
            int count = cmd.ExecuteNonQuery();
            con.Close();
            if (count > 0)
            {
                MessageBox.Show("Leave is declined..!!", "Success",
MessageBoxButtons.OK, MessageBoxIcon.Information);
                gridview();
            }
            else
            {
                MessageBox.Show("Some Issues are Here..!!", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        else
        {
            if (empstatus == "APPROVE")
            {
                MessageBox.Show("You have already approve this leave..!!", "Warning",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
            else if (empstatus == "DECLINE")
            {
                MessageBox.Show("You have already Decline this leave..!!", "Warning",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
            }
            else
            {
                MessageBox.Show("Some Issues are Here..!!", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
        }
        cleardata();
    }
    else
    {
        MessageBox.Show("Please Select Record To Decline..!!", "Warning",
MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }
}

```

```
}
```

Generate Salary.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Configuration;
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 WindowsFormsApp1
{
    public partial class GenerateSalary : Form
    {
        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);
        public GenerateSalary()
        {
            InitializeComponent();
            SqlDataAdapter da = new SqlDataAdapter("select eid, name from Employee", con);
            con.Open();
            DataTable dt = new DataTable();
            da.Fill(dt);
            int count = dt.Rows.Count;
            if (count > 0)
            {
                //Insert the Default Item to DataTable.
                DataRow row = dt.NewRow();
```

```

        row[0] = 0;
        row[1] = "Please select";
        dt.Rows.InsertAt(row, 0);
        ename.DataSource = dt;
        ename.DisplayMember = "name";
        ename.ValueMember = "eid";
        con.Close();
    }
    else
    {
        //Empty.Visible = true;
        //Empty.Text = "Kindly Add Department..!!";
        //panel1.Visible = false;
    }

    string[] installs = new string[] { "January", "February", "March", "April", "May",
    "June", "July", "August", "September", "October", "November", "December"};
    month.Text = "Choose Month";
    month.Items.AddRange(installs);
}

private void TextBox1_KeyPress(object sender, KeyPressEventArgs e)
{
    if ((e.KeyChar != (char)8) && (!Char.IsDigit(e.KeyChar)))
    {
        MessageBox.Show("Digits expected", "Error", MessageBoxButtons.OK,
        MessageBoxIcon.Error);
        e.Handled = true;
    }
}

private void Generate_Click(object sender, EventArgs e)
{
    string empname = ename.Text;

```

```

string empmonth = month.Text.Trim();
string empyear = year.Text.Trim();
string empearning = earnings.Text.Trim();
string providentfund = deductions.Text.Trim();
string leavededuc = leavededuction.Text.Trim();
string latededuc = latededuction.Text.Trim();

if(empname != "Please select" && empmonth != "" && empyear != "" &&
empearning != "" && providentfund != "" && leavededuc != "" && latededuc != "")
{
    SqlDataAdapter cmd = new SqlDataAdapter("select s.*, e.name from salary s inner
join employee e on e.eid = s.eid where name = '"+empname+"' and month = '"+empmonth+"'
and year = '"+empyear+"'", con);

    DataSet drr = new DataSet();
    cmd.Fill(drr);
    int count = drr.Tables[0].Rows.Count;
    if (count > 0)
    {
        MessageBox.Show("You have already generated salary slip for this userof this
year-month..!!", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    else
    {
        int net = Convert.ToInt32(empearning) - (Convert.ToInt32(providentfund) +
Convert.ToInt32(leavededuc) + Convert.ToInt32(latededuc));

        SqlCommand da = new SqlCommand("select e.eid, e.name, e.contact, e.email,
e.gender, e.address, d.dep_name, e.salary, e.designation, e.depid from Employee e inner join
Department d on d.dep_id = e.depid where name = " + empname + "'", con);

        con.Open();
        SqlDataReader dr = da.ExecuteReader();
        if (dr.HasRows)
        {
            dr.Read();
            panel1.Visible = true;

```



```

name.Text = dr["name"].ToString();
designation.Text = dr["designation"].ToString();
department.Text = dr["dep_name"].ToString();
time.Text = empmonth + "-" + empyear;
earn.Text = empearning;
deduc.Text = providentfund;
ldeduc.Text = leavededuc;
mdeduc.Text = latededuc;

```

Employee login.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.Diagnostics;
using System.Configuration;
using System.Data.SqlClient;
using System.IO;

namespace WindowsFormsApp1
{

    public partial class Employeelogin : Form
    {
        public static string UserID;

        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);
        public Employeelogin()
        {
            InitializeComponent();
            this.BackColor = Color.LimeGreen;
            this.TransparencyKey = Color.LimeGreen;
        }

        private void LinkLabel1_LinkClicked(object sender, LinkLabelLinkClickedEventArgs
e)
        {

```

```

        Home obj = new Home();
        obj.Show();
        this.Hide();
    }

    private void Button1_Click(object sender, EventArgs e)
    {
        try
        {
            string id = textBox1.Text.Trim();
            string password = textBox2.Text.Trim();
            if (id != "" && password != "")
            {
                SqlCommand cmd = new SqlCommand("select * from employee where email = 
" + id.ToUpper() + " and password = " + password + "'", con);
                con.Open();
                SqlDataReader sr = cmd.ExecuteReader();
                if (sr.HasRows)
                {
                    sr.Read();
                    UserID = sr["eid"].ToString();
                    EmployeeMaster obj = new EmployeeMaster();
                    obj.Show();
                    this.Hide();

                }
                else
                {
                    MessageBox.Show("You are not Authorized..!!", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                }
                con.Close();
            }
            else
            {
                MessageBox.Show("Kindly enter details..!!", "Warning",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
            }
        }
        catch (Exception ex)
        {
            throw (ex);
        }
    }
}

```

EmployeeMaster.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;

namespace WindowsFormsApp1
{
    public partial class EmployeeMaster : Form
    {
        public EmployeeMaster()
        {
            InitializeComponent();
        }

        private void AboutToolStripMenuItem_Click(object sender, EventArgs e)
        {
            pictureBox1.Visible = false;
            ViewSalarySlip vs = new ViewSalarySlip();
            vs.MdiParent = this;
            DisposeAllButThis(vs);
            vs.Show();
        }

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

        public void DisposeAllButThis(Form form)
        {
            foreach (Form frm in this.MdiChildren)
            {
                if (frm != form)
                {
                    frm.Close();
                }
            }
            return;
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            return;
        }
    }
}
```

```

        pictureBox1.Visible = false;
        ApplyLeaves emp = new ApplyLeaves();
        emp.MdiParent = this;
        DisposeAllButThis(emp);
        emp.Show();
    }
}
}

```

ApplyLeave.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.Diagnostics;
using System.Configuration;
using System.Data.SqlClient;
using System.IO;

namespace WindowsFormsApp1
{
    public partial class ApplyLeaves : Form
    {
        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);
        public ApplyLeaves()
        {
            InitializeComponent();
            fromDate.Format = DateTimePickerFormat.Short;
            fromDate.Value = DateTime.Today;
            fromDate.MinDate = DateTime.Today;
            todate.Value = DateTime.Today;
            todate.Format = DateTimePickerFormat.Short;
            todate.MinDate = DateTime.Today;
            gridview();
        }

        private void Button1_Click(object sender, EventArgs e)
        {
            string FromDate = fromDate.Text;
            //string EndDate = todate.Text;
            string EndDate = "";
            string leavetype = "";
            if (half.Checked)

```

```

        {
            if(first.Checked)
            {
                leavetype = "First Half";
            }
            else
            {
                leavetype = "Second Half";
            }
            EndDate = fromDate.Text;
        }
        else if (full.Checked)
        {
            leavetype = "1";
            EndDate = fromDate.Text;
        }
        else if (multiple.Checked)
        {
            EndDate = toDate.Text;
            TimeSpan tp = Convert.ToDateTime(EndDate) - Convert.ToDateTime(FromDate);
            double days = tp.TotalDays + 1;
            leavetype = days.ToString(); ;
        }
        string Reason = reason.Text;
        if(FromDate != "" && EndDate != "" && Reason != "")
        {
            if(Convert.ToDateTime(EndDate) >= Convert.ToDateTime(FromDate))
            {
                SqlCommand cmd = new SqlCommand("LeaveApply", con);
                cmd.CommandType = CommandType.StoredProcedure;
                SqlParameter eid = cmd.Parameters.Add("@eid", SqlDbType.VarChar, 50);
                eid.Value = EmployeeeLogin.UserID;
                SqlParameter fromdate = cmd.Parameters.Add("@fromdate",
SqlDbType.DateTime, 50);
                fromdate.Value = Convert.ToDateTime(FromDate);
                SqlParameter enddate = cmd.Parameters.Add("@enddate",
SqlDbType.DateTime, 50);
                enddate.Value = Convert.ToDateTime(EndDate);
                SqlParameter reason = cmd.Parameters.Add("@reason", SqlDbType.NVarChar,
-1);
                reason.Value = Reason;
                SqlParameter leavetypes = cmd.Parameters.Add("@leavetype",
SqlDbType.VarChar, 50);
                leavetypes.Value = leavetype;
                SqlParameter op = cmd.Parameters.Add("@op", SqlDbType.Int, 50);
                op.Direction = ParameterDirection.Output;
                con.Open();
                cmd.ExecuteNonQuery();
                con.Close();
            }
        }
    }
}

```

```

        string data = op.Value.ToString();
        if (data != "0")
        {
            MessageBox.Show("You have already apply leave for those days..!!",
"Warning", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        }
        else
        {
            MessageBox.Show("Successfully Apply Leave ..!!", "Success",
MessageBoxButtons.OK, MessageBoxIcon.Information);
            cleardata();
            gridview();
        }
    }
    else
    {
        MessageBox.Show("Last Date should be greater then start Date..!!", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
else
{
    MessageBox.Show("Kindly Provide Details..!!", "Warning",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
}

private void gridview()
{
    SqlCommand cmd = new SqlCommand("select * from Leave where eid = '" +
Employeeelogin.UserID + "' order by leave_id desc", con);
    con.Open();
    SqlDataReader dr = cmd.ExecuteReader();
    if (dr.HasRows)
    {
        DataTable dt = new DataTable();
        dt.Columns.Add("From Date");
        dt.Columns.Add("To Date");
        dt.Columns.Add("Reason");
        dt.Columns.Add("Total Days");
        dt.Columns.Add("Status");
        while (dr.Read())
        {
            DateTime date = Convert.ToDateTime(dr["from_date"].ToString());
            string from = date.ToString("dd/MM/yyyy");
            date = Convert.ToDateTime(dr["to_date"].ToString());
            string to = date.ToString("dd/MM/yyyy");
            dt.Rows.Add(from, to, dr["reason"].ToString(), dr["leave_type"].ToString(),
dr["status"].ToString().ToUpper());
        }
    }
}

```

ViewSalarySlip.cs

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

namespace WindowsFormsApp1
{
    public partial class ViewSalarySlip : Form
    {
        SqlConnection con = new
        SqlConnection(ConfigurationManager.ConnectionStrings["Payroll"].ConnectionString);

        public ViewSalarySlip()
        {
            InitializeComponent();
            string[] installs = new string[] { "January", "February", "March", "April", "May",
"June", "July", "August", "September", "October", "November", "December" };
            month.Text = "Choose Month";
            month.Items.AddRange(installs);
        }

        private void Year_KeyPress_1(object sender, KeyPressEventArgs e)
        {
            if ((e.KeyChar != (char)8) && (!Char.IsDigit(e.KeyChar)))
            {
                MessageBox.Show("Digits expected", "Error", MessageBoxButtons.OK,
                MessageBoxIcon.Error);
                e.Handled = true;
            }
        }

        private void Generate_Click(object sender, EventArgs e)
        {
            string empname = Employeelogin.UserID;
            string empmonth = month.Text.Trim();
            string empyear = year.Text.Trim();

            if (empmonth != "Choose Month" && empyear != "")
```

```

        {
            SqlCommand da = new SqlCommand("select s.*, e.name, d.dep_name from salary
s inner join employee e on e.eid = s.eid inner join Department d on d.dep_id = e.depid where
s.eid = '" + empname + "' and s.month = '" + empmnth + "' and s.year = '" + empyear + "'", con);
            con.Open();
            SqlDataReader dr = da.ExecuteReader();
            if (dr.HasRows)
        }

```

Home.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;

namespace WindowsFormsApp1
{
    public partial class Home : Form
    {
        public Home()
        {
            InitializeComponent();
        }

        private void Button1_Click(object sender, EventArgs e)
        {
            Adminlogin fo = new Adminlogin();
            fo.Show();
            Visible = false;
        }

        private void Button2_Click(object sender, EventArgs e)
        {
            Employeelogin fo1 = new Employeelogin();
            fo1.Show();
            Visible = false;
        }
    }
}

```


EmployeeMaster.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;

namespace WindowsFormsApp1
{
    public partial class EmployeeMaster : Form
    {
        public EmployeeMaster()
        {
            InitializeComponent();
        }

        private void AboutToolStripMenuItem_Click(object sender, EventArgs e)
        {
            pictureBox1.Visible = false;
            ViewSalarySlip vs = new ViewSalarySlip();
            vs.MdiParent = this;
            DisposeAllButThis(vs);
            vs.Show();
        }

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

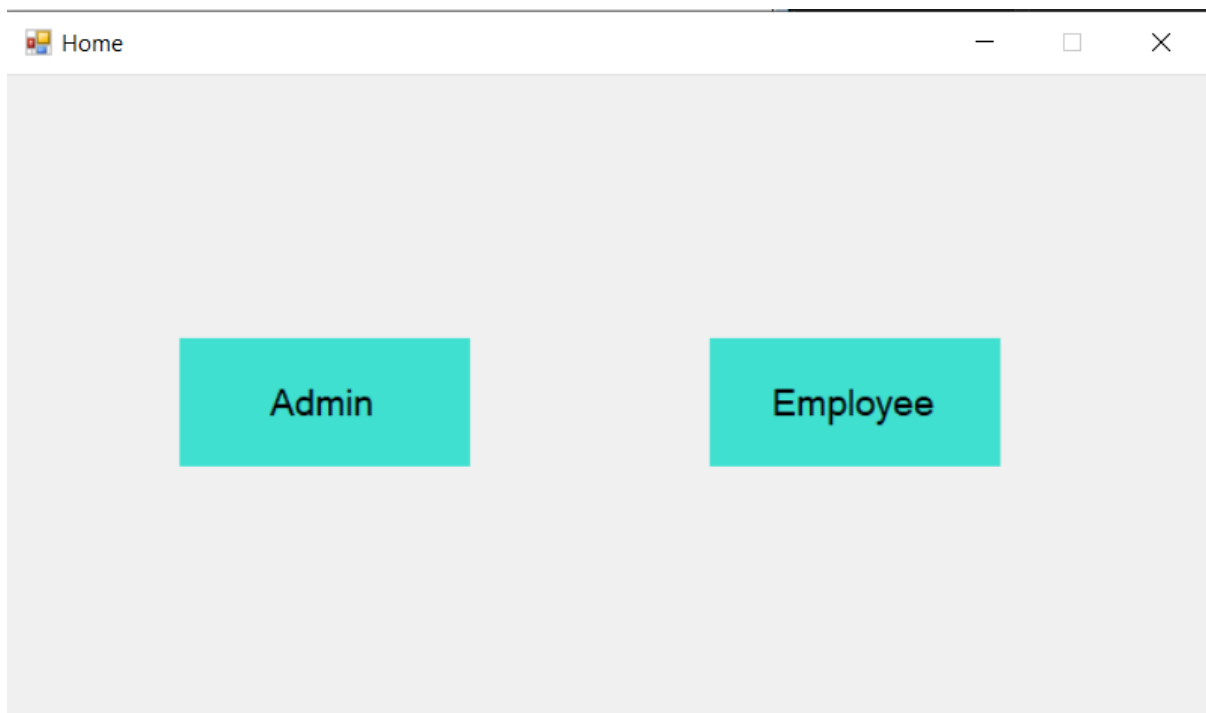
        public void DisposeAllButThis(Form form)
        {
            foreach (Form frm in this.MdiChildren)
            {
                if (frm != form)
                {
                    frm.Close();
                }
            }
            return;
        }

        private void HomeToolStripMenuItem_Click(object sender, EventArgs e)
        {
            return;
        }
    }
}
```

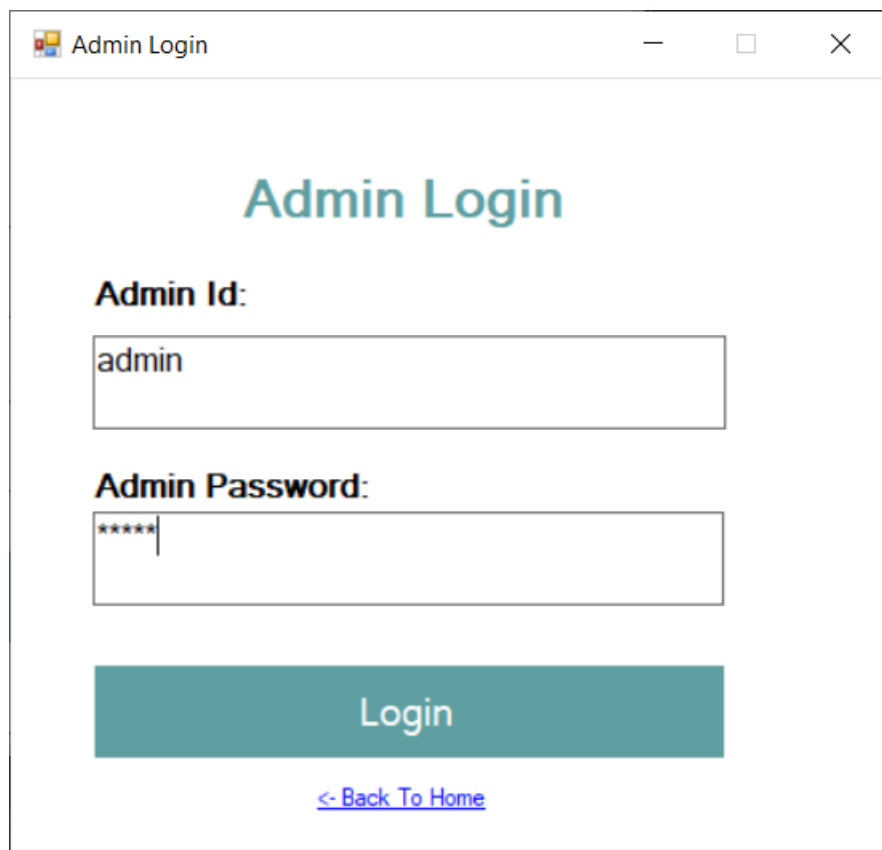
```
        pictureBox1.Visible = false;  
        ApplyLeaves emp = new ApplyLeaves();  
        emp.MdiParent = this;  
        DisposeAllButThis(emp);  
        emp.Show();  
    }  
}  
}
```

4.2 Screenshots

1. Home Screen



2.Admin Login



A screenshot of a web application window titled "Admin Login". The window has a standard title bar with minimize, maximize, and close buttons. The main content area has a light gray background. At the top, the text "Admin Login" is displayed in a large, bold, teal font. Below this, there are two labels: "Admin Id:" and "Admin Password:". Under "Admin Id:", there is a text input field containing the word "admin". Under "Admin Password:", there is a password input field with six asterisks "*****" and a cursor at the end. Below the password field is a large, teal button with the text "Login" in white. At the bottom of the window, there is a blue hyperlink that says "<- Back To Home".

3.Master Page



4. Manage Department

Form2 - [Manage Department]

Manage Department Manage Employee Manage Leaves Manage Salary Logout

Manage Department

Department Name:

Search By Department Name:

Department Name

- SECURITY
- IT
- ACCOUNTS & FINANCE
- MARKETING & SALES
- HUMAN RESOURCE

5. Manage Employee

Form2 - [Manage Employee]

Manage Department Manage Employee Manage Leaves Manage Salary Logout

Manage Employee

Personal Details

Id

Name:

Contact:

Email-id:

Gender: ☒ Male ☐ Female

Address:

Department:

Designation:

Salary:

Search By Name:

Search By Id:

id	Name	Email Id	Contact	Gender	Address	Department	Salary
40	pranav	pranav@yahoo.	7788664429	Male	Mumbai Central	ACCOUNTS & FINANCE	50000
39	Akhtar	akhtar@yahoo.	9323450987	Male	Dadar (west)	ACCOUNTS & FINANCE	100000
38	Gautam	gautam12@gm	8879564590	Male	Matunga (west)	ACCOUNTS & FINANCE	60000
37	Vikram	vikram@gmail.c	7788994352	Male	goregaon (east)	ACCOUNTS & FINANCE	45000
36	Vikas	vikas@gmail.co	9876543210	Male	bandra (east)	ACCOUNTS & FINANCE	45000
35	Ashish	ashish@rediff.c	9000223398	Male	Mulund	ACCOUNTS & FINANCE	65000
34	Abhishek	abhi@yahoo.co	8879584650	Male	kurla	ACCOUNTS & FINANCE	60000
33	Atif	atif15470@gm	7878234591	Male	bandra (west)	ACCOUNTS & FINANCE	150000

6. Manage Leave

employee leave request

Form2 - [ApproveLeave]

Manage Department Manage Employee Manage Leaves Manage Salary Logout

Manage Leaves

Search By Name:

	Employee	Designation	From Date	To Date	Reason	Total Days	Status
>	pranav	Accountant	16-09-2019	16-09-2019	not well	1	REQUESTED
	AryaanSharma	Senior Operations Specialist	05-10-2019	05-10-2019	gtfjh	Second Half	REQUESTED
	AryaanSharma	Senior Operations Specialist	16-09-2019	27-09-2019	thtutut	12	REQUESTED
	AryaanSharma	Senior Operations Specialist	15-09-2019	15-09-2019	t7u76u67	1	REQUESTED
	AryaanSharma	Senior Operations	14-09-2019	14-09-2019	utyty	First Half	APPROVE

Approve Decline

Admin approving leave request

Form2 - [ApproveLeave]

Manage Department Manage Employee Manage Leaves Manage Salary Logout

Manage Leaves

Search By Name:

	Employee	Designation	From Date	To Date	Reason	Total Days	Status
>	AryaanSharma	Senior Operations Specialist	14-09-2019	14-09-2019	utyty	First Half	REQUESTED
	Amaan	Technical Consultant	09-11-2019	09-11-2019	raininn	First Half	APPROVE
	Nasheed	Security Consultant	08-09-2019	12-09-2019			APPROVE
	Nasheed	Security Consultant	07-09-2019	07-09-2019			DECLINE
	Nasheed	Security Consultant	06-09-2019	06-09-2019	sous	First Half	APPROVE

Approve Decline

Success
Leave is approved.!!
OK

7. Manage salary & Generating Salary slip

Form2 - [GenerateSalary]

Manage Department Manage Employee Manage Leaves Manage Salary Logout

Generate Salary Slip

Employee Name:

Choose Month:

Year:

Earnings:

Provident Fund:

Leave Deductions:

Income tax Deduction:

KHAN PVT LIMITED
Bandra (west) Mumbai -400050 Maharashtra

Salary Slip

Employee Name: pranav

Designation: Accountant Department: ACCOUNTS FINANCE

Month and Year: January-2019

Earnings:	50000	Provident Fund:	1000
		Leave Deduction:	100
		Income Tax	1000
		Net Salary:	47900

Signature and Stamp

Form2 - [GenerateSalary]

Manage Department Manage Employee Manage Leaves Manage Salary Logout

Generate Salary Slip

Employee Name:

Choose Month:

Year:

Earnings:

Provident Fund:

Leave Deductions:

Income tax Deduction:

KHAN PVT LIMITED
Bandra (west) Mumbai -400050 Maharashtra

Salary Slip

Employee Name: pranav

Designation: Accountant Department: ACCOUNTS FINANCE

Month and Year: January-2019

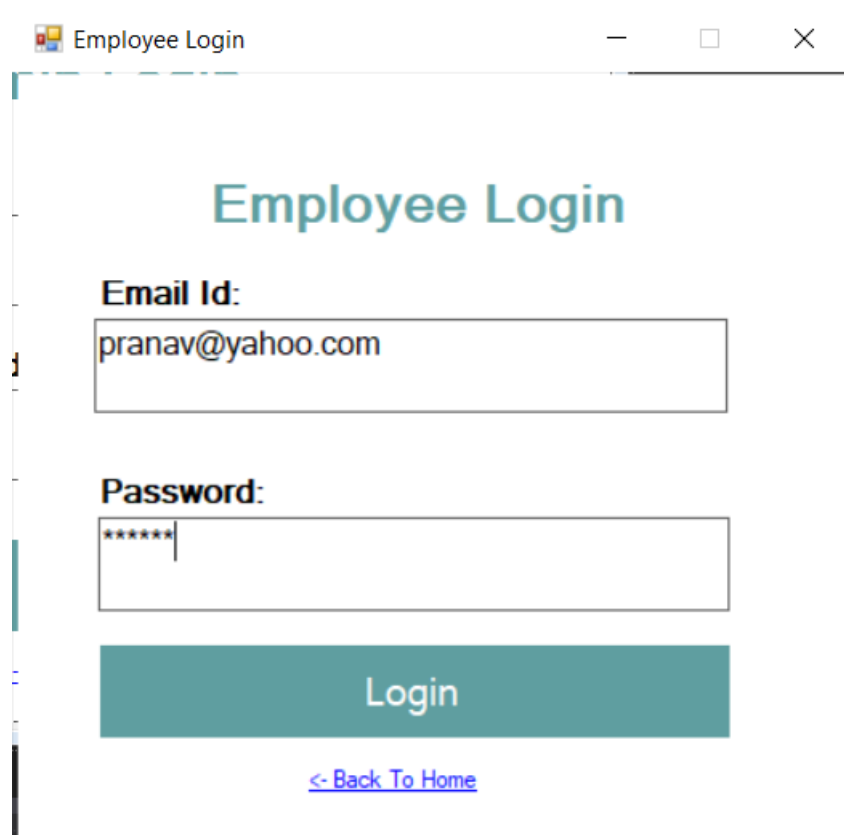
Earnings:	50000	Provident Fund:	1000
		Leave Deduction:	100
		Income Tax	1000
		Net Salary:	47900

Signature and Stamp

Success

Generated Slip..!!

8.Employee Login



A screenshot of a web browser window titled "Employee Login". The window contains a login form with the following elements:

- Employee Login** (Section Header)
- Email Id:** A text input field containing "pranav@yahoo.com".
- Password:** A text input field with masked characters "*****".
- Login** (A large teal button).
- [< Back To Home](#) (A blue link).

9.Employee MasterPage



10.Employee Applying leave

EmployeeMaster - [ApplyLeaves]

Apply leave Salary Slip Logout

Apply Leaves

☐ Half Day ☒ Full Day ☐ Multiple Day

From Date: 16-09-2019 Reason: not well

Apply

Success

Successfully Apply Leave ...!!

OK

11. Leave Request Approved

EmployeeMaster - [ApplyLeaves]

Apply leave Salary Slip Logout

Apply Leaves

☐ Half Day ☒ Full Day ☐ Multiple Day

From Date: 16-09-2019 Reason: not well

Apply

	From Date	To Date	Reason	Total Days	Status
+	16-09-2019	16-09-2019	not well	1	APPROVE

12. Viewing & Print Salary slip

EmployeeMaster - [ViewSalarySlip]

Apply leave Salary Slip Logout

Generate Salary Slip

Choose Month: Year:

KHAN PVT LIMITED
Bandra (West) Mumbai -400050 Maharashtra

Salary Slip

Employee Name: pranav

Designation: A Department: A

Month and Year: January-2019

Earnings:	50000	Provident Fund:	1000
		Leave Deduction:	100
		Income Tax	1000
		Net Salary:	47900

Signature and Stamp

Print preview

January 2019

KHAN PVT LIMITED
Bandra (West) Mumbai -400050 Maharashtra

Salary Slip

Employee Name: pranav

Designation: A Department: A

Month and Year: January-2019

Earnings:	50000	Provident Fund:	1000
		Leave Deduction:	100
		Income Tax	1000
		Net Salary:	47900

Signature and Stamp

4.3 Testing Approach

As the project is bit large we always need testing to make it successful. If each component work properly in all respect and gives desired output for all kind of inputs then project is said to be successful. So the conclusion is-to make the project successful, it needs to be tested.

The testing done here was System Testing checking whether the user requirements were satisfied. The code for the new system has been written completely using C# as the coding language, .Net Framework as the interface for front-end designing. The new system has been tested well with the help of the users and all the applications have been verified from every nook and corner of the user.

Types of Testing

1.Unit Testing

Unit testing focuses verification efforts on the smallest unit of the software design, the module. This is also known as “Module Testing”. The modules are tested separately. This testing carried out during programming stage itself. In this testing each module is found to be working satisfactorily as regards to the expected output from the module.

2.Integration Testing

Data can be grossed across an interface; one module can have adverse efforts on another. Integration testing is systematic testing for construction the program structure while at the same time conducting tests to uncover errors associated with in the interface. The objective is to take unit tested modules and build a program structure. All the modules are combined and tested as a whole. Here correction is difficult because the isolation of cause is complicate by the vast expense of the entire program. Thus in the integration testing stop, all the errors uncovered are corrected for the text testing steps.

3.System testing

System testing is the stage of implementation that is aimed at ensuring that the system works accurately and efficiently for live operation commences. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, then goal will be successfully achieved.

4.Validation Testing

At the conclusion of integration testing software is completely assembled as a package, interfacing errors have been uncovered and corrected and a final series of software tests begins, validation test begins. Validation test can be defined in many ways. But the simple definition is that validation succeeds when the software function in a manner that can reasonably expected by the customer. After validation test has been conducted one of two possible conditions exists. One is the function or performance characteristics confirm to specifications and are accepted and the other is deviation from specification is uncovered and a deficiency list is created. Proposed system under consideration has been tested by using validation testing and found to be working satisfactorily.

5.Output Testing

After performing 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 specified format. Asking the users about the format required by them tests the outputs generated by the system under consideration. Here the output format is considered in two ways, one is on the screen and other is the printed format. The output format on the screen is found to be correct as the format was designed in the system designed phase according to the user needs. For the hard copy also the output comes as the specified requirements by the users. Hence output testing does not result any corrections in the system.

6.User Acceptance Testing

User acceptance of a system is the key factor of the success of any system. The system under study is tested for the user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required.

4.4 Test Case

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Running the project	Click on the start button to run project	Project should start	Project started	Pass
2	Click on admin or employee button to display login page of admin or employee	Clicking on admin or employee button for displaying login page of employee or admin	Login page should be displayed	Login page appeared	Pass
3	Enter login id and password for logging in as employee or admin	Entering Login id and password	Should get logged in as either admin or employee	Logged in	Pass

4	(Admin) Managing department i.e entering Department Details and can even add update and delete the departments	Adding department And even updating ,deleting ,and adding changes	Department should be managed i.e things can be added updated &deleted in it	Changes can be made	Pass
5	Admin can add employee and their details and can makes changes	Admin adding employee and their details	Employee should be added with their details	Employee added	Pass
6	Admin can approve and decline leave request of employee	Admin can be approving or declining leave request of employee	Leave request should work in both cases i.e it can even approve and decline	Leave requested can be approved and even declined	Pass
7	Admin can calculate employee salary and generate salary slip	Admin calculating and generating salary slip of employee	Salary should be calculated and payslip should be generated	Salary calculated and payslip generated	Pass
8	Employee Can apply for Leave	Employee applying for leave	Leave should be applied	Leave applied	Pass

9	Employee can view and print salary slip	Employee clicking on generate slip button to view and to print the slip	Slip should appear and can even be printed	Slip appeared and can even be printed	Pass
---	--	--	---	--	------

CHAPTER 5. RESULTS & DISCUSSION

Manage Employee

Search By Name: Search By Id:

Personal Details

Id:

Name:

Contact:

Email-id:

Gender: ☒ Male ☐ Female

Address:

Department:

Designation:

Salary:

id	Name	Email Id	Contact	Gender	Address	Department	Salary
40	pranav	pranav@yahoo	7788664429	Male	Mumbai Central	ACCOUNTS & FINANCE	50000
39	Akhtar	akhtar@yahoo	9323450987	Male	Dadar (west)	ACCOUNTS & FINANCE	100000
38	Gautam	gautam12@gm	8879564590	Male	Matunga (west)	ACCOUNTS & FINANCE	60000
37	Vikram	vikram@gmail.c	7788994352	Male	goregaon (east)	ACCOUNTS & FINANCE	45000
36	Vikas	vikas@gmail.co	9876543210	Male	bandra (east)	ACCOUNTS & FINANCE	45000
35	Ashish	ashish@rediff.c	9000223398	Male	Mulund	ACCOUNTS & FINANCE	65000
34	Abhishek	abhi@yahoo.co	8879584650	Male	kurla	ACCOUNTS & FINANCE	60000
33	Atif	atif15470@gm	7878234591	Male	bandra (west)	ACCOUNTS & FINANCE	150000

1. Admin can manage employee data by adding the employee details and can even search any employee by their name and through their id and every employee that has been registered their entry can be displayed on the screen

Manage Leaves

Search By Name:

Employee	Designation	From Date	To Date	Reason	Total Days	Status
AryaanSharma	Senior Operations Specialist	14-09-2019	14-09-2019	uty	First Half	REQUESTED
Amaan	Technical Consultant	09-11-2019	09-11-2019	rain	First Half	APPROVE
Nasheed	Security Consultant	08-09-2019	12-09-2019			APPROVE
Nasheed	Security Consultant	07-09-2019	07-09-2019			DECLINE
Nasheed	Security Consultant	06-09-2019	06-09-2019	sous	First Half	APPROVE

Success
Leave is approved...!

2. Admin can even approve and decline leaves accordingly of the employee incase of many leaves admin can search the selected employee through their name and can approve it.

Generate Salary Slip

Employee Name: pranav
 Choose Month: January
 Year: 2019
 Earnings: 50000
 Provident Fund: 1000
 Leave Deductions: 100
 Income tax Deduction: 1000

Apply

Success
 Generated Slip..!!
OK

KHAN PVT LIMITED
 Bandra (west) Mumbai -400050 Maharashtra

Salary Slip
 Employee Name: pranav
 Department: ACCOUNTS FINANCE

Designation		Provident Fund:	1000
Month		Leave Deduction:	100
Earnings		Income Tax	1000
		Net Salary:	47900

Signature and Stamp

Generate Slip

3.Admin can calculate salary of the employee and can even generate salary slip of the employee for a particular month.

Apply Leaves

☐ Half Day ☒ Full Day ☐ Multiple Day

From Date: 16-09-2019
 Reason: not well

Apply

Success
 Successfully Apply Leave ..!!
OK

4.Employee can apply for their leaves and by just entering the details and the reason for it

EmployeeMaster - [ViewSalarySlip]
 Apply leave Salary Slip Logout

Generate Salary Slip

Choose Month: January Year: 2019 Generate Slip

KHAN PVT LIMITED
 Bandra (West) Mumbai -400050 Maharashtra

Salary Slip
 Employee Name: pranav
 Designation: A Department: A
 Month and Year: January-2019

Earnings:	50000	Provident Fund:	1000
		Leave Deduction:	100
		Income Tax	1000
		Net Salary:	47900

Print Signature and Stamp

5.Admin can generate salary slip for viewing and can even print it.

Print preview Close Page 1/1

January 2019 Generate Slip

KHAN PVT LIMITED
 Bandra (West) Mumbai -400050 Maharashtra

Salary Slip
 Employee Name: pranav
 Designation: A Department: A
 Month and Year: January-2019

Earnings:	50000	Provident Fund:	1000
		Leave Deduction:	100
		Income Tax	1000
		Net Salary:	47900

Print Signature and Stamp

6.From here the salary slip can be printed.

CHAPTER 6. CONCLUSION & FUTURE ENHANCEMENT

6.1 Conclusion

The detailed analysis performed on each process has revealed the complexity and sometimes the simplicity of the payroll process. As a result of this the process designed on automation the system will benefit from a top down design. It will allow for exception time off the phone to be easily coded as paid or unpaid. Such changes will reduce the time spent by managers checking time accuracy, payroll clerks keying individual entries and thus by spending less time for each of these activities the cost of payroll can be reduced and the accuracy of the process increased.

6.2 Future Enhancement

Till now the payroll management system project can calculate salary generate receipt add employee data approve leaves etc. The future enhancement of the project can include the incentives of the employee, their extra time work data and their increase in their salary.

CHAPTER 7. REFERENCES

1. <https://www.nibusinessinfo.co.>
2. <https://www.hr.com>
3. <https://youtube.be/d5zWwwwxj0lg>
4. <https://youtube.be/y20QqWpcJdA>
5. <https://youtube/d5zWwwwxj0lg>
6. <https://youtu.be/y20QqWpcJdA>
7. <https://www.atsaccountinginc.com>