Table of Contents

Introduction

ABC Security Company is a new established company that provides security services to various business organizations. The company has many part-time security guards and deploy these guards to its customers based on the service order received. The deployment of security guards need to be very flexible as it has to fulfil the customers' requirement at very short notices. In this situation, maintaining the attendance history and generating payslip for its security guard employees has been a very difficult task for the administration and human resource department.

To improve the scenario, the company has decided to implement a computerized system from the next fiscal year that maintains the list of employees, their attendance (deployment in customers' premise) and generate the payslip based on UK rules (Calculate basic, tax free allowance, tax deduction, NI deduction etc.).

As I have been hired as the developer I have to design and develop a information system for the company. The major business needs required to address with the new system are as follows:

1. Maintaining the list of employees (security guards)
2. Maintaining the attendance (deployment of security guards in customers' premises)
3. Generating monthly summary and detailed report of attendance list (number of hours worked, days etc.)
4. Generating the Pay slip of the employees.

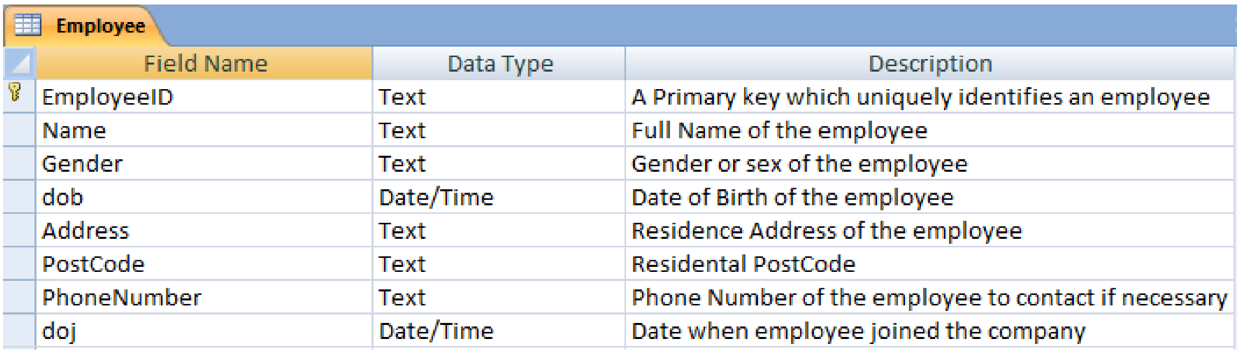
Choice of Technology

For this problem I thought to use .net C# as dot net is very powerful and gives us the platform to create application without worrying much.

Database Design – Logical Design

Table: Employee

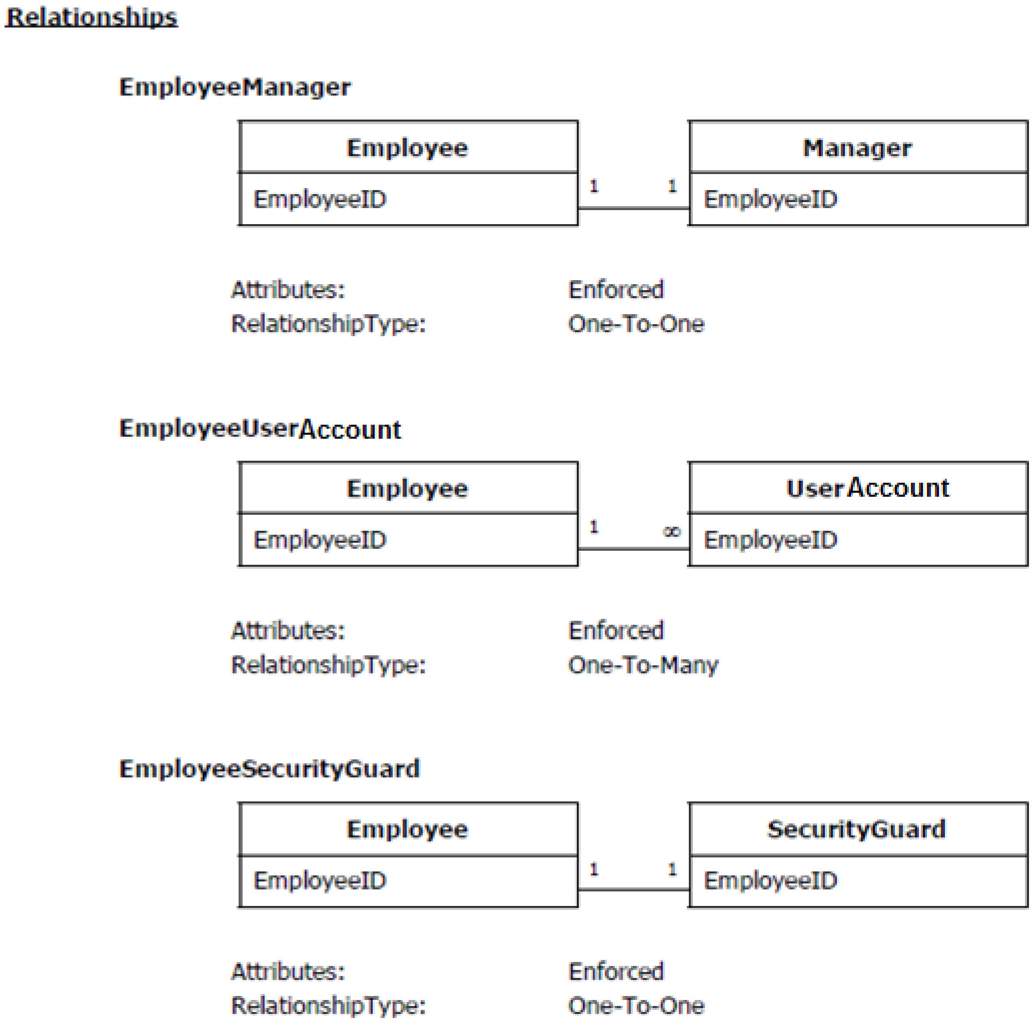
Below is the attributes Description, Image showing 🔑 sign as Primary Key.



Employee

	Field Name	Data Type	Description
🔑	EmployeeID	Text	A Primary key which uniquely identifies an employee
	Name	Text	Full Name of the employee
	Gender	Text	Gender or sex of the employee
	dob	Date/Time	Date of Birth of the employee
	Address	Text	Residence Address of the employee
	PostCode	Text	Residental PostCode
	PhoneNumber	Text	Phone Number of the employee to contact if necessary
	doj	Date/Time	Date when employee joined the company

Below is the Relationship diagram showing relationships or Employee Table with other tables.



Relationships

EmployeeManager

Attributes: Enforced
RelationshipType: One-To-One

EmployeeUserAccount

Attributes: Enforced
RelationshipType: One-To-Many

EmployeeSecurityGuard

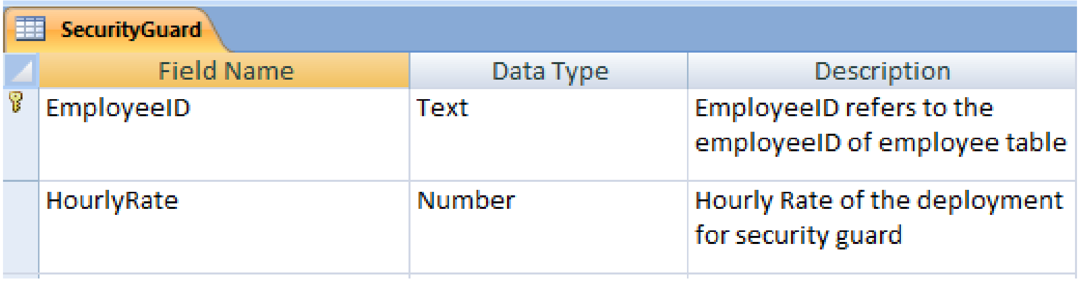
Attributes: Enforced
RelationshipType: One-To-One

Note: Data Type Given as Text, Date/Time according to the Ms Access DBMS for Standard data type I have given in the full relationship diagram later.

Database Design – Logical Design

Table: SecurityGuard

Below is the attributes Description, Image showing 🔑 sign as Primary Key.



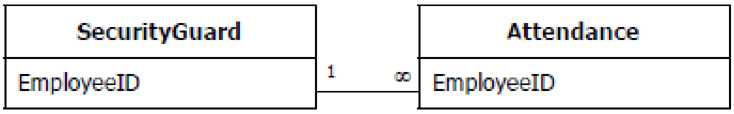
SecurityGuard

	Field Name	Data Type	Description
🔑	EmployeeID	Text	EmployeeID refers to the employeeID of employee table
	HourlyRate	Number	Hourly Rate of the deployment for security guard

Below is the Relationship diagram showing relationships or SecurityGuard Table with other tables.

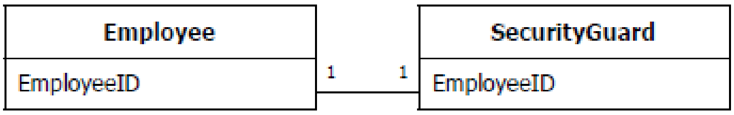
Relationships

SecurityGuardAttendance



Attributes:	Enforced
RelationshipType:	One-To-Many

EmployeeSecurityGuard

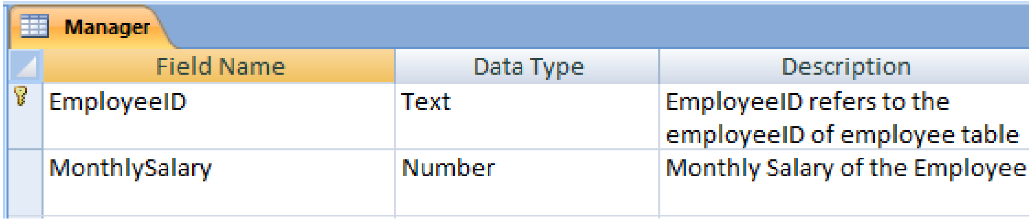


Attributes:	Enforced
RelationshipType:	One-To-One

Database Design – Logical Design

Table: Manager

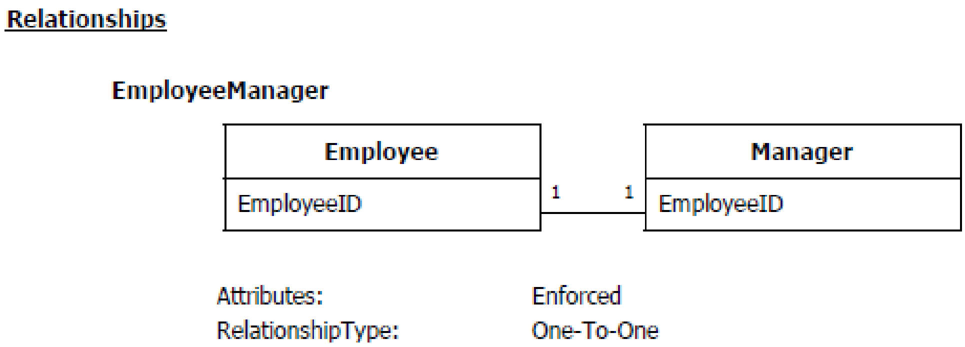
Below is the attributes Description, Image showing 🔑 sign as Primary Key.



Manager

	Field Name	Data Type	Description
🔑	EmployeeID	Text	EmployeeID refers to the employeeID of employee table
	MonthlySalary	Number	Monthly Salary of the Employee

Below is the Relationship diagram showing relationships or **Manager** Table with other tables.



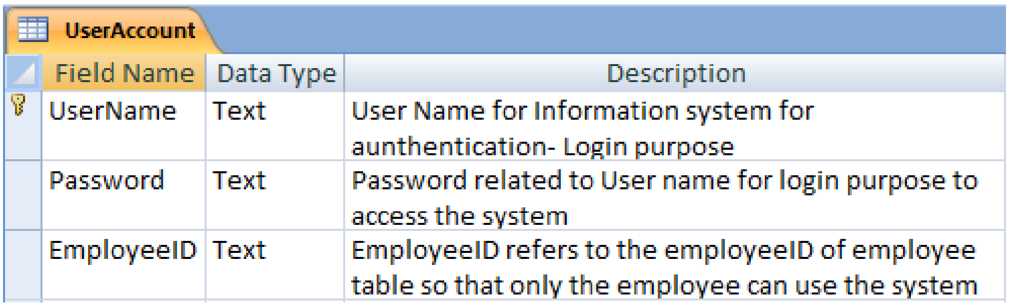
Relationships

EmployeeManager

Attributes: Enforced
RelationshipType: One-To-One

Table: UserAccount

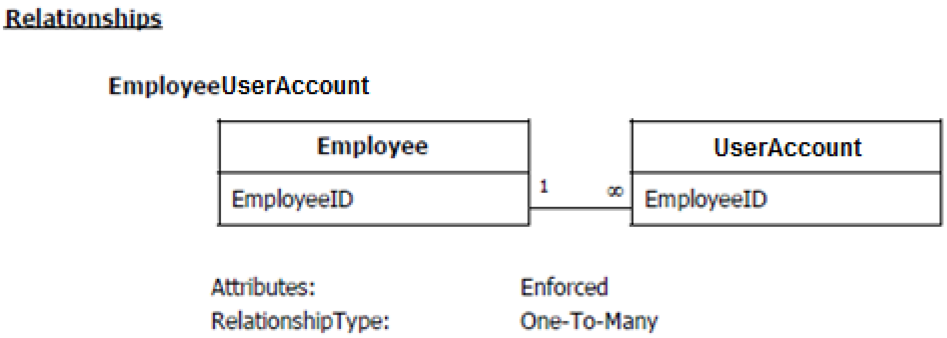
Below is the attributes Description, Image showing 🔑 sign as Primary Key.



UserAccount

	Field Name	Data Type	Description
🔑	UserName	Text	User Name for Information system for aunthentication- Login purpose
	Password	Text	Password related to User name for login purpose to access the system
	EmployeeID	Text	EmployeeID refers to the employeeID of employee table so that only the employee can use the system

Below is the Relationship diagram showing relationships or **UserAccount** table with other tables.



Relationships

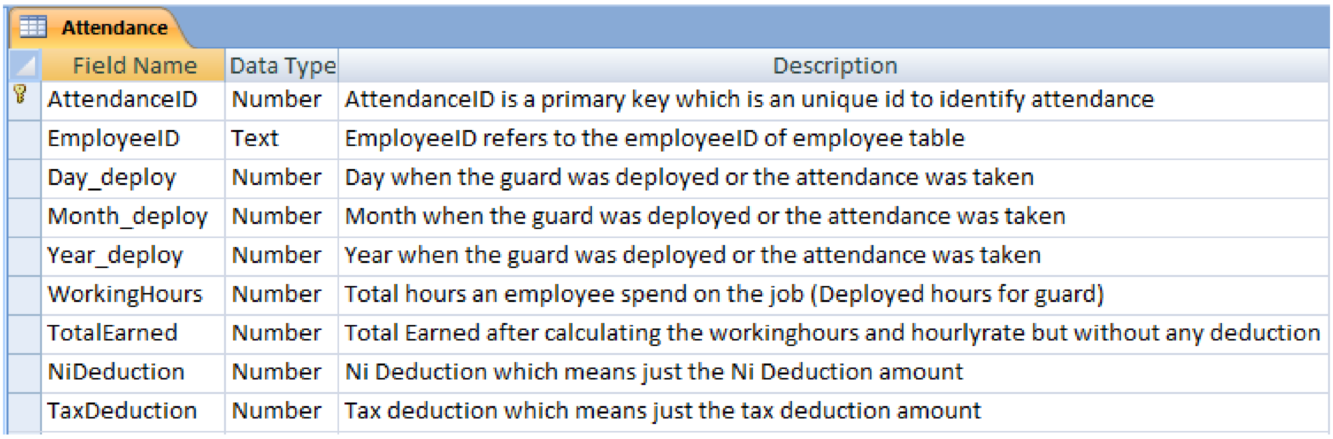
EmployeeUserAccount

Attributes: Enforced
RelationshipType: One-To-Many

Database Design – Logical Design

Table: Attendance

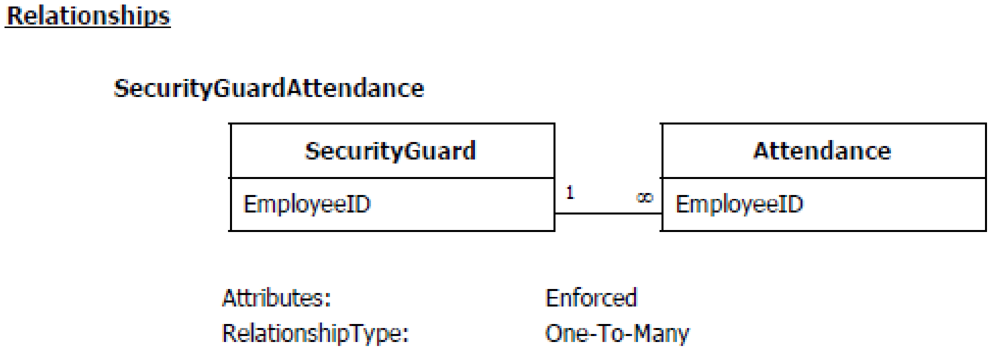
Below is the attributes Description, Image showing 🔑 sign as Primary Key.



Attendance

	Field Name	Data Type	Description
🔑	AttendanceID	Number	AttendanceID is a primary key which is an unique id to identify attendance
	EmployeeID	Text	EmployeeID refers to the employeeID of employee table
	Day_deploy	Number	Day when the guard was deployed or the attendance was taken
	Month_deploy	Number	Month when the guard was deployed or the attendance was taken
	Year_deploy	Number	Year when the guard was deployed or the attendance was taken
	WorkingHours	Number	Total hours an employee spend on the job (Deployed hours for guard)
	TotalEarned	Number	Total Earned after calculating the workinghours and hourlyrate but without any deduction
	NiDeduction	Number	Ni Deduction which means just the Ni Deduction amount
	TaxDeduction	Number	Tax deduction which means just the tax deduction amount

Below is the Relationship diagram showing relationships or **Attendance** table with other tables.



Attributes: Enforced
RelationshipType: One-To-Many

In Above Attribute Description Text Means Varchar(n) where as Date/Time Resembles to any Date or Time But here it is Date. Relationship diagram with standard data type are given in next page.

Here I have created a table with Manager to show that we can use specialization in this way as well and as well as to show why I did generalization specialization. UserAccount table is a table which contains the usernames and password for a given employee with their employeeID which is a login credential which means it is needed to login or access the system.

Database Design- Logical Design

Er- Diagram

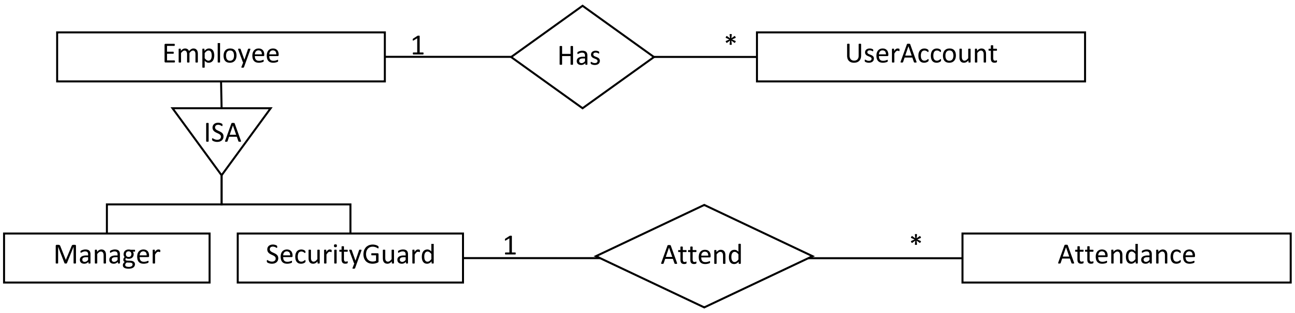


Fig. 1.d.i Above showing the Er-diagram (Attributes omitted for clearance)

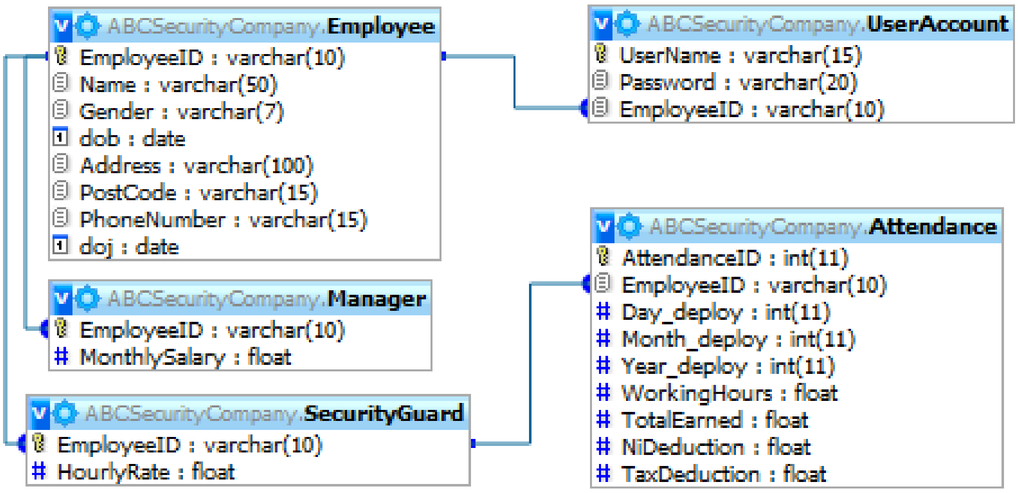


Fig 1.e.i: Diagram showing the attributes and the relation between tables.

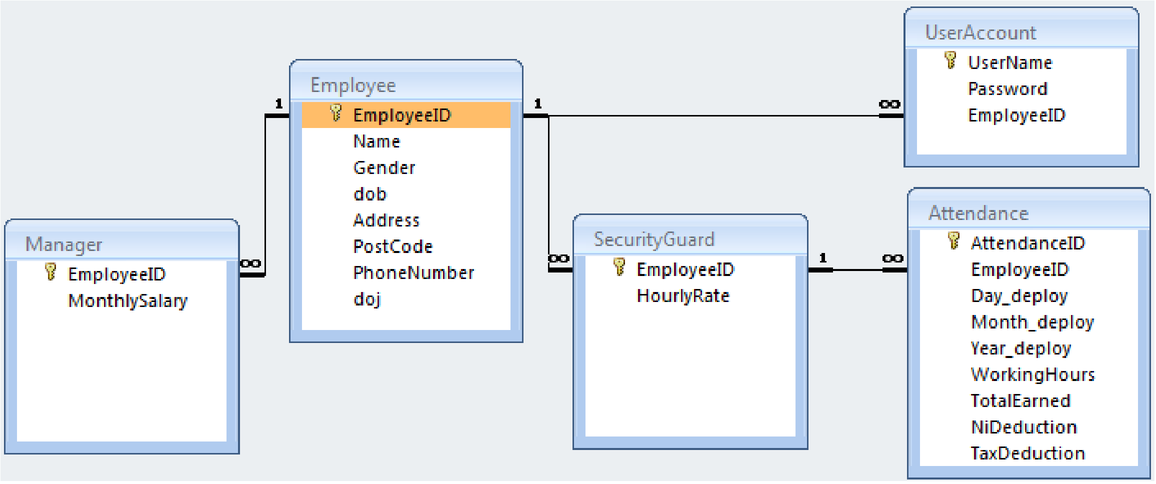


Fig. 1.e.ii Diagram showing the attributes and the relation between tables.

Database Design- Physical Design

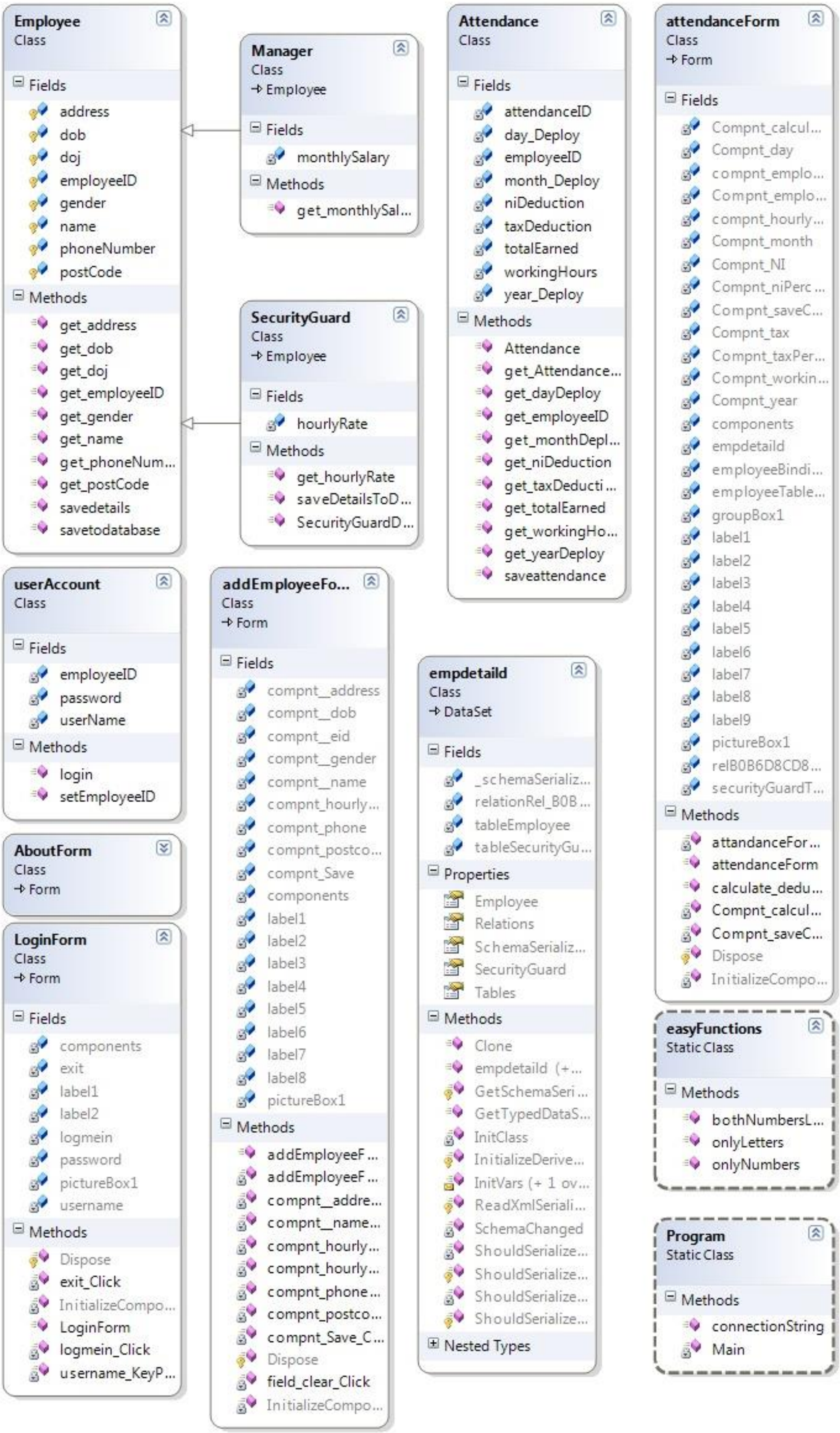
Omitting the screenshots below is the SQL Query for each table.

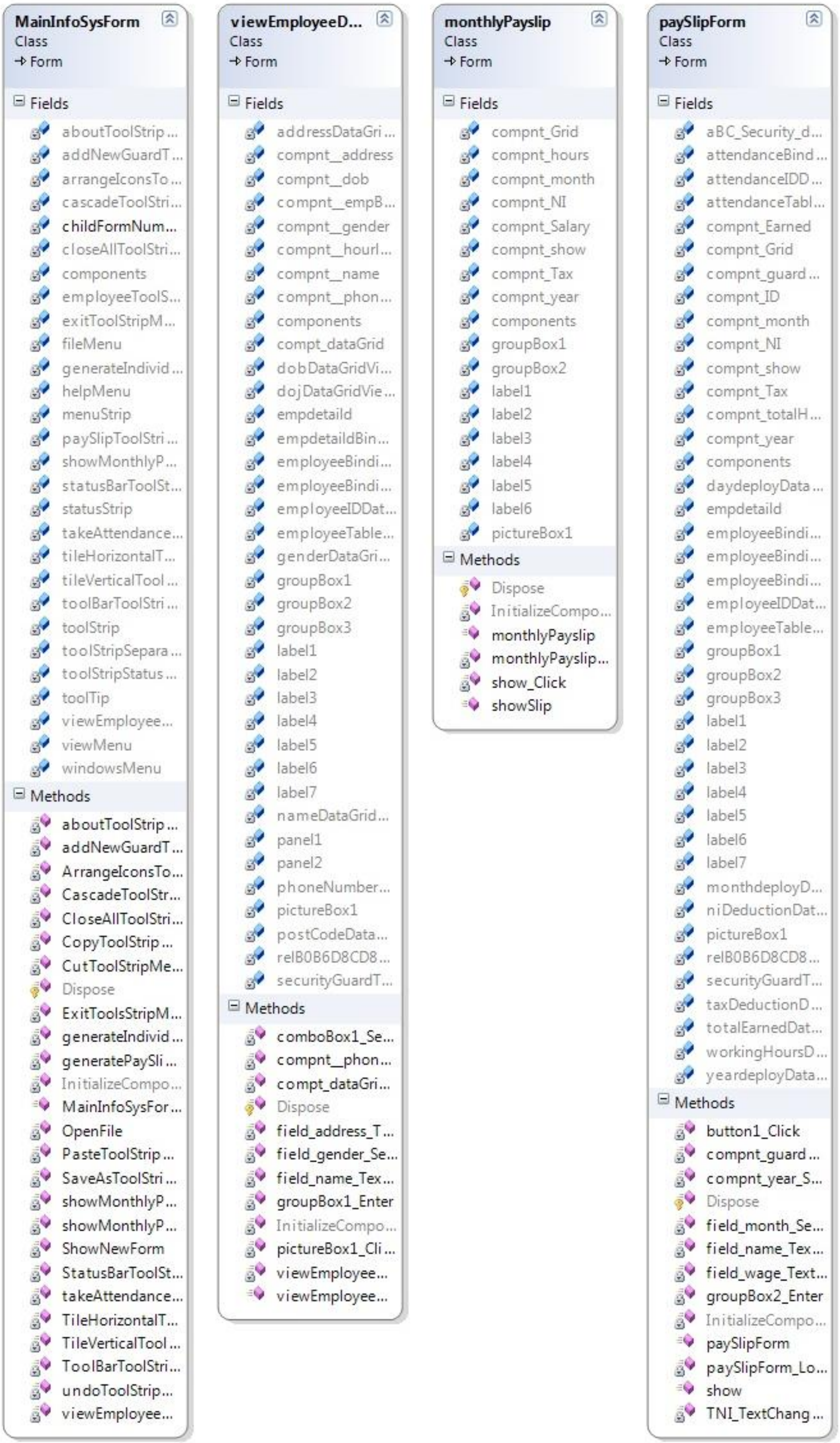
CREATE DATABASE ABC_Security_db;

```
-- ......................................................................
--/* Employee Table */
CREATE TABLE `Employee` (
`EmployeeID` varchar(10) NOT NULL,
`Name`  varchar(50) NULL ,
`Gender`  varchar(7) NULL ,
`dob`  date NULL ,
`Address`  varchar(100) NULL ,
`PostCode`  varchar(15) NULL ,
`PhoneNumber`  varchar(15) NULL ,
`doj`  date NULL ,
PRIMARY KEY (`EmployeeID`)
);

-- ......................................................................
--/* SecurityGuard Table*/
CREATE TABLE `SecurityGuard` (
`EmployeeID`  varchar(10) NULL ,
`HourlyRate`  float NULL ,
PRIMARY KEY (`EmployeeID`),
FOREIGN KEY (`EmployeeID`) REFERENCES `Employee` (`EmployeeID`)
);

-- ......................................................................
--/* Manager Table*/
CREATE TABLE `Manager` (
`EmployeeID`  varchar(10) NOT NULL ,
`MonthlySalary`  float NOT NULL ,
PRIMARY KEY (`EmployeeID`),
FOREIGN KEY (`EmployeeID`) REFERENCES `Employee` (`EmployeeID`)
);

-- ......................................................................
--/* User Table*/
CREATE TABLE `User` (
`UserName`  varchar(15) NOT NULL ,
`Password`  varchar(20) NOT NULL ,
`EmployeeID`  varchar(10) NOT NULL ,
PRIMARY KEY (`UserName`),
FOREIGN KEY (`EmployeeID`) REFERENCES `Employee` (`EmployeeID`)
);
```

```
-- .......................................................................
--/* Attendance Table*/
CREATE TABLE `Attendance` (
`AttendanceID`  int NOT NULL ,
`EmployeeID`  varchar(10) NOT NULL ,
`Day_deploy`  int NULL ,
`Month_deploy`  int NULL ,
`Year_deploy`  int NULL ,
`WorkingHours`  float NULL ,
`TotalEarned`  float NULL ,
`NiDeduction`  float NULL ,
`TaxDeduction`  float NULL ,
PRIMARY KEY (`AttendanceID`),
FOREIGN KEY (`EmployeeID`) REFERENCES `Employee` (`EmployeeID`)
);
--.......................................................................
```

Class Diagram





MainInfoSysForm
Class
→ Form

Fields
- aboutToolStrip...
- addNewGuardT...
- arrangeIconsTo...
- cascadeToolStri...
- childFormNum...
- closeAllToolStri...
- components
- employeeToolS...
- exitToolStripM...
- fileMenu
- generateIndivid...
- helpMenu
- menuStrip
- paySlipToolStri...
- showMonthlyP...
- statusBarToolSt...
- statusStrip
- takeAttendance...
- tileHorizontalT...
- tileVerticalTool...
- toolBarToolStri...
- toolStrip
- toolStripSepara...
- toolStripStatus...
- toolTip
- viewEmployee...
- viewMenu
- windowsMenu

Methods
- aboutToolStrip...
- addNewGuardT...
- ArrangeIconsTo...
- CascadeToolStr...
- CloseAllToolStri...
- CopyToolStrip...
- CutToolStripMe...
- Dispose
- ExitToolsStripM...
- generateIndivid...
- generatePaySli...
- InitializeCompo...
- MainInfoSysFor...
- OpenFile
- PasteToolStrip...
- SaveAsToolStri...
- showMonthlyP...
- showMonthlyP...
- ShowNewForm
- StatusBarToolSt...
- takeAttendance...
- TileHorizontalT...
- TileVerticalTool...
- ToolBarToolStri...
- undoToolStrip...
- viewEmployee...

viewEmployeeD...
Class
→ Form

Fields
- addressDataGri...
- compnt_address
- compnt_dob
- compnt_empB...
- compnt_gender
- compnt_hourl...
- compnt_name
- compnt_phon...
- components
- compt_dataGrid
- dobDataGridVi...
- dojDataGridVie...
- empdetaild
- empdetaildBin...
- employeeBindi...
- employeeBindi...
- employeeIDDat...
- employeeTable...
- genderDataGri...
- groupBox1
- groupBox2
- groupBox3
- label1
- label2
- label3
- label4
- label5
- label6
- label7
- nameDataGrid...
- panel1
- panel2
- phoneNumber...
- pictureBox1
- postCodeData...
- relB0B6D8CD8...
- securityGuardT...

Methods
- comboBox1_Se...
- compnt_phon...
- compt_dataGri...
- Dispose
- field_address_T...
- field_gender_Se...
- field_name_Tex...
- groupBox1_Enter
- InitializeCompo...
- pictureBox1_Cli...
- viewEmployee...
- viewEmployee...

monthlyPayslip
Class
→ Form

Fields
- compnt_Grid
- compnt_hours
- compnt_month
- compnt_NI
- compnt_Salary
- compnt_show
- compnt_Tax
- compnt_year
- components
- groupBox1
- groupBox2
- label1
- label2
- label3
- label4
- label5
- label6
- pictureBox1

Methods
- Dispose
- InitializeCompo...
- monthlyPayslip
- monthlyPayslip...
- show_Click
- showSlip

paySlipForm
Class
→ Form

Fields
- aBC_Security_d...
- attendanceBind...
- attendanceIDD...
- attendanceTabl...
- compnt_Earned
- compnt_Grid
- compnt_guard...
- compnt_ID
- compnt_month
- compnt_NI
- compnt_show
- compnt_Tax
- compnt_totalH...
- compnt_year
- components
- daydeployData...
- empdetaild
- employeeBindi...
- employeeBindi...
- employeeBindi...
- employeeIDDat...
- employeeTable...
- groupBox1
- groupBox2
- groupBox3
- label1
- label2
- label3
- label4
- label5
- label6
- label7
- monthdeployD...
- niDeductionDat...
- pictureBox1
- relB0B6D8CD8...
- securityGuardT...
- taxDeductionD...
- totalEarnedDat...
- workingHoursD...
- yeardeployData...

Methods
- button1_Click
- compnt_guard...
- compnt_year_S...
- Dispose
- field_month_Se...
- field_name_Tex...
- field_wage_Text...
- groupBox2_Enter
- InitializeCompo...
- paySlipForm
- paySlipForm_Lo...
- show
- TNI_TextChang...

Implementation

Validation

For the validation purpose I have created the **easyfunction** class which has methods which like onlyNumbers(KeyPressEventArgs e) which takes the e which is the character which is sent from the control (textbox), and this function returns true if the character enter is only numbers which in a sense is helpful as it prevents users from typing invalid character.

```
static class easyFunctions
  {
    public static Boolean onlyNumbers(KeyPressEventArgs e)
    {
      if (((char.IsNumber(e.KeyChar) || (Keys)e.KeyChar == Keys.Back)))
      {
        return false;
      }
      return true;
    }

    public static Boolean onlyLetters(KeyPressEventArgs e)
    {
      if ((char.IsLetter(e.KeyChar) || (Keys)e.KeyChar == Keys.Back) || ((Keys)e.KeyChar == Keys.Decimal) ||
((Keys)e.KeyChar == Keys.Space))
      {
        return false;
      }
      return true;
    }

    public static Boolean bothNumbersLetters(KeyPressEventArgs e)
    {
      if ((onlyLetters(e) == false || onlyNumbers(e) == false))
      {
        return false;
      }
      return true;
    }
  }
```

Where as the other functions bothNumbersLetters and onlyLetters does as the name suggest.

Here in the below code

```
        private void compnt__name_KeyPress(object sender, KeyPressEventArgs e)
        {
            e.Handled = easyFunctions.onlyLetters(e);
        }
```

The onlyLetters functions is called as shown when the keypress event triggers.

Login to the system

Here in login we have two main attribute that is username and password which has to match the database.

```
        private void logmein_Click(object sender, EventArgs e)
        {
            userAccount l=new userAccount();
            if (l.login(username.Text, password.Text))
            {
                MainInfoSysForm m = new MainInfoSysForm();
                m.Show();
                this.Hide();
            }
            else
            {
                MessageBox.Show("Username or password incorrect, Try again.");
            }
        }
```

Above is the event which is triggered when the user clicks the login button where the object is created of userAccount and then the login(…) function is called to check the login credentials match the data in database or not. If it matches then the user is allowed to login if not then they are not allowed to login.

Add Employee

Here in the addEmployeeForm the following code which is the main part as it saves the employee details is given below.

```
        private void compnt_Save_Click(object sender, EventArgs e)
        {
            DateTime d =DateTime.Now;
            string dt = d.ToShortDateString();
            SecurityGuard s = new SecurityGuard();
            s.SecurityGuardDetails(compnt__eid.Text, compnt__name.Text,
compnt__gender.Text, compnt__dob.Text, compnt__address.Text, compnt_postcode.Text,
compnt_phone.Text, dt, float.Parse(compnt_hourlyrate.Text));
            if (s.saveDetailsToDatabase())
            {
                MessageBox.Show("Employee Record has been added Successfully!");
                this.Hide();
            }
            else
            {
                MessageBox.Show("Please enter valid data or user exist with similar
details.");
            }
```

The compnt__Save_Click event is triggered when the user click the save button, here the new object of securityGuard is created which is s and then the method SecurityGuardDetails (…) is called to save the details and then saveDetailsToDatabase() functions is called to save the details which was provided earlier.

Attendance form

Attendance form is the form which allows user to save attendance record.

```
        public void calculate_deduction()
        {
            float ni = 0, tax = 0;
            float wHours = 0, defaultValue = 0;
            wHours = float.Parse("0" + Compnt_workingHours.Text);
            defaultValue = float.Parse("0" + Compnt_niPercentage.Text);
            ni = (wHours * defaultValue) / 100;
            Compnt_NI.Text = "" + ni;

            defaultValue = float.Parse("0" + Compnt_taxPercentage.Text);
            tax = (wHours * defaultValue) / 100;
            Compnt_tax.Text = "" + tax;
        }

        private void Compnt_saveCalculate_Click(object sender, EventArgs e)
        {
            float ni = 0, tax = 0;
            float wHours = 0, defaultValue = 0;
            wHours = float.Parse("0" + Compnt_workingHours.Text);
            defaultValue = float.Parse("0" + Compnt_niPercentage.Text);
            ni = (wHours * defaultValue) / 100;

            defaultValue = float.Parse("0" + Compnt_taxPercentage.Text);
            tax = (wHours * defaultValue) / 100;

            wHours = float.Parse("0" + Compnt_workingHours.Text);

            calculate_deduction();
            float totalEarned = 0;
            totalEarned = (wHours * float.Parse("0" + compnt_hourlyRate.Text))-
(ni+tax);
            MessageBox.Show(" "+totalEarned);
            Attendance a = new Attendance(-1,compnt_employeeID.Text,
int.Parse(Compnt_day.Text),int.Parse(Compnt_month.Text),int.Parse(Compnt_year.Text)
,double.Parse("0"+Compnt_workingHours.Text), totalEarned, ni,tax);
            if(a.saveattendance())
            {
                MessageBox.Show("Attendance has been taken.");
            }
            else
            {
                MessageBox.Show("Attendance wasnt taken!! Please try again.");
            }
        }
```

The `calculate_deduction()` function calculates the Ni deduction and tax deduction where as the `Compnt_saveCalculate_Click(object sender, EventArgs e)` calculate and saves the record to database.

Individual Payslip

Below is the code which generates the individual payslip.

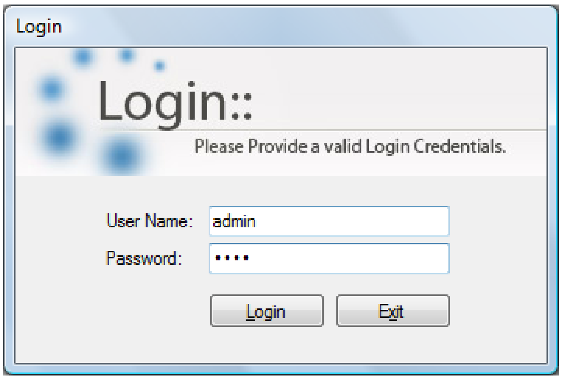
```
        public void show()
        {
            double totalHours = 0, totalEarning = 0, totalNi = 0, totalTax = 0;
            string query = "";
            OleDbConnection connection = new
OleDbConnection(Program.connectionString());
            int payslipMonth = int.Parse(this.compnt_month.Text);
            int payslipYear = int.Parse(this.compnt_year.Text);
            query = "SELECT * FROM Attendance WHERE EmployeeID='" + compnt_ID.Text
+ "' and month_Deploy=" + payslipMonth + " and year_Deploy=" + payslipYear + ";";
            OleDbCommand cmd = new OleDbCommand(query, connection);
            connection.Open();
            OleDbDataReader rd;
            rd = cmd.ExecuteReader();
            while (rd.Read())
            {
                totalHours = totalHours + double.Parse(rd.GetValue(5).ToString());
                totalEarning = totalEarning +
double.Parse(rd.GetValue(6).ToString());
                totalNi = totalNi + double.Parse(rd.GetValue(7).ToString());
                totalTax = totalTax + double.Parse(rd.GetValue(8).ToString());
            }
            rd.Close();
            this.compnt_totalHWorkd.Text = "" + totalHours;
            this.compnt_Earned.Text = totalEarning.ToString();
            this.compnt_Tax.Text = totalTax.ToString();
            this.compnt_NI.Text = totalNi.ToString();
            connection.Close();
            OleDbDataAdapter dAdp = new OleDbDataAdapter(@query, connection);
            DataSet dset = new DataSet("dataGrid");
            dAdp.Fill(dset, "dataGrid");
            compnt_Grid.DataSource = dset.Tables[0];
            compnt_Grid.Update();
        }
```

Here the show function creates the query and then calculates the total Hours, earning, ni deduction, Tax deduction of all the employee as a grand total

Other implementation codes are in appendix.

Testing

1. Login to the system using wrong credential.

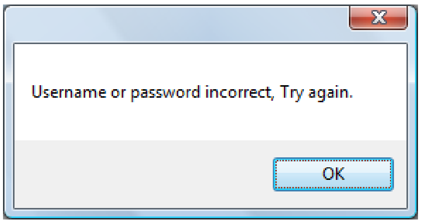


Please Provide a valid Login Credentials.

User Name: admin

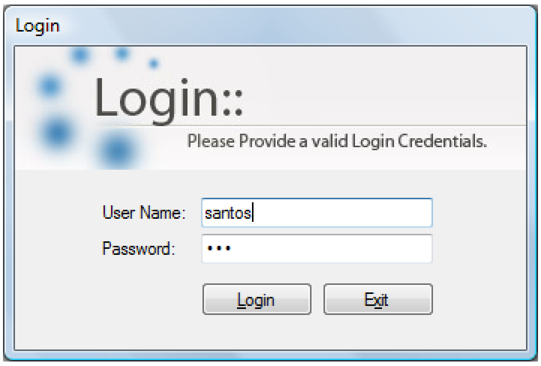
Password: ••••

Below: The system gives notification to the user asking to provide valid login credentials.



Username or password incorrect, Try again.

Below: Login to the system with valid login credentials (Username= “santos” and password =”123”)

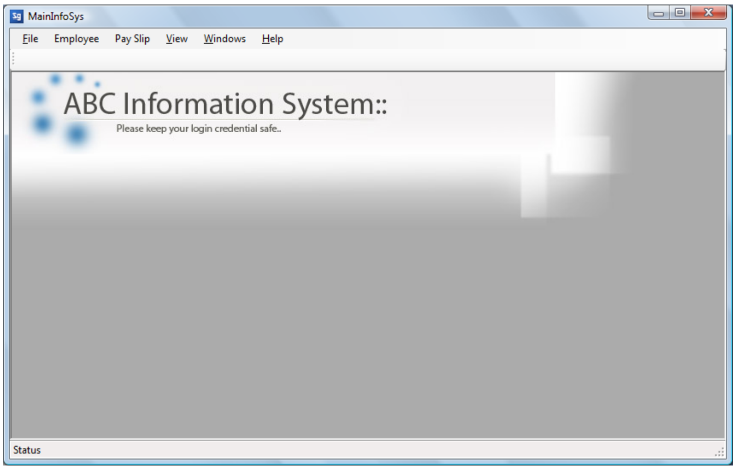


Please Provide a valid Login Credentials.

User Name: santos

Password: •••

As we login to the system with valid login credentials the main MDI window will be shown

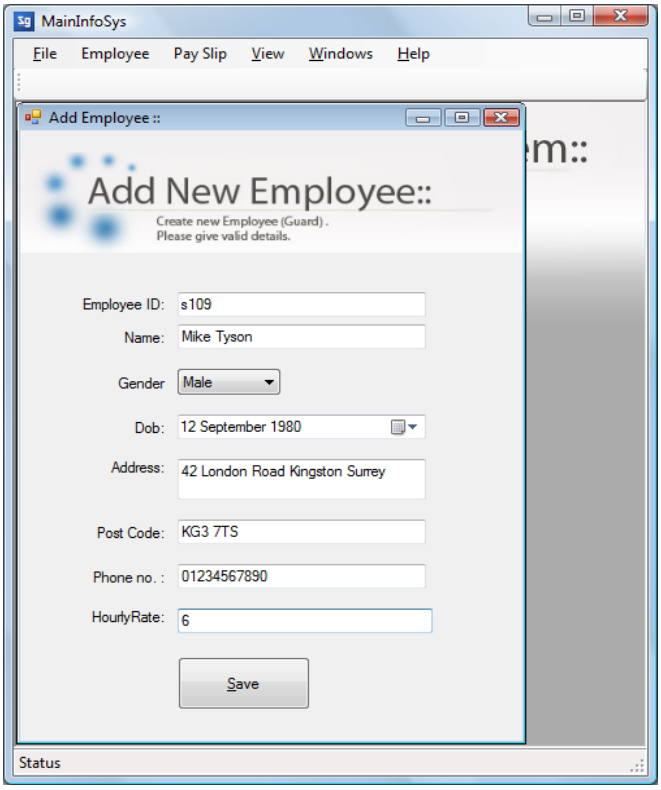


ABC Information System::

Please keep your login credential safe..

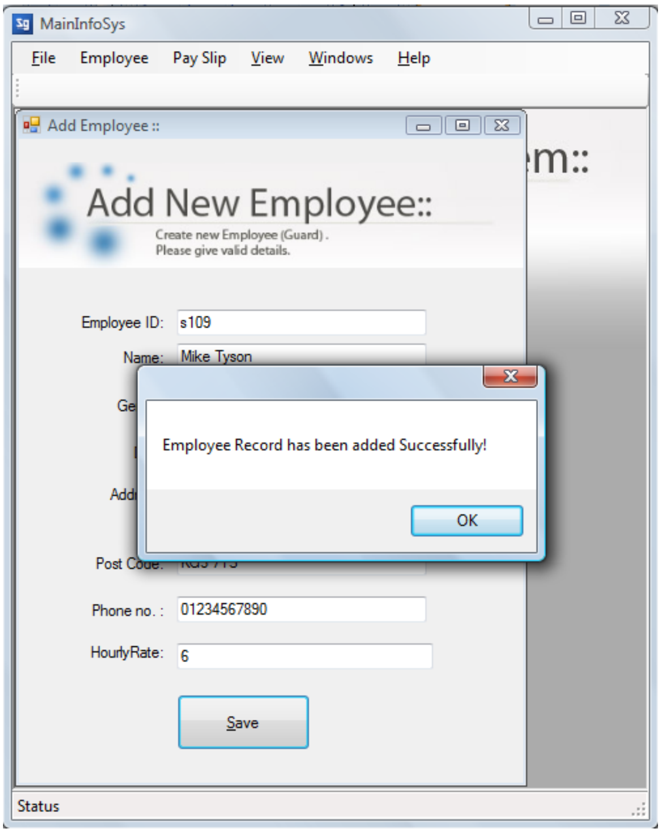
2. Add new Employee (Add new Guard)

Below is the screenshot showing the adding of new Employee with details.



Employee ID: s109
Name: Mike Tyson
Gender: Male
Dob: 12 September 1980
Address: 42 London Road Kingston Surrey
Post Code: KG3 7TS
Phone no. : 01234567890
HourlyRate: 6

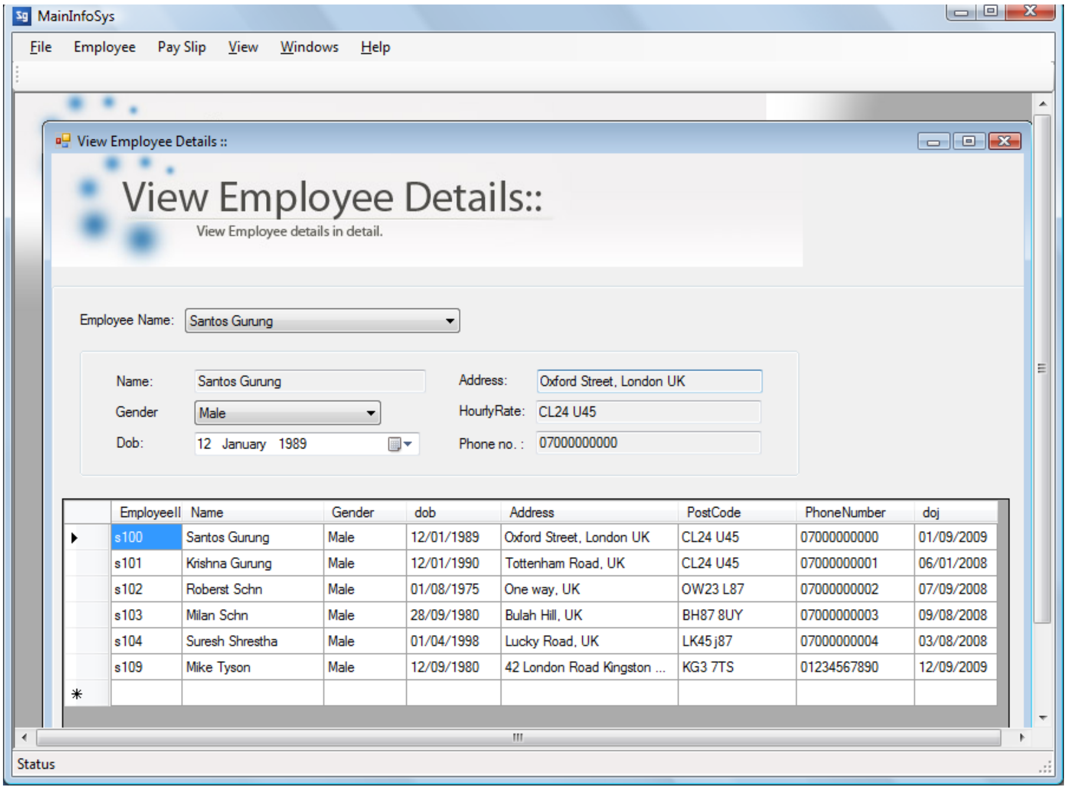
If the system was able to add new employee or Guard it gives notification as shown below.



Employee Record has been added Successfully!

3. View Employee Details

Below is report showing the details of the individual and all the details of the employees.



View Employee Details::

View Employee details in detail.

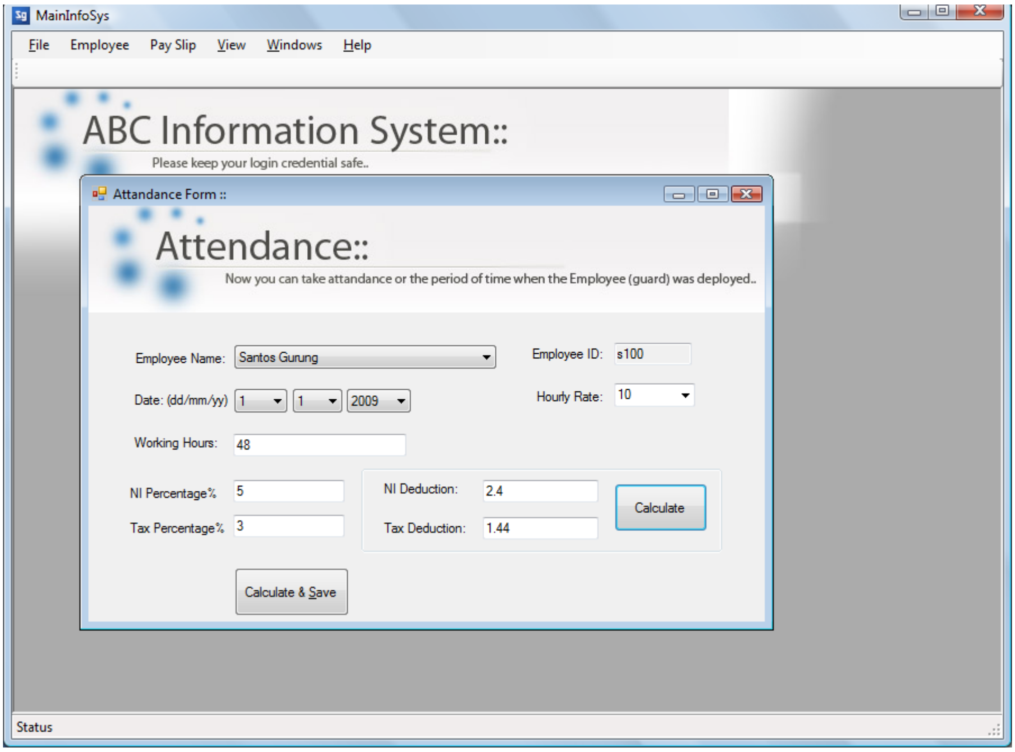
Employee Name: Santos Gurung

Name:	Santos Gurung	Address:	Oxford Street, London UK
Gender	Male	HourlyRate:	CL24 U45
Dob:	12 January 1989	Phone no. :	07000000000

EmployeeI	Name	Gender	dob	Address	PostCode	PhoneNumber	doj
s100	Santos Gurung	Male	12/01/1989	Oxford Street, London UK	CL24 U45	07000000000	01/09/2009
s101	Krishna Gurung	Male	12/01/1990	Tottenham Road, UK	CL24 U45	07000000001	06/01/2008
s102	Roberst Schn	Male	01/08/1975	One way, UK	OW23 L87	07000000002	07/09/2008
s103	Milan Schn	Male	28/09/1980	Bulah Hill, UK	BH87 8UY	07000000003	09/08/2008
s104	Suresh Shrestha	Male	01/04/1998	Lucky Road, UK	LK45 j87	07000000004	03/08/2008
s109	Mike Tyson	Male	12/09/1980	42 London Road Kingston ...	KG3 7TS	01234567890	12/09/2009

4. Taking the attendance of the employee (Guard)

Below is the screenshot showing the attendance being taken where the NI percentage and tax has been calculated and shown to the user.



ABC Information System::

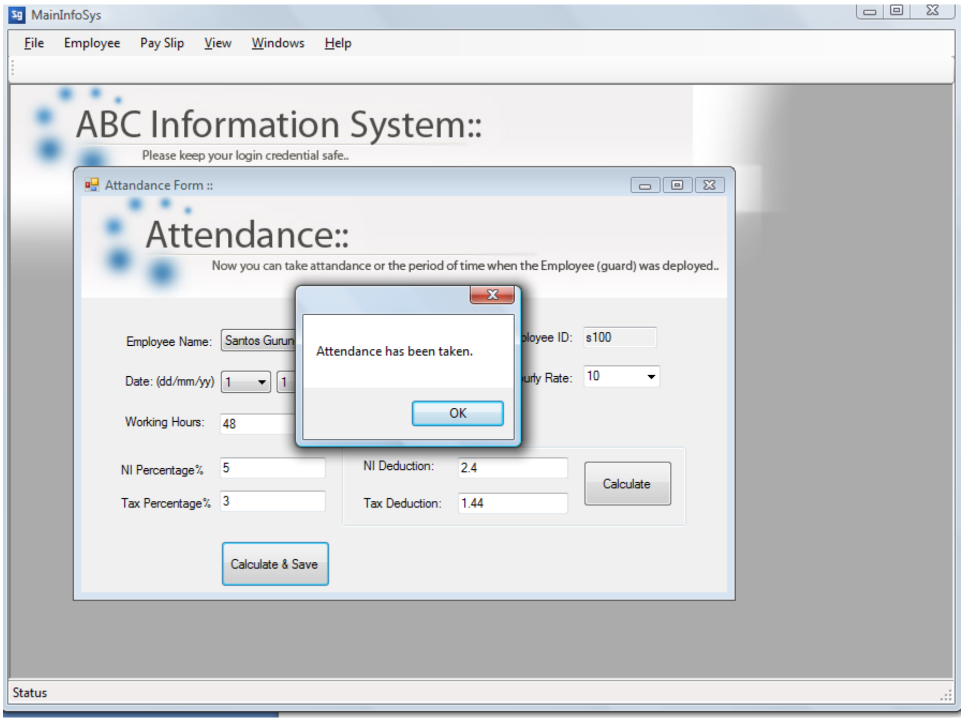
Please keep your login credential safe..

Attendance::

Now you can take attandance or the period of time when the Employee (guard) was deployed..

Employee Name:	Santos Gurung	Employee ID:	s100
Date: (dd/mm/yy)	1 1 2009	Hourly Rate:	10
Working Hours:	48		
NI Percentage%	5	NI Deduction:	2.4
Tax Percentage%	3	Tax Deduction:	1.44

Below is the notification saying the user that the records were saved properly which means the attendance was taken successfully.



ABC Information System::

Please keep your login credential safe..

Attendance::

Now you can take attandance or the period of time when the Employee (guard) was deployed..

Attendance has been taken.

Employee Name: Santos Gurun

Date: (dd/mm/yy) 1 1

Working Hours: 48

ployee ID: s100

urly Rate: 10

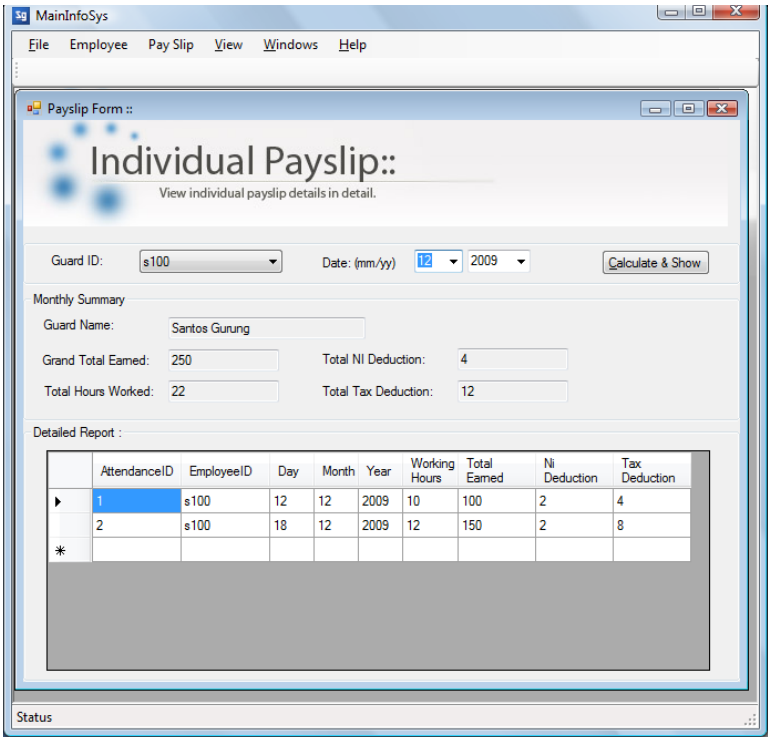
NI Percentage% 5

Tax Percentage% 3

NI Deduction: 2.4

Tax Deduction: 1.44

5. Generate Payslip for individual employee



Individual Payslip::

View individual payslip details in detail.

Guard ID: s100

Date: (mm/yy) 12 2009

Monthly Summary

Guard Name: Santos Gurung

Grand Total Earned: 250

Total Hours Worked: 22

Total NI Deduction: 4

Total Tax Deduction: 12

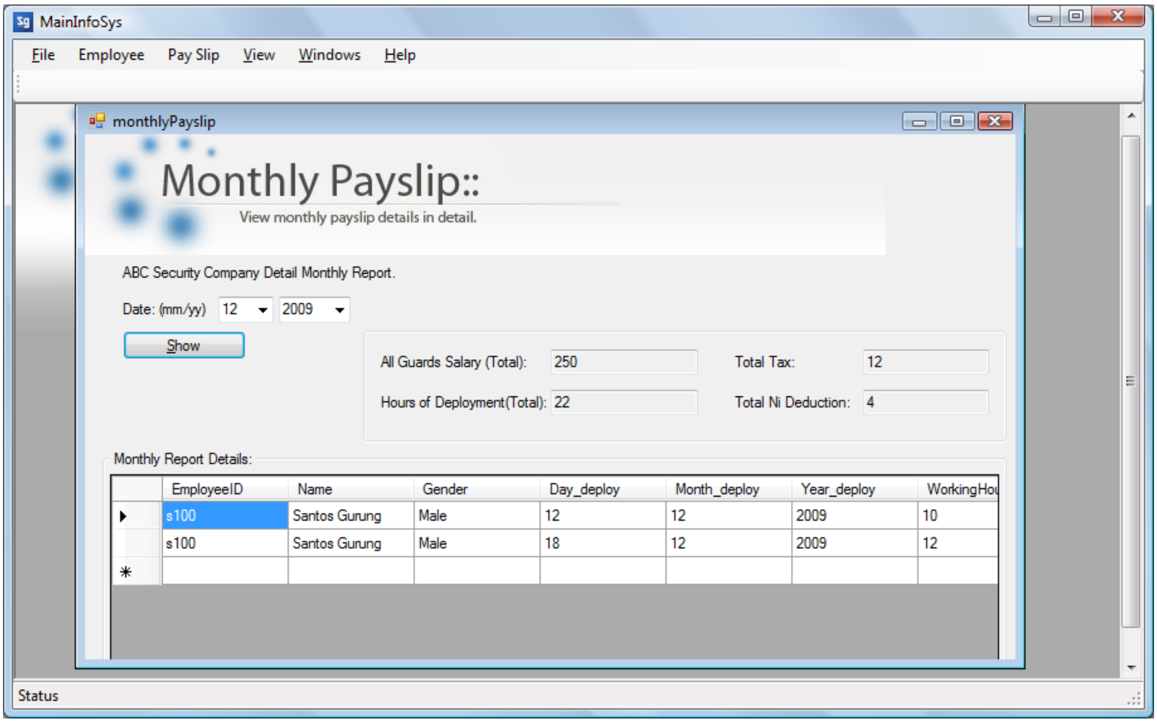
Detailed Report :

	AttendanceID	EmployeeID	Day	Month	Year	Working Hours	Total Earned	Ni Deduction	Tax Deduction
▸	1	s100	12	12	2009	10	100	2	4
	2	s100	18	12	2009	12	150	2	8
*									

Above is the payslip for an individual employee where total earning, hours worked and other deduction are shown. As well as the details are shown below.

6. Generating the monthly payslip

Below is the monthly payslip showing the detail monthly



Monthly Payslip::

View monthly payslip details in detail.

ABC Security Company Detail Monthly Report.

Date: (mm/yy) 12 2009

All Guards Salary (Total): 250

Total Tax: 12

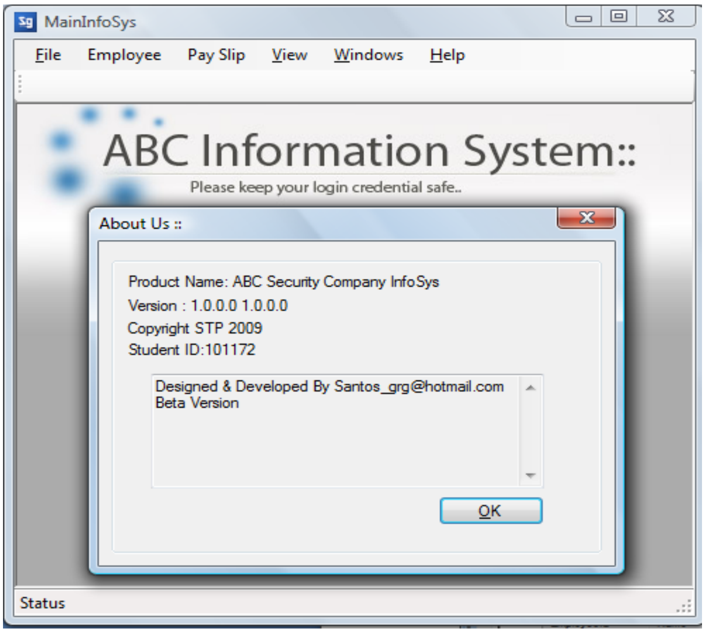
Hours of Deployment(Total): 22

Total Ni Deduction: 4

Monthly Report Details:

EmployeeID	Name	Gender	Day_deploy	Month_deploy	Year_deploy	WorkingHou
s100	Santos Gurung	Male	12	12	2009	10
s100	Santos Gurung	Male	18	12	2009	12

7. About us Form



ABC Information System::

Please keep your login credential safe..

About Us ::

Product Name: ABC Security Company InfoSys
Version : 1.0.0.0 1.0.0.0
Copyright STP 2009
Student ID:101172

Designed & Developed By Santos_grg@hotmail.com
Beta Version

Conclusion

I have learned about .net platform and learned c# and with this I have learned a lot about gui and programming that is visual programming and I can create any information system or any other normal application in .net platform

Appendix

Codes here

Attandance class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.OleDb;
using System.Windows.Forms;

namespace ABC_Security_Company_InfoSys
{
    public class Attendance
    {
        private int attendanceID;
        private string employeeID;
        private int day_Deploy;
        private int month_Deploy;
        private int year_Deploy;
        private double workingHours;
        private double totalEarned;
        private float niDeduction;
        private float taxDeduction;

        public int get_AttendanceID(int attendance_ID)
        {
            return attendance_ID;
        }
        public string get_employeeID(string employee_ID)
        {
            return employee_ID;
        }
        public int get_dayDeploy(int dayDeploy)
        {
            return dayDeploy;
        }
        public int get_monthDeploy(int monthDeploy)
        {
            return monthDeploy;
        }
        public int get_yearDeploy(int yearDeploy)
        {
            return yearDeploy;
        }
        public double get_workingHours(double working_Hours)
        {
            return working_Hours;
        }
        public double get_totalEarned(double total_Earned)
        {
            return total_Earned;
        }
        public float get_niDeduction(float ni_Deduction)
        {
            return ni_Deduction;
        }
        public float get_taxDeduction(float tax_Deduction)
        {
            return tax_Deduction;
        }
        public Attendance(int attendance_ID, string employee_ID, int
deploy_day, int deploy_month, int deploy_year, double working_Hours, double
total_Earned, float ni_Deduction, float tax_Deduction)
        {
            this.attendanceID= attendance_ID;
            this.employeeID = employee_ID;
            this.day_Deploy = deploy_day;
            this.month_Deploy = deploy_month;
            this.year_Deploy = deploy_year;
            this.workingHours = working_Hours;
            this.totalEarned = total_Earned;
            this.niDeduction = ni_Deduction;
            this.taxDeduction = tax_Deduction;
        }

        public Boolean saveattendance()
        {
            Boolean r = false;
            OleDbConnection connection = new
OleDbConnection(Program.connectionString());
            string query;
            query = "INSERT INTO Attendance (EmployeeID, Day_deploy,
Month_deploy, Year_deploy, WorkingHours, TotalEarned, NiDeduction,
TaxDeduction) VALUES ('" + this.employeeID + "'," + this.day_Deploy + "," +
this.month_Deploy + "," + this.year_Deploy + "," + this.workingHours + ","
+ this.totalEarned + "," +this.niDeduction+","+this.taxDeduction+");";

            connection.Open();
            OleDbCommand Cmd = new OleDbCommand(query, connection);
            Cmd.CommandText = query;

            Cmd.CommandType = CommandType.Text;
            if (Cmd.ExecuteNonQuery() > 0)
            {
                r = true;
            }
            else
            {
                r = false;
            }
            connection.Close();
            return r;
        }
    }
}
```

Easyfunction class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace ABC_Security_Company_InfoSys
{
    static class easyFunctions
    {
        public static Boolean onlyNumbers(KeyPressEventArgs e)
        {
            if (((char.IsNumber(e.KeyChar) || (Keys)e.KeyChar ==
Keys.Back)))
            {
                return false;
            }
            return true;
        }

        public static Boolean onlyLetters(KeyPressEventArgs e)
        {
            if ((char.IsLetter(e.KeyChar) || (Keys)e.KeyChar == Keys.Back)
|| ((Keys)e.KeyChar == Keys.Decimal) || ((Keys)e.KeyChar == Keys.Space))
            {
                return false;
            }
            return true;
        }

        public static Boolean bothNumbersLetters(KeyPressEventArgs e)
        {
            if ((onlyLetters(e) == false || onlyNumbers(e) == false))
            {
                return false;
            }
            return true;
        }
    }
}
```

Employee Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.OleDb;
using System.Windows.Forms;

namespace ABC_Security_Company_InfoSys
{
    public class Employee
    {
        //Attributes
        protected string employeeID="";
        protected string name = "";
        protected string gender = "";
        protected string dob = "";
        protected string address = "";
        protected string postCode = "";
        protected string phoneNumber = "";
        protected string doj = "";
        //Functions started
        public string get_employeeID()
        {
            return this.employeeID;
        }
        public string get_name()
        {
            return this.name;
        }
        public string get_gender()
        {
            return this.gender;
        }
        public string get_dob()
        {
            return this.dob;
        }
        public string get_address()
        {
            return this.address;
        }
        public string get_postCode()
        {
            return this.postCode;
        }
        public string get_phoneNumber()
        {
            return this.phoneNumber;
        }
        public string get_doj()
        {
            return this.doj;
        }

        public void savedetails(string employee_ID, string employee_name,
string employee_gender, string employee_dob, string employee_address,
string employee_postCode, string employee_phoneNumber, string employee_doj)
         {
             this.employeeID = employee_ID;
             this.name = employee_name;
             this.gender = employee_gender;
             this.dob = employee_dob;
             this.address = employee_address;
             this.postCode = employee_postCode;
             this.phoneNumber= employee_phoneNumber;
             this.doj= employee_doj;
         }

        public Boolean savetodatabase()
        {
            Boolean r = false;
            OleDbConnection connection = new
OleDbConnection(Program.connectionString());
            string query;
            query = "INSERT INTO Employee(EmployeeID,Name, Gender, dob,
Address, PostCode, PhoneNumber, doj) VALUES ('" +this.employeeID+"','"+
this.name + "','" + this.gender + "','" + this.dob + "','" + this.address +
"','" + this.postCode +"','"+ this.phoneNumber+"','"+this.doj+ "')";
            connection.Open();
            OleDbCommand command = new OleDbCommand(query, connection);
            command.CommandText = query;

            command.CommandType = CommandType.Text;

            if (command.ExecuteNonQuery() > 0)
            {
                r = true;
            }
            else
            {
                r = false;
            }
            connection.Close();
            return r;

        }


    }
}
```

Manager Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
//Class jus for clearance of securityGuard as an inheritance
namespace ABC_Security_Company_InfoSys
{
    public class Manager : Employee
    {
        private double monthlySalary = 0;
        public double get_monthlySalary()
        {
            return this.monthlySalary;
        }
    }
}
```

SecurityGuard Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.OleDb;
using System.Windows.Forms;

namespace ABC_Security_Company_InfoSys
{
    public class SecurityGuard : Employee
    {
        public void SecurityGuardDetails(string employee_ID, string
employee_name, string employee_gender, string employee_dob, string
employee_address, string employee_postCode, string employee_phoneNumber,
string employee_doj, float securityGuard_hourlyRate)
        {

savedetails(employee_ID,employee_name,employee_gender,employee_dob,employee
_address,employee_postCode,employee_phoneNumber, employee_doj);

            this.hourlyRate = securityGuard_hourlyRate;
        }
        private float hourlyRate = 0;

        public float get_hourlyRate()
        {
            return this.hourlyRate;
        }
        public Boolean saveDetailsToDatabase()
        {
            savetodatabase();
            Boolean r = false;
            OleDbConnection connection = new
OleDbConnection(Program.connectionString());
            string query;
            string ed = this.employeeID;
            query = "INSERT INTO SecurityGuard(EmployeeID,HourlyRate)
VALUES ('" + ed+"',"+this.hourlyRate+")";
            connection.Open();
            OleDbCommand command = new OleDbCommand(query, connection);
            command.CommandText = query;

            command.CommandType = CommandType.Text;
            if (command.ExecuteNonQuery() > 0)
            {
                r = true;
            }
            else
            {
                r = false;
            }
            connection.Close();
            return r;
        }
    }
}
```

UserAccounts Class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data;
using System.Data.OleDb;
using System.Windows.Forms;

namespace ABC_Security_Company_InfoSys
{
    class userAccount
    {
        private string userName;
        private string password;
        private string employeeID;

        public void setEmployeeID(string emp)
        {
            employeeID = emp;
        }
        public Boolean login(string user_name, string passwrd)
        {
            userName= user_name;
            password = passwrd;
            Boolean r = false;
            OleDbConnection connection = new
OleDbConnection(Program.connectionString());
            string query;
            query = "SELECT count(*) FROM UserAccount where UserName='" +
userName + "' AND Password='" + password + "'";

            connection.Open();
            OleDbCommand command = new OleDbCommand(query, connection);
            command.CommandText = query;

            command.CommandType = CommandType.Text;
            int cnt=(Int32)command.ExecuteScalar();
            if ( cnt> 0)
            {
                r = true;
            }
            else
            {
                r = false;
            }
            connection.Close();
            return r;
        }
    }
}
```

MonthlyPayslip Form

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.OleDb;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace ABC_Security_Company_InfoSys
{
    public partial class monthlyPayslip : Form
    {
        public monthlyPayslip()
        {
            InitializeComponent();
        }

        private void monthlyPayslip_Load(object sender, EventArgs e)
        {
            // TODO: This line of code loads data into the
'aBC_Security_dbDataSet.Attendance' table. You can move, or remove it, as
needed.

//this.attendanceTableAdapter.Fill(this.aBC_Security_dbDataSet.Attendance);

        }

        private void show_Click(object sender, EventArgs e)
        {
            showSlip();
        }

        public void showSlip()
        {
            string query="";
            OleDbConnection connection = new
OleDbConnection(Program.connectionString());
            query = "SELECT Employee.EmployeeID, Employee.Name,
Employee.Gender,";
```

```
            query=query+"Attendance.Day_deploy, Attendance.Month_deploy, 
Attendance.Year_deploy,";
            query=query+"Attendance.WorkingHours, Attendance.NiDeduction, 
Attendance.TaxDeduction, Attendance.TotalEarned";
            query = query + " FROM Attendance INNER JOIN Employee ON 
Attendance.EmployeeID = Employee.EmployeeID";
            query = query + " WHERE Attendance.Month_deploy=" + 
int.Parse(this.compnt_month.Text) + " And  Attendance.Year_deploy=" + 
int.Parse(this.compnt_year.Text)+";";
            connection.Open();
            OleDbCommand command = new OleDbCommand(query, connection);
            OleDbDataReader reader;
            reader = command.ExecuteReader();
            double totalHours = 0,totalEarning = 0, totalNi = 0, totalTax = 
0;
            while (reader.Read())
            {
                totalHours = totalHours + 
double.Parse(reader.GetValue(6).ToString());
                totalNi = totalNi + 
double.Parse(reader.GetValue(7).ToString());
                totalTax = totalTax + 
double.Parse(reader.GetValue(8).ToString());
                totalEarning = totalEarning + 
double.Parse(reader.GetValue(9).ToString());
            }

            reader.Close();
            this.compnt_hours.Text = "" + totalHours;
            this.compnt_Salary.Text = totalEarning.ToString();
            this.compnt_Tax.Text = totalTax.ToString();
            this.compnt_NI.Text = totalNi.ToString();
            connection.Close();
            OleDbDataAdapter dA = new OleDbDataAdapter(@query, connection);
            DataSet ds = new DataSet("dataGrid");
            dA.Fill(ds, "dataGrid");
            compnt_Grid.DataSource = ds.Tables[0];
            compnt_Grid.Update();
        }
    }
}
```