



UNIVERSITI  
TEKNOLOGI  
PETRONAS

**TFB1033/TEB1043: OBJECT ORIENTED PROGRAMMING**  
**SEMESTER MAY 2025**

**PROJECT REPORT**

**TITLE: HR MANAGEMENT SYSTEM**

**LECTURER: DR NORDIN ZAKARIA**

NAME	ID
Lee Jie Yi	22011597
Aminudin Razif Bin Arman	22007578
Muhammad Danish Bin Jamal	22011473
Syed Muhammad Kadzim Alattas Bin Syed Sheikh Alattas	22011248
Desmond Sua Anak Tony	24007648

## TABLE OF CONTENT

TABLE OF CONTENT.....	2
1.0 DESCRIPTION.....	3
2.0 TEAM ORGANISATION.....	4
3.0 UNIFIED MODELLING LANGUAGE(UML).....	6
3.1 UML DIAGRAM.....	6
3.2 UML DETAILS.....	7
4.0 IMPLEMENTATION.....	13
4.1 CODE.....	13
4.2 OUTPUT.....	22

## 1.0 DESCRIPTION

This project, Human Resources (HR) Management System is developed using C# programming language with Windows Form (WinForms) Framework. The system purpose is to automate essential HR task in a company. For example, such as employee data management, leave application tracking system, staff attendance tracking and payroll processing.

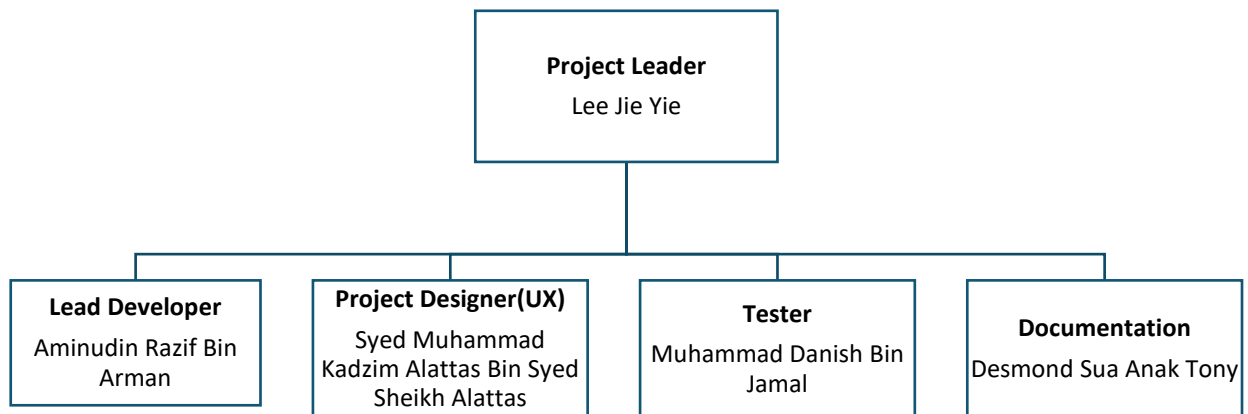
The application feature user friendly graphical user interface (GUI) that enables HR staff to perform their task efficiently. This also includes adding, updating, deleting and finding employee records. The reason is to ensure all employee related information is securely stored and easily accessible.

Key features of the system include:

- **Employee Management:** Maintain employee personal information such as contact details and job description
- **User Login Panel:** For HR personnel to access the mainframe of the program for their record.
- **Leave Management:** Tracking employee leave request, approval and leave balances.
- **Attendance Tracking:** Tracking employee attendance status.
- **Payroll Module System:** Initiate the salary of the employee by calculating hours worked, bonuses and penalty deduction.

To sum up, this system helps reduce manual workload, minimizes errors in record keeping, and improves overall HR process efficiency within an organization. It is suitable for small and medium enterprises (SME) seeking for a desktop-based HR management solution.

## 2.0 TEAM ORGANISATION



### Description

#### 1. Project Leader

Person: Lee Jie Yi

Role

- The main coordinator and overall decision-maker for the project.
- Responsible for planning, organizing, and ensuring the team meets project goals and deadlines.
- Communicates with stakeholders and resolves any major issues.

#### 2. Lead Developer

Person: Aminudin Razif Bin Arman

Role:

- Responsible for coding, designing the software architecture, and solving technical problems.
- Guides other developers (if any) and ensures code quality.
- Implements core features according to the project requirements.

### 3. Project Designer (UX)

Person: Syed Muhammad Kadzim Alattas Bin Syed Sheikh Alattas

Role:

- Focuses on user experience (UX) and the design aspect of the project.
- Ensures the application is user-friendly and visually appealing.

### 4. Tester

Person: Muhammad Danish Bin Jamal

Role:

- Responsible for testing the software for bugs and errors.
- Develops test cases and scenarios.
- Ensures the software works as intended and meets quality standards.

### 5. Documentation

Person: Desmond Sua Anak Tony

Role:

- Prepares and maintains all project documentation
- Ensures records are up to date and comprehensive for future reference.
- Helps team members and users understand the system and processes.

## 3.0 UNIFIED MODELLING LANGUAGE(UML)

### 3.1 UML DIAGRAM

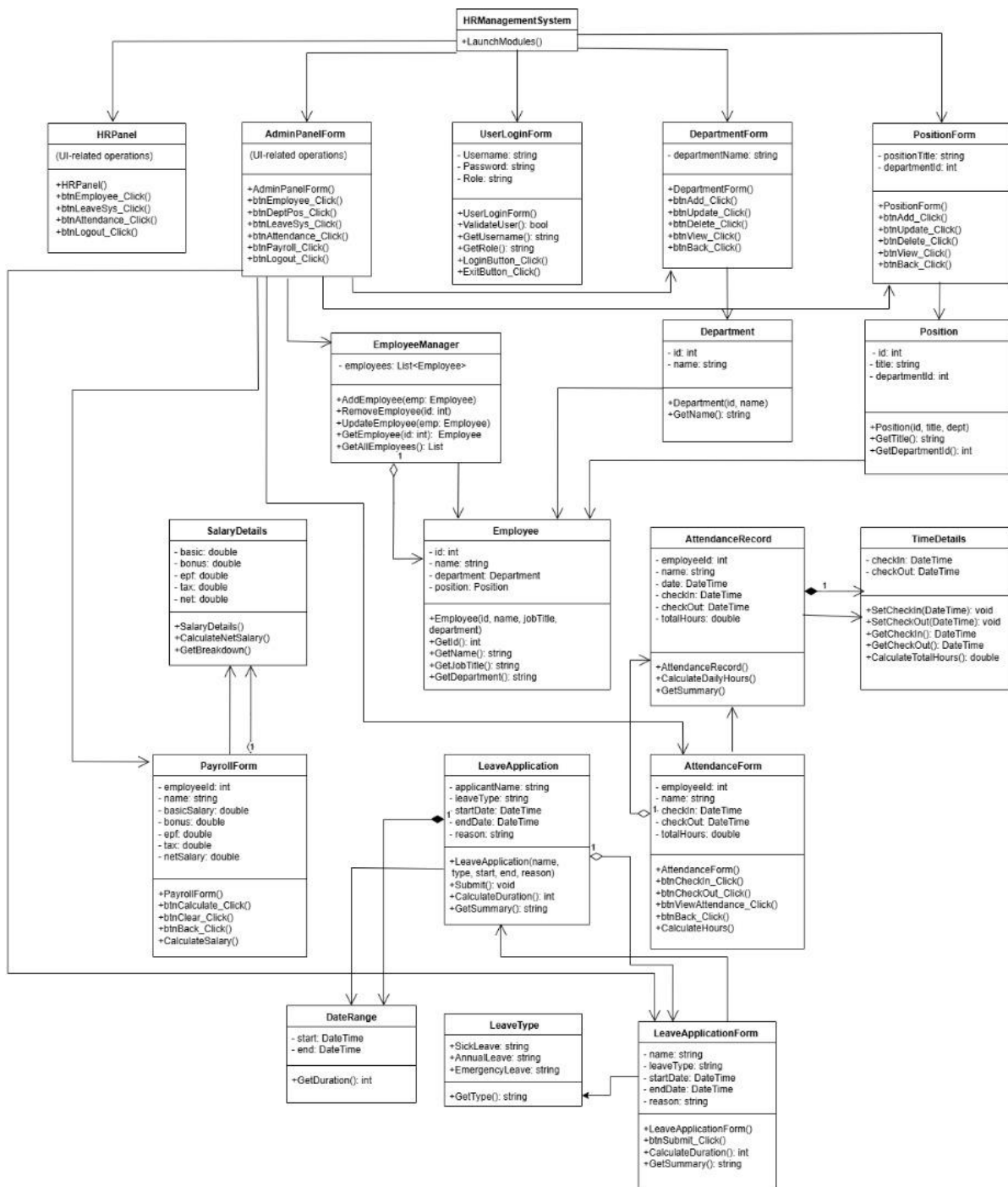


Figure 1: UML Diagram

## 3.2 UML DETAILS

### 1. Main Components & UI Classes

HRManagementSystem

- Method: LaunchModules()
- Role: Central system class, responsible for launching different modules/forms.

Panels and Forms (UI Layer)

- HRPanel / AdminPanelForm / UserLoginForm / DepartmentForm / PositionForm
  - Role: Handle UI-related operations (buttons and user actions).
  - Methods: Button click event handlers (e.g., btnEmployee\_Click(), btnAdd\_Click()).

### 2. Core Domain Classes

EmployeeManager

- Attributes: employees: List<Employee>
- Methods:
  - AddEmployee(emp: Employee)
  - RemoveEmployee(id: int)
  - UpdateEmployee(emp: Employee)
  - GetEmployee(id: int): Employee
  - GetAllEmployees(): List<Employee>
- Relationship: Aggregates/manages multiple Employee objects.

Employee

- Attributes:
  - id: int
  - name: string

- department: Department
  - position: Position
- Methods:
  - GetId()
  - GetName()
  - GetJobTitle()
  - GetDepartment()
- Relationships:
  - Association with Department and Position
  - Linked to: SalaryDetails, AttendanceRecord, LeaveApplication

## Department

- Attributes:
  - id: int
  - name: string
- Methods:
  - GetName()
  - GetId()
- Relationship: Used by Employee, referenced in DepartmentForm.

## Position

- Attributes:
  - id: int
  - title: string
  - departmentId: int



- Methods:
  - GetId()
  - GetTitle()
  - GetDepartmentId()
- Relationship: Used by Employee, referenced in PositionForm.

### 3. Salary & Payroll

#### SalaryDetails

- Attributes:
  - basic, bonus, ot, net, total (all double)
- Methods:
  - CalculateNetSalary()
  - GetBreakdown()
- Relationship: Associated with PayrollForm and Employee.

#### PayrollForm

- Attributes: Employee details and salary breakdown.
- Methods: Button event handlers for payroll actions, salary calculation.
- Relationship: Uses SalaryDetails.

### 4. Attendance Management

#### AttendanceRecord

- Attributes:
  - employeeId: int
  - name: string
  - date: DateTime

- checkIn: DateTime
  - checkOut: DateTime
  - totalHours: double
- Methods:
  - CalculateDailyHours()
  - GetSummary()
- Relationship:
  - Composition with TimeDetails (AttendanceRecord uses TimeDetails).

#### TimeDetails

- Attributes:
  - checkIn: DateTime
  - checkOut: DateTime
- Methods:
  - SetCheckIn(DateTime)
  - SetCheckOut(DateTime)
  - CalculateTotalHours()
- Relationship: Used by AttendanceRecord.

#### AttendanceForm

- Attributes: Attendance data.
- Methods: Button event handlers for attendance, CalculateHours().
- Relationship: Connects UI to AttendanceRecord.

## 5. Leave Management

### LeaveApplication

- Attributes:
  - applicantName: string
  - leaveType: string
  - startDate: DateTime
  - endDate: DateTime
  - reason: string
- Methods:
  - CalculateDuration()
  - GetSummary()
- Relationship:
  - Association with LeaveType and DateRange.

### DateRange

- Attributes:
  - start: DateTime
  - end: DateTime
- Methods:
  - GetDuration()
- Relationship: Used by LeaveApplication.

### LeaveType

- Attributes:
  - SickLeave: string

- AnnualLeave: string
  - EmergencyLeave: string
- Methods:
  - GetType()
- Relationship: Used by LeaveApplication.

#### LeaveApplicationForm

- Attributes: All leave application fields.
- Methods:
  - btnSubmit\_Click()
  - btnClear\_Click()
  - btnBack\_Click()
  - GetSummary()
- Relationship: Connects UI to LeaveApplication

## 4.0 IMPLEMENTATION

### 4.1 CODE

#### Employee Management

```
namespace EmployeeManagementWinForms
{
    4 references
    public class Employee
    {
        6 references
        public int Id { get; set; }
        5 references
        public string Name { get; set; }
        5 references
        public string JobTitle { get; set; }
        5 references
        public string Department { get; set; }

        1 reference
        public Employee(int id, string name, string jobTitle, string department)
        {
            Id = id;
            Name = name;
            JobTitle = jobTitle;
            Department = department;
        }
    }
}
```

Figure 2: Employee.cs

```
internal class EmployeeManager
{
    List<Employee> list = new List<Employee>();

    0 references
    public void AddEmployee(Employee employee)
    {
        list.Add(employee);
    }

    0 references
    public void RemoveEmployee(int id)
    {
        var employee = list.FirstOrDefault(e => e.Id == id);
        if (employee != null)
        {
            list.Remove(employee);
        }
    }

    0 references
    public void UpdateEmployee(int id, Employee updatedEmployee)
    {
        var employee = list.FirstOrDefault(e => e.Id == id);
        if (employee != null)
        {
            employee.Name = updatedEmployee.Name;
            employee.JobTitle = updatedEmployee.JobTitle;
            employee.Department = updatedEmployee.Department;
        }
    }

    0 references
    public List<Employee> GetAllEmployees()
    {
        return list;
    }
}
```

Figure 3: EmployeeManager.cs

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Windows.Forms;
5
6  namespace EmployeeManagementWinForms
7  {
8      3 references
9      public partial class MainForm : Form
10     {
11         private List<Employee> employees = new List<Employee>();
12         private int nextId = 1;
13
14         1 reference
15         public MainForm()
16         {
17             InitializeComponent();
18         }
19
20         1 reference
21         private void btnAdd_Click(object sender, EventArgs e)
22         {
23             string name = txtName.Text.Trim();
24             string job = txtJob.Text.Trim();
25             string dept = txtDept.Text.Trim();
26
27             if (string.IsNullOrEmpty(name))
28             {
29                 MessageBox.Show("Name cannot be empty.");
30                 return;
31             }
32
33             employees.Add(new Employee(nextId++, name, job, dept));
34
35             RefreshGrid();
36             ClearInputs();
37         }
38     }
39 }

```

Figure 4 : Mainform.cs

```

36     private void btnEdit_Click(object sender, EventArgs e)
37     {
38         if (gridEmployees.SelectedRows.Count == 0) return;
39
40         int id = (int)gridEmployees.SelectedRows[0].Cells[0].Value;
41         var emp = employees.FirstOrDefault(e => e.Id == id);
42         if (emp != null)
43         {
44             emp.Name = txtName.Text.Trim();
45             emp.JobTitle = txtJob.Text.Trim();
46             emp.Department = txtDept.Text.Trim();
47             RefreshGrid();
48         }
49     }
50
51     1 reference
52     private void btnDelete_Click(object sender, EventArgs e)
53     {
54         if (gridEmployees.SelectedRows.Count == 0) return;
55
56         int id = (int)gridEmployees.SelectedRows[0].Cells[0].Value;
57         employees.RemoveAll(e => e.Id == id);
58         RefreshGrid();
59         ClearInputs();
60     }
61
62     1 reference
63     private void gridEmployees_SelectionChanged(object sender, EventArgs e)
64     {
65         if (gridEmployees.SelectedRows.Count == 0) return;
66
67         int id = (int)gridEmployees.SelectedRows[0].Cells[0].Value;
68         var emp = employees.FirstOrDefault(e => e.Id == id);
69         if (emp != null)
70         {
71             txtName.Text = emp.Name;
72             txtJob.Text = emp.JobTitle;
73             txtDept.Text = emp.Department;
74         }
75     }
76 }

```

Figure 5: Mainform.cs(cont')

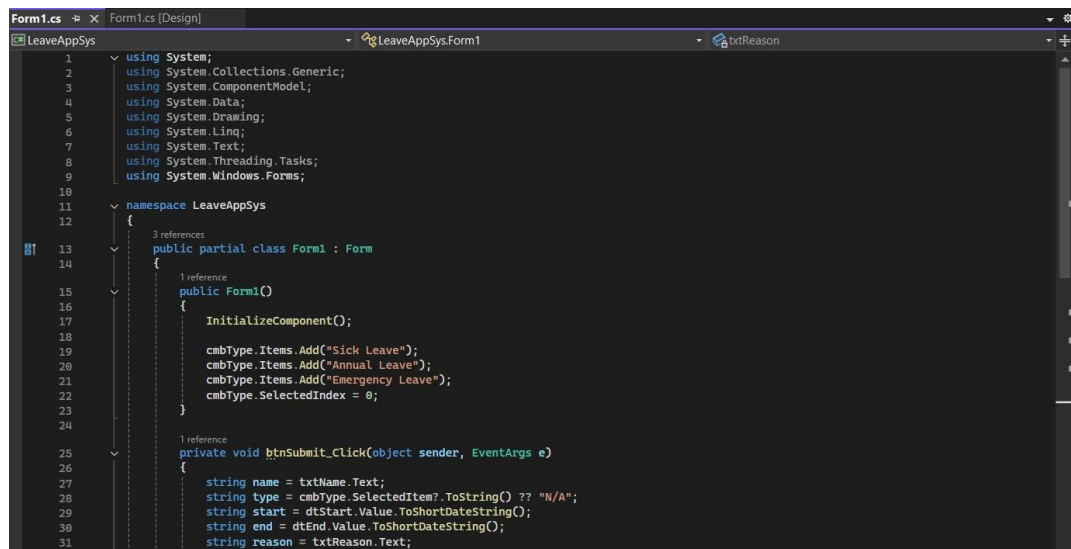
```

75 3 references
76 private void RefreshGrid()
77 {
78     gridEmployees.DataSource = null;
79     gridEmployees.DataSource = employees.Select(e => new
80     {
81         e.Id,
82         e.Name,
83         e.JobTitle,
84         e.Department
85     }).ToList();
86 }
87 2 references
88 private void ClearInputs()
89 {
90     txtName.Clear();
91     txtJob.Clear();
92     txtDept.Clear();
93 }
94 1 reference
95 private void txtDept_TextChanged(object sender, EventArgs e)
96 {
97 }
98 }
99 }
100

```

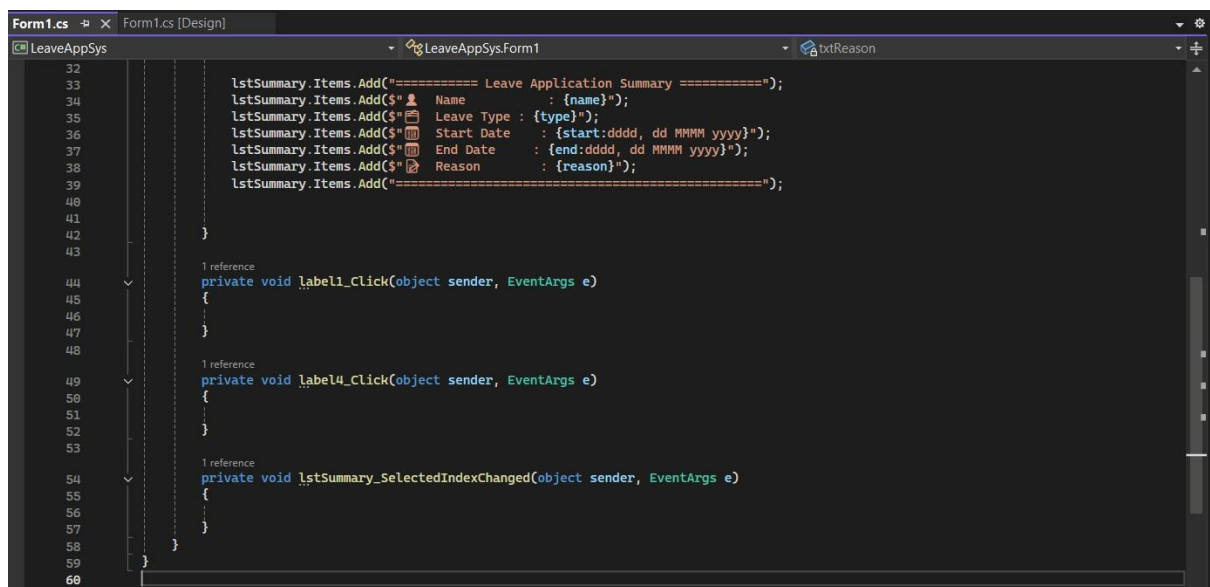
Figure 6: Mainform.cs(cont')

## Leave Application



```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace LeaveAppSys
12 {
13     3 references
14     public partial class Form1 : Form
15     {
16         1 reference
17         public Form1()
18         {
19             InitializeComponent();
20             cmbType.Items.Add("Sick Leave");
21             cmbType.Items.Add("Annual Leave");
22             cmbType.Items.Add("Emergency Leave");
23             cmbType.SelectedIndex = 0;
24         }
25
26         1 reference
27         private void btnSubmit_Click(object sender, EventArgs e)
28         {
29             string name = txtName.Text;
30             string type = cmbType.SelectedItem?.ToString() ?? "N/A";
31             string start = dtStart.Value.ToShortDateString();
32             string end = dtEnd.Value.ToShortDateString();
33             string reason = txtReason.Text;
```

Figure 7: LeaveAppSys.cs



```
32
33     lstSummary.Items.Add("===== Leave Application Summary =====");
34     lstSummary.Items.Add($"{name} Name : {name}");
35     lstSummary.Items.Add($"{type} Leave Type : {type}");
36     lstSummary.Items.Add($"{start} Start Date : {start:dddd, dd MMMM yyyy}");
37     lstSummary.Items.Add($"{end} End Date : {end:dddd, dd MMMM yyyy}");
38     lstSummary.Items.Add($"{reason} Reason : {reason}");
39     lstSummary.Items.Add("=====");
40
41
42
43
44     1 reference
45     private void lbl1_Click(object sender, EventArgs e)
46     {
47     }
48
49     1 reference
50     private void lbl4_Click(object sender, EventArgs e)
51     {
52     }
53
54     1 reference
55     private void lstSummary_SelectedIndexChanged(object sender, EventArgs e)
56     {
57     }
58
59
60
```

Figure 8: LeaveAppSys.cs (cont ')



## Payroll Module

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace PayrollSystem_
8  {
9      1 reference
10     public class PaySlip
11     {
12         2 references
13         public string Name { get; set; }
14         2 references
15         public string Position { get; set; }
16         5 references
17         public decimal BasicSalary { get; set; }
18         3 references
19         public decimal Bonus { get; set; }
20         3 references
21         public decimal Allowance { get; set; }
22         3 references
23         public decimal OvertimeHours { get; set; }
24         3 references
25         public decimal TaxDeduction { get; set; }
26         3 references
27         public decimal LateDeduction { get; set; }
28
29         2 references
30         public decimal OvertimeRate => 20m;
31         2 references
32         public decimal OvertimePay => OvertimeHours * OvertimeRate;
33         2 references
34         public decimal EPF => BasicSalary * 0.11m;
35         2 references
36         public decimal SOCSO => BasicSalary * 0.005m;
37         2 references
38         public decimal GrossSalary => BasicSalary + Bonus + Allowance + OvertimePay;
39         2 references
40         public decimal TotalDeductions => TaxDeduction + LateDeduction + EPF + SOCSO;
41         1 reference
42         public decimal NetSalary => GrossSalary - TotalDeductions;
43
44     1 reference
45     public override string ToString()
46     {
47         return
48             "===== 📄 PAY SLIP =====\n" +
49             $" Name           : {Name}\n" +
50             $" Position        : {Position}\n" +
51             "-----\n" +
52             " EARNINGS\n" +
53             $" • Basic Salary   : RM{BasicSalary,10:0.00}\n" +
54             $" • Bonus          : RM{Bonus,10:0.00}\n" +
55             $" • Allowance      : RM{Allowance,10:0.00}\n" +
56             $" • Overtime ({OvertimeHours} hrs @ RM{OvertimeRate}): RM{OvertimePay,10:0.00}\n" +
57             $" • Gross Salary   : RM{GrossSalary,10:0.00}\n" +
58             "-----\n" +
59             " DEDUCTIONS\n" +
60             $" • EPF (11%)      : RM{EPF,10:0.00}\n" +
61             $" • SOCSO (0.5%)   : RM{SOCSO,10:0.00}\n" +
62             $" • Tax Deduction  : RM{TaxDeduction,10:0.00}\n" +
63             $" • Late Deduction : RM{LateDeduction,10:0.00}\n" +
64             $" • Total Deductions : RM{TotalDeductions,10:0.00}\n" +
65             "-----\n" +
66             $" Net Salary      : RM{NetSalary,10:0.00}\n" +
67             "===== ";
68     }
69 }

```

Figure 9: Payslip.cs

## Attendance tracking

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Windows.Forms;
5
6  namespace AttendanceApp
7  {
8      public partial class HRAttendanceTrackingWFForm : Form
9      {
10         private List<AttendanceRecord> records = new List<AttendanceRecord>();
11
12         public HRAttendanceTrackingWFForm()
13         {
14             InitializeComponent();
15
16             // Setup grid columns
17             dgvRecords.Columns.Add("EmployeeId", "Employee ID");
18             dgvRecords.Columns.Add("CheckInTime", "Check-In Time");
19             dgvRecords.Columns.Add("CheckOutTime", "Check-Out Time");
20             dgvRecords.Columns.Add("WorkedTime", "Worked Hours");
21         }
22
23         private void btnCheckIn_Click(object sender, EventArgs e)
24         {
25             if (!int.TryParse(txtEmployeeId.Text, out int id))
26             {
27                 MessageBox.Show("Invalid Employee ID");
28                 return;
29             }
30
31             if (records.Any(r => r.EmployeeId == id && !r.CheckOutTime.HasValue))
32             {
33                 MessageBox.Show("Already checked in");
34                 return;
35             }
36
37             records.Add(new AttendanceRecord
38             {
39                 EmployeeId = id,
40                 CheckInTime = DateTime.Now
41             });
```

Figure 10: HRAttendanceTracking.cs

```
43         MessageBox.Show($"Employee {id} checked in.");
44         txtEmployeeId.Clear();
45     }
46
47     private void btnCheckOut_Click(object sender, EventArgs e)
48     {
49         if (!int.TryParse(txtEmployeeId.Text, out int id))
50         {
51             MessageBox.Show("Invalid Employee ID");
52             return;
53         }
54
55         var rec = records.LastOrDefault(r => r.EmployeeId == id && !r.CheckOutTime.HasValue);
56         if (rec == null)
57         {
58             MessageBox.Show("No active check-in found.");
59             return;
60         }
61
62         rec.CheckOutTime = DateTime.Now;
63         MessageBox.Show($"Employee {id} checked out.\nWorked: {rec.TotalHoursWorked?.ToString(@"hh\:mm")}");
64         txtEmployeeId.Clear();
65     }
66
67     private void btnShowRecords_Click(object sender, EventArgs e)
68     {
69         dgvRecords.Rows.Clear();
70
71         foreach (var rec in records)
72         {
73             string checkIn = rec.CheckInTime.ToString(@"hh\:mm tt");
74             string checkOut = rec.CheckOutTime.HasValue ? rec.CheckOutTime.Value.ToString(@"hh\:mm tt") : "N/A";
75             string worked = rec.TotalHoursWorked.HasValue ? rec.TotalHoursWorked.Value.ToString(@"hh\:mm") : "N/A";
76
77             dgvRecords.Rows.Add(rec.EmployeeId, checkIn, checkOut, worked);
78         }
79     }
80 }
81 }
```

Figure 11: HRAttendanceTracking.cs (cont')

## Admin + HR Login

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using PayrollSystem_;
11 using LeaveAppSys;
12
13 namespace HRManagementSystem
14 {
15     public partial class AdminPanelForm : Form
16     {
17         public AdminPanelForm()
18         {
19             InitializeComponent();
20
21             // Attach Click Event Handlers
22             btnEmployeeManagement.Click += btnEmployeeManagement_Click;
23             btnDepartmentPosition.Click += btnDepartmentPosition_Click;
24             btnLeaveSystem.Click += btnLeaveSystem_Click;
25             btnAttendanceTracking.Click += btnAttendanceTracking_Click;
26             btnPayrollSystem.Click += btnPayrollSystem_Click;
27             btnLogout.Click += btnLogout_Click;
28         }
29
30         private void AdminPanelForm_Load(object sender, EventArgs e)
31         {
32             // You can add any initialization code here if needed.
33         }
34     }
35 }
```

Figure 12: AdminPanelForm.cs

```

35 private void btnEmployeeManagement_Click(object sender, EventArgs e)
36 {
37     MessageBox.Show("Employee Management is not implemented yet.");
38     // TODO: Open Employee Management Form
39 }
40
41 private void btnDepartmentPosition_Click(object sender, EventArgs e)
42 {
43     MessageBox.Show("Department & Position is not implemented yet.");
44     // TODO: Open Department & Position Form
45 }
46
47 private void btnLeaveSystem_Click(object sender, EventArgs e)
48 {
49     private void btnLeaveSystem_Click(object sender, EventArgs e)
50     {
51         Form1 leaveForm = new Form1(); // or LeaveAppSysForm if renamed
52         leaveForm.Show();
53     }
54 }
55 private void btnAttendanceTracking_Click(object sender, EventArgs e)
56 {
57     MessageBox.Show("Attendance Tracking is not implemented yet.");
58     // TODO: Open Attendance Tracking Form
59 }
60
61 private void btnPayrollSystem_Click(object sender, EventArgs e)
62 {
63     PayRollForm payrollForm = new PayRollForm();
64     payrollForm.Show();
65 }
66
67
68 private void btnLogout_Click(object sender, EventArgs e)
69 {
70     // Logout
71     UserLoginForm loginForm = new UserLoginForm();
72     loginForm.Show();
73     this.Hide();
74 }
75 }
76

```

Figure 13: AdminPanelForm.cs (cont')

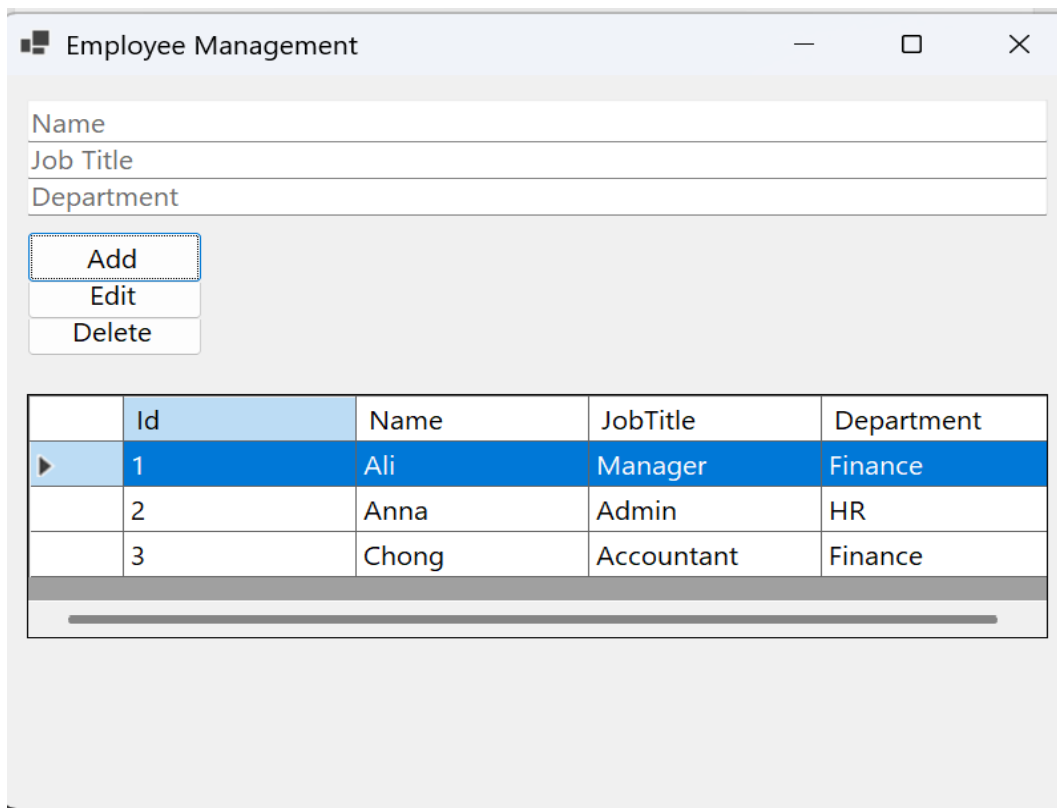
## Department & Position

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 namespace HRManagementSystem
11 {
12     public partial class Department_and_Position : Form
13     {
14         public Department_and_Position()
15         {
16             InitializeComponent();
17         }
18
19         private void btnAddDepartment_Click(object sender, EventArgs e)
20         {
21             // TODO: Add your code for adding departments
22             MessageBox.Show($"Department '{txtDepartmentName.Text}' added!");
23         }
24
25         private void btnAssignPosition_Click(object sender, EventArgs e)
26         {
27             // TODO: Add your code for assigning positions
28             MessageBox.Show($"Position '{txtPositionName.Text}' assigned!");
29         }
30
31         private void btnLinkEmployee_Click(object sender, EventArgs e)
32         {
33             // TODO: Add your code for linking employee
34             MessageBox.Show($"Employee ID '{txtEmployeeID.Text}' linked!");
35         }
36     }
37 }
```

Figure 14:Department\_and\_Position.cs

## 4.2 OUTPUT

### Employee Management

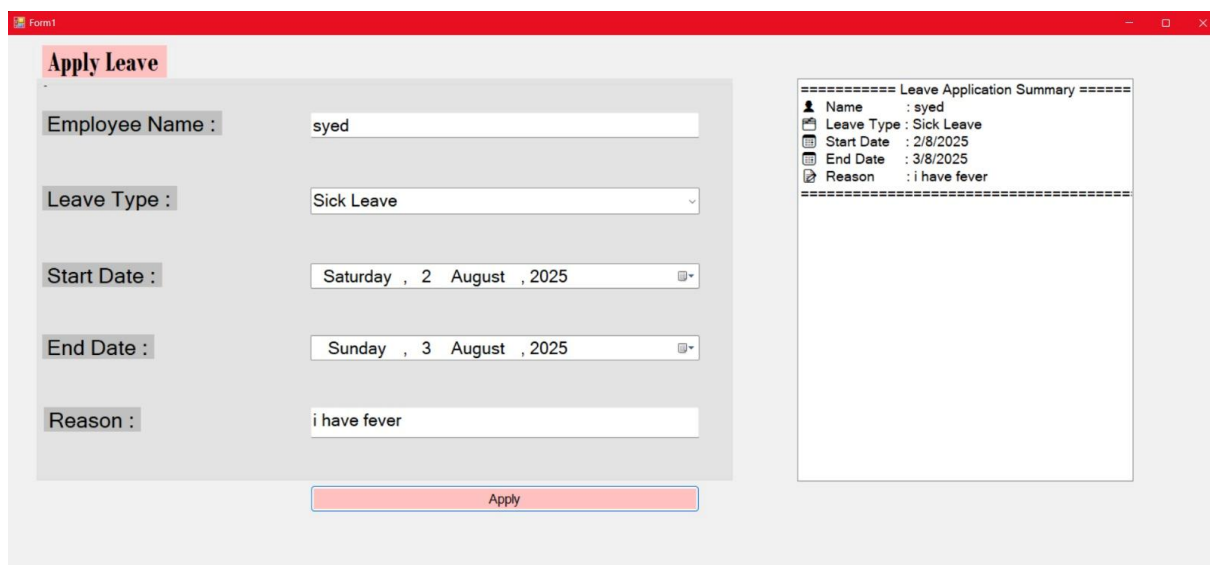


The screenshot shows a window titled "Employee Management" with a standard Windows title bar. Inside, there are three text input fields labeled "Name", "Job Title", and "Department". Below these fields are three buttons: "Add", "Edit", and "Delete". At the bottom, there is a table with five columns: "Id", "Name", "JobTitle", and "Department". The first row is highlighted in blue and contains the values "1", "Ali", "Manager", and "Finance". The second row contains "2", "Anna", "Admin", and "HR". The third row contains "3", "Chong", "Accountant", and "Finance".

	Id	Name	JobTitle	Department
▶	1	Ali	Manager	Finance
	2	Anna	Admin	HR
	3	Chong	Accountant	Finance

Figure 15: Employee Management Output

### Leave Application



The screenshot shows a window titled "Form1" with a red title bar. The main content area is titled "Apply Leave" in a red box. Below this title, there are five labeled input fields: "Employee Name" with the value "syed", "Leave Type" with a dropdown menu showing "Sick Leave", "Start Date" with a date picker showing "Saturday , 2 August , 2025", "End Date" with a date picker showing "Sunday , 3 August , 2025", and "Reason" with the text "i have fever". At the bottom center is a red "Apply" button. On the right side of the form, there is a text area titled "Leave Application Summary" containing the following text: "Name : syed", "Leave Type : Sick Leave", "Start Date : 2/8/2025", "End Date : 3/8/2025", and "Reason : i have fever".

Figure 16: Leave Application Output

## Payroll Module

**PAYROLL SYSTEM**

**Employee Details**

Full Name: AMINUDDIN RAZIF  
Position: CEO

**Deductions**

Tax Deduction (RM): 200  
Late Deduction (RM): 50

**Salary Earnings**

Basic Salary (RM): 5000  
Bonus (RM): 2000  
Allowance (RM): 1200  
Overtime Hours: 8 (RM20/hour)

**Generate Pay Slip**

**Pay Slip:**

===== PAY SLIP =====  
Name : AMINUDDIN RAZIF  
Position : CEO

**EARNINGS**

- Basic Salary : RM 5000.00
- Bonus : RM 2000.00
- Allowance : RM 1200.00
- Overtime (8 hrs @ RM20): RM 160.00
- Gross Salary : RM 8360.00

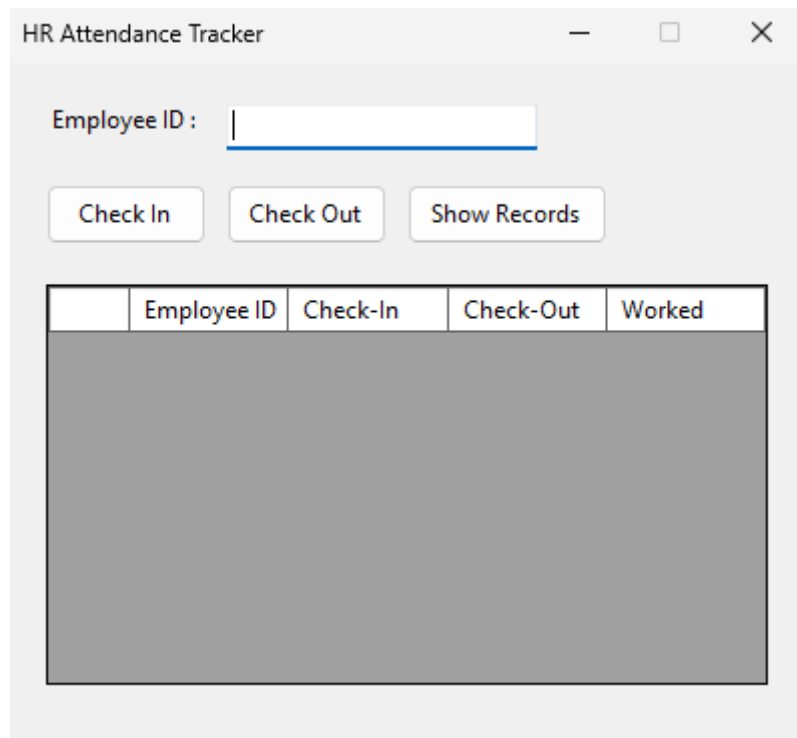
**DEDUCTIONS**

- EPF (11%) : RM 550.00
- SOCSO (0.5%) : RM 25.00
- Tax Deduction : RM 200.00
- Late Deduction : RM 50.00
- Total Deductions : RM 825.00

Net Salary : RM 7535.00  
=====

Figure 17: Payroll Module Output

## Attendance tracking

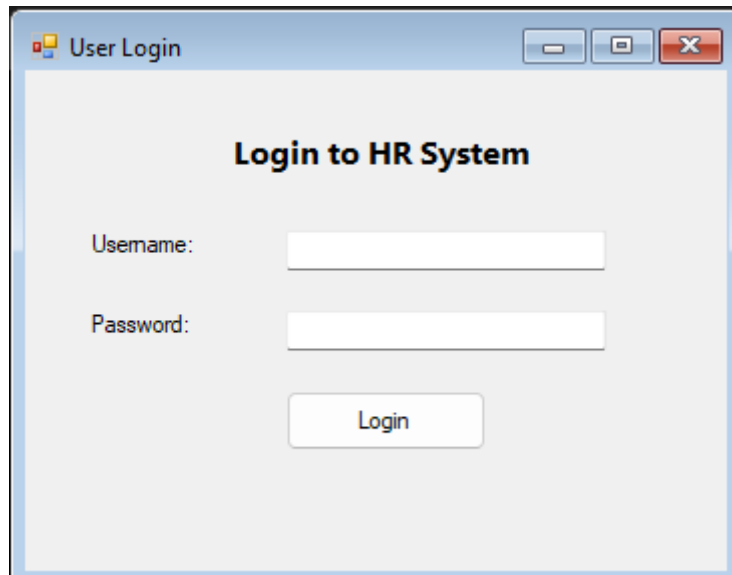


The HR Attendance Tracker window features a title bar with the text "HR Attendance Tracker" and standard window controls. Below the title bar is a text input field labeled "Employee ID :". Underneath the input field are three buttons: "Check In", "Check Out", and "Show Records". At the bottom of the window is a table with five columns: "Employee ID", "Check-In", "Check-Out", and "Worked". The table body is currently empty.

	Employee ID	Check-In	Check-Out	Worked
--	-------------	----------	-----------	--------

Figure 18: Attendance Tracking

## HR Login

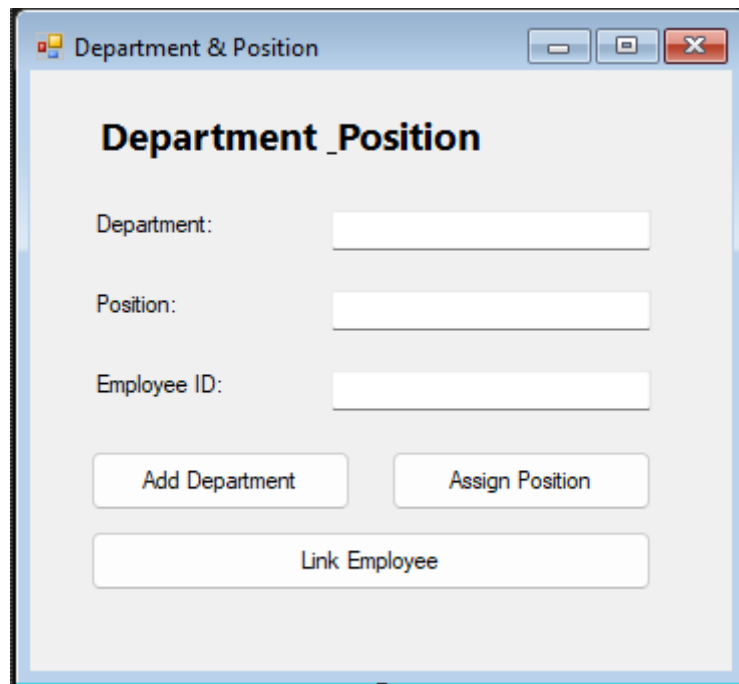


The User Login window has a title bar with the text "User Login" and standard window controls. The main content area is titled "Login to HR System". It contains two text input fields: "Username:" and "Password:". Below these fields is a "Login" button.

Figure 19: HR Login



## Department & Position



The image shows a software dialog box titled "Department & Position". It features a title bar with standard window controls (minimize, maximize, close). The main content area has a title "Department \_Position" in bold. Below the title are three input fields labeled "Department:", "Position:", and "Employee ID:". At the bottom, there are three buttons: "Add Department", "Assign Position", and "Link Employee".

**Department \_Position**

Department:

Position:

Employee ID:

*Figure 20: Department & Position*