Payroll Management System Report

Ali Mostafa 2022074

December 15, 2024

Contents

1 Introduction

The purpose of this report is to document a **Payroll Management System** implemented in C++. This system manages employee data, generates salary slips, and provides functionalities for adding, editing, and listing employees. The program demonstrates key concepts of object-oriented programming such as encapsulation, inheritance, and abstraction.

2 Features

The program includes the following features:

- Adding new employees.
- Displaying details of an employee using their unique code.
- Listing all employees in the system.
- Generating salary slips for employees based on their worked days and overtime hours.
- Editing details of existing employees.

3 Classes and Methods

3.1 Class: `Employee`

The `Employee` class encapsulates the details of an employee and provides methods to:

- Add employee details.
- Display employee details.
- Generate a salary slip.
- Edit employee details.

3.2 Class: `PayrollSystem`

The `PayrollSystem` class manages a collection of employees and provides functionalities for:

- Adding new employees.
- Displaying individual employee details.
- Listing all employees.
- Generating salary slips.
- Editing employee information.

4 Code Explanation

4.1 Key Concepts

- **Encapsulation:** Employee details and related operations are encapsulated within the `Employee` class.
- **Data Structures:** The program uses a `vector` to maintain a dynamic list of employees.
- **User Input Handling:** Functions use standard input/output to interact with the user.
- **Salary Calculation:** Salary slips are generated based on the employee's basic salary, house allowance, overtime pay, and loan deductions.

4.2 Flow of Execution

1. The `main()` function initializes the payroll system and displays a menu to the user.
2. Based on user input, relevant methods from the `PayrollSystem` class are invoked.
3. Operations such as adding, editing, and listing employees are handled seamlessly.

5 Complete Code

The complete source code for the Payroll Management System is provided below:

```
#include <iostream>
#include <string>
#include <vector>
#include <iomanip>
using namespace std;

struct Date {
    int day;
    int month;
    int year;
};

class Employee {
private:
    int code;
    string name;
    string address;
    string phone;
    string designation;
    char grade;
    double basicSalary;
    double houseAllowance;
    double loan;
    Date joiningDate;
```

```
25
26  public:
27      Employee() {
28          code = 0;
29          grade = 'A';
30          basicSalary = 0;
31          houseAllowance = 0;
32          loan = 0;
33      }
34
35      void addEmployee() {
36          cout << "\nEnter Employee Code: ";
37          cin >> code;
38          cin.ignore();
39          cout << "Enter Name: ";
40          getline(cin, name);
41          cout << "Enter Address: ";
42          getline(cin, address);
43          cout << "Enter Phone: ";
44          getline(cin, phone);
45          cout << "Enter Designation: ";
46          getline(cin, designation);
47          cout << "Enter Joining Date (DD MM YYYY): ";
48          cin >> joiningDate.day >> joiningDate.month >>
                joiningDate.year;
49          cout << "Enter Grade (A/B/C/D/E): ";
50          cin >> grade;
51          cout << "Enter Basic Salary: ";
52          cin >> basicSalary;
53          cout << "Enter House Allowance: ";
54          cin >> houseAllowance;
55          cout << "Enter Loan (if any): ";
56          cin >> loan;
57      }
58
59      void displayEmployee() {
60          cout << "\nEmployee Details:" << endl;
61          cout << "Code: " << code << endl;
62          cout << "Name: " << name << endl;
63          cout << "Address: " << address << endl;
64          cout << "Phone: " << phone << endl;
65          cout << "Designation: " << designation << endl;
66          cout << "Joining Date: " << joiningDate.day << "/"
67               << joiningDate.month << "/" << joiningDate.year <<
                    endl;
68          cout << "Grade: " << grade << endl;
69          cout << "Basic Salary: $" << basicSalary << endl;
70          cout << "House Allowance: $" << houseAllowance << endl;
71          cout << "Loan: $" << loan << endl;
72      }
73
```

```
74      void generateSalarySlip() {
75          int daysWorked, overtimeHours;
76          cout << "\nEnter Days Worked: ";
77          cin >> daysWorked;
78          cout << "Enter Overtime Hours: ";
79          cin >> overtimeHours;
80
81          double overtimeRate = 100.0;
82          double overtimePay = overtimeHours * overtimeRate;
83          double grossSalary = basicSalary + houseAllowance +
                overtimePay;
84          double deductions = loan;
85          double netSalary = grossSalary - deductions;
86
87          cout << "\n-------- SALARY SLIP --------" << endl;
88          cout << "Employee Code: " << code << endl;
89          cout << "Name: " << name << endl;
90          cout << "Designation: " << designation << endl;
91          cout << "Days Worked: " << daysWorked << endl;
92          cout << "Overtime Hours: " << overtimeHours << endl;
93          cout << "\nEarnings:" << endl;
94          cout << "Basic Salary: $" << basicSalary << endl;
95          cout << "House Allowance: $" << houseAllowance << endl;
96          cout << "Overtime Pay: $" << overtimePay << endl;
97          cout << "Gross Salary: $" << grossSalary << endl;
98          cout << "\nDeductions:" << endl;
99          cout << "Loan: $" << loan << endl;
100         cout << "\nNet Salary: $" << netSalary << endl;
101         cout << "----------------------------" << endl;
102     }
103
104     int getCode() { return code; }
105     void editEmployee() {
106         cin.ignore();
107         cout << "\nEnter New Name: ";
108         getline(cin, name);
109         cout << "Enter New Address: ";
110         getline(cin, address);
111         cout << "Enter New Phone: ";
112         getline(cin, phone);
113         cout << "Enter New Designation: ";
114         getline(cin, designation);
115         cout << "Enter New Joining Date (DD MM YYYY): ";
116         cin >> joiningDate.day >> joiningDate.month >>
                joiningDate.year;
117         cout << "Enter New Grade (A/B/C/D/E): ";
118         cin >> grade;
119         cout << "Enter New Basic Salary: ";
120         cin >> basicSalary;
121         cout << "Enter New House Allowance: ";
122         cin >> houseAllowance;
```

```
123         cout << "Enter New Loan Amount: ";
124         cin >> loan;
125     }
126 
127     string getName() { return name; }
128 };
129 
130 class PayrollSystem {
131 private:
132     vector<Employee> employees;
133 
134 public:
135     void addNewEmployee() {
136         Employee emp;
137         emp.addEmployee();
138         employees.push_back(emp);
139         cout << "\nEmployee Added Successfully!" << endl;
140     }
141 
142     void displayEmployee() {
143         int code;
144         cout << "\nEnter Employee Code: ";
145         cin >> code;
146 
147         for (Employee& emp : employees) {
148             if (emp.getCode() == code) {
149                 emp.displayEmployee();
150                 return;
151             }
152         }
153         cout << "\nEmployee not found!" << endl;
154     }
155 
156     void listEmployees() {
157         if (employees.empty()) {
158             cout << "\nNo employees in the system!" << endl;
159             return;
160         }
161 
162         cout << "\nList of All Employees:" << endl;
163         cout << setw(10) << "Code" << setw(20) << "Name" << endl;
164         cout << "--------------------------------" << endl;
165 
166         for (Employee& emp : employees) {
167             cout << setw(10) << emp.getCode() << setw(20) <<
                    emp.getName() << endl;
168         }
169     }
170 
171     void generateSalarySlip() {
172         int code;
```

```
        cout << "\nEnter Employee Code: ";
        cin >> code;

        for (Employee& emp : employees) {
            if (emp.getCode() == code) {
                emp.generateSalarySlip();
                return;
            }
        }
        cout << "\nEmployee not found!" << endl;
    }

    void editEmployeeDetails() {
        int code;
        cout << "\nEnter Employee Code to Edit: ";
        cin >> code;

        for (Employee& emp : employees) {
            if (emp.getCode() == code) {
                emp.editEmployee();
                cout << "\nEmployee Details Updated
                   Successfully!" << endl;
                return;
            }
        }
        cout << "\nEmployee not found!" << endl;
    }
};

int main() {
    PayrollSystem payroll;
    int choice;

    do {
        cout << "\n=== PAYROLL MANAGEMENT SYSTEM ===" << endl;
        cout << "1: NEW EMPLOYEE" << endl;
        cout << "2: DISPLAY EMPLOYEE" << endl;
        cout << "3: LIST OF EMPLOYEES" << endl;
        cout << "4: SALARY SLIP" << endl;
        cout << "5: EDIT" << endl;
        cout << "0: QUIT" << endl;
        cout << "ENTER YOUR CHOICE: ";
        cin >> choice;

        switch (choice) {
            case 1:
                payroll.addNewEmployee();
                break;
            case 2:
                payroll.displayEmployee();
                break;
```

```
            case 3:
                payroll.listEmployees();
                break;
            case 4:
                payroll.generateSalarySlip();
                break;
            case 5:
                payroll.editEmployeeDetails();
                break;
            case 0:
                cout << "\nThank you for using the system!" <<
                    endl;
                break;
            default:
                cout << "\nInvalid choice! Please try again." <<
                    endl;
        }
    } while (choice != 0);

    return 0;
}
```

6 Conclusion

The Payroll Management System is a comprehensive example of object-oriented programming in C++. It showcases real-world applications and is scalable for further enhancements such as database integration or GUI support.