PROJECT REPORT

Name : Ashwini M. Jadhav
Date : 19/01/2026

Employee Attendance & Salary Calculator

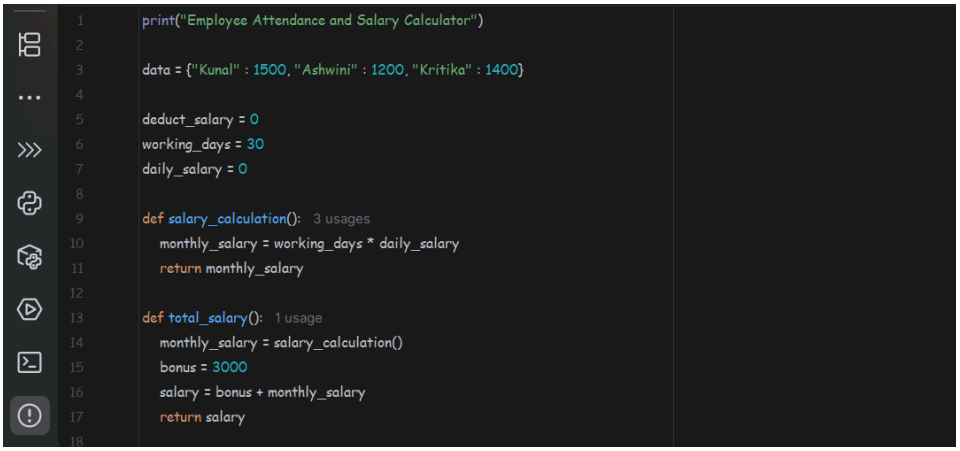
---

1. Project Overview

This program is designed to calculate the monthly salary of an employee based on:

- Total working days in a month
- Days actually worked
- Daily wage
- Bonus eligibility
- Salary deductions for absenteeism

It helps employers automate salary calculation without manual math.



```python
print("Employee Attendance and Salary Calculator")

data = {"Kunal" : 1500, "Ashwini" : 1200, "Kritika" : 1400}

deduct_salary = 0
working_days = 30
daily_salary = 0

def salary_calculation():  3 usages
    monthly_salary = working_days * daily_salary
    return monthly_salary

def total_salary():  1 usage
    monthly_salary = salary_calculation()
    bonus = 3000
    salary = bonus + monthly_salary
    return salary
```

2. Data Used

The program contains a dictionary of employees and their daily salary:

```
data = {"Kunal": 1500, "Ashwini": 1200, "Kritika": 1400}
```

- The **key** represents the employee's name
- The **value** represents their per-day salary

This allows easy reference to different employee salaries.

---

3. Defined Variables

```
working_days = 30
daily_salary = 0
deduct_salary = 0
```

- **working_days** is fixed to 30 per month
- **daily_salary** starts at 0 and is updated by the user
- **deduct_salary** stores deduction values later

These variables are used across functions.

---

4. Function Breakdowns

A. salary_calculation()

```
def salary_calculation():
    monthly_salary = working_days * daily_salary
    return monthly_salary
```

Purpose:
To calculate salary based on total working days and daily wages.

How it works:
Monthly salary = daily salary × 30 working days
It returns the calculated total base salary.

B. total_salary()

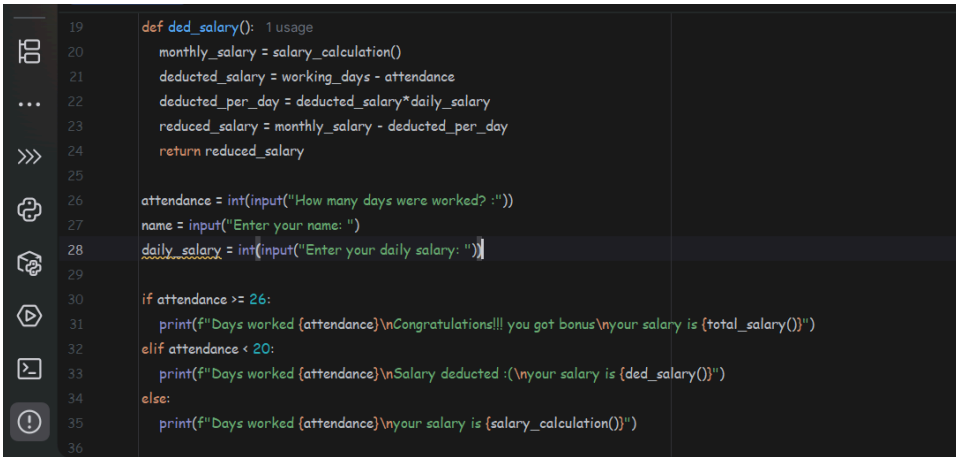
```
def total_salary():
    monthly_salary = salary_calculation()
    bonus = 3000
    salary = bonus + monthly_salary
    return salary
```

Purpose:
To reward employees who worked 26 days or more.

How it works:

1. Calls **`salary_calculation()`** to compute base salary
2. Adds a fixed bonus of ₹3000
3. Returns the new salary (base + bonus)

Used when the employee has **excellent attendance**.



```
def ded_salary():  1 usage
    monthly_salary = salary_calculation()
    deducted_salary = working_days - attendance
    deducted_per_day = deducted_salary*daily_salary
    reduced_salary = monthly_salary - deducted_per_day
    return reduced_salary

attendance = int(input("How many days were worked? :"))
name = input("Enter your name: ")
daily_salary = int(input("Enter your daily salary: "))

if attendance >= 26:
    print(f"Days worked {attendance}\nCongratulations!!! you got bonus\nyour salary is {total_salary()}")
elif attendance < 20:
    print(f"Days worked {attendance}\nSalary deducted :(\nyour salary is {ded_salary()}")
else:
    print(f"Days worked {attendance}\nyour salary is {salary_calculation()}")
```

C. ded_salary()

```
def ded_salary():
    monthly_salary = salary_calculation()
    deducted_salary = working_days - attendance
    deducted_per_day = deducted_salary * daily_salary
    reduced_salary = monthly_salary - deducted_per_day
    return reduced_salary
```

Purpose:
To calculate reduced salary when attendance is poor.

Step–by–step flow:

1. Calculate full monthly salary
2. Find how many days the employee was absent
3. Multiply absent days × daily pay
4. Subtract deduction from monthly salary
5. Return the final reduced amount

Used when employee works **less than 20 days**.

---

5. User Inputs

```
attendance = int(input("How many days were worked? :"))
name = input("Enter your name: ")
daily_salary = int(input("Enter your daily salary: "))
```

The user enters:

- Their name
- Number of days worked
- Salary per day

This customizes the salary calculation for each employee.

---

6. Decision Conditions

```
if attendance >= 26:

    print(f"Days worked {attendance}\nCongratulations!!! you got
bonus\nyour salary is {total_salary()}")
```

```
elif attendance < 20:

    print(f"Days worked {attendance}\nSalary deducted :(\nyour salary is {ded_salary()}")

else:

    print(f"Days worked {attendance}\nyour salary is {salary_calculation()}")
```

Logic Applied

Attendance Range	result
26–30 days	Full salary + bonus
Under 20 days	Salary deduction
20–25 days	Normal salary, no bonus

Each condition calls the matching function:

- **`total_salary()`** for bonus
- **`ded_salary()`** for salary cut
- **`salary_calculation()`** for normal case

---

7. Final Output

Based on attendance, the program prints:

- Days worked
- Message showing bonus or deduction
- Final payable salary amount

This gives quick payroll results for the employee.

---

8. Scope for Improvements

Future upgrades can include:

- Auto-select daily salary from dictionary
- Store records for multiple employees
- Export salary slips to Excel, PDF, or CSV
- Add tax and PF deduction
- Convert program into a GUI application

---

9. Conclusion

This project demonstrates:

- Use of variables and user input
- Dictionaries to store employee data
- Modular programming with functions
- Conditional logic based on attendance
- Practical applications of Python in real-life payroll systems

It successfully automates salary calculations based on attendance, bonus eligibility, and absent-day deductions.