Week 5: Thursday InClass Assignment

Ashok Nath

Rina Katuwal

Prashant Ojha

Meena Ale Magar

Bikram Khatiwada

Kaushal Kishore Rai

Master Information Technology, Atlantis University

MIT 563: Programming and Applications Development

Professor: Alex Lima

Sep 26, 2024

1 Customer Service module and integration with HR module –

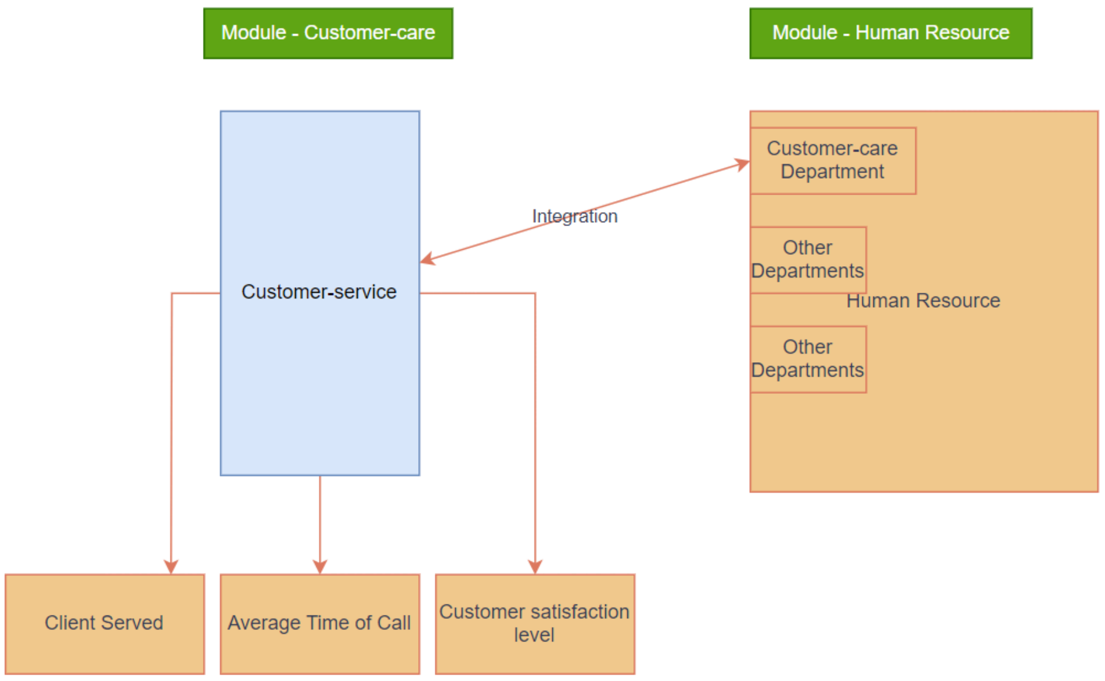

Fig: Customer Service module with HR Integration

2 Document: Integration of Customer Service Module and Human Resources in

2.1 Overview

This document describes the integration of the Human Resources (HR) module with the Customer Service (CS) module. The integrated system tracks employee management and specific customer service metrics, demonstrating how the two systems collaborate to provide a complete overview of an organization's HR and customer service operations. The program ensures that HR employees belonging to the customer service department can be tracked for their customer-facing performance metrics, while other employees are managed through general HR operations.

2.2 System Architecture

1. **HR Module**:
   - Handles employee details like ID, name, role, hours worked, overtime, unavailable hours, training hours, promotion status, and vacation status.
   - Provides functionality to log these details for all employees across the organization.
2. **Customer Service Module**:
   - Extends HR functionality specifically for customer service employees.
   - Tracks clients served, average service time, customer satisfaction, and the specific service area.
3. **Integration**:
   - The integration ensures that employees added to the HR module who belong to the customer service department are also tracked in the CS module.
   - HR manages employee attributes such as hours worked, promotions, etc., while CS tracks customer service performance metrics.

2.3 Integration Details

- The CustomerServiceManager class interacts with employees managed by the HRManager class.
- When an employee is hired or added to the HR system, they can be registered in the CustomerServiceManager if they belong to the customer service department.
- Variables like employee ID, name, and role are common between both modules. HR focuses on general employee management (hours worked, promotions), and the CS module focuses on client interaction metrics (clients served, service quality, etc.).

---

3 Python Code Implementation for Customer Service:

```
# -*- coding: utf-8 -*-
"""CustomerService
This module manages employees in the customer-service department
and tracks customer service metrics such as service quality,
number of clients served, average service time, and customer
satisfaction.
"""

class CustomerServiceEmployee:
    def __init__(self, emp_id, name, role):
        self.emp_id = emp_id
        self.name = name
        self.role = role
        self.clients_served = 0
        self.total_service_time = 0
        self.customer_satisfaction = []
        self.service_area = ""

    def serve_client(self, service_time, satisfaction_level):
        self.clients_served += 1
        self.total_service_time += service_time
        self.customer_satisfaction.append(satisfaction_level)


    def set_service_area(self, service_area):
        self.service_area = service_area


    def calculate_average_service_time(self):
        return self.total_service_time / self.clients_served if
self.clients_served > 0 else 0


    def calculate_average_satisfaction(self):
        return sum(self.customer_satisfaction) /
len(self.customer_satisfaction) if self.customer_satisfaction else
0


    def display_customer_service_info(self):
        print(f"Customer Service Employee ID: {self.emp_id}")
        print(f"Name: {self.name}")
        print(f"Role: {self.role}")
        print(f"Clients Served: {self.clients_served}")
        print(f"Average Service Time: 
{self.calculate_average_service_time()} minutes")
        print(f"Average Customer Satisfaction Level: 
{self.calculate_average_satisfaction()}")
        print(f"Service Area: {self.service_area}")
        print("\n")


class CustomerServiceManager:
    def __init__(self):
        self.cs_employees = {}


    def add_cs_employee(self, emp_id, name, role):
        self.cs_employees[emp_id] = 
CustomerServiceEmployee(emp_id, name, role)


    def log_service(self, emp_id, service_time, 
satisfaction_level):
        if emp_id in self.cs_employees:
            self.cs_employees[emp_id].serve_client(service_time, 
satisfaction_level)


    def set_service_area(self, emp_id, service_area):
        if emp_id in self.cs_employees:

self.cs_employees[emp_id].set_service_area(service_area)


    def display_cs_employee_info(self, emp_id):
        if emp_id in self.cs_employees:

self.cs_employees[emp_id].display_customer_service_info()


# Example usage
```

```
if __name__ == "__main__":
    customer_service_manager = CustomerServiceManager()


    # Add customer service employees
    customer_service_manager.add_cs_employee(1, "Chris Brown", 
"Customer Service Rep")
    customer_service_manager.add_cs_employee(2, "Emma Wilson", 
"Customer Support Specialist")
    customer_service_manager.add_cs_employee(3, "Derek Robinson", 
"Customer Support Specialist")
    customer_service_manager.add_cs_employee(4, "Rachel Raynolds", 
"Customer Support Specialist")


    # Log service information
    customer_service_manager.log_service(1, 15, 4.5)  # 15 minutes 
call with 4.5 satisfaction
    customer_service_manager.log_service(1, 10, 4.0)  # 10 minutes 
call with 4.0 satisfaction
    customer_service_manager.log_service(2, 20, 5.0)  # 20 minutes 
call with 5.0 satisfaction
    customer_service_manager.log_service(3, 25, 4.8)  # 25 minutes 
call with 4.8 satisfaction
    customer_service_manager.log_service(4, 30, 4.7)  # 30 minutes 
call with 4.7 satisfaction
    customer_service_manager.log_service(4, 18, 4.3)  # 18 minutes 
call with 4.3 satisfaction


    # Set service area
    customer_service_manager.set_service_area(1, "Product 
Support")
    customer_service_manager.set_service_area(2, "Technical 
Assistance")
    customer_service_manager.set_service_area(3, "Technical 
Support")
    customer_service_manager.set_service_area(4, "Customer 
Relations")

    # Display customer service info for employees
    customer_service_manager.display_cs_employee_info(1)
    customer_service_manager.display_cs_employee_info(2)
    customer_service_manager.display_cs_employee_info(3)
    customer_service_manager.display_cs_employee_info(4)
```

Output:

```
Customer Service Employee ID: 1
Name: Chris Brown
Role: Customer Service Rep
Clients Served: 2
Average Service Time: 12.5 minutes
Average Customer Satisfaction Level: 4.25
Service Area: Product Support


Customer Service Employee ID: 2
Name: Emma Wilson
Role: Customer Support Specialist
Clients Served: 1
Average Service Time: 20.0 minutes
Average Customer Satisfaction Level: 5.0
Service Area: Technical Assistance


Customer Service Employee ID: 3
Name: Derek Robinson
Role: Customer Support Specialist
Clients Served: 1
Name: Derek Robinson
Role: Customer Support Specialist
Name: Derek Robinson
Name: Derek Robinson
Role: Customer Support Specialist
```

```
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Service Area: Technical Support


Customer Service Employee ID: 4
Name: Rachel Raynolds
Role: Customer Support Specialist
Clients Served: 2
Average Service Time: 24.0 minutes
Name: Derek Robinson
Role: Customer Support Specialist
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Service Area: Technical Support


Customer Service Employee ID: 4
Name: Rachel Raynolds
Role: Customer Support Specialist
Name: Derek Robinson
Role: Customer Support Specialist
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Service Area: Technical Support


Customer Service Employee ID: 4
Name: Derek Robinson
Role: Customer Support Specialist
```

```
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Service Area: Technical Support


Name: Derek Robinson
Role: Customer Support Specialist
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Name: Derek Robinson
Role: Customer Support Specialist
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Service Area: Technical Support
Name: Derek Robinson
Role: Customer Support Specialist
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Service Area: Technical Support

Name: Derek Robinson
Role: Customer Support Specialist
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Name: Derek Robinson
Role: Customer Support Specialist
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Clients Served: 1
Average Service Time: 25.0 minutes
Average Customer Satisfaction Level: 4.8
Average Customer Satisfaction Level: 4.8
Service Area: Technical Support


Customer Service Employee ID: 4
Name: Rachel Raynolds
Role: Customer Support Specialist
Clients Served: 2
Average Service Time: 24.0 minutes
Average Customer Satisfaction Level: 4.5
Service Area: Customer Relations
```

4 Employee Information from HR:

```python
class Employee:
    def __init__(self, emp_id, name, role):
        self.emp_id = emp_id
        self.name = name
        self.role = role
        self.hours_worked = 0
        self.overtime_hours = 0
        self.unavailable_hours = 0
        self.training_hours = 0
        self.promoted = False
        self.on_vacation = False

    def work(self, hours):
        self.hours_worked += hours

    def log_overtime(self, hours):
        self.overtime_hours += hours
    def mark_unavailable(self, hours):
        self.unavailable_hours += hours

    def log_training(self, hours):
        self.training_hours += hours

    def promote(self):
        self.promoted = True

    def start_vacation(self):
        self.on_vacation = True

    def end_vacation(self):
        self.on_vacation = False

class HRManager:
    def __init__(self):
        self.employees = {}
        self.to_be_hired = []

    def add_employee(self, emp_id, name, role):
        self.employees[emp_id] = Employee(emp_id, name, role)

    def remove_employee(self, emp_id):
        if emp_id in self.employees:
            del self.employees[emp_id]

    def hire_employee(self, name, role):
        self.to_be_hired.append((name, role))

    def start_vacation(self, emp_id):
        if emp_id in self.employees:
            self.employees[emp_id].start_vacation()

    def end_vacation(self, emp_id):
        if emp_id in self.employees:
            self.employees[emp_id].end_vacation()

    def log_hours(self, emp_id, hours):
        if emp_id in self.employees:
            self.employees[emp_id].work(hours)
    def log_overtime(self, emp_id, hours):
        if emp_id in self.employees:
            self.employees[emp_id].log_overtime(hours)

    def log_unavailable_hours(self, emp_id, hours):
        if emp_id in self.employees:
            self.employees[emp_id].mark_unavailable(hours)

    def log_training_hours(self, emp_id, hours):
        if emp_id in self.employees:
            self.employees[emp_id].log_training(hours)

    def promote_employee(self, emp_id):
        if emp_id in self.employees:
            self.employees[emp_id].promote()

    def display_employee_info(self, emp_id):
        if emp_id in self.employees:
            emp = self.employees[emp_id]
            print(f"Employee ID: {emp.emp_id}")
            print(f"Name: {emp.name}")
            print(f"Role: {emp.role}")
            print(f"Hours Worked: {emp.hours_worked}")
            print(f"Overtime Hours: {emp.overtime_hours}")
            print(f"Unavailable Hours: {emp.unavailable_hours}")
            print(f"Training Hours: {emp.training_hours}")
            print(f"On Vacation: {'Yes' if emp.on_vacation else
'No'}")
            print(f"Promoted: {'Yes' if emp.promoted else 'No'}")
            print("\n")

    def display_to_be_hired(self):
        print("Employees to be Hired:")
        for name, role in self.to_be_hired:
            print(f"Name: {name}, Role: {role}")

# Example usage
if __name__ == "__main__":
    hr_manager = HRManager()

    # Add some employees
    hr_manager.add_employee(1, "John Doe", "Developer")
    hr_manager.add_employee(2, "Jane Smith", "Designer")

    # Log hours worked
    hr_manager.log_hours(1, 40)
    hr_manager.log_overtime(1, 5)
    hr_manager.log_training_hours(2, 8)

    # Start and end vacation
    hr_manager.start_vacation(2)
    hr_manager.end_vacation(2)

    # Promote an employee
    hr_manager.promote_employee(1)

    # Display employee info
    hr_manager.display_employee_info(1)
    hr_manager.display_employee_info(2)

    # Hire new employees
    hr_manager.hire_employee("Alice Johnson", "Product Manager")
    hr_manager.hire_employee("Bob Lee", "QA Engineer")
    hr_manager.display_to_be_hired()
```

Output:

```
Employee ID: 1

Name: John Doe

Role: Developer

Hours Worked: 40

Overtime Hours: 5

Unavailable Hours: 0

Training Hours: 0

On Vacation: No

Promoted: Yes
```

```
Employee ID: 2

Name: Jane Smith

Role: Designer

Hours Worked: 0

Overtime Hours: 0

Unavailable Hours: 0

Training Hours: 8

On Vacation: No

Promoted: No


Employees to be Hired:

Name: Alice Johnson, Role: Product Manager

Name: Bob Lee, Role: QA Engineer
```

5 Integrated Program for Customer Service and HR:

```python
# -*- coding: utf-8 -*-
"""CustomerService

This module manages employees in the customer-service department
and tracks customer service metrics such as service quality,
number of clients served, average service time, and customer
satisfaction.
"""
class Employee:
    def __init__(self, emp_id, name, role):
        self.emp_id = emp_id
        self.name = name
        self.role = role
        self.hours_worked = 0
        self.overtime_hours = 0
        self.unavailable_hours = 0
        self.training_hours = 0
        self.promoted = False
        self.on_vacation = False

    def work(self, hours):
        self.hours_worked += hours

    def log_overtime(self, hours):
        self.overtime_hours += hours

    def mark_unavailable(self, hours):
        self.unavailable_hours += hours

    def log_training(self, hours):
        self.training_hours += hours

    def promote(self):
        self.promoted = True

    def start_vacation(self):
        self.on_vacation = True

    def end_vacation(self):
        self.on_vacation = False

class HRManager:
    def __init__(self):
        self.employees = {}
        self.to_be_hired = []

    def add_employee(self, emp_id, name, role):
        self.employees[emp_id] = Employee(emp_id, name, role)

    def remove_employee(self, emp_id):
        if emp_id in self.employees:
            del self.employees[emp_id]
```

```python
def hire_employee(self, name, role):
    self.to_be_hired.append((name, role))

def start_vacation(self, emp_id):
    if emp_id in self.employees:
        self.employees[emp_id].start_vacation()

def end_vacation(self, emp_id):
    if emp_id in self.employees:
        self.employees[emp_id].end_vacation()

def log_hours(self, emp_id, hours):
    if emp_id in self.employees:
        self.employees[emp_id].work(hours)

def log_overtime(self, emp_id, hours):
    if emp_id in self.employees:
        self.employees[emp_id].log_overtime(hours)

def log_unavailable_hours(self, emp_id, hours):
    if emp_id in self.employees:
        self.employees[emp_id].mark_unavailable(hours)

def log_training_hours(self, emp_id, hours):
    if emp_id in self.employees:
        self.employees[emp_id].log_training(hours)

def promote_employee(self, emp_id):
    if emp_id in self.employees:
        self.employees[emp_id].promote()

def display_employee_info(self, emp_id):
    if emp_id in self.employees:
        emp = self.employees[emp_id]
        if isinstance(emp, CustomerServiceEmployee):
            emp.display_customer_service_info()
        else:
            print(f"Employee ID: {emp.emp_id}")
            print(f"Name: {emp.name}")
            print(f"Role: {emp.role}")
            print(f"Hours Worked: {emp.hours_worked}")
            print(f"Overtime Hours: {emp.overtime_hours}")
```

```
                print(f"Unavailable Hours:
{emp.unavailable_hours}")
                print(f"Training Hours: {emp.training_hours}")
                print(f"On Vacation: {'Yes' if emp.on_vacation
else 'No'}")
                print(f"Promoted: {'Yes' if emp.promoted else
'No'}")
                print("\n")

    def display_to_be_hired(self):
        print("Employees to be Hired:")
        for name, role in self.to_be_hired:
            print(f"Name: {name}, Role: {role}")

class CustomerServiceEmployee:
    def __init__(self, emp_id, name, role):
        self.emp_id = emp_id
        self.name = name
        self.role = role
        self.clients_served = 0
        self.total_service_time = 0
        self.customer_satisfaction = []
        self.service_area = ""

    def serve_client(self, service_time, satisfaction_level):
        self.clients_served += 1
        self.total_service_time += service_time
        self.customer_satisfaction.append(satisfaction_level)

    def set_service_area(self, service_area):
        self.service_area = service_area

    def calculate_average_service_time(self):
        return self.total_service_time / self.clients_served if
self.clients_served > 0 else 0

    def calculate_average_satisfaction(self):
        return sum(self.customer_satisfaction) /
len(self.customer_satisfaction) if self.customer_satisfaction else
0

    def display_customer_service_info(self):
        print("\n")
        print(f"Customer Service Employee ID: {self.emp_id}")
        print(f"Name: {self.name}")
        print(f"Role: {self.role}")
        print(f"Clients Served: {self.clients_served}")
        print(f"Average Service Time:
{self.calculate_average_service_time()} minutes")
        print(f"Average Customer Satisfaction Level:
{self.calculate_average_satisfaction()}")
        print(f"Service Area: {self.service_area}")

class CustomerServiceManager:
    def __init__(self):
        self.cs_employees = {}

    def add_cs_employee(self, emp_id, name, role):
        self.cs_employees[emp_id] =
CustomerServiceEmployee(emp_id, name, role)

    def log_service(self, emp_id, service_time,
satisfaction_level):
        if emp_id in self.cs_employees:
            self.cs_employees[emp_id].serve_client(service_time,
satisfaction_level)

    def set_service_area(self, emp_id, service_area):
        if emp_id in self.cs_employees:

self.cs_employees[emp_id].set_service_area(service_area)

    def display_cs_employee_info(self, emp_id):
        if emp_id in self.cs_employees:

self.cs_employees[emp_id].display_customer_service_info()

# Example usage
if __name__ == "__main__":
    customer_service_manager = CustomerServiceManager()

    hr_manager = HRManager()

    # Add some employees
    print("Employee Information")
    hr_manager.add_employee(1, "John Doe", "Developer")
    hr_manager.add_employee(2, "Jane Smith", "Designer")

    # Log hours worked
    hr_manager.log_hours(1, 40)
    hr_manager.log_overtime(1, 5)
    hr_manager.log_training_hours(2, 8)

    # Display employee info
    hr_manager.display_employee_info(1)
    hr_manager.display_employee_info(2)
    hr_manager.display_employee_info(3)



    # Log hours worked
    hr_manager.log_hours(1, 40)
    hr_manager.log_overtime(1, 5)
    hr_manager.log_training_hours(2, 8)


    # Start and end vacation
    hr_manager.start_vacation(2)
    hr_manager.end_vacation(2)

    # Promote an employee
    hr_manager.promote_employee(1)


    # Hire new employees
    hr_manager.hire_employee("Alice Johnson", "Product Manager")
    hr_manager.hire_employee("Bob Lee", "QA Engineer")
    hr_manager.display_to_be_hired()


    # Add customer service employees
    customer_service_manager.add_cs_employee(1, "Chris Brown", 
"Customer Service Rep")
    customer_service_manager.add_cs_employee(2, "Emma Wilson", 
"Customer Support Specialist")
    customer_service_manager.add_cs_employee(3, "Derek Robinson", 
"Customer Support Specialist")
    customer_service_manager.add_cs_employee(4, "Rachel Raynolds",
"Customer Support Specialist")

    # Log service information
    customer_service_manager.log_service(1, 15, 4.5)  # 15 minutes
call with 4.5 satisfaction
    customer_service_manager.log_service(1, 10, 4.0)  # 10 minutes
call with 4.0 satisfaction
    customer_service_manager.log_service(2, 20, 5.0)  # 20 minutes
call with 5.0 satisfaction
    customer_service_manager.log_service(3, 25, 4.8)  # 25 minutes
call with 4.8 satisfaction
    customer_service_manager.log_service(4, 30, 4.7)  # 30 minutes
call with 4.7 satisfaction
    customer_service_manager.log_service(4, 18, 4.3)  # 18 minutes
call with 4.3 satisfaction

    # Set service area
    customer_service_manager.set_service_area(1, "Product
Support")
    customer_service_manager.set_service_area(2, "Technical
Assistance")
    customer_service_manager.set_service_area(3, "Technical
Support")
    customer_service_manager.set_service_area(4, "Customer
Relations")

    # Display customer service info for employees
    customer_service_manager.display_cs_employee_info(1)
    customer_service_manager.display_cs_employee_info(2)
    customer_service_manager.display_cs_employee_info(3)
    customer_service_manager.display_cs_employee_info(4)
```

Output:

```
Employee Information

Employee ID: 1

Name: John Doe
```

```
Role: Developer
Hours Worked: 50
Overtime Hours: 25
Unavailable Hours: 30
Training Hours: 40
On Vacation: No
Promoted: No

Employee ID: 2
Name: Jane Smith
Role: Designer
Hours Worked: 10
Overtime Hours: 20
Unavailable Hours: 30
Training Hours: 48
On Vacation: No
Promoted: No

Employees to be Hired:
Name: Alice Johnson, Role: Product Manager
Name: Bob Lee, Role: QA Engineer

Customer Service Employee ID: 1
Name: Chris Brown
```

Role: Customer Service Rep

Clients Served: 2

Average Service Time: 12.5 minutes

Average Customer Satisfaction Level: 4.25

Service Area: Product Support

Customer Service Employee ID: 2

Name: Emma Wilson

Role: Customer Support Specialist

Clients Served: 1

Average Service Time: 20.0 minutes

Average Customer Satisfaction Level: 5.0

Service Area: Technical Assistance

Customer Service Employee ID: 3

Name: Derek Robinson

Role: Customer Support Specialist

Clients Served: 1

Average Service Time: 25.0 minutes

Average Customer Satisfaction Level: 4.8

Service Area: Technical Support

Customer Service Employee ID: 4

Name: Rachel Raynolds

```
Role: Customer Support Specialist

Clients Served: 2

Average Service Time: 24.0 minutes

Average Customer Satisfaction Level: 4.5

Service Area: Customer Relations
```

6 Conclusion:

This integrated program provides a complete solution to manage both general HR and customer service metrics in one system, producing concrete results about employee performance and customer service quality. This allows businesses to maintain efficient employee tracking and customer service optimization.