# Object Modelling

Level 1 Problems

1. Vehicle Info System

**Class Diagram:**   **Use Case Diagram:**

Vehicle
-brand : string   -model : string
+**init**(brand, model)   +display()
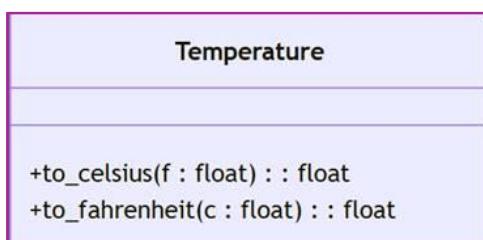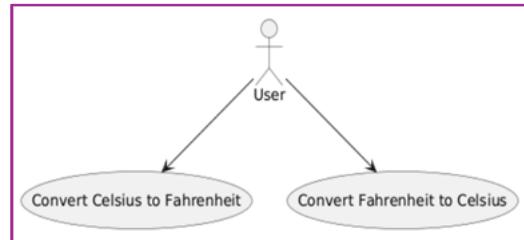
2. Temperature Converter Class

Class Diagram: **Use Case Diagram:**

Temperature
+to_celsius(f : float) : : float   +to_fahrenheit(c : float) : : float
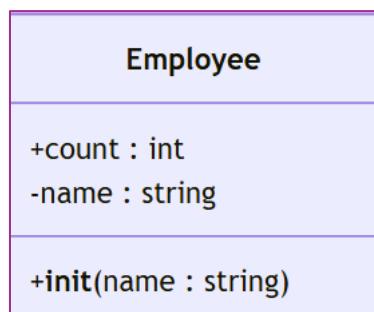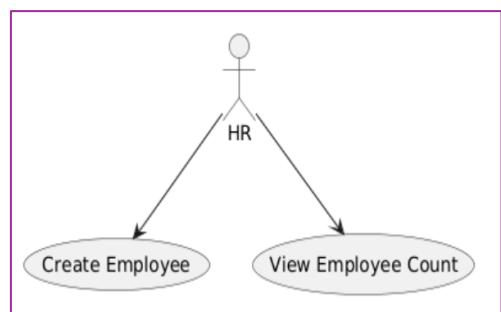
3. Employee Count Tracker

Class Diagram: **Use Case Diagram:**

Employee
+count : int   -name : string
+**init**(name : string)
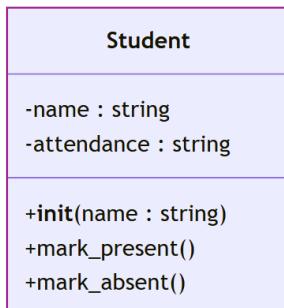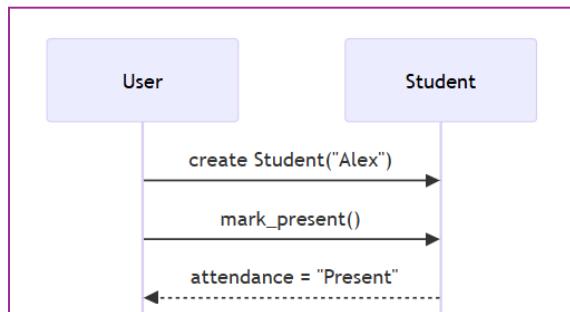
Level 2 Problems

4. Student Attendance System

Class Diagram

Student
-name : string
-attendance : string
+**init**(name : string)
+mark_present()
+mark_absent()

Sequence Diagram

User → Student: create Student("Alex")
User → Student: mark_present()
Student --> User: attendance = "Present"

Activity Diagram

Start → Create Student → Mark Present → Attendance becomes Present → End
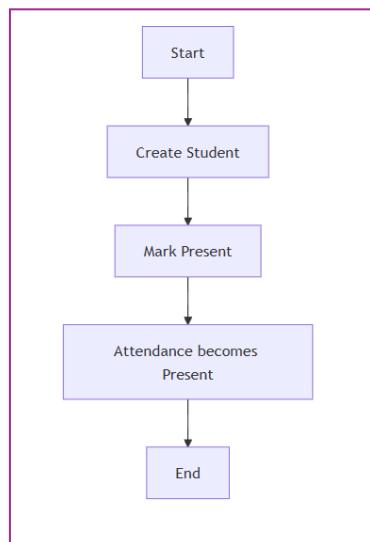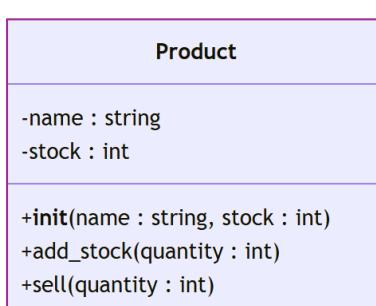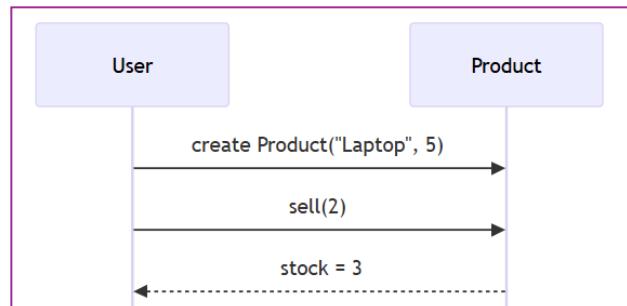
5. Online Store Product Catalog

Class Diagram

Product
-name : string
-stock : int
+**init**(name : string, stock : int)
+add_stock(quantity : int)
+sell(quantity : int)

Sequence Diagram

User → Product: create Product("Laptop", 5)
User → Product: sell(2)
Product --> User: stock = 3

Activity Diagram

```
                    ┌─────────┐
                    │  Start  │
                    └─────────┘
                         │
                ┌──────────────┐
                │ Create Product│
                └──────────────┘
                         │
                ┌──────────────┐
                │ Call sell method│
                └──────────────┘
                         │
                      ╱─────╲
                    ╱         ╲
                  ╱  Is quantity ╲
                 ╲   <= stock?   ╱
                    ╲         ╱
                      ╲─────╱
                   Yes │     │ No
          ┌─────────────────┐  ┌──────────────────┐
          │ Reduce stock by │  │ Print Not Enough │
          │    quantity     │  │      Stock       │
          └─────────────────┘  └──────────────────┘
                       │           │
                     ┌─────────┐
                     │   End   │
                     └─────────┘
```
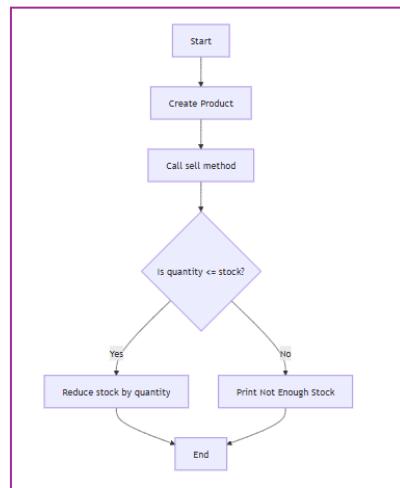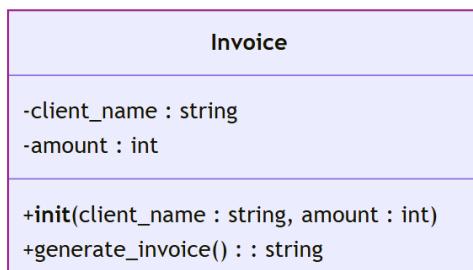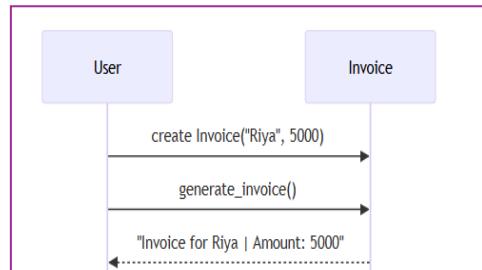
6. Invoice Generator

Class Diagram

Invoice
-client_name : string
-amount : int
+**init**(client_name : string, amount : int)
+generate_invoice() : : string

Sequence Diagram

```
  ┌──────┐                    ┌─────────┐
  │ User │                    │ Invoice │
  └──────┘                    └─────────┘
     │    create Invoice("Riya", 5000)  │
     │ ───────────────────────────────> │
     │       generate_invoice()         │
     │ ───────────────────────────────> │
     │  "Invoice for Riya | Amount: 5000"│
     │ <·······························  │
```

Activity Diagram

```
              ┌─────────┐
              │  Start  │
              └─────────┘
                   │
           ┌──────────────┐
           │Create Invoice │
           │    Object      │
           └──────────────┘
                   │
           ┌──────────────┐
           │Call generate_invoice│
           │     method     │
           └──────────────┘
                   │
           ┌──────────────┐
           │Display invoice │
           │    string      │
           └──────────────┘
                   │
              ┌─────────┐
              │   End   │
              └─────────┘
```
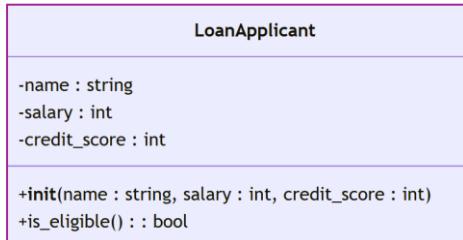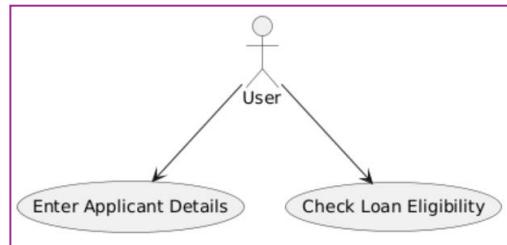
Level 3 Problems

7. Bank Loan Eligibility System

Class Diagram

LoanApplicant

-name : string
-salary : int
-credit_score : int

+**init**(name : string, salary : int, credit_score : int)
+is_eligible() : : bool

Use Case Diagram

User

Enter Applicant Details

Check Loan Eligibility

Sequence Diagram

User → LoanApplicant

create LoanApplicant("Riya", 80000, 750)

is_eligible()

True

State Diagram

Evaluating

Eligible

NotEligible

Activity Diagram

Start

Create Applicant

Check Eligibility

Meets rules

Yes → Return True

No → Return False

End
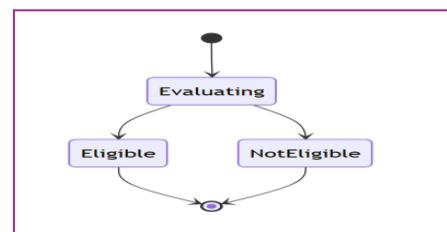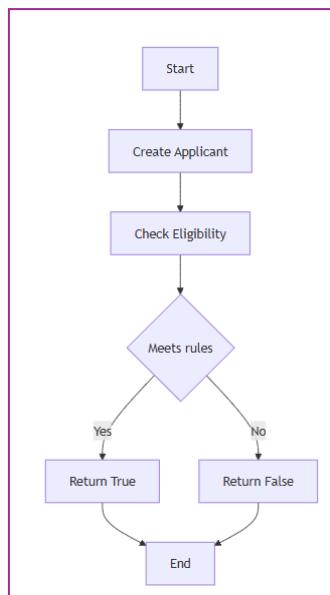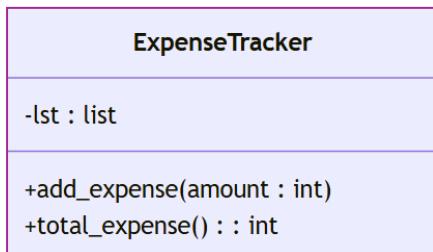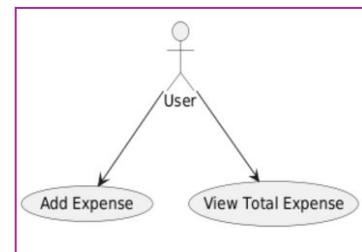
8. Expense Tracker

Class Diagram

ExpenseTracker
-lst : list
+add_expense(amount : int) +total_expense() : : int

Use Case Diagram

User

Add Expense   View Total Expense

Sequence Diagram

User   ExpenseTracker

create tracker

add expense 200

add expense 300

get total expense

total is 500

State Diagram

Empty

Recording

Calculated

Activity Diagram

Start

Create Expense Tracker

Add Expense

Yes

More expenses

No

Calculate Total Expense

Display Total

End
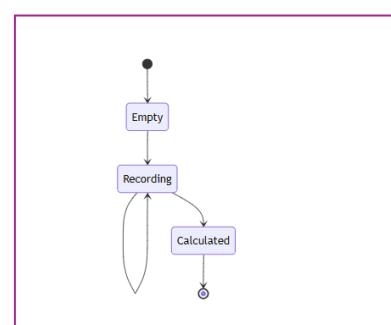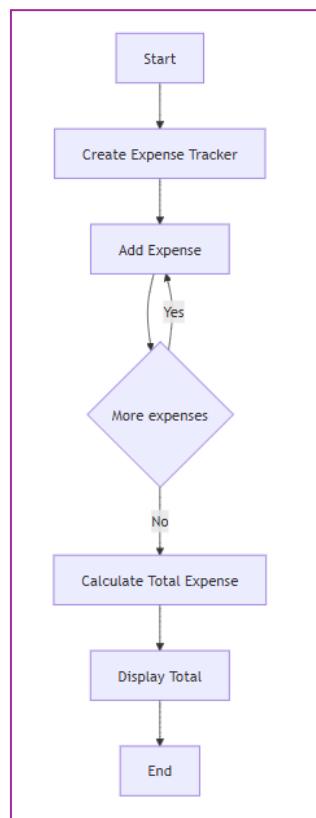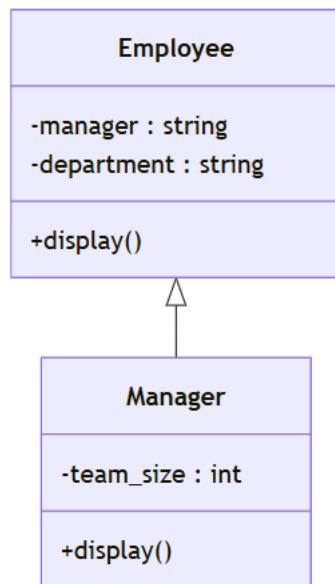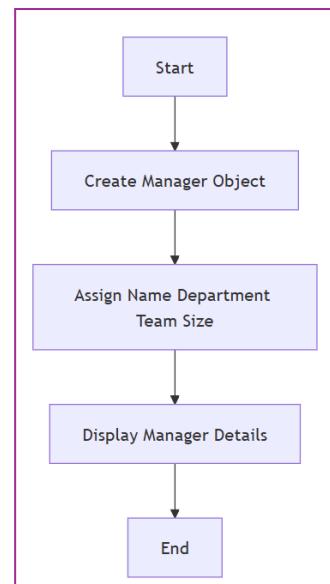
9. Employee Hierarchy Manager

Class Diagram

Employee

-manager : string
-department : string

+display()

△

Manager

-team_size : int

+display()

Activity Diagram

Start

↓

Create Manager Object

↓

Assign Name Department Team Size

↓

Display Manager Details

↓

End

Sequence Diagram

User | Manager

create manager object →

call display →

← show manager details

Use Case Diagram

User

Create Employee | Create Manager | View Employee Details | View Manager Details

State Diagram