**Experiment No. 5: Use Case and Gantt Chart Development**

### **Aim:**

To create use case diagrams and Gantt charts for the project, representing functional requirements and project timelines effectively.

### **Theory:**

**1. Use Case Diagrams**  
A use case diagram is a visual representation of the interactions between the system and its users (actors). It illustrates the functional requirements of the **Expense Splitter App**, helping to understand how users interact with the system.

#### **Key Components:**

* **Actors:** The entities interacting with the system, such as **Users** (who add and manage expenses) and **Admins** (who oversee user activities).
* **Use Cases:** The core functionalities of the app, such as **Register/Login, Add Expense, Split Expense, Settle Payments, Manage Users, and View Expense Reports**.
* **Relationships:** Associations between actors and use cases, including:
  + **Include:** Indicates mandatory functions within a primary use case (e.g., "Split Expense" includes "Calculate Shares").
  + **Extend:** Represents optional or conditional functionality (e.g., "Send Reminder" extends "Settle Payment" for overdue transactions).

**2. Gantt Charts :**  
A Gantt chart is a project management tool that visually represents the development timeline of the **Expense Splitter App**. It outlines tasks, their durations, dependencies, and progress over time.

#### **Key Components:**

* **Tasks:** Activities involved in developing the app, such as **Planning, Backend Development (Node.js & Database Setup), Frontend Development (React & Tailwind CSS), API Integration, Feature Implementation (Expense Tracking, Payment Handling), Testing, and Deployment**.
* **Duration:** The estimated time required for each task, ensuring smooth project progression.
* **Dependencies:** The relationship between tasks (e.g., **"Create Database Schema" must be completed before "Database Setup" begins**).
* **Timeline:** A visual schedule that tracks milestones, development phases, and deadlines.

**Learning Objective:**

* To understand the role of use case diagrams in capturing functional requirements.
* To gain proficiency in creating Gantt charts for effective project scheduling and tracking.
* To apply analytical and design skills for project management and system design.

### **Learning Outcome:**

At the end of this experiment, students will be able to:

1. Develop use case diagrams to represent functional requirements of their project.
2. Create Gantt charts to manage project timelines and dependencies.
3. Demonstrate the ability to plan, schedule, and track project progress effectively.

### **Course Outcomes (COs):**

* **CO3:** Identify and formulate functional and non-functional requirements for software systems.
* **CO5**: Evaluate and mitigate software project risks and apply configuration management practices to maintain project integrity.

### **Cognitive Levels of Attainment as per Bloom’s Taxonomy:**

* **L3 (Apply):** Apply knowledge to create use case diagrams and Gantt charts for project planning.
* **L5 (Evaluate):** Evaluate project timelines and dependencies using Gantt charts to ensure efficient scheduling.

### **Programme Outcome (POs:**

* **PO2: Problem Analysis:** Analyze project requirements and represent them using use case diagrams.
* **PO3: Design/Development of Solutions:** Design Gantt charts to plan and manage project deliverables.
* **PO9: Communication:** Communicate project requirements and timelines through visual representations.
* **PO10: Project Management and Finance:** Utilize project management tools to organize tasks and resources effectively.

### **Programme Specific Outcome (PSOs):**

* **PSO1:** Apply software engineering principles to create sustainable IT solutions by effectively managing project timelines and requirements.

### **Result & Discussion:**

* **Result**: Developed a comprehensive use case diagram to represent functional requirements and a Gantt chart to manage project tasks and timelines.
* **Discussion:**
  + The use case diagram identified key interactions between actors and the system.
  + The Gantt chart highlighted task durations, dependencies, and the critical path for project completion.

### **Conclusion:**

Use case diagrams and Gantt charts are essential tools for understanding system requirements and managing project timelines. By developing these artifacts, students gained insights into system design and project management, ensuring better planning and execution of their project.

### **Steps for the Experiment**

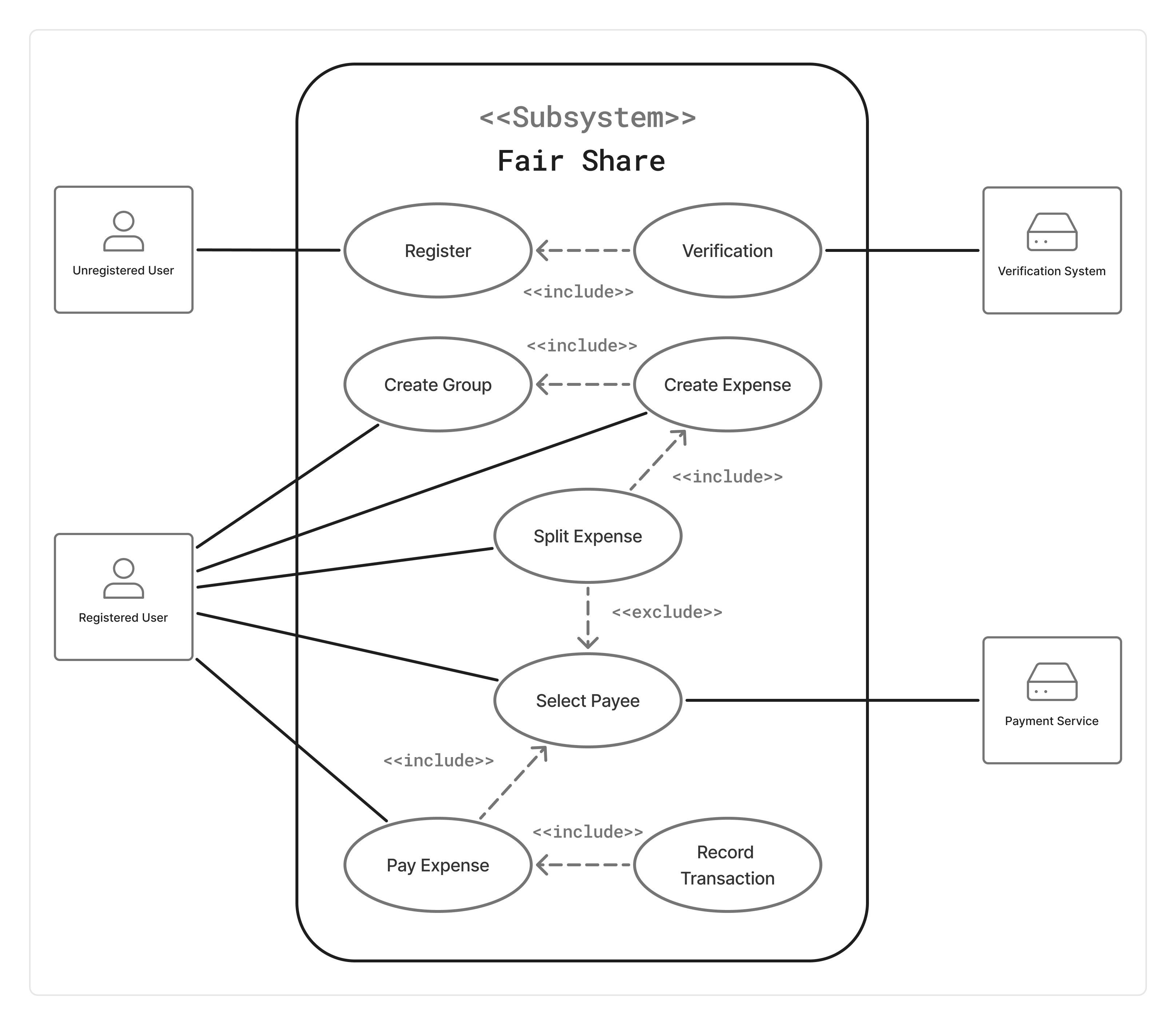
#### ****1. Use Case Diagram Development****

1. Identify all actors interacting with the system.
2. List the primary use cases or functionalities of the system.
3. Establish relationships (e.g., include, extend) between actors and use cases.
4. Use tools like Lucidchart, Microsoft Visio, or draw.io to create the diagram.

#### ****2. Gantt Chart Development****

1. Break down the project into tasks or activities.
2. Assign durations and dependencies to each task.
3. Create a Gantt chart using tools like MS Project, Excel, or online Gantt chart tools.
4. Review and refine the chart to ensure it reflects the project schedule accurately.

### **Use Case Diagram for Expense Splitter App:**



**Gantt Chart:**

