Taskeen Abdoola – u22698681
Karishma Boodhoo – u23538199
Sabira Karie – u23591481
Rene Reddy – u23558572
Kiolin Gounden – u22647300

TASK 1: PRACTICAL ASSIGNMENT 5

1.1 GREENHOUSE/ GARDEN AREA

Functional requirements:

Design pattern: **Command**
FR1: Plant care -
The system shall execute different care commands (watering, fertilizing, sunlight exposure) based on the specific needs of each plant type.The system shall allow gardeners to undo the last care action performed on a plant in case of errors.

Design pattern: **State**
FR2: Plant life cycle -
The system shall transition plants through distinct life cycle states (Seedling, Flowering, Mature, Ready for Sale, Sold) based on age, care received, and environmental conditions.

Design pattern: **Iterator**
FR3: Inventory -
The system shall provide a mechanism to traverse through all plants in the inventory without exposing the underlying collection structure.The system shall allow staff to iterate through plants that are ready for sale separately from plants that are still growing.

Design pattern: **Composite**
FR4: Stock -
The system must allow users to work with individual plants and plant groups using identical operations (display, get price, count items).

Design pattern: **Factory Method**
FR5: Stock -
The system must allow users to create plants by selecting a category (Succulent, Flower, or Tree) without knowing the specific plant type that will be created.

Design pattern: **Observer**
FR6: Stock -
The system must automatically notify the inventory tracker and staff when plants are added to or removed from stock.

Non-functional requirements:

NFR1: Maintainability -
The system shall be designed such that adding a new plant type requires only the creation of a new factory class and plant subclass, with no modifications to existing code.

1.2 THE STAFF

<u>Functional requirements:</u>

Design pattern: **Command**
FR1: Plant care -
The system shall allow the staff to execute plant care routines such as: watering, pruning and fertilizing, as discrete commands can be queued, scheduled and undone.

Design pattern: **Chain of Responsibility**
FR2: Plant life cycle -
The system shall automatically escalate plant health issues through a chain of staff members based on the severity. The members will range from junior gardeners to plant specialists.

Design pattern: **Mediator**
FR3: Inventory -
The system shall coordinate inventory updates between greenhouse and sales floor through a central mediator which ensures that the staff receives timely stock notifications.

Design pattern: **Chain of Responsibility**
FR4: Customer query handling system -
The system shall route customer queries through a help chain where different staff members will handle questions based on their expertise and authority level.

Design pattern: **Factory Method** -
FR5: The system shall create different staff roles such as: gardeners, sales assistants and managers, which uses the factory method to support future role extensions.

<u>Non-functional requirements:</u>

NFR1: Maintainability -
The system shall be designed with modular components which allows new plant types, staff roles and care routines to be added with minimal code changes.

1.3 THE CUSTOMERS AND THE SALES FLOOR

<u>Functional requirements:</u>

Design pattern: **Iterator**
FR1: Customer Browsing -
This system provides traversal of items ready for sale and all items without exposing internal collections.

Design pattern: **Strategy**
FR2: Customer Browsing -
This system will provide the Customer with different recommendations/ strategies based on their inputs or not.

Design pattern: **Builder**
FR3: Personalization -
The system shall assemble multi-component arrangements stepwise via a builder (base plant(s), pot, wrap, tag), validating order and mandatory components.

FR4: Personalization -
The system shall provide pre-defined “recipes” (Director) to construct common arrangements (e.g., Gift) with one call.

Design pattern: **Prototype**
FR5: Personalization -
The system shall clone plant/arrangement templates from a registry to create personalised variants without reconfiguring from scratch.

Design pattern: **Decorator**
FR6: Personalization -
The system shall allow a customer to apply multiple personalisation layers to an item (e.g., DecorativePot, GiftWrap), such that each layer wraps the previous one and the final personalised item is still usable wherever an Item is expected.

Design pattern: **State**
FR7: Sales and Transactions -
The system shall represent each order with a state machine: CartOpen → PendingPayment → (PaymentAuthorized|PaymentFailed) → Completed with terminal state Cancelled.

FR8: Sales and Transactions -
Each state shall enforce allowed transitions and reject invalid actions (e.g., CapturePayment is disallowed unless in PaymentAuthorized).

Design pattern: **Command**
FR9: Sales and Transactions -
Sales operations shall be executed as commands (AddToCart, RemoveFromCart, AuthorizePayment, CapturePayment, CommitOrder, CancelOrder) that mutate the order and may emit domain events.

FR10: Sales and Transactions -
If a command in a multi-step transaction fails, the system shall execute configured compensation commands (e.g., CancelOrder after failed CommitOrder).

Design Pattern: **Strategy**
FR11: Pricing Policy -
The system shall compute price via a pluggable PricingStrategy selected at checkout time (e.g., base, promo code, bulk discount).

FR12: Pricing Policy -
The system shall allow runtime switching of the active pricing policy per order (e.g., apply promotional strategy when a valid coupon is added).

Non-functional requirements:

NFR1: Maintainability -
Concerns regarding different types of recommendations are separated. This makes modifying and adding recommendations easier.

NFR2: Scalability -
The pricing subsystem must support dynamic switching between pricing strategies (e.g., base, bulk, promo) without requiring system downtime or redeployment, maintaining 100% service availability during pricing policy updates.

NFR3: Usability and Flexibility -
The arrangement-builder component must enable customers to construct customised plant arrangements in user interactions (steps) on the interface, and allow for re-ordering or skipping steps without errors, ensuring high usability and flexibility in the personalisation workflow.

Constraints:

- Team Composition: 5
- Design Patterns : 12 (7 Behavioural, 3 Creational and 2 Structural)
- UML Documentation Provided