Templanter: A Plant Nursery Simulator

COS 214 Final Project Report

Team Templation
- Jo Reardon – u24597652
- Paul Hofmeyr – u24618391
- Finnley Wyllie – u24754120
- Mutombo Kabau – u24957102
- Alessandro Paravano – u24713122

University of Pretoria
- November 2025

1. Project Overview

Templanter is a software simulation of a plant nursery, built in C++. The system models the core operations of a greenhouse, including plant growth, staff management, and customer interactions. The primary objective was to design a maintainable and scalable system by applying object oriented principles and design patterns to solve specific architectural problems. The simulation is structured around three main subsystems: Greenhouse, Staff, and Customers.

2. Research Brief

Effective nursery management requires coordination between plant care, inventory tracking, and customer service. Our research into real world nurseries highlighted the need for systems that can manage different plant lifecycles, employee schedules, and customer preferences. This informed our design to prioritize flexibility and scalability. This is achieved by providing the ability to easily specialise or generalise parts of the system. We simplified real world problems, such as pest control and weather, in order to focus on the core challenge of creating a cohesive system.

3. System Architecture

The system is divided into three subsystems that communicate through well defined interfaces.

Greenhouse Subsystem: Manages all plant related logic. This includes plant creation, growth cycles, health status, and inventory tracking. It is the core of the simulation.

Staff Subsystem: Handles worker entities. Staff are assigned tasks such as watering, fertilizing, and harvesting plants, and assisting customers.

Customer Subsystem: Manages customer entities. Customers can browse available plants, interact with staff, and make purchases.

4. Design Patterns Application

We implemented 11 design patterns to address specific challenges in the system's design. The five required patterns from the project specification are indicated.

4.1 Creational Patterns

Factory Pattern: Used to create different types of plants and customers. The PlantFactory generates specific plant objects like Tomato or Sunflower, while the CustomerFactory creates different customer types such as Regular, VIP, and Robber.

Singleton Pattern: Ensures a single instance of the core Game class exists. This provides a global point of access for the game state, inventory, and greenhouse, preventing inconsistencies.

4.2 Structural Patterns

Adapter Pattern: Converts the Plant interface to the StoreItem interface. This allows plant objects to be displayed and sold on the sales floor without modifying the original Plant class.

Facade Pattern: Provides a simplified interface, the Demo class, to the complex subsystems of the game loop. This makes the system easier to use and reduces dependencies.

Flyweight Pattern: Shares common, immutable data like customer images across multiple objects. This significantly reduces the memory footprint when handling many customers.

4.3 Behavioral Patterns

Command Pattern: Encapsulates actions like watering or harvesting as objects. These command objects are passed between workers and plants, allowing for flexible task execution and queuing.

Observer Pattern: Allows the Greenhouse to notify Workers when plants need care. This creates a one to many dependency so that multiple workers can be alerted automatically without the plants having direct references to them.

State Pattern: Manages a plant's behavior through its lifecycle. A plant can be in different states such as Seed, Growing, Ripe, or Decaying, with each state defining its specific behavior.

Strategy Pattern: Defines a family of interchangeable algorithms for rendering plants. Different visual strategies can be applied to a plant without changing its core class.

Template Method Pattern: Defines the skeleton of the plant growth algorithm in a base class. Subclasses can override specific steps, allowing different plants to have unique growth cycles while following a common structure.

Memento Pattern: Captures and externalizes the game's internal state. This enables save and load functionality, allowing the simulation state to be restored for day night cycles or different sessions.

5. Implementation

The system was implemented in C++. We used Git for version control and GitHub for collaboration, following a feature branch workflow. All team members contributed regularly, with each member achieving over 10 commits. The code is documented using Doxygen and includes a suite of unit tests.

6. Conclusion

The Templanter project successfully meets the requirements for the COS 214 final project. We developed a functional plant nursery simulator that effectively models greenhouse operations, staff activities, and customer interactions. The application of 11 design patterns, including all required ones, resulted in a codebase that is flexible, maintainable, and demonstrates a strong understanding of software design principles. The project provided valuable experience in object oriented design, team collaboration, and professional development practices.

References

1. Acovino, V. and Summers, J. (2025) *The legacy - and future - of the farming game Stardew Valley*, *NPR*. Available at: https://www.npr.org/2025/01/24/g-s1-44510/the-legacy-and-future-of-the-farming-game-stardew-valley (Accessed: 20 October 2025).

2. FarmERP (2025) *Advanced Nursery Management software to boost production and profitability*, *FarmERP*. Available at: https://www.farmerp.com/blog/advanced-nursery-management-software-to-boost-production-and-profitability/ (Accessed: 21 October 2025).

3. (No date) *Nursery management - an overview | sciencedirect topics*. Available at: https://www.sciencedirect.com/topics/agricultural-and-biological-sciences/nursery-management (Accessed: 21 October 2025).

4. (2017) © *July 2020| IJIRT | Volume 7 issue 2 | ISSN: 2349-6002 IJIRT 150053*. Available at: https://ijirt.org/publishedpaper/IJIRT150053_PAPER.pdf (Accessed: 20 October 2025).

5.*Development of Web Based Plant Nursery Management* (no date) *ISJEM Journal*. Available at: https://isjem.com/download/development-of-web-based-plant-nursery-management/ (Accessed: 04 November 2025).