

# Module #5 Report | CSE 310 – Applied Programming

**Name**      **Date**      **Teacher**  
Ethan Trent 12/06/2025 Bro McGary

## Project Repository Link

Copy the link to your [Github Repository \(https://github.com/ethantrent/SmallBizInventory\)](https://github.com/ethantrent/SmallBizInventory)

## Module

Mark an **X** next to the module you completed

Module	Language
Cloud Databases	Java
Data Analysis	Kotlin
Game Framework	R
GIS Mapping	Erlang
Mobile App	JavaScript
Networking	C#
Web Apps	TypeScript
Language – C++	X Rust
SQL Relational Databases	Choose Your Own Adventure

## Fill Out the Checklist

Complete the following checklist to make sure you completed all parts of the module. Mark your response with **Yes** or **No**. If the answer is **No** then additionally describe what was preventing you from completing this step.

Question	Your Response	Comments
Did you implement the entire set of unique requirements as described in the Module Description document in I-Learn?	Yes	Implemented OOP with inheritance, STL containers (vector/map), file I/O, dynamic memory management, virtual functions, and full CRUD operations
Did you write at least 100 lines of code in your software and include useful comments?	Yes	Over 1500 lines of documented code across all source files with comprehensive comments explaining C++ concepts
Did you use the correct README.md template from the Module Description document in I-Learn?	Yes	Used the C++ README template with all required sections
Did you completely populate the README.md template?	Yes	All sections completed including overview, installation, usage, code structure, and design decisions
Did you create the video, publish it on YouTube, and reference it in the README.md file?	Yes	Video recorded, link added to README
Did you publish the code with the README.md (in the top-level folder) into a public GitHub repository?	Yes	Pushed to GitHub

## Did you complete a Stretch Challenge

If you completed a stretch challenge, describe what you completed.

### Yes - Completed Multiple Stretch Challenges:

- File I/O for Data Persistence:** Implemented CSV-based save/load functionality with automatic saving on exit. Products are serialized to CSV format with type-prefixed records for proper deserialization.
- Inheritance with Virtual Functions:** Created a complete Product class hierarchy with pure virtual functions (`display()`, `getType()`) and overridable virtual functions (`calculateValue()`, `applyDiscount()`). Demonstrated runtime polymorphism throughout the application.
- Dynamic Memory Management:** Used `new/delete` for creating/destroying products dynamically. Implemented proper memory cleanup in destructor and `clearAll()` method to prevent memory leaks. Deleted copy constructor/assignment operator to prevent double-deletion issues.
- Advanced STL Integration:** Utilized both `std::vector<Product*>` and `std::map<std::string, Product*>` containers simultaneously - vector for ordered iteration and sorting, map for O(log n) SKU lookups. Implemented multiple sorting algorithms using `std::sort` with lambda functions.

## Record your time

How many hours did you spend on this module and the team project this Sprint?

*Include all time including planning, researching, implementation, troubleshooting, documentation, video production, and publishing.*

Hours
Individual Module 12
Team Project 4

## Retrospective

- What learning strategies worked well in this module?

Breaking down the project into smaller components (base class first, then derived classes, then inventory manager, then UI) helped me understand C++ OOP concepts incrementally. Using `cppreference.com` for STL container documentation was invaluable. Creating the detailed plan with a schedule helped me stay on track and allocate time effectively. Testing each component as I built it prevented bugs from accumulating.

- What strategies (or lack of strategy) did not work well?

I initially struggled with memory management concepts and spent extra time debugging pointer issues. I should have reviewed RAII principles and smart pointers earlier. Also, I underestimated the time needed for comprehensive documentation and should have documented as I coded rather than at the end.

- How can you improve in the next module?

For future modules, I will: (1) Start with a small proof-of-concept to test the core technology before building the full application, (2) Document code and design decisions immediately while they're fresh in my mind, (3) Allocate specific time for video planning and recording early in the sprint, (4) Use debugging tools (like Valgrind or Visual Studio's memory profiler) proactively rather than reactively.

<!-- Create this Markdown to a PDF and submit it. In visual studio code you can convert this to a pdf with any one of the extensions. -->