<u>Final Project Proposal: Inventory Management System</u>

1. Project Title

Efficient Inventory Management System Using Dynamic Arrays and Category Search Trees

2. Team Members

1. Seiya Genda
2. Kwan Ho
3. Zach Madison

---

3. Project Overview

This project aims to use learned data structures and performance analysis techniques to improve the performance of an Inventory Management System. The application this project will be modifying is a Gemini developed IMS available in our group repository in App\old at https://github.com/zacMadison/CYBR330-InentoryManagement. We chose this because AI becomes more relevant in the world of programming everyday, and we wanted to highlight how it's output can be improved as well as to display its pitfalls. The modified system will utilize data structures and algorithms learned from class, such as dynamic array for storage, Binary search for improved searching efficiency, and heap implementation for sorting. Each item in the inventory is represented as an object containing:

- Name
- Price
- Quantity
- Date

Additionally, a Category Tree will be used to optimize search operations based on product categories, improving retrieval performance compared to linear search.

Primary implementation language: **Python**

Optional performance analysis and visualization language: **JAVA**

---

4. Objectives

The main objectives of this project include:

- Implement a dynamic array that automatically resizes to support expanding inventory data.
- Find and implement efficiencies in the code base.
- Utilize learned data structures to improve speed and space efficiency.
- Develop a tree-based category search algorithm for fast query operations.
- Efficient implementation of essential inventory functions such as add, remove, update, and search.
- Conduct performance analysis comparing operations with different data sizes.
- Maintain clear documentation in a shared repository.

---

5. Methodology

Our solution will incorporate multiple algorithms and data structures:

Algorithm Implementation

- Dynamic Array for storing items, including automatic resizing using doubling strategy.
- Binary Search Tree (BST) or General Tree for category indexing and fast searching.
- Standard algorithms for array insertions, deletions, and searches.
- Binary Search for improved searching speed
- Heaps and Heap sorting for improved sorting efficiency

Performance Analysis

- Measure execution time for:
    - Item insertion into dynamic array
    - Category-based tree search

    - Updating item data such as quantity and price
    - Any additional identified efficiencies
- Evaluate memory usage differences as the dataset scales (100 → 10,000 items)

Tools & Documentation

- Code stored on GitHub for collaboration
- Time and memory benchmarking scripts included in repository
- Commented code with a detailed README and performance results

---

6. Expected Outcomes

By the end of the project, we expect to achieve:

- A fully functional inventory system capable of handling large data efficiently
- Quantified performance metrics that show benefits of implemented data structures
- Real-world application example demonstrating learned data structure techniques
- Improved teamwork and version control experience

---

7. Timeline

Phase	Task	Deadline
Week One	Identify and Implement Efficiencies and Implement New Data Structures	Nov 23
Week Two	Perform Performance Analysis and Create Performance Analysis Documentation	Nov 29
Week Three	Final Documentation and Code Review	Dec 7

Conclusion

This project will provide hands-on experience applying data structure concepts to a practical and scalable software solution. With measurable performance improvements and clear system functionality, our Inventory Management System will demonstrate the importance of algorithm efficiency in real-world computing.