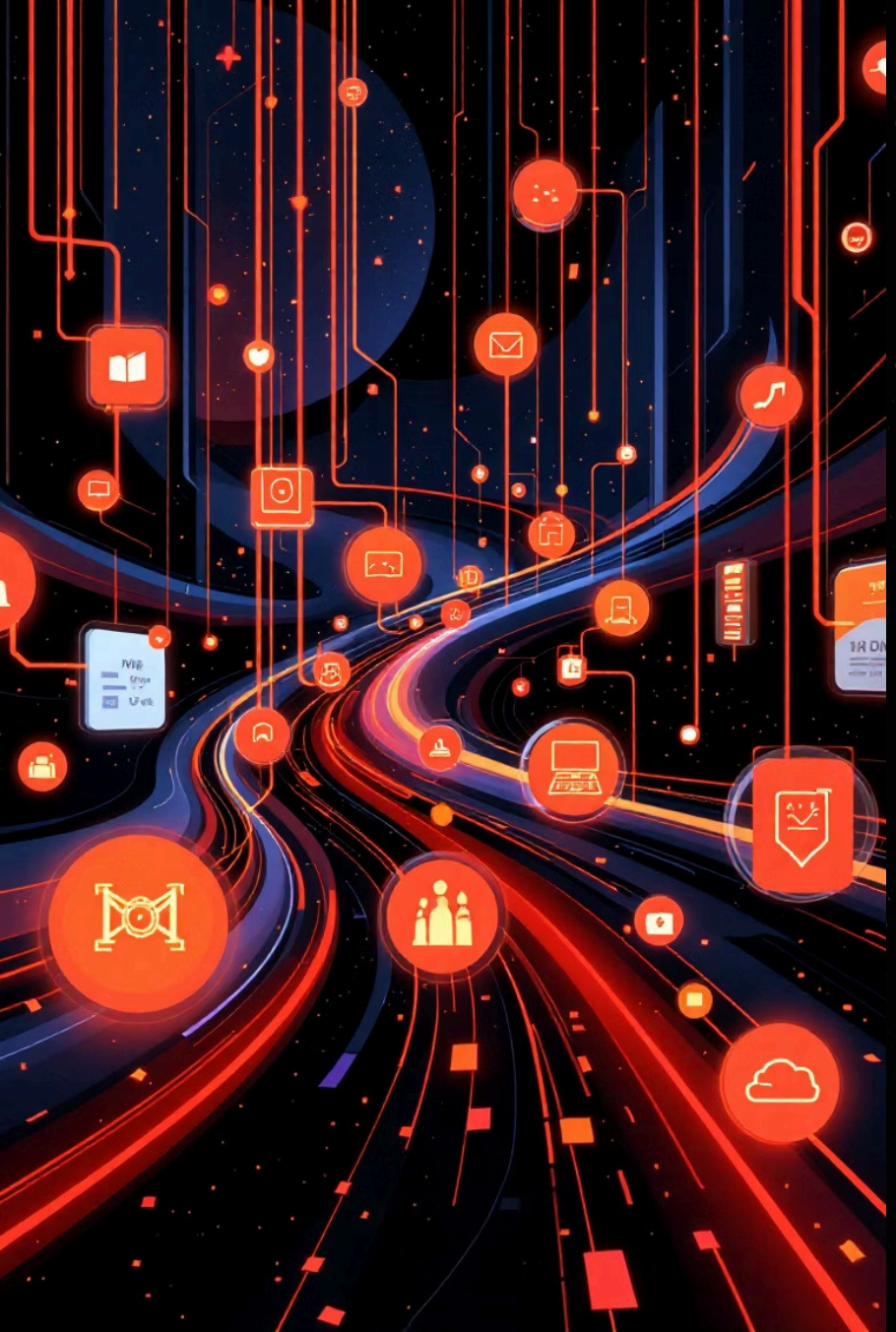


Online Marketplace Inventory & Order Matching Engine

Course: Data Structures & Algorithms

Team: Bilal Azfar & Huzaifa Soomar

Made with GAMMA

The Challenge: Problem Statement

Rapid Matching Demands

Online marketplaces require instantaneous buyer-seller matching to ensure user satisfaction and transaction speed.

Scalable Inventory

Tracking vast and dynamic product inventories at scale presents significant algorithmic and architectural challenges.

Intelligent Recommendations

Effective recommendation systems are crucial for enhancing user experience and driving increased sales conversions.

Integrated Solution

Our goal is to build an efficient engine that integrates matching, inventory, and personalized recommendations.

Made with GAMMA

Project Objectives

Efficient Order Matching

Develop a system to match buy and sell orders based on defined priority rules.

Dynamic Inventory Management

Implement robust mechanisms for real-time product inventory tracking and updates.

Recommendation System

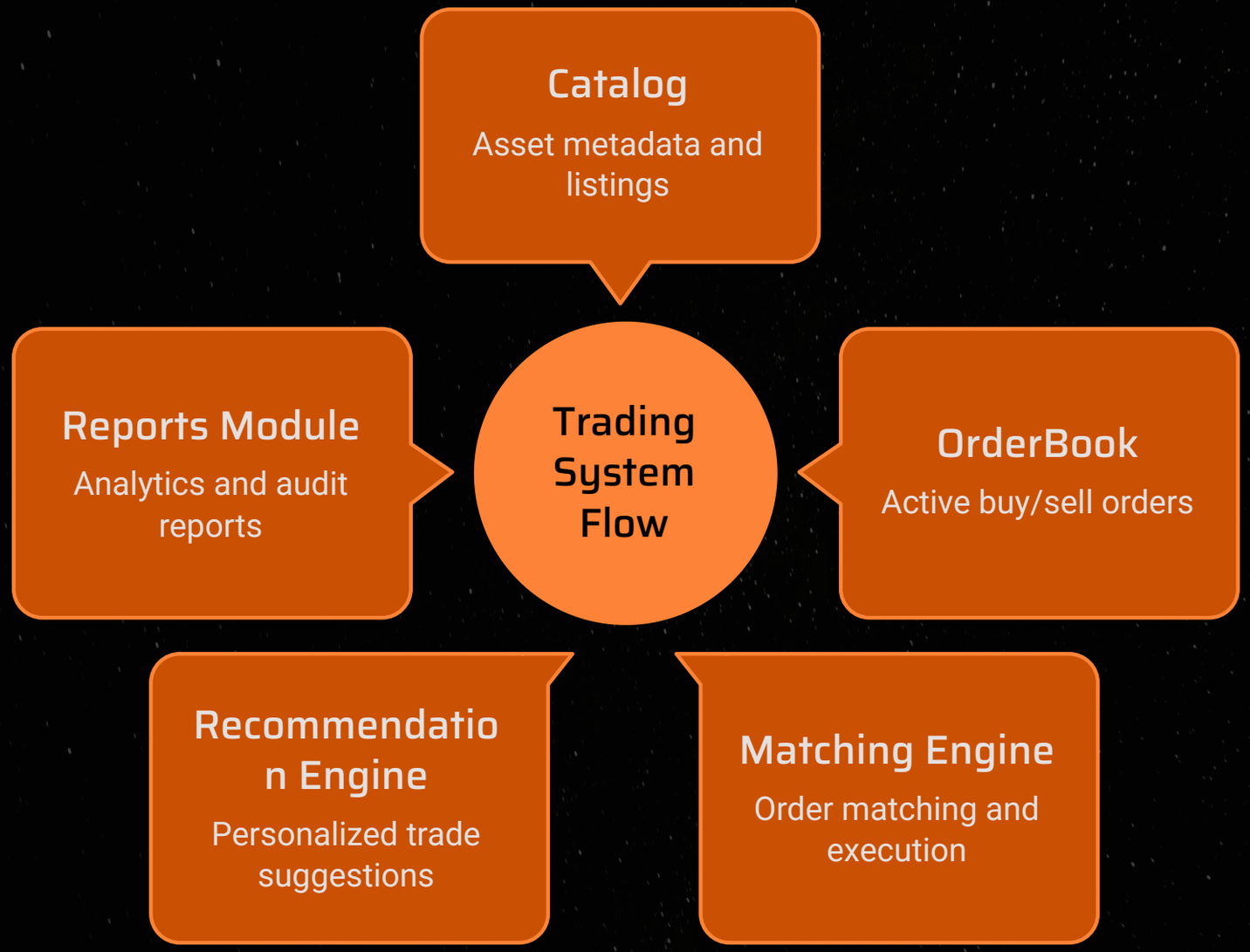
Design a "Frequently Bought Together" bundling feature to boost sales.

DSA Application

Strategically apply fundamental Data Structures and Algorithms for optimal performance.

Made with GAMMA

System Architecture Overview



The architecture is modular, allowing for independent development and scalable integration of core functionalities.

Made with GAMMA

Core Data Structures Applied

Hash Tables

For rapid product catalog lookups and efficient key-value storage.

Heaps (Priority Queues)

To manage and prioritize buy and sell orders for optimal matching.

Graphs

Modeling co-purchase relationships for intelligent product recommendations.

Trees

Organizing hierarchical data like product categories for efficient navigation.

Made with GAMMA

Order Matching Logic: Heaps in Action

Our matching engine leverages specialized heaps to achieve optimal matching efficiency:

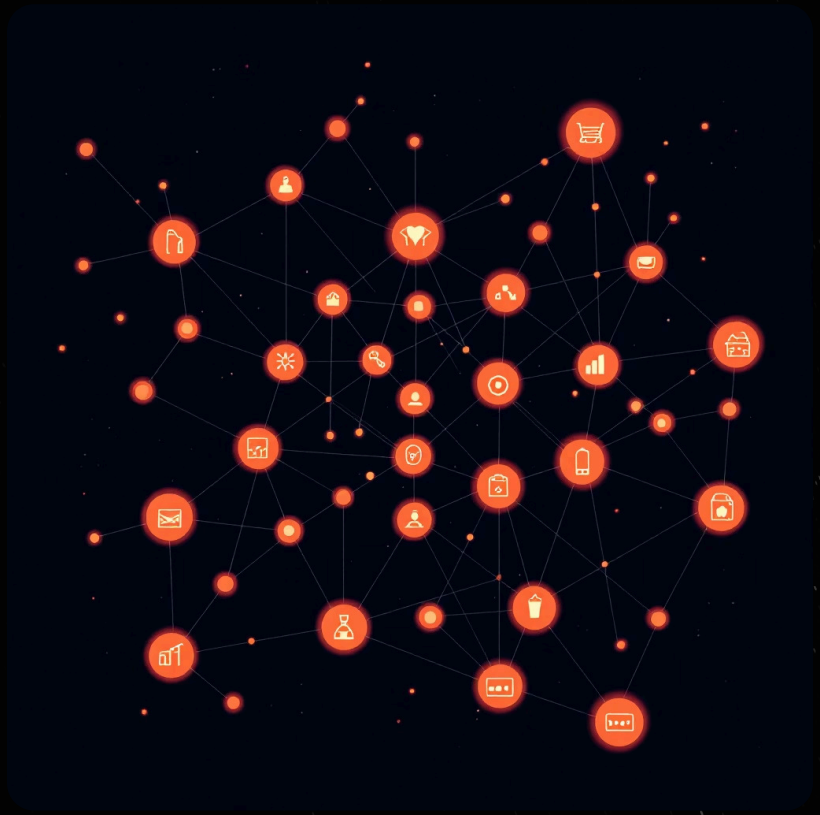
- **Buy Orders:** Stored in a **Max-Heap**, prioritizing highest price first.
- **Sell Orders:** Stored in a **Min-Heap**, prioritizing lowest price first.
- **Priority:** Orders are matched based on price-time priority.
- **Efficiency:** Matching operations are executed in **O(log n)** time, ensuring real-time performance.
- A custom heap implementation was developed to handle specific order attributes and rules.

Big-O Complexity

Heap operations (insertion, deletion, peek) are highly efficient, maintaining logarithmic time complexity.

Made with GAMMA

Recommendation System: Graph-Based Bundles



We implement a graph-based model to identify "Frequently Bought Together" product bundles:

- **Nodes:** Represent individual products within the marketplace.
- **Edges:** Indicate co-purchase relationships, weighted by frequency or strength of association.
- **Discovery:** Carries out graph-based analysis using weighted adjacency lists.
- **Output:** Extracts **Top-K** related products based on traversal results, offering relevant recommendations.

This approach provides dynamic and context-aware product suggestions.

Made with GAMMA

Inventory Management & Reporting

Real-time Updates

Accurate maintaining of product availability, instantly updating after every order matched.

Order Status Visibility

Matched orders are logged during execution; pending orders remain in the order book.

Summary Reports

Generation of comprehensive inventory summaries for operational insights.

Category Insights

Detailed breakdowns and analytics by product category for strategic decision-making.

Made with GAMMA

Testing & Validation

Rigorous testing was conducted to ensure the correctness, efficiency, and reliability of the engine.

Matching Correctness

Verified that all buy and sell orders were matched according to priority rules.

Concurrency Simulation

Simulated sequential concurrent order placements to test system stability.

Recommendation Accuracy

Evaluated the relevance and effectiveness of the "Frequently Bought Together" suggestions.

Performance Validation

Measured the Big-O complexity of critical operations and benchmarked response times.

Made with GAMMA

Conclusion & Future Work

Efficient & Scalable

Achieved efficient, real-time order matching with a scalable design using fundamental DSA principles.

Future Enhancements

- Implement true concurrency with thread-safe data structures.
- Integrate persistent storage for order books and inventory.
- Develop a user interface for real-time visualization and interaction.

This project demonstrates a robust foundation for building high-performance marketplace systems.

Made with GAMMA