CODING SKILLS - 1

PRESENTATION

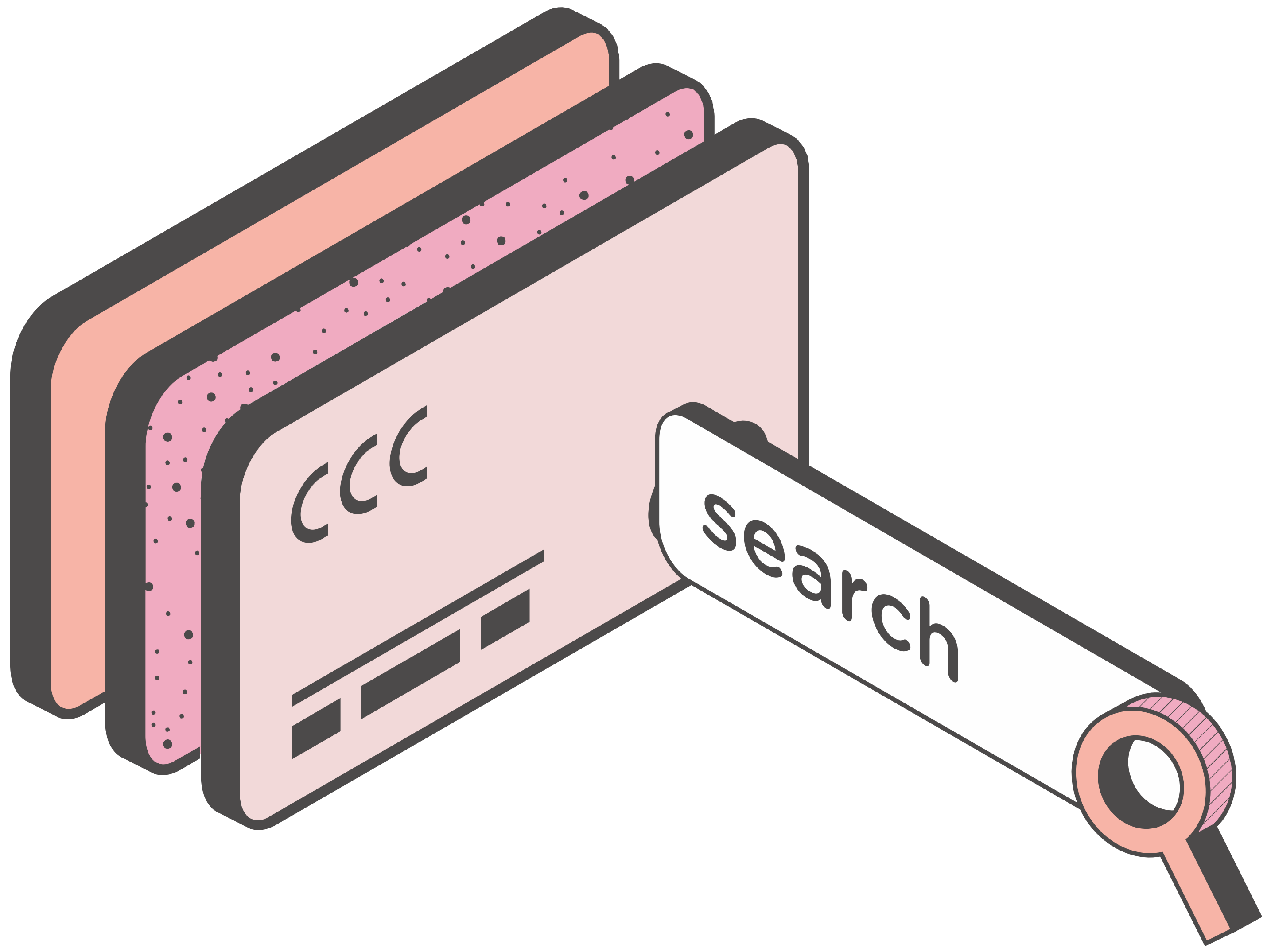
CIRCULAR LINKED LIST SIMULATION

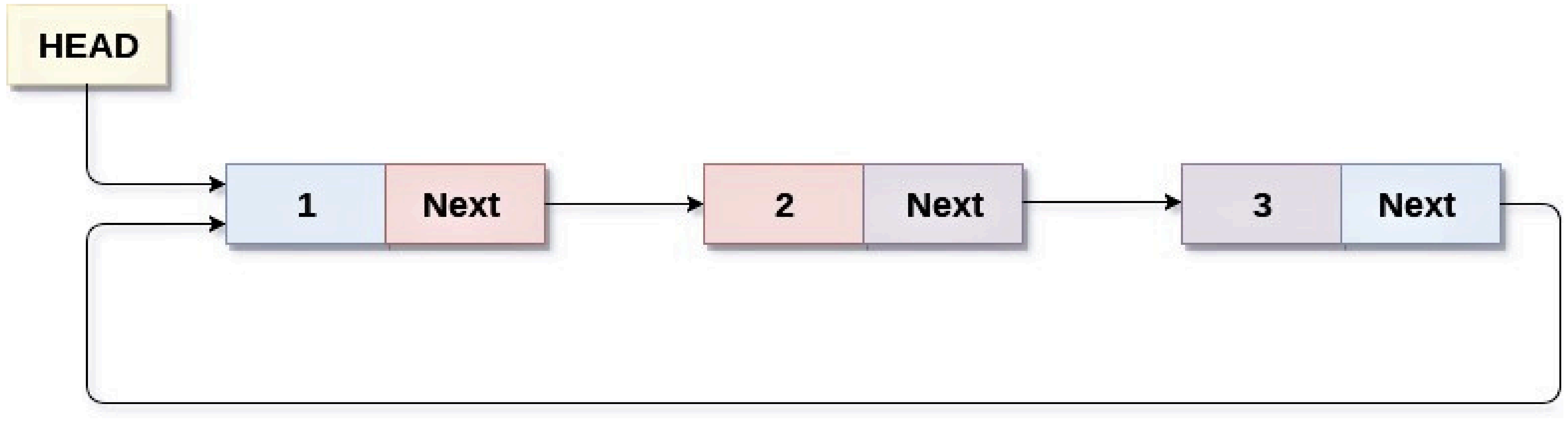
PRESENTED BY:

1. Y. HARDHIK AP24110011453
2. P. DHATRI AP24110011459
3. P. KIRANMAI AP24110011466
4. B. SRI BHAVYA AP24110011431
5. A. KEERTHAN AP24110011464
6. SK. YUSUF AP24110011408

INTRODUCTION

- A circular linked list is a variation of a linked list where the last node points back to the first node.
- Used in round-robin scheduling, buffering, and cyclic processes.
- Students often struggle to visualize how nodes connect in a loop.
- This project provides a GUI-based simulation for better understanding.

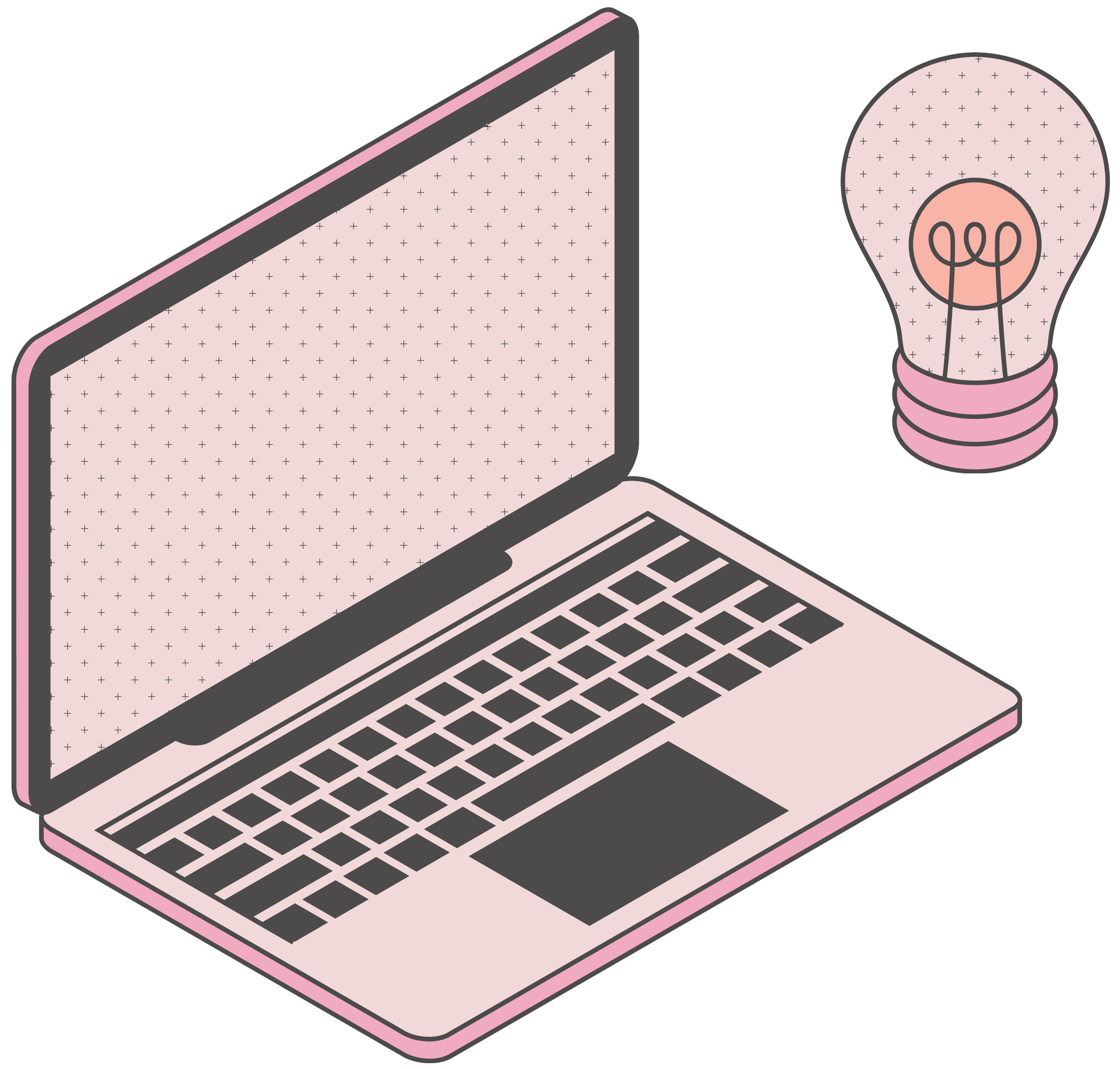




Circular Singly Linked List

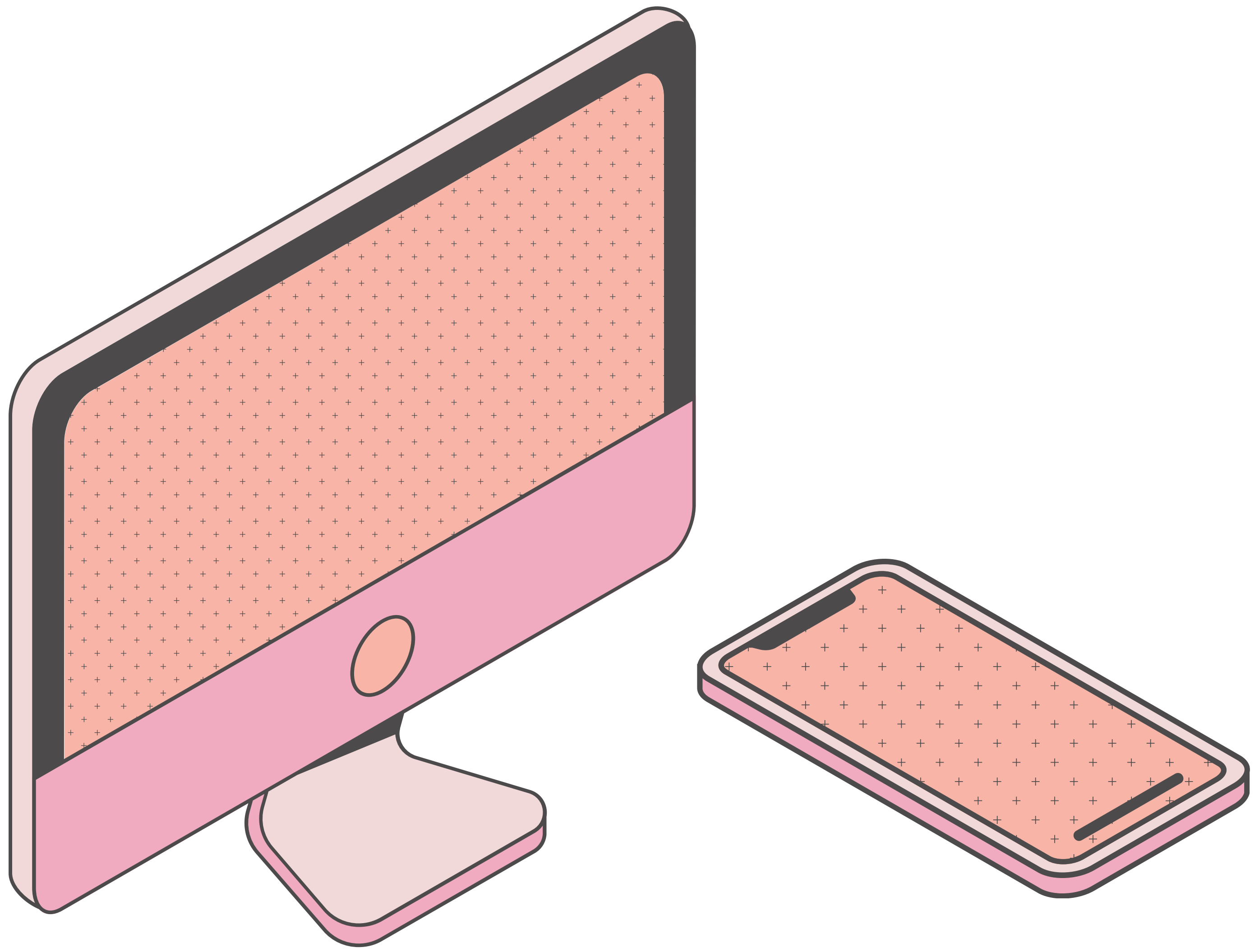
Problem Statement

- Difficult for learners to imagine circular pointer connections using theory alone.
- No simple interactive tools available for beginners.
- Need for a system that visually demonstrates insert, delete, search, and traversal operations.
- Goal: Create an intuitive GUI simulator to represent circular linked lists in real time.



OBJECTIVES

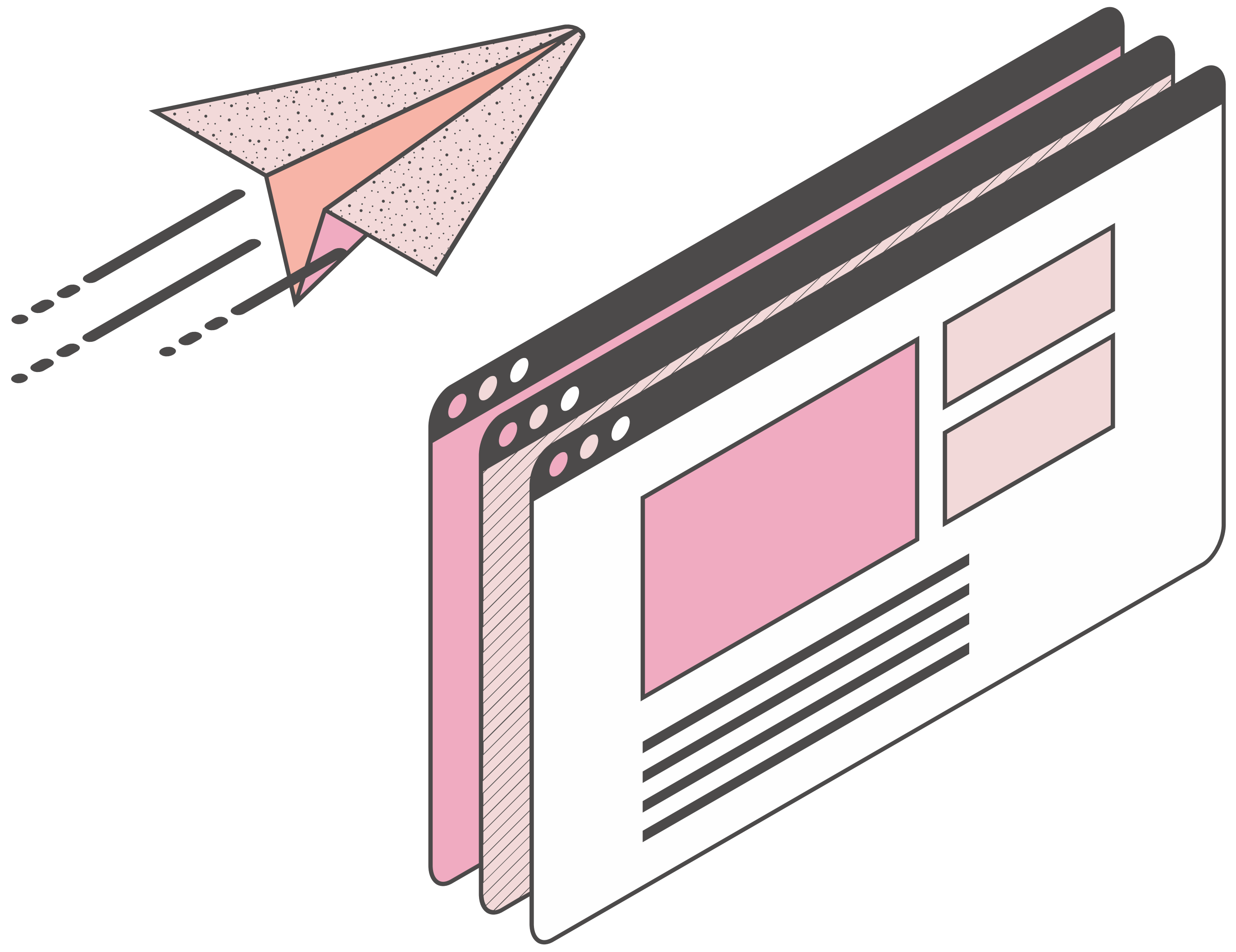
- Implement a fully functional Circular Linked List.
- Support insertion, deletion, searching, and traversal.
- Design an interactive GUI using Python Tkinter.
- Provide real-time visual updates of list operations.
- Improve conceptual clarity for students and beginners.



SYSTEM ARCHITECTURE

Two-Layer Architecture:

1. Data Structure Logic Layer
   - Node class
   - CircularLinkedList class
2. GUI Layer
   - Input field
   - Operation buttons
   - Output display panel



DATA STRUCTURE LOGIC

Node Structure:

- Each node contains data and a next pointer.
- The last node's pointer always links back to the head.

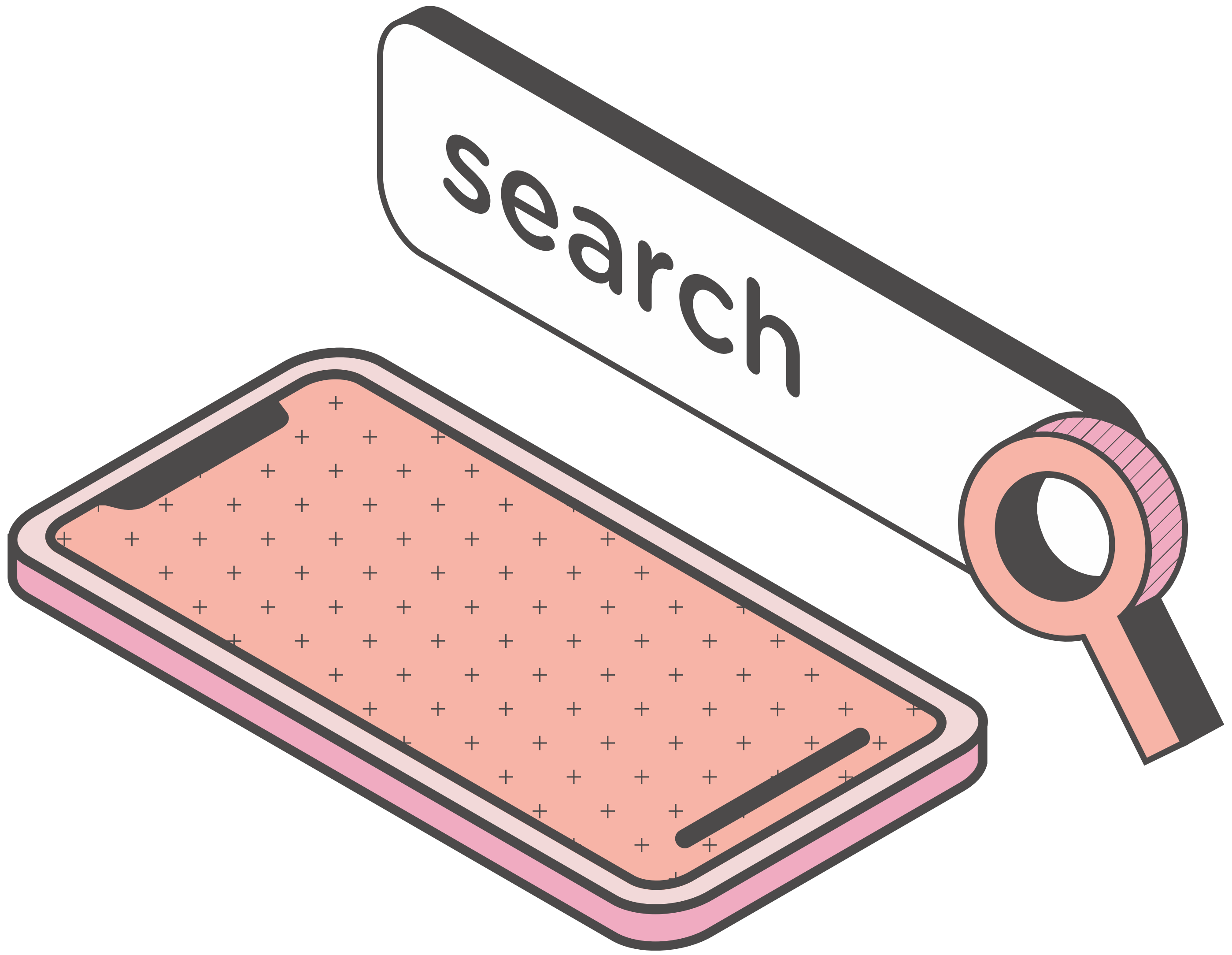
Maintains the circular connection throughout the list.

Operations Implemented:

- Insert at Front: Adds a new node at the beginning
- Insert at End: Appends a node and reconnects it to the head.
- Delete by Value: Removes the target node and adjusts surrounding pointers.
- Search: Traverses the list to find the value.
- Traversal: Visits each node until it loops back to the head.
- Ensures the circular linked structure remains intact after every update.

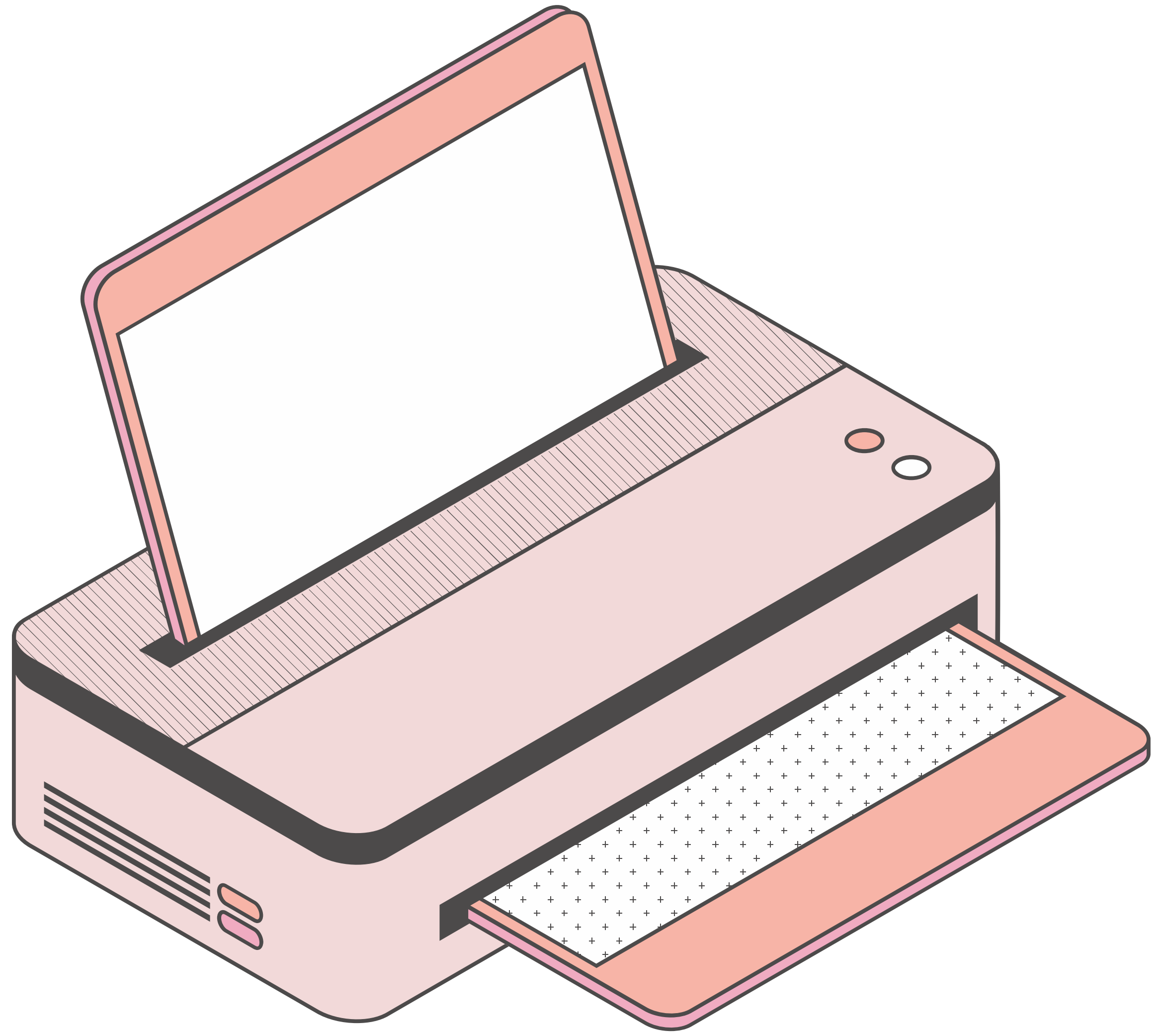
GUI DESIGN

- GUI created using Python Tkinter.
- Input box for entering values.
- Buttons for:
 1. Insert Front
 2. Insert End
- Delete Value
- Search Value
- Reset
- Output panel shows circular linked list in sequence.



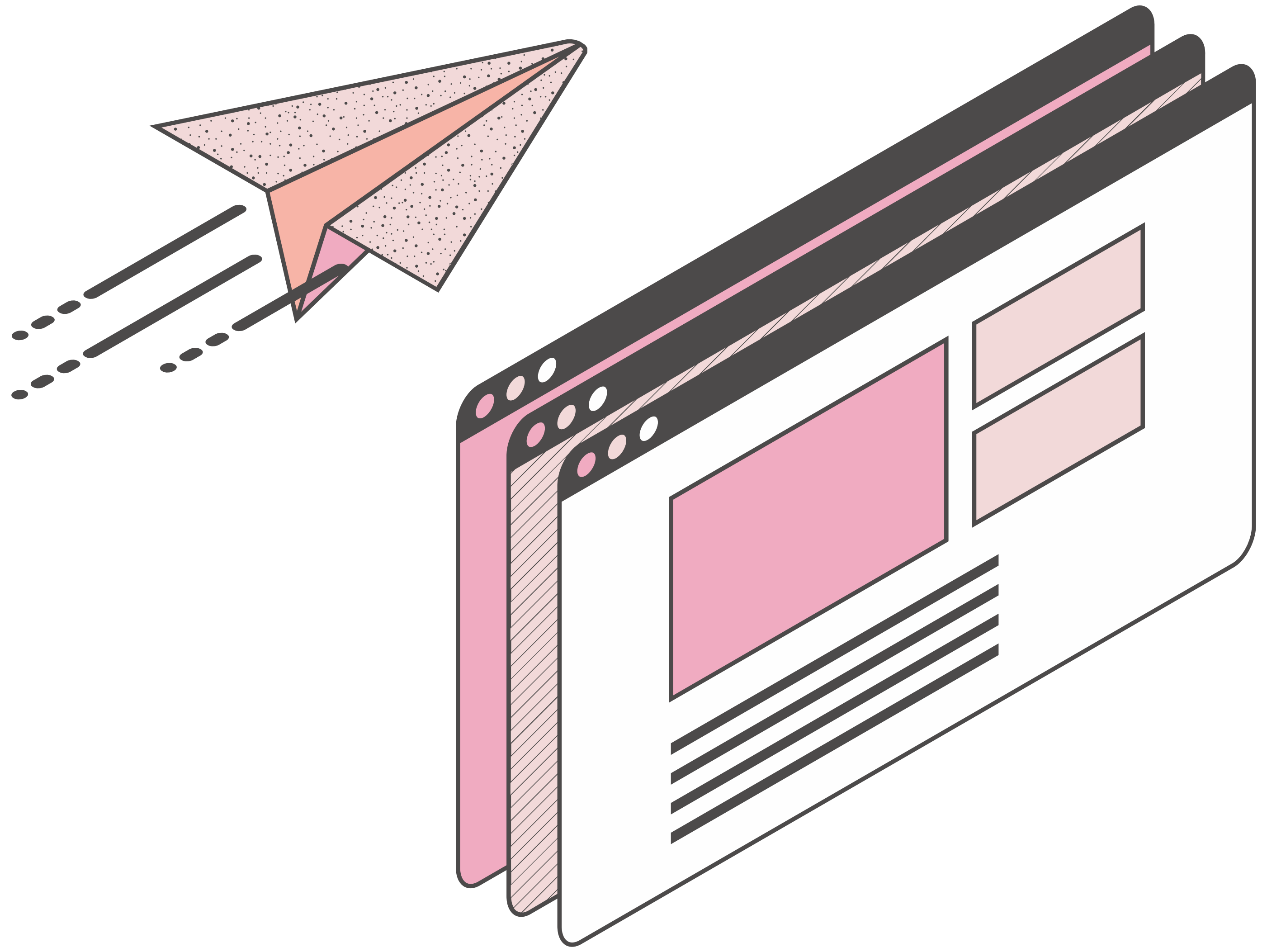
MODULES IMPLEMENTED

- **Node Module:** Creates each node.
- **Circular Linked List Operations Module:** Handles all list operations.
- **GUI Interaction Module:** Connects buttons and backend logic.
- **System Control & Flow Module:** Manages communication between layers.
- **Error Handling Module:** Manages invalid inputs and empty list cases.



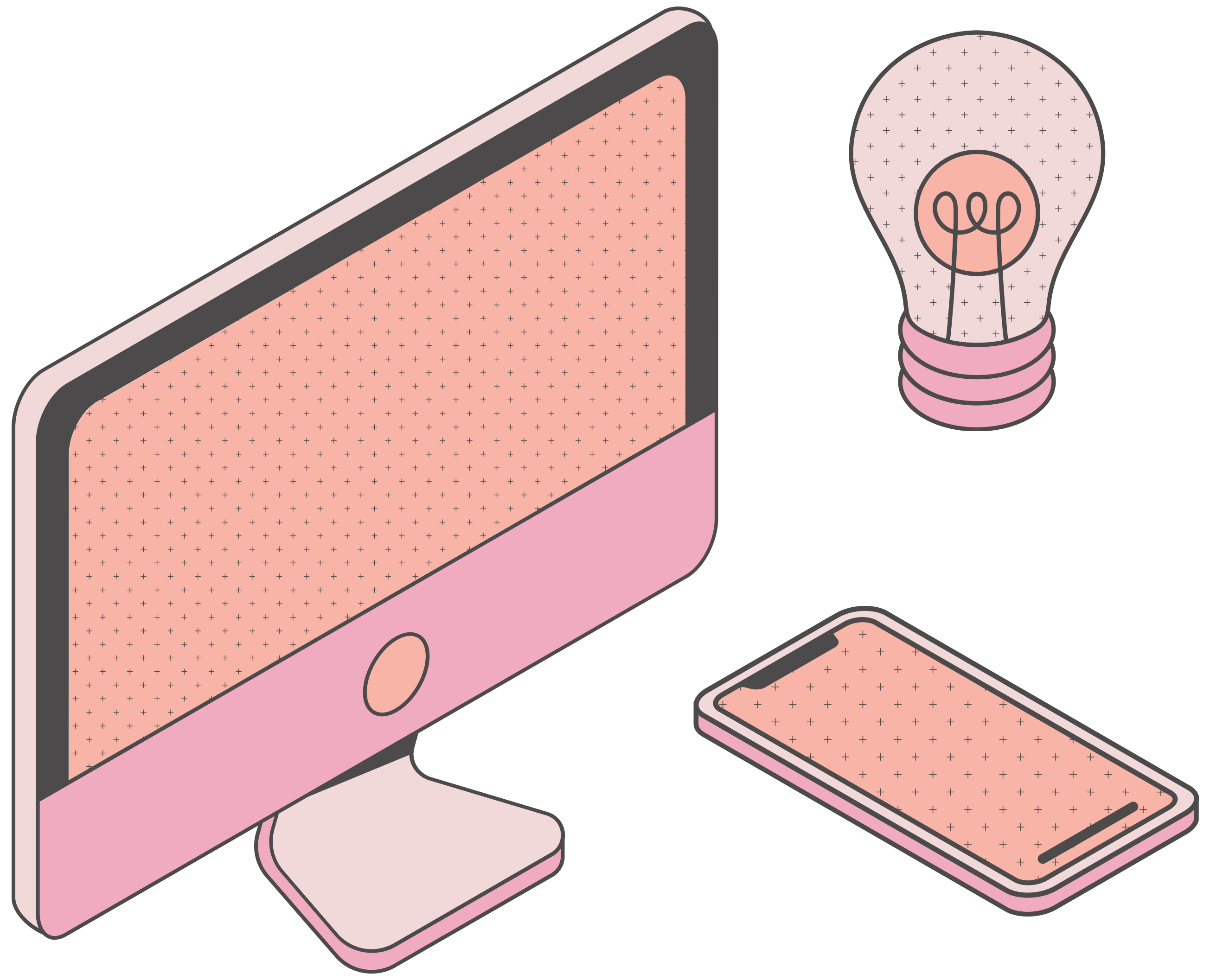
FEATURES IMPLEMENTED

- User-friendly Tkinter GUI.
- Insert front & end operations.
- Delete node by value.
- Search function with message alerts.
- Real-time display of circular structure.
- Reset feature to clear list.
- Handles invalid inputs gracefully.
- Modular and maintainable code.



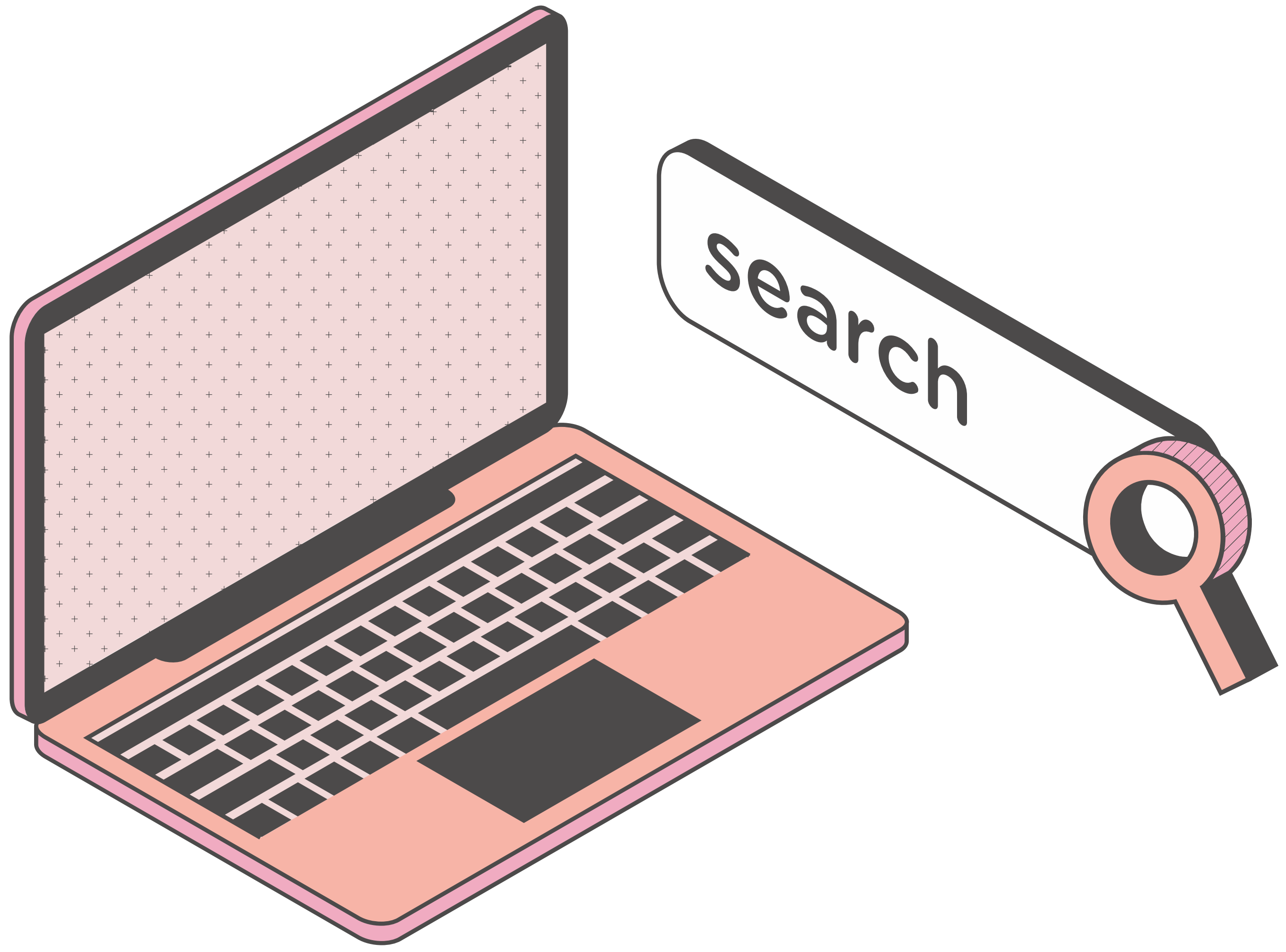
CONCLUSIONS

- Successfully built a circular linked list simulator with GUI.
- Provides clear visualization of pointer updates and structure changes.
- Enhances understanding of data structures for beginners.
- Achieved modular, clean, and maintainable implementation.



FUTURE ENHANCEMENTS

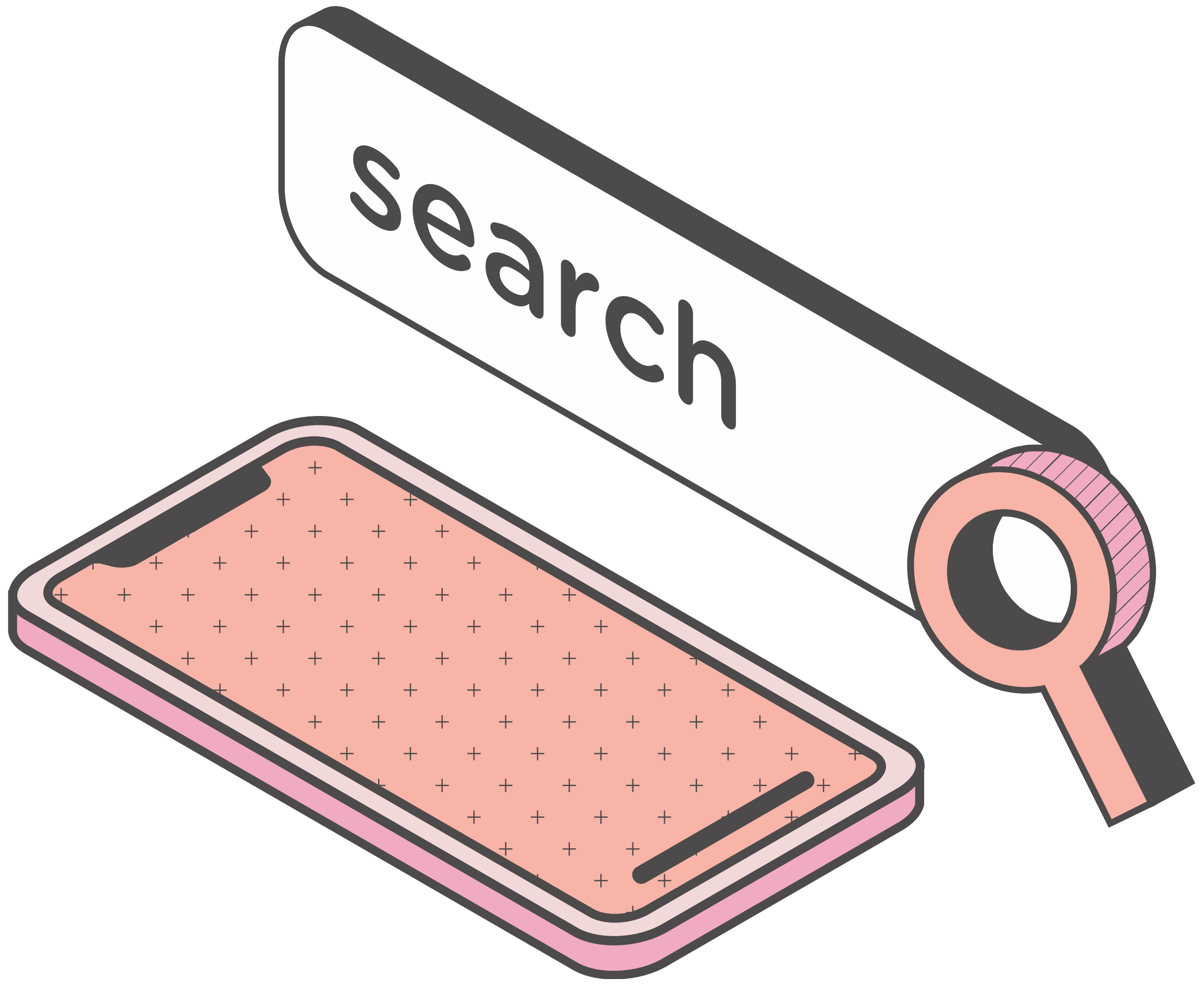
- Graphical visualization of nodes with arrows.
- Step-by-step pointer animation.
- Support for doubly circular linked lists.
- Better UI with themes and highlighting.
- Export/import functionality.
- Web-based version for wider accessibility.



CODING SKILLS - 1

PRESENTATION

THANK YOU FOR YOUR ATTENTION



WWW.REALLYGREATSITE.COM