PROJECT OVERVIEW

Objective

Develop a Singly Linked List in C to manage a dynamic sequence of roll numbers, with functionality to insert and delete at specified positions, and to display the list contents.

Team Collaboration

The team collaborated using GitHub, leveraging version control features like branches, commits, and pull requests to manage contributions effectively.

Visualization

Figma was used to create clear and interactive diagrams representing the internal working of linked list operations for easier understanding.

Documentation

A detailed README file is included, covering:

- Project purpose
- Setup instructions
- Code structure
- Sample input and output
- Linked list behavior explanation

KEY FEATURES

Singly Linked List Operations

The program supports:

- Insertion at a specific position: Add a new roll number at a user-defined index in the list
- Deletion at a specific position: Remove a node from a specified position
- Traversal and Display: Print all roll numbers in order from head to tail

---

SAMPLE OUTPUT

Program execution demonstrates:

Insertion of 1st roll no. : 24 -> NULL
Insertion of 2nd roll no. : 24 -> 29 -> NULL
Insertion of 3rd roll no. : 24 -> 29 -> 39 -> NULL
Insertion of 4th roll no. : 24 -> 29 -> 39 -> 52 -> NULL

GITHUB COLLABORATION

Branching
Each member created a branch named after their roll number (for example, branch-24) to work independently without interfering with the main branch.

Commits
Team members made frequent, descriptive commits to track progress and changes in each function or feature.

Pull Requests
All completed features were merged into the main branch via pull requests, allowing for:

- Peer reviews
- Bug identification
- Code improvement suggestions

Merge Conflict Handling
In case of overlapping edits, the team worked together to resolve merge conflicts to maintain code consistency and stability.

VISUALIZATION

Using Figma, the following operations were diagrammatically represented:

- Insert at Position: Shows how a node is added by updating next pointers
- Delete at Position: Illustrates removal of a node and pointer redirection

- Traversal: Demonstrates step-by-step navigation from head to tail

These diagrams helped team members understand and debug the list behavior more effectively.