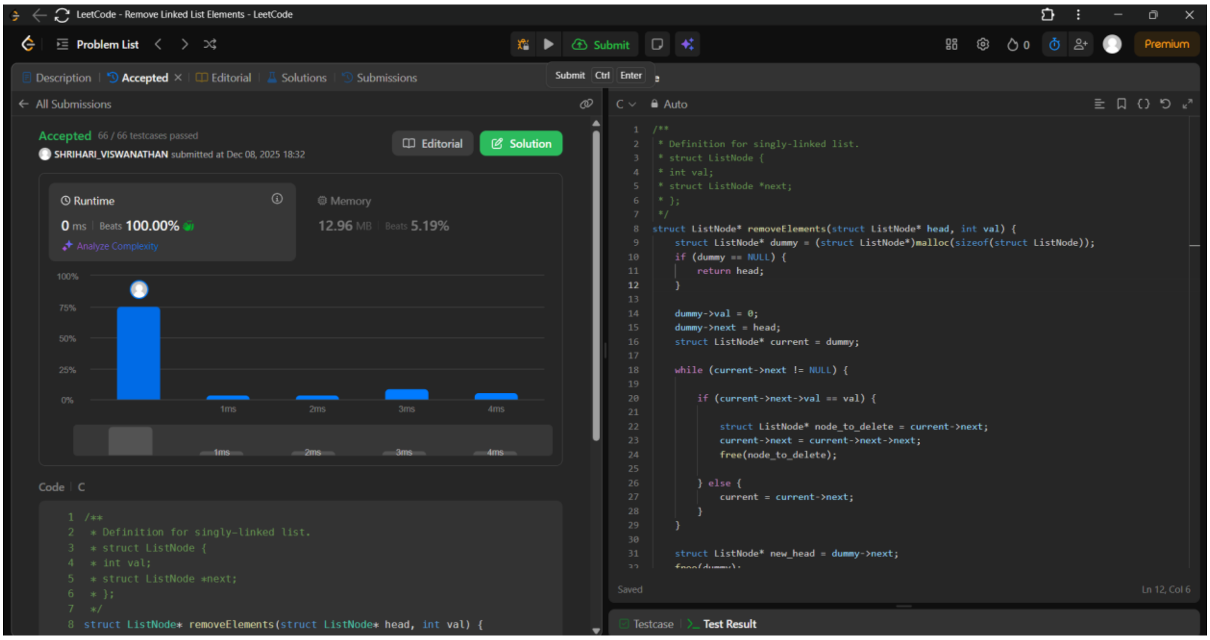
Lab_05_LEETCODE_ 203. Remove Linked List Elements

```
/**
 * Definition for singly-linked list.
 * struct ListNode {
 * int val;
 * struct ListNode *next;
 * };
 */
struct ListNode* removeElements(struct ListNode* head, int
val) {
    struct ListNode* dummy = (struct
ListNode*)malloc(sizeof(struct ListNode));
    if (dummy == NULL) {
        return head;
    }

    dummy->val = 0;
    dummy->next = head;
    struct ListNode* current = dummy;

    while (current->next != NULL) {

        if (current->next->val == val) {

            struct ListNode* node_to_delete = current->next;
            current->next = current->next->next;
            free(node_to_delete);

        } else {
            current = current->next;
        }
    }

    struct ListNode* new_head = dummy->next;
    free(dummy);

    return new_head;
}
```



Accepted 66 / 66 testcases passed

SHRIHARI_VISWANATHAN submitted at Dec 08, 2025 18:32

Runtime

0 ms | Beats 100.00%

Memory

12.96 MB | Beats 5.19%

Code | C

```c
/**
 * Definition for singly-linked list.
 * struct ListNode {
 * int val;
 * struct ListNode *next;
 * };
 */
struct ListNode* removeElements(struct ListNode* head, int val) {
```

```c
/**
 * Definition for singly-linked list.
 * struct ListNode {
 * int val;
 * struct ListNode *next;
 * };
 */
struct ListNode* removeElements(struct ListNode* head, int val) {
    struct ListNode* dummy = (struct ListNode*)malloc(sizeof(struct ListNode));
    if (dummy == NULL) {
        return head;
    }

    dummy->val = 0;
    dummy->next = head;
    struct ListNode* current = dummy;

    while (current->next != NULL) {

        if (current->next->val == val) {

            struct ListNode* node_to_delete = current->next;
            current->next = current->next->next;
            free(node_to_delete);

        } else {
            current = current->next;
        }
    }

    struct ListNode* new_head = dummy->next;
    free(dummy);
```

Saved

Ln 12, Col 6