```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// Define structure for a bus seat
typedef struct Seat {
    int seat_number;
    char passenger_name[50];
    int age; // New age field
    char source[50];
    char destination[50];
    char journey_date[20];
    char bus_type[10];
    struct Seat *next;
} Seat;

// Function to create a new seat node
Seat *createSeat(int seat_number, char *passenger_name, int age, char *source, char
*destination, char *journey_date, char *bus_type) {
    Seat *new_seat = (Seat *)malloc(sizeof(Seat));
    if (new_seat != NULL) {
        new_seat->seat_number = seat_number;
        strcpy(new_seat->passenger_name, passenger_name);
        new_seat->age = age; // Set the age field
        strcpy(new_seat->source, source);
        strcpy(new_seat->destination, destination);
        strcpy(new_seat->journey_date, journey_date);
        strcpy(new_seat->bus_type, bus_type);
        new_seat->next = NULL;
    }
    return new_seat;
}

// Function to display an e-ticket
void displayETicket(Seat *seat) {
    printf("\n--- E-Ticket ---\n");
    printf("Journey Date: %s\n", seat->journey_date);
    printf("Source: %s\n", seat->source);
    printf("Destination: %s\n", seat->destination);
    printf("Seat Number: %d\n", seat->seat_number);
    printf("Passenger Name: %s\n", seat->passenger_name);
    printf("Age: %d\n", seat->age); // Display age
    printf("Bus Type: %s\n", seat->bus_type);
    printf("----------------\n");
}

// Function to book a new seat
void bookSeat(Seat **head, int seat_number, char *passenger_name, int age, char *source,
char *destination, char *journey_date, char *bus_type, int *available_seats) {
    Seat *new_seat = createSeat(seat_number, passenger_name, age, source, destination,
journey_date, bus_type);
    if (new_seat == NULL) {
        printf("Memory allocation failed. Could not book seat.\n");
        return;
    }
    } else {
        Seat *current = *head;
        while (current->next != NULL) {
            current = current->next;
        }
        current->next = new_seat;
    }
    (*available_seats)--;
    printf("Seat %d booked successfully.\n", seat_number);
    displayETicket(new_seat);
}

// Function to cancel a booked seat
void cancelSeat(Seat **head, int seat_number, int *available_seats) {
    Seat *current = *head;
    Seat *prev = NULL;
    while (current != NULL && current->seat_number != seat_number) {
        prev = current;
        current = current->next;
    }
    if (current == NULL) {
        printf("Seat not found.\n");
        return;
    }
    if (prev == NULL) {
        *head = current->next;
    } else {
        prev->next = current->next;
    }
    free(current);
    (*available_seats)++;
    printf("Seat %d canceled successfully.\n", seat_number);
}

// Function to free memory allocated to seat nodes
void freeSeats(Seat *head) {
    Seat *current = head;
    while (current != NULL) {
        Seat *temp = current;
        current = current->next;
        free(temp);
    }
}

int main() {
    // Login credentials
    char username[] = "shree";
    char password[] = "shree123";

    // User input
    char input_username[50];
    char input_password[50];

    printf("Login\n");
    printf("Username: ");
    scanf("%s", input_password);

    // Check login credentials
    if (strcmp(username, input_username) != 0 || strcmp(password, input_password) != 0) {
        printf("Invalid username or password. Exiting program.\n");
        return 0;
    }

    Seat *head = NULL;
    int choice, seat_number, age;
    char passenger_name[50];
    char source[50];
    char destination[50];
    char journey_date[20];
    char bus_type[10];
    int total_seats = 50; // Assuming 50 seats in the bus
    int available_seats = total_seats;

    do {
        printf("\nBus Reservation System\n");
        printf("1. Book a seat\n");
        printf("2. Cancel a seat\n");
        printf("3. Display available seats\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                if (available_seats > 0) {
                    printf("Enter journey date (DD/MM/YYYY): ");
                    scanf("%s", journey_date);
                    printf("Enter source: ");
                    scanf("%s", source);
                    printf("Enter destination: ");
                    scanf("%s", destination);
                    printf("Enter passenger name: ");
                    scanf("%s", passenger_name);
                    printf("Enter age: ");
                    scanf("%d", &age);
                    printf("Enter bus type (AC/Non-AC): ");
                    scanf("%s", bus_type);
                    printf("Enter seat number: ");
                    scanf("%d", &seat_number);
                    if (seat_number < 1 || seat_number > total_seats) {
                        printf("Invalid seat number. Please enter a number between 1 and
%d.\n", total_seats);
                    } else {
                        bookSeat(&head, seat_number, passenger_name, age, source,
destination, journey_date, bus_type, &available_seats);
                    }
                } else {
                    printf("No seats available.\n");
                }
                break;
                scanf("%d", &seat_number);
                if (seat_number < 1 || seat_number > total_seats) {
                    printf("Invalid seat number. Please enter a number between 1 and %d.\n",
total_seats);
                } else {
                    cancelSeat(&head, seat_number, &available_seats);
                }
                break;
case 3:
    printf("Available seats:\n");
    for (int i = 1; i <= total_seats; ++i) {
        Seat *current = head;
        int is_available = 1;
        while (current != NULL) {
            if (current->seat_number == i) {
                printf("Seat %d is unavailable\n", i);
                is_available = 0;
                break;
            }
            current = current->next;
        }
        if (is_available) {
            printf("Seat %d is available\n", i);
        }
    }
    break;
case 4:
                freeSeats(head);
                printf("Exiting program.\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 4);

    return 0;
}
```