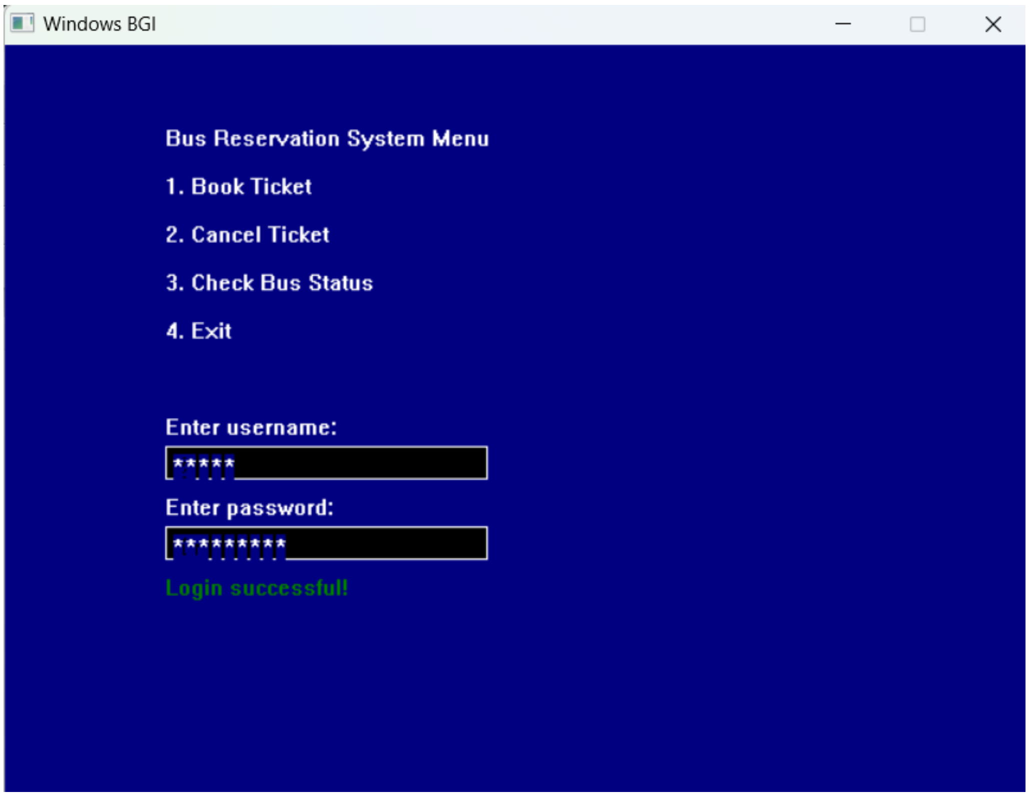
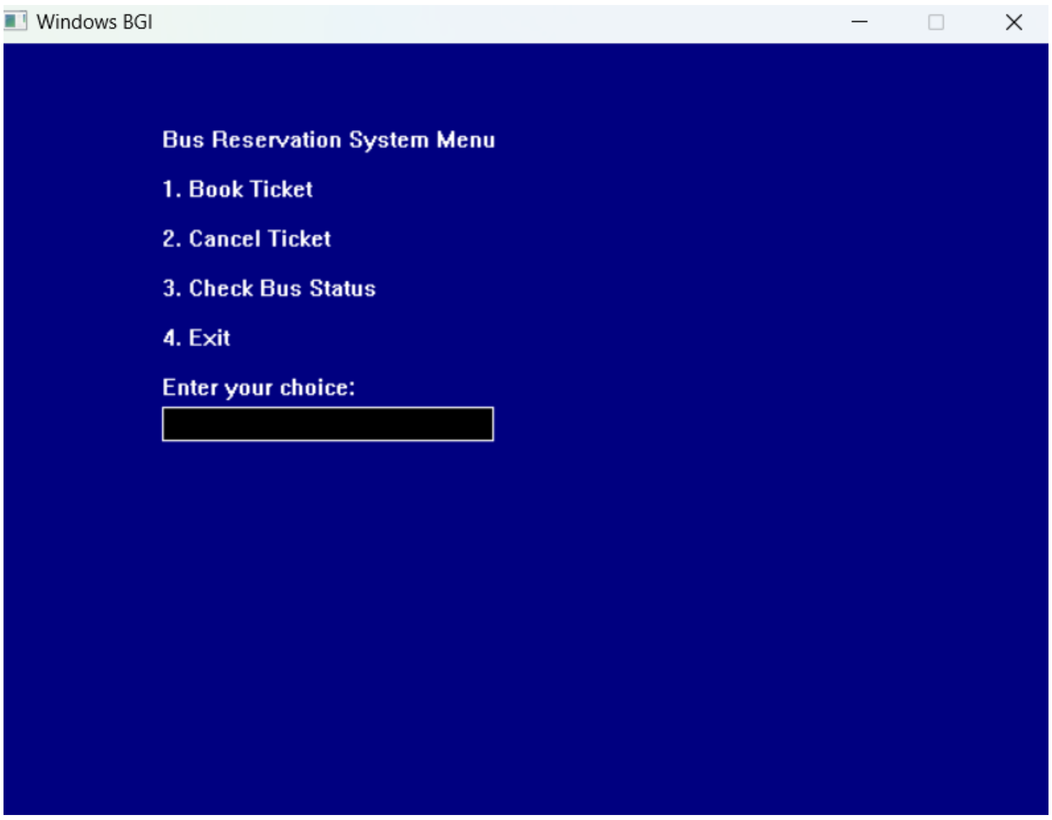
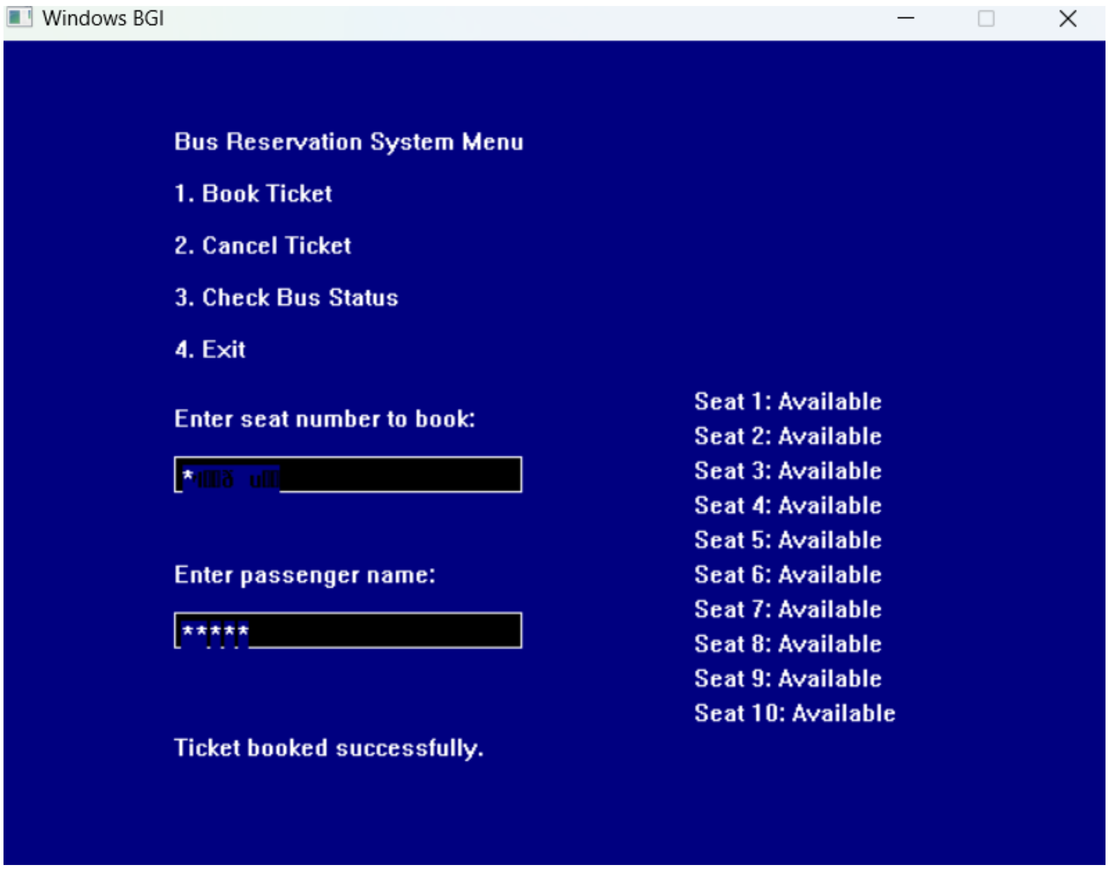
PROJECT NAME: BUS RESERVATION SYSTEM



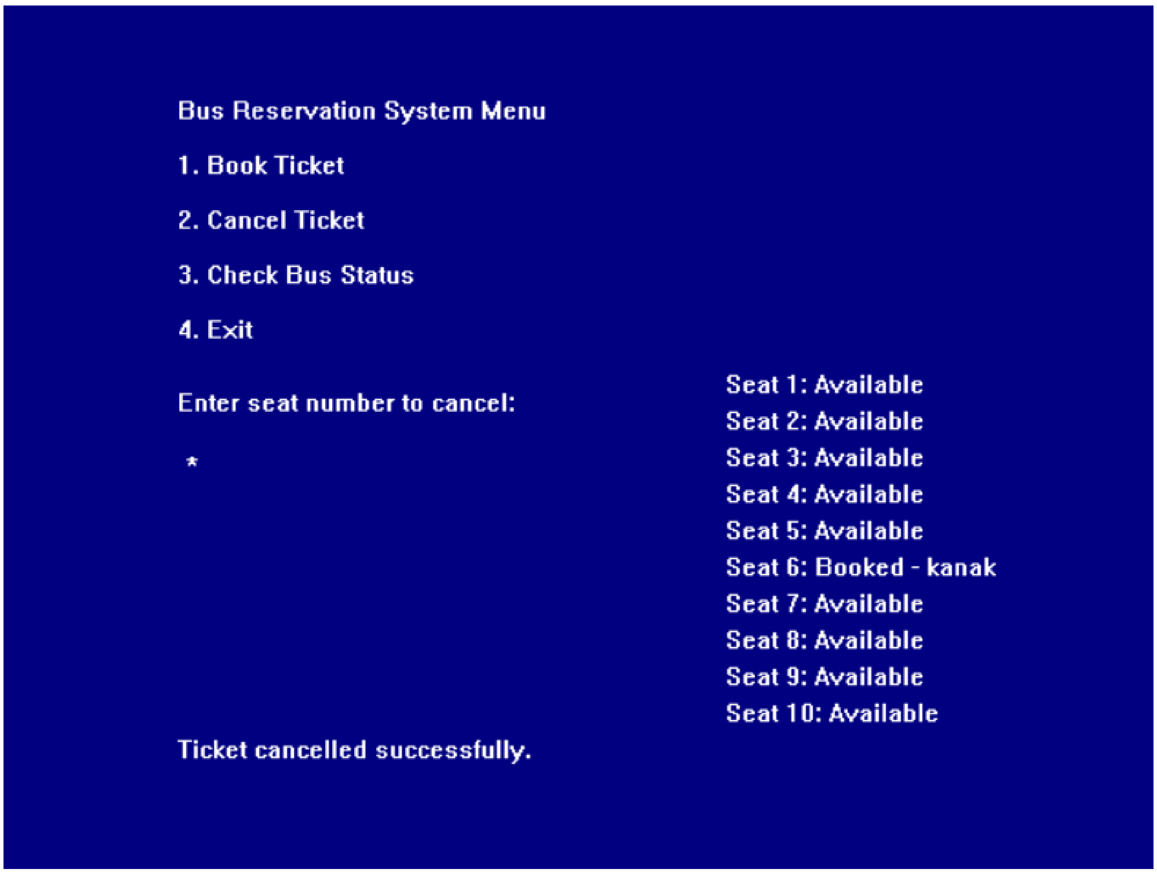
```
Windows BGI

Bus Reservation System Menu
1. Book Ticket
2. Cancel Ticket
3. Check Bus Status
4. Exit

Enter username:
*****
Enter password:
*********
Login successful!
```



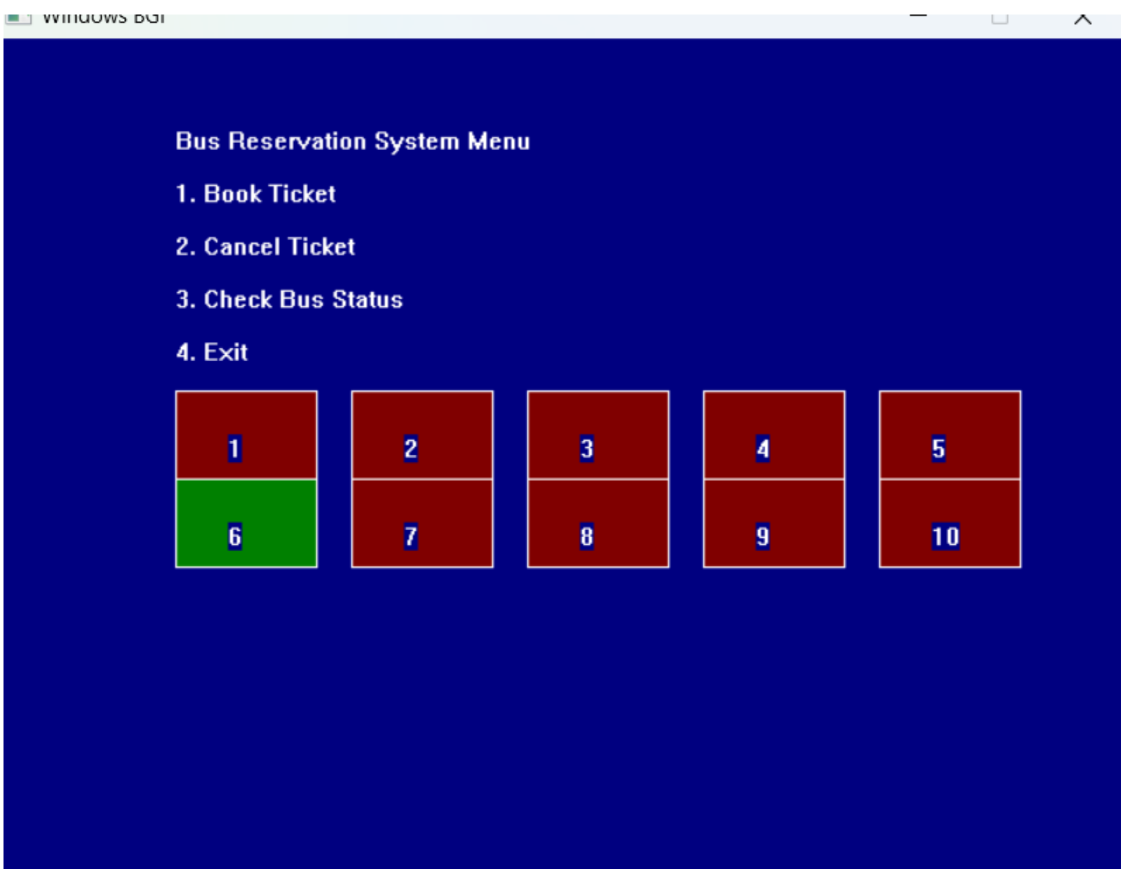
```
Windows BGI

Bus Reservation System Menu
1. Book Ticket
2. Cancel Ticket
3. Check Bus Status
4. Exit
Enter your choice:
```



```
Windows BGI

Bus Reservation System Menu
1. Book Ticket
2. Cancel Ticket
3. Check Bus Status
4. Exit

Enter seat number to book:            Seat 1: Available
                                      Seat 2: Available
*                                     Seat 3: Available
                                      Seat 4: Available
                                      Seat 5: Available
Enter passenger name:                 Seat 6: Available
                                      Seat 7: Available
*****                                 Seat 8: Available
                                      Seat 9: Available
                                      Seat 10: Available
Ticket booked successfully.
```



```
Bus Reservation System Menu
1. Book Ticket
2. Cancel Ticket
3. Check Bus Status
4. Exit

                                      Seat 1: Available
Enter seat number to cancel:          Seat 2: Available
                                      Seat 3: Available
 *                                    Seat 4: Available
                                      Seat 5: Available
                                      Seat 6: Booked - kanak
                                      Seat 7: Available
                                      Seat 8: Available
                                      Seat 9: Available
                                      Seat 10: Available
Ticket cancelled successfully.
```


Windows BGI

Bus Reservation System Menu

1. Book Ticket

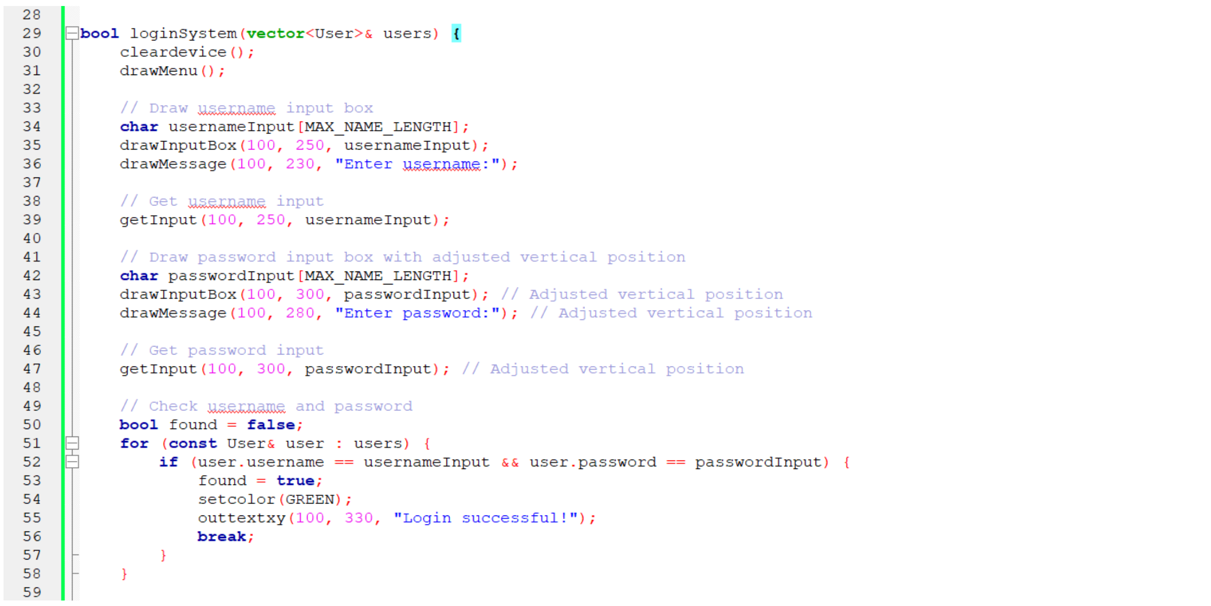
2. Cancel Ticket

3. Check Bus Status

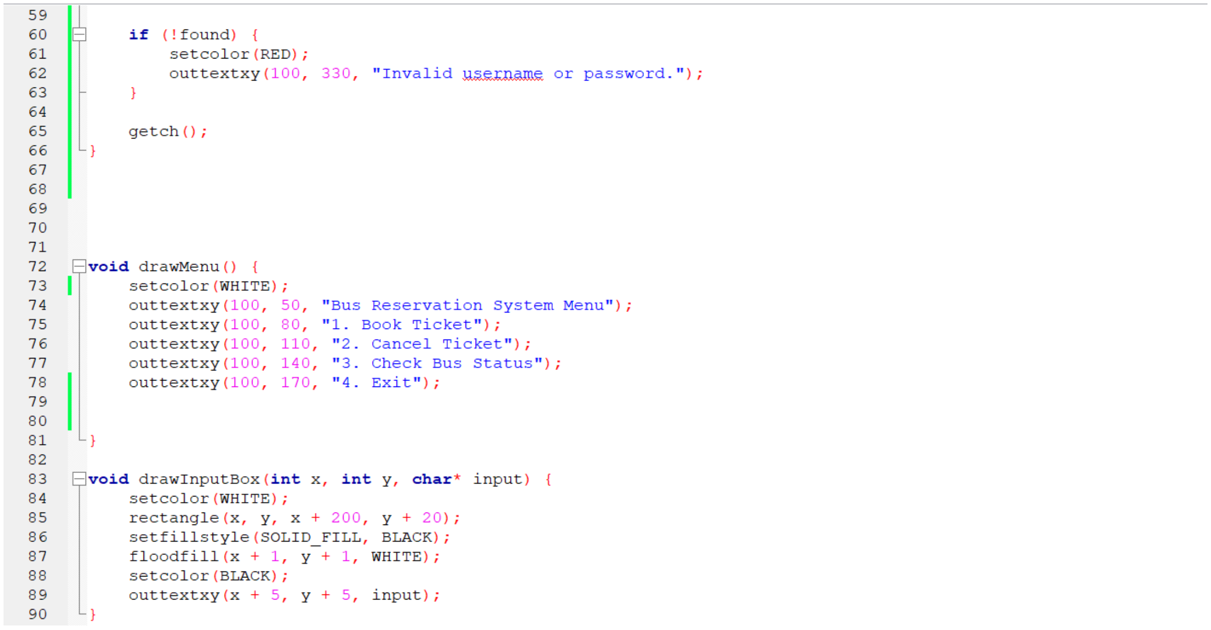
4. Exit

CODE:

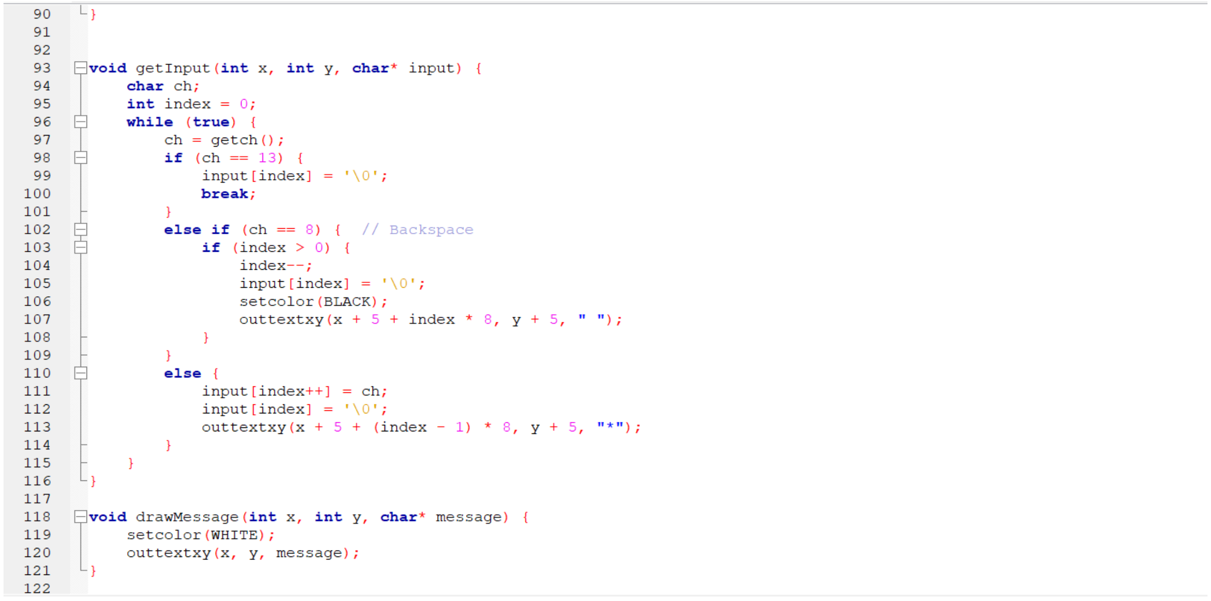
```
#include <iostream>
#include <graphics.h>
#include <cstring>
#include<vector>
#include<conio.h>
using namespace std;
//KANAK MAHALWAL 22BIT0694
const int MAX_SEATS = 10;
const int MAX_NAME_LENGTH = 50;
void drawInputBox(int x, int y, char* input);
void drawMessage(int x, int y, char* message);
void drawMenu();
void getInput(int x, int y, char* input);


struct Ticket {
    int seatNumber;
    char passengerName[MAX_NAME_LENGTH];
    bool booked;
};

// User structure for login
struct User {
    string username;
    string password;
};

```



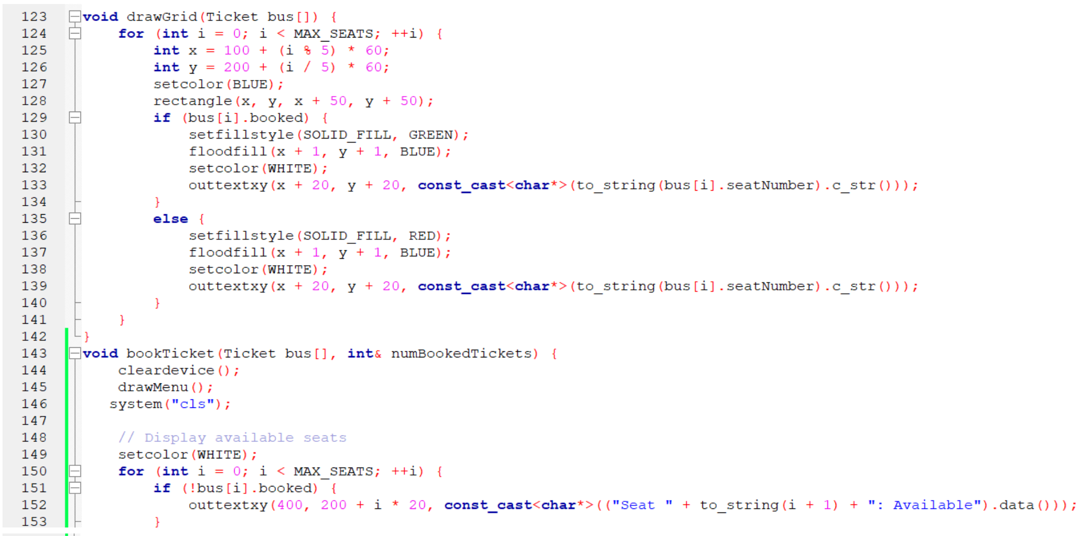
```cpp

bool loginSystem(vector<User>& users) {
    cleardevice();
    drawMenu();

    // Draw username input box
    char usernameInput[MAX_NAME_LENGTH];
    drawInputBox(100, 250, usernameInput);
    drawMessage(100, 230, "Enter username:");

    // Get username input
    getInput(100, 250, usernameInput);

    // Draw password input box with adjusted vertical position
    char passwordInput[MAX_NAME_LENGTH];
    drawInputBox(100, 300, passwordInput); // Adjusted vertical position
    drawMessage(100, 280, "Enter password:"); // Adjusted vertical position

    // Get password input
    getInput(100, 300, passwordInput); // Adjusted vertical position

    // Check username and password
    bool found = false;
    for (const User& user : users) {
        if (user.username == usernameInput && user.password == passwordInput) {
            found = true;
            setcolor(GREEN);
            outtextxy(100, 330, "Login successful!");
            break;
        }
    }

```



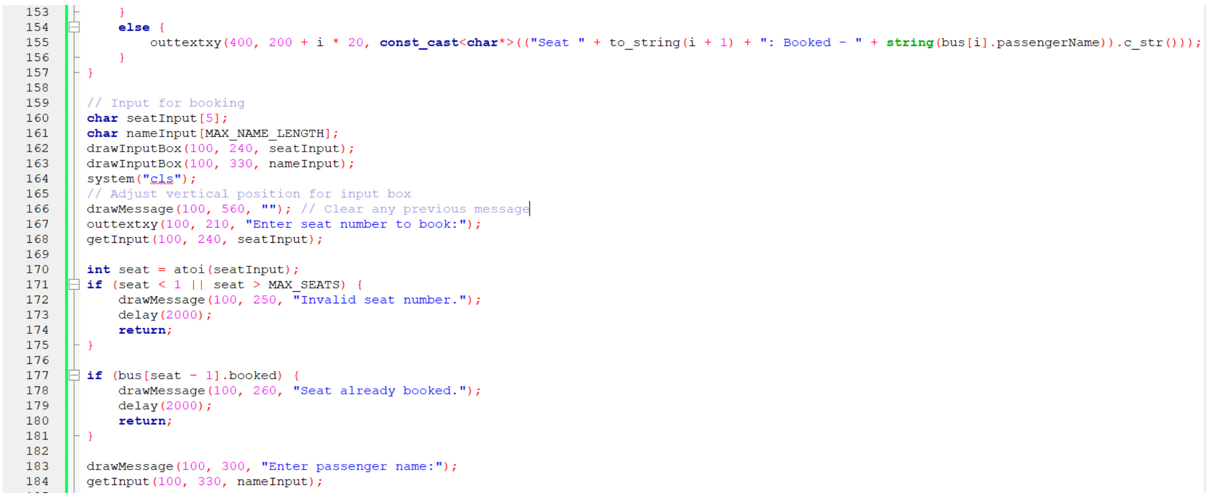
```cpp

    if (!found) {
        setcolor(RED);
        outtextxy(100, 330, "Invalid username or password.");
    }

    getch();
}





void drawMenu() {
    setcolor(WHITE);
    outtextxy(100, 50, "Bus Reservation System Menu");
    outtextxy(100, 80, "1. Book Ticket");
    outtextxy(100, 110, "2. Cancel Ticket");
    outtextxy(100, 140, "3. Check Bus Status");
    outtextxy(100, 170, "4. Exit");


}

void drawInputBox(int x, int y, char* input) {
    setcolor(WHITE);
    rectangle(x, y, x + 200, y + 20);
    setfillstyle(SOLID_FILL, BLACK);
    floodfill(x + 1, y + 1, WHITE);
    setcolor(BLACK);
    outtextxy(x + 5, y + 5, input);
}
```



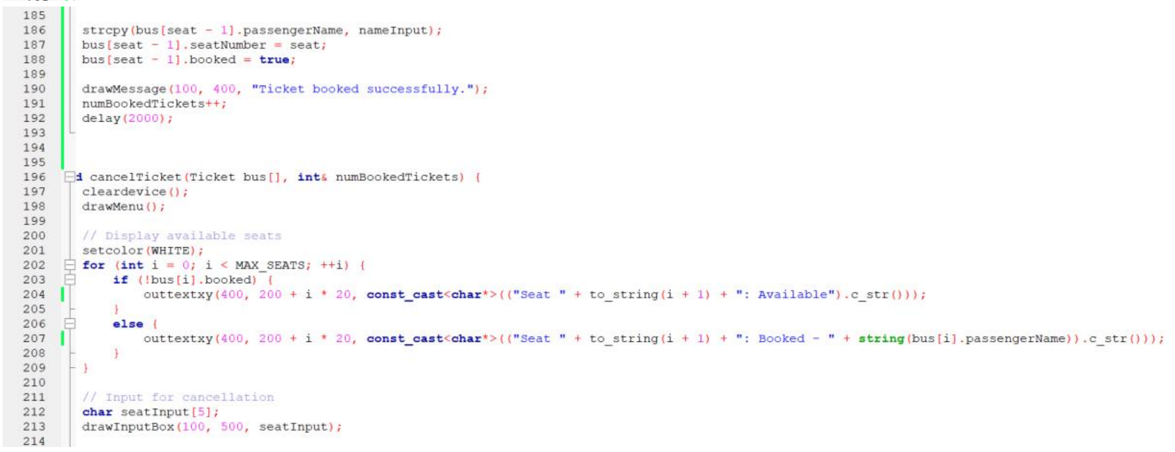
```cpp
}


void getInput(int x, int y, char* input) {
    char ch;
    int index = 0;
    while (true) {
        ch = getch();
        if (ch == 13) {
            input[index] = '\0';
            break;
        }
        else if (ch == 8) {  // Backspace
            if (index > 0) {
                index--;
                input[index] = '\0';
                setcolor(BLACK);
                outtextxy(x + 5 + index * 8, y + 5, " ");
            }
        }
        else {
            input[index++] = ch;
            input[index] = '\0';
            outtextxy(x + 5 + (index - 1) * 8, y + 5, "*");
        }
    }
}

void drawMessage(int x, int y, char* message) {
    setcolor(WHITE);
    outtextxy(x, y, message);
}

```



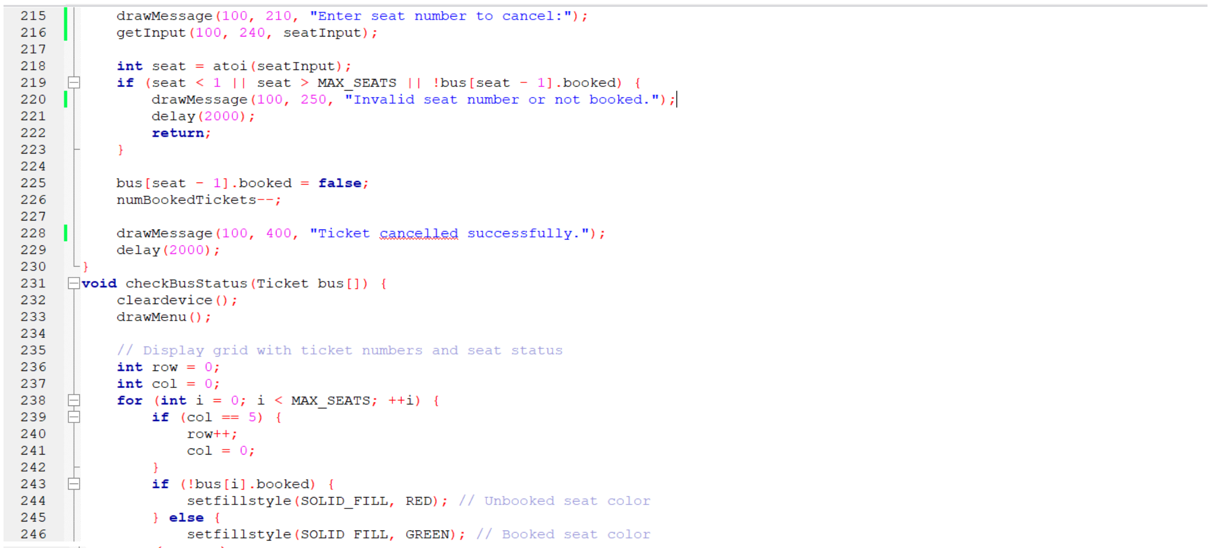
```cpp
void drawGrid(Ticket bus[]) {
    for (int i = 0; i < MAX_SEATS; ++i) {
        int x = 100 + (i % 5) * 60;
        int y = 200 + (i / 5) * 60;
        setcolor(BLUE);
        rectangle(x, y, x + 50, y + 50);
        if (bus[i].booked) {
            setfillstyle(SOLID_FILL, GREEN);
            floodfill(x + 1, y + 1, BLUE);
            setcolor(WHITE);
            outtextxy(x + 20, y + 20, const_cast<char*>(to_string(bus[i].seatNumber).c_str()));
        }
        else {
            setfillstyle(SOLID_FILL, RED);
            floodfill(x + 1, y + 1, BLUE);
            setcolor(WHITE);
            outtextxy(x + 20, y + 20, const_cast<char*>(to_string(bus[i].seatNumber).c_str()));
        }
    }
}
void bookTicket(Ticket bus[], int& numBookedTickets) {
    cleardevice();
    drawMenu();
   system("cls");

    // Display available seats
    setcolor(WHITE);
    for (int i = 0; i < MAX_SEATS; ++i) {
        if (!bus[i].booked) {
            outtextxy(400, 200 + i * 20, const_cast<char*>(("Seat " + to_string(i + 1) + ": Available").data()));
        }
```



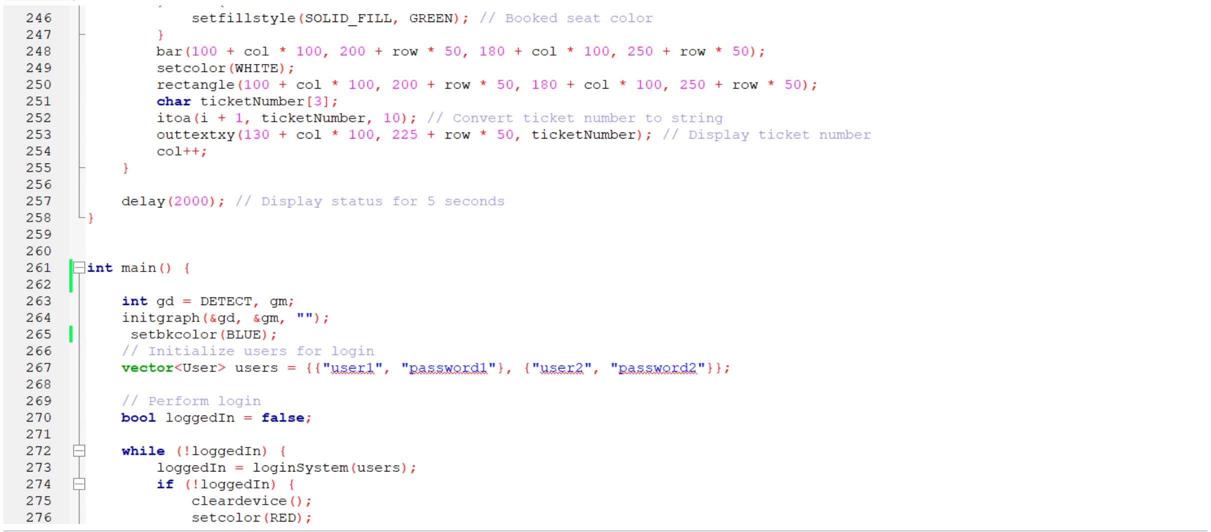
```cpp
        }
        else {
            outtextxy(400, 200 + i * 20, const_cast<char*>(("Seat " + to_string(i + 1) + ": Booked - " + string(bus[i].passengerName)).c_str()));
        }
    }

    // Input for booking
    char seatInput[5];
    char nameInput[MAX_NAME_LENGTH];
    drawInputBox(100, 240, seatInput);
    drawInputBox(100, 330, nameInput);
    system("cls");
    // Adjust vertical position for input box
    drawMessage(100, 560, ""); // Clear any previous message
    outtextxy(100, 210, "Enter seat number to book:");
    getInput(100, 240, seatInput);

    int seat = atoi(seatInput);
    if (seat < 1 || seat > MAX_SEATS) {
        drawMessage(100, 250, "Invalid seat number.");
        delay(2000);
        return;
    }

    if (bus[seat - 1].booked) {
        drawMessage(100, 260, "Seat already booked.");
        delay(2000);
        return;
    }

    drawMessage(100, 300, "Enter passenger name:");
    getInput(100, 330, nameInput);
```



```cpp

    strcpy(bus[seat - 1].passengerName, nameInput);
    bus[seat - 1].seatNumber = seat;
    bus[seat - 1].booked = true;

    drawMessage(100, 400, "Ticket booked successfully.");
    numBookedTickets++;
    delay(2000);



d cancelTicket(Ticket bus[], int& numBookedTickets) {
    cleardevice();
    drawMenu();

    // Display available seats
    setcolor(WHITE);
    for (int i = 0; i < MAX_SEATS; ++i) {
        if (!bus[i].booked) {
            outtextxy(400, 200 + i * 20, const_cast<char*>(("Seat " + to_string(i + 1) + ": Available").c_str()));
        }
        else {
            outtextxy(400, 200 + i * 20, const_cast<char*>(("Seat " + to_string(i + 1) + ": Booked - " + string(bus[i].passengerName)).c_str()));
        }
    }

    // Input for cancellation
    char seatInput[5];
    drawInputBox(100, 500, seatInput);

```

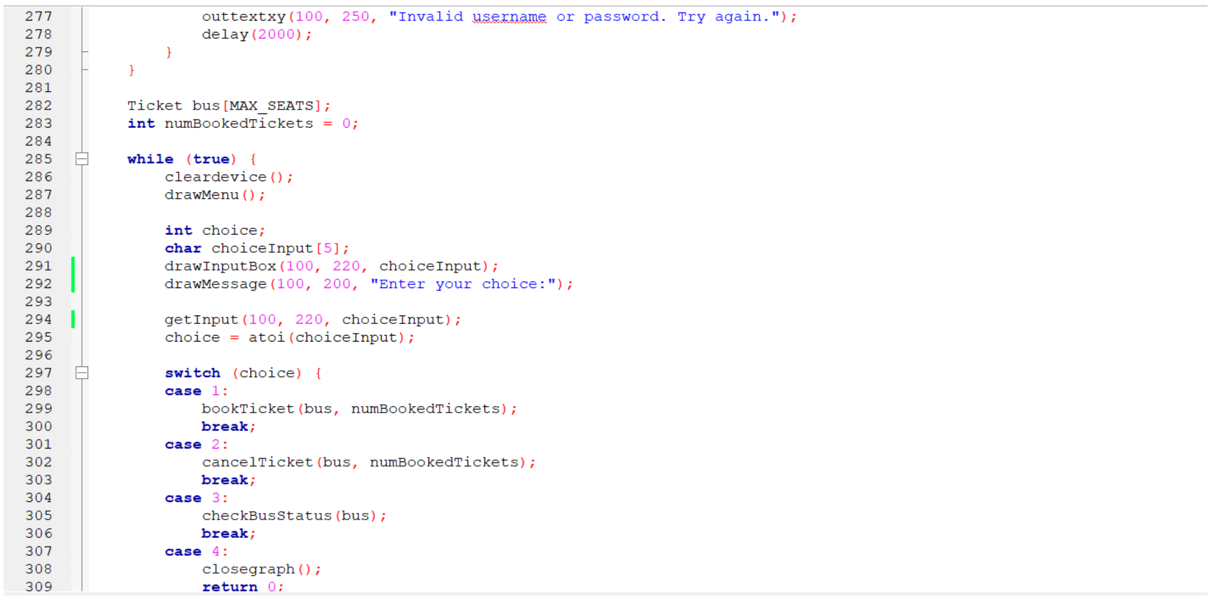


```
    drawMessage(100, 210, "Enter seat number to cancel:");
    getInput(100, 240, seatInput);

    int seat = atoi(seatInput);
    if (seat < 1 || seat > MAX_SEATS || !bus[seat - 1].booked) {
        drawMessage(100, 250, "Invalid seat number or not booked.");
        delay(2000);
        return;
    }

    bus[seat - 1].booked = false;
    numBookedTickets--;

    drawMessage(100, 400, "Ticket cancelled successfully.");
    delay(2000);
}
void checkBusStatus(Ticket bus[]) {
    cleardevice();
    drawMenu();

    // Display grid with ticket numbers and seat status
    int row = 0;
    int col = 0;
    for (int i = 0; i < MAX_SEATS; ++i) {
        if (col == 5) {
            row++;
            col = 0;
        }
        if (!bus[i].booked) {
            setfillstyle(SOLID_FILL, RED); // Unbooked seat color
        } else {
            setfillstyle(SOLID FILL, GREEN); // Booked seat color
```



```
            setfillstyle(SOLID_FILL, GREEN); // Booked seat color
```

```
        }
        bar(100 + col * 100, 200 + row * 50, 180 + col * 100, 250 + row * 50);
        setcolor(WHITE);
        rectangle(100 + col * 100, 200 + row * 50, 180 + col * 100, 250 + row * 50);
        char ticketNumber[3];
        itoa(i + 1, ticketNumber, 10); // Convert ticket number to string
        outtextxy(130 + col * 100, 225 + row * 50, ticketNumber); // Display ticket number
        col++;
    }

    delay(2000); // Display status for 5 seconds
}


int main() {

    int gd = DETECT, gm;
    initgraph(&gd, &gm, "");
     setbkcolor(BLUE);
    // Initialize users for login
    vector<User> users = {{"user1", "password1"}, {"user2", "password2"}};

    // Perform login
    bool loggedIn = false;

    while (!loggedIn) {
        loggedIn = loginSystem(users);
        if (!loggedIn) {
            cleardevice();
            setcolor(RED);
```



```
            outtextxy(100, 250, "Invalid username or password. Try again.");
            delay(2000);
        }
    }

    Ticket bus[MAX_SEATS];
    int numBookedTickets = 0;

    while (true) {
        cleardevice();
        drawMenu();

        int choice;
        char choiceInput[5];
        drawInputBox(100, 220, choiceInput);
        drawMessage(100, 200, "Enter your choice:");

        getInput(100, 220, choiceInput);
        choice = atoi(choiceInput);

        switch (choice) {
        case 1:
            bookTicket(bus, numBookedTickets);
            break;
        case 2:
            cancelTicket(bus, numBookedTickets);
            break;
        case 3:
            checkBusStatus(bus);
            break;
        case 4:
            closegraph();
            return 0;
```

```c
        default:
            drawMessage(100, 300, "Invalid choice. Please try again.");
            delay(2000);
            break;
        }
    }

    closegraph();
    return 0;
}

```