**PROJECT REPORT ON**

Vehicle Parking Management System

Submitted By:

Team Number: 17

Sohan Agashimani, USN: 2JI19CS054

Hundre Aniket, USN: 2JI19CS019

Department of Computer Science & Engineering,

Jain College Of Engineering,

Belagavi.

About:

Our project on vehicle parking management system is a simple CLI application which helps the user to maintain proper data of the vehicles getting parked in his/her parking lot, and gives him/her required well organized data.

Here we have classified vehicles into three types, they are two wheelers, lmvs and hmvs. So the user of this application can do the entry of vehicles as per their category. The user needs to enter the vehicle number of the vehicle getting parked. We have decided predefined amount and parking blocks(10 for each) for each vehicle class. The user also needs to enter the vehicle number at the time of its departure. After doing the entry the user can view/check the status of his/her parking lot, where we provide him/her,

* The total number of vehicles parked.
* Total number of vehicles parked as per their classifications (two wheeler/lmv/hmv).
* Vehicle registration numbers along with the slot numbers they are parked in.
* Generates vehicle number whenever user enters a particular slot number to check.
* Total amount earned.
* Can clear all the data at the end of the day or whenever he/she wishes to.

Approach:

This is a Menu driven CLI app made using C language. We have used the data structures, ***Singly linked lists and Queues*** to develop the program.

Here we have used to three linked lists to store the data(vehicle number) of the respective class of vehicles(two wheeler/lmv/hmv), also we have used to three queues to maintain a slot number for these parking spaces.

The functions that we are using and their use is mentioned below,

* ***append():*** Used to create a node and input the vehicle number into the node.
* ***departure():*** Used when a vehicle is departing from the parking lot. Here we are using another function ***“search()”*** to check the node data with the vehicle number which is going to depart and using the search function we delete the particular node.
* ***checkstatus():*** Used to display all the necessary data to the user.
* ***checkslot():*** Used to display vehicle number present at a particular slot.
* ***clear():*** Used to clear all the data as well all the three linked lists. Here we are using another function ***“clearlist()”*** recursively to delete the nodes of the linked list.
* ***insertq() and deleteq()*** functions to maintain the slot numbers of the parking lot separately for all three types of vehicles(two wheelers/lmv/hmv). Again to clear these queues we use ***“clearlist()”***  function.

**Source code:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#define max 10

typedef struct node

{

    char vehno[20];

    struct node \*next;

    int slot;

}N;

typedef struct queue

{

    int slotno;

    struct queue \*next;

}P;

N \*lmv=NULL,\*hmv=NULL,\*two=NULL;

P \*lrear=NULL,\*lfront=NULL,\*hrear=NULL,\*hfront=NULL,\*trear=NULL,\*tfront=NULL;

int twoc=0,lmvc=0,hmvc=0,totc=0,amount=0;

char no[20];

void insertq(int i,int flag)

{

    P \*temp=(P\*)malloc(sizeof(P));

    temp->next=NULL;

    temp->slotno=i;

    if(flag==1)

    {

        if(tfront==trear && trear==NULL)

            tfront=trear=temp;

        else

        {

            tfront->next=temp;

            tfront=temp;

        }

    }

    else if(flag==2)

    {

        if(lfront==lrear && lrear==NULL)

            lfront=lrear=temp;

        else

        {

            lfront->next=temp;

            lfront=temp;

        }

    }

    else

    {

        if(hfront==hrear && hrear==NULL)

            hfront=hrear=temp;

        else

        {

            hfront->next=temp;

            hfront=temp;

        }

    }

}

int deleteq(int flag)

{

    int x;

    if(flag==1)

    {

        x=trear->slotno;

        P \*temp=trear;

        if(tfront==trear)

            tfront=trear=NULL;

        else

            trear=trear->next;

        free(temp);

    }

    else if(flag==2)

    {

        x=lrear->slotno;

        P \*temp=lrear;

        if(lfront==lrear)

            lfront=lrear=NULL;

        else

            lrear=lrear->next;

        free(temp);

    }

    else

    {

        x=hrear->slotno;

        P \*temp=hrear;

        if(hfront==hrear)

            hfront=hrear=NULL;

        else

            hrear=hrear->next;

        free(temp);

    }

    return x;

}

void append(int ch)

{

    int slot;

    if((ch==1 && twoc<max )||(ch==2 && lmvc<max)||(ch==3 && hmvc<max))

    {

        N \*t=(N\*)malloc(sizeof(N));

        t->next=NULL;

        printf("Enter the vehicle number: ");

        scanf("%s",t->vehno);

        if(ch==1)

        {

            slot=deleteq(1);

            t->slot=slot;

            printf("Park your vehicle at slot number A%d.\n",slot);

            twoc++;

            totc++;

            amount+=20;

            if(!two)

                two=t;

            else

            {

                N\* p = two;

                while(p->next!=NULL){

                    p = p->next;

                }

                p->next = t;

            }

        }

        if(ch==2)

        {

            slot=deleteq(2);

            t->slot=slot;

            printf("Park your vehicle at slot number B%d.\n",slot);

            lmvc++;

            totc++;

            amount+=50;

            if(!lmv)

                lmv=t;

            else

            {

                N\* p = lmv;

                while(p->next!=NULL){

                    p = p->next;

                }

                p->next = t;

            }

        }

        if(ch==3)

        {

            slot=deleteq(3);

            t->slot=slot;

            printf("Park your vehicle at slot number C%d.\n",slot);

            hmvc++;

            totc++;

            amount+=100;

            if(!hmv)

                hmv=t;

            else

            {

                N\* p = hmv;

                while(p->next!=NULL){

                    p = p->next;

                }

                p->next = t;

            }

        }

    }

    else

        printf("Parking is full.\n");

}

void search()

{

    N \*t=two,\*temp=two;

    int slot;

    if(two && !strcmp(no,two->vehno))

    {

        insertq(two->slot,1);

        slot=two->slot;

        two=two->next;

        free(t);

        twoc--;

        totc--;

        printf("Vehicle has departed from slot number A%d.\n",slot);

        return;

    }

    while(temp && temp->next)

    {

        if(strcmp(no,temp->next->vehno)==0)

        {

            t=temp->next;

            insertq(t->slot,1);

            slot=t->slot;

            temp->next=temp->next->next;

            free(t);

            twoc--;

            totc--;

            printf("Vehicle has departed from slot number A%d.\n",slot);

            return;

        }

        temp=temp->next;

    }

    t=lmv;temp=lmv;

    if(lmv && !strcmp(no,lmv->vehno))

    {

        insertq(lmv->slot,2);

        slot=lmv->slot;

        lmv=lmv->next;

        free(t);

        lmvc--;

        totc--;

        printf("Vehicle has departed from slot number B%d.\n",slot);

        return;

    }

    while(temp && temp->next)

    {

        if(strcmp(no,temp->next->vehno)==0)

        {

            t=temp->next;

            insertq(t->slot,2);

            slot=t->slot;

            temp->next=temp->next->next;

            free(t);

            lmvc--;

            totc--;

            printf("Vehicle has departed from slot number B%d.\n",slot);

            return;

        }

        temp=temp->next;

    }

    t=hmv;temp=hmv;

    if(hmv && !strcmp(no,hmv->vehno))

    {

        insertq(hmv->slot,3);

        slot=hmv->slot;

        hmv=hmv->next;

        free(t);

        hmvc--;

        totc--;

        printf("Vehicle has departed from slot number C%d.\n",slot);

        return;

    }

    while(temp && temp->next)

    {

        if(strcmp(no,temp->next->vehno)==0)

        {

            t=temp->next;

            insertq(t->slot,3);

            slot=t->slot;

            temp->next=temp->next->next;

            free(t);

            hmvc--;

            totc--;

            printf("Vehicle has departed from slot number C%d.\n",slot);

            return;

        }

        temp=temp->next;

    }

    printf("Vehicle not found.\n");

}

void display()

{

    if(!two)

        printf("No two wheelers.\n");

    else

    {

        printf("Two wheelers parked in the lot: \n");

        N \*t=two;

        while(t)

        {

            printf("%d.%s  ",t->slot,t->vehno);

            t=t->next;

        }

        printf("\n");

    }

    if(!lmv)

        printf("\nNo lmvs.\n");

    else

    {

        printf("\nlmvs parked in the lot: \n");

        N \*t=lmv;

        while(t)

        {

            printf("%d.%s  ",t->slot,t->vehno);

            t=t->next;

        }

        printf("\n");

    }

    if(!hmv)

        printf("\nNo hmvs.\n");

    else

    {

        printf("\nhmvs parked in the lot: \n");

        N \*t=hmv;

        while(t)

        {

            printf("%d.%s  ",t->slot,t->vehno);

            t=t->next;

        }

        printf("\n");

    }

}

void departure()

{

    if(!two&&!lmv&&!hmv){

        printf("No vehicles in the lot\n");

        return;

    }

    display();

    printf("\nEnter your vehicle number: ");

    scanf("%s",no);

    search();

}

void checkstatus()

{

    printf("Two-Wheeler count: %d\n",twoc);

    if(!two)

        printf("No vehicles.\n");

    else

    {

        printf("Vehicles in the lot: ");

        N \*t=two;

        while(t)

        {

            printf("%d.%s  ",t->slot,t->vehno);

            t=t->next;

        }

        printf("\n");

    }

    printf("\nLMV count: %d\n",lmvc);

    if(!lmv)

        printf("No vehicles.\n");

    else

    {

        printf("Vehicles in the lot: ");

        N \*t=lmv;

        while(t)

        {

            printf("%d.%s  ",t->slot,t->vehno);

            t=t->next;

        }

        printf("\n");

    }

    printf("\nHMV count: %d\n",hmvc);

    if(!hmv)

        printf("No vehicles.\n");

    else

    {

        printf("Vehicles in the lot: ");

        N \*t=hmv;

        while(t)

        {

            printf("%d.%s  ",t->slot,t->vehno);

            t=t->next;

        }

        printf("\n");

    }

    printf("\nTotal count: %d\n",totc);

    printf("Total amount earned: Rs.%d\n\n",amount);

}

N\* clearlist(N \*t)

{

    if(t==0)

        return NULL;

    else

    {

        clearlist(t->next);

        free(t);

        return NULL;

    }

}

P\* clearlistq(P \*t)

{

    if(t==0)

        return NULL;

    else

    {

        clearlistq(t->next);

        free(t);

        return NULL;

    }

}

void clear()

{

    printf("Your data has been reset\n");

    lmvc=hmvc=twoc=totc=amount=0;

    two=clearlist(two);

    lmv=clearlist(lmv);

    hmv=clearlist(hmv);

    trear=clearlistq(trear);

    lrear=clearlistq(lrear);

    hrear=clearlistq(hrear);

    tfront=lfront=hfront=NULL;

    for(int i=1;i<=max;i++)

    {

        insertq(i,1);

        insertq(i,2);

        insertq(i,3);

    }

}

void checkslot()

{

    int flag,found=0;

    int slot;

    N \*temp;

    printf("Enter type of vehicle(1 for two wheeler, 2 for lmv, 3 for hmv): ");

    scanf("%d",&flag);

    printf("Enter slot number: ");

    scanf("%d",&slot);

    if(flag==1)

    {

        temp=two;

        while(temp && !found)

        {

            if(slot==temp->slot)

            {

                found=1;

                printf("Vehicle at that slot is: %s\n",temp->vehno);

            }

            else

                temp=temp->next;

        }

        if(!found)

            printf("Slot is empty.\n");

    }

    else if(flag==2)

    {

        temp=lmv;

        while(temp && !found)

        {

            if(slot==temp->slot)

            {

                found=1;

                printf("Vehicle at that slot is: %s\n",temp->vehno);

            }

            else

                temp=temp->next;

        }

        if(!found)

            printf("Slot is empty.\n");

    }

    else

    {

        temp=hmv;

        while(temp && !found)

        {

            if(slot==temp->slot)

            {

                found=1;

                printf("Vehicle at that slot is: %s\n",temp->vehno);

            }

            else

                temp=temp->next;

        }

        if(!found)

            printf("Slot is empty.\n");

    }

}

int main()

{

    for(int i=1;i<=max;i++)

    {

        insertq(i,1);

        insertq(i,2);

        insertq(i,3);

    }

    while(1)

    {

        printf("---------\n\*\*\*MENU\*\*\*\n1.Two-Wheeler\n2.LMV\n3.HMV\n4.Departure\n5.Check Status\n6.Check Slot\n7.Clear\n8.Exit\n-----------\n");

        char ch;

        scanf("%s",&ch);

        switch(ch)

        {

            case '1':append(1);

                    break;

            case '2':append(2);

                    break;

            case '3':append(3);

                    break;

            case '4':departure();

                    break;

            case '5':checkstatus();

                    break;

            case '6':checkslot();

                    break;

            case '7':clear();

                    break;

            case '8':exit(0);

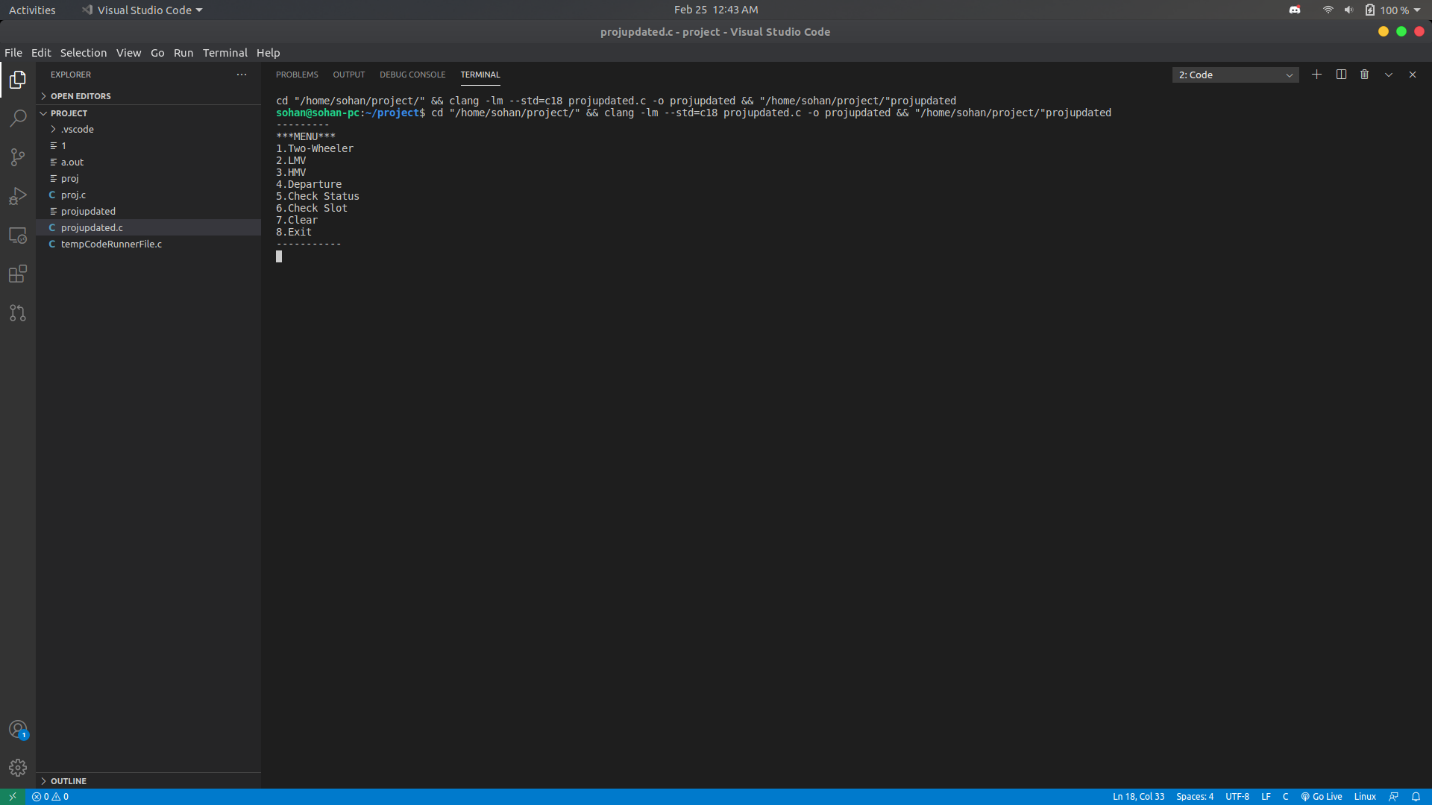
            default:printf("\nInvalid Input\n\n");

        }

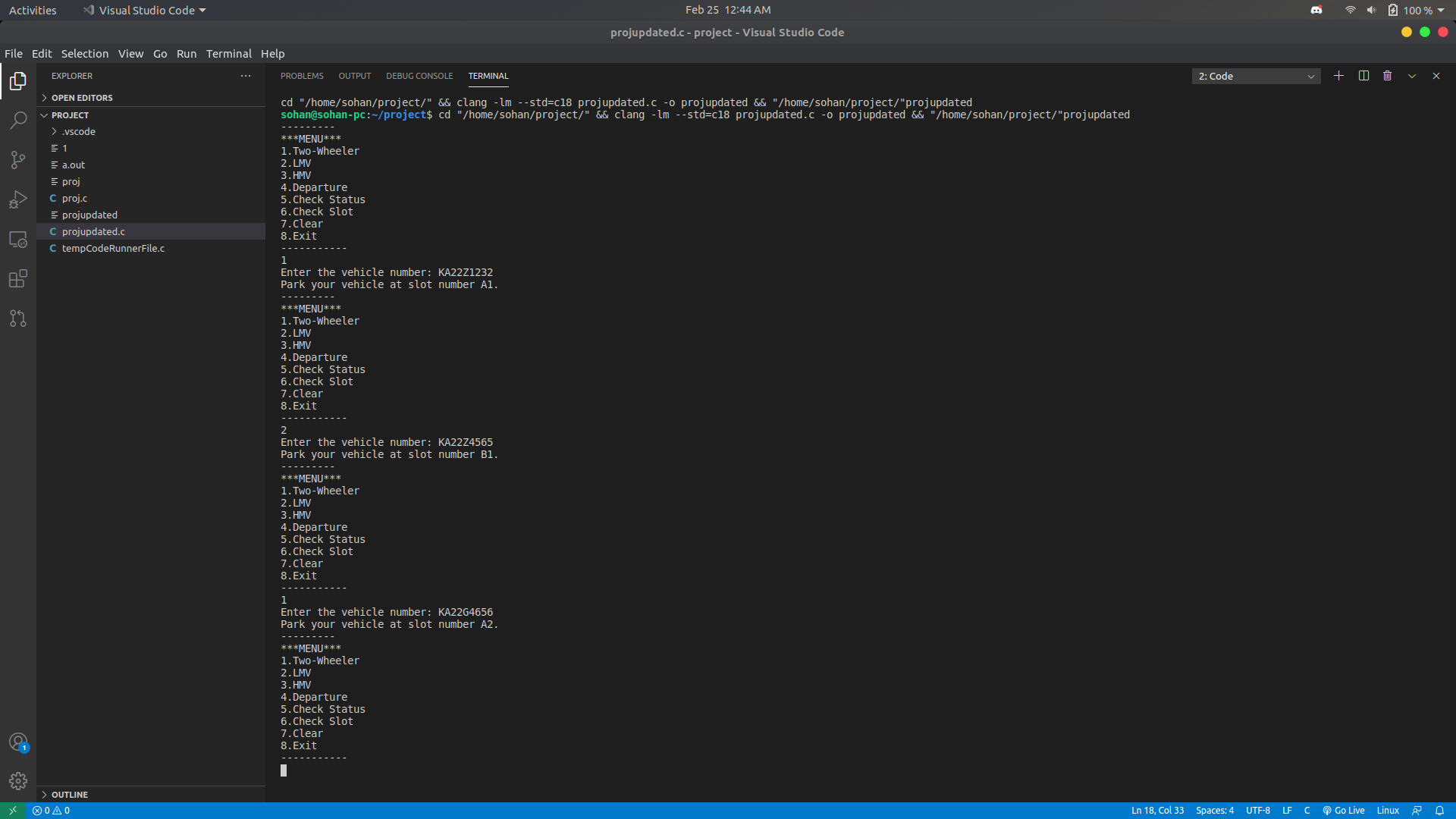
    }

}

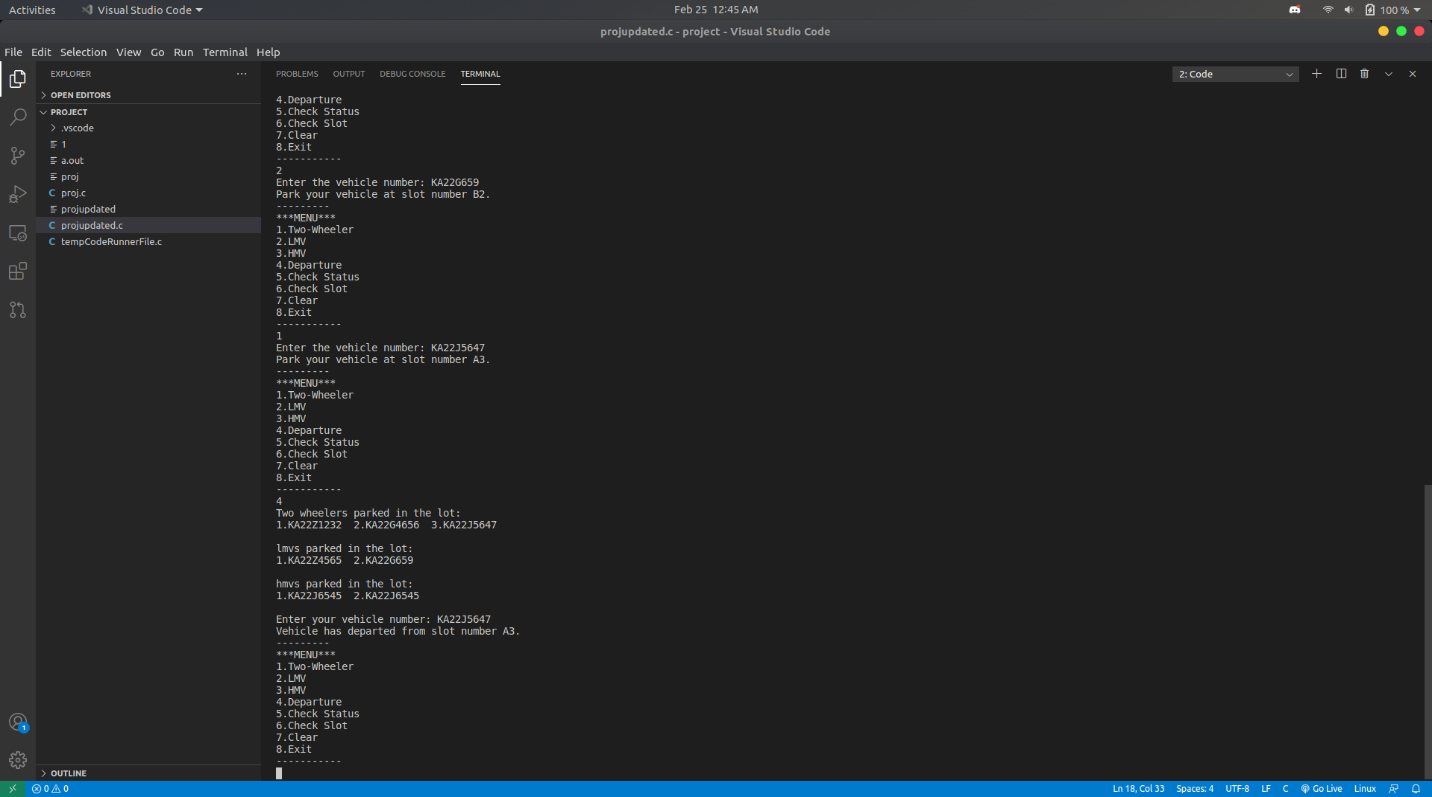
**Output:**

****

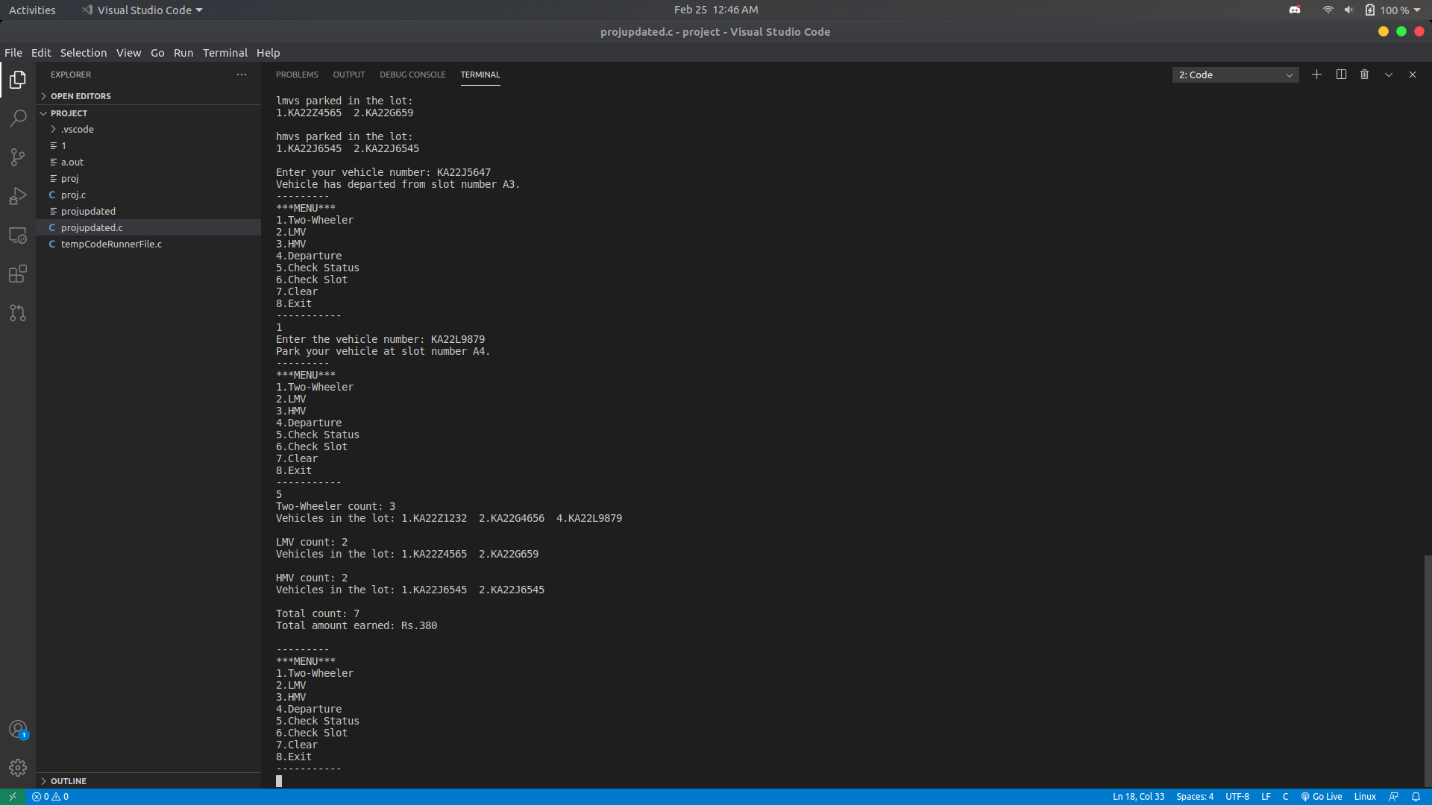
**Vehicle append:**

****

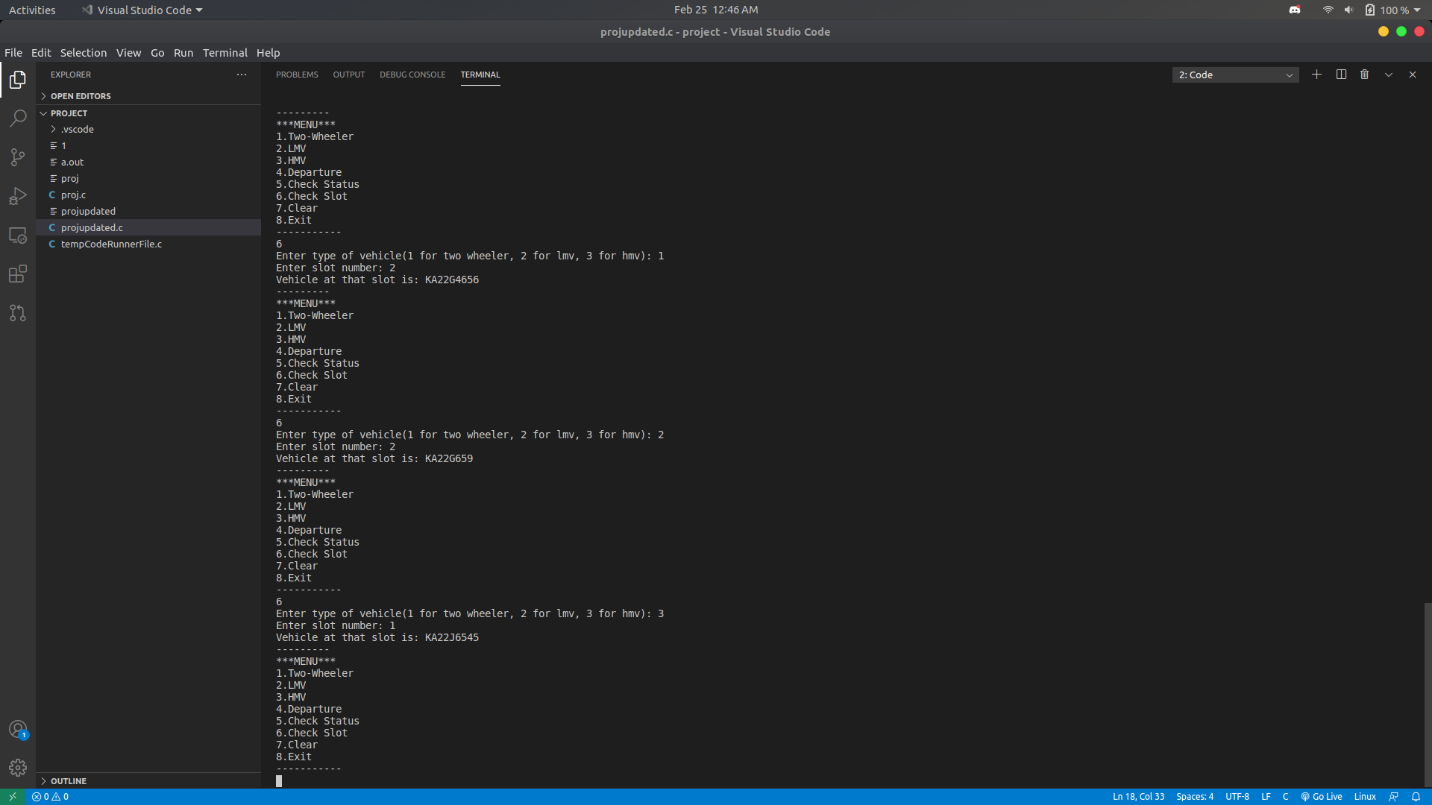
**Vehicle departure:**

****

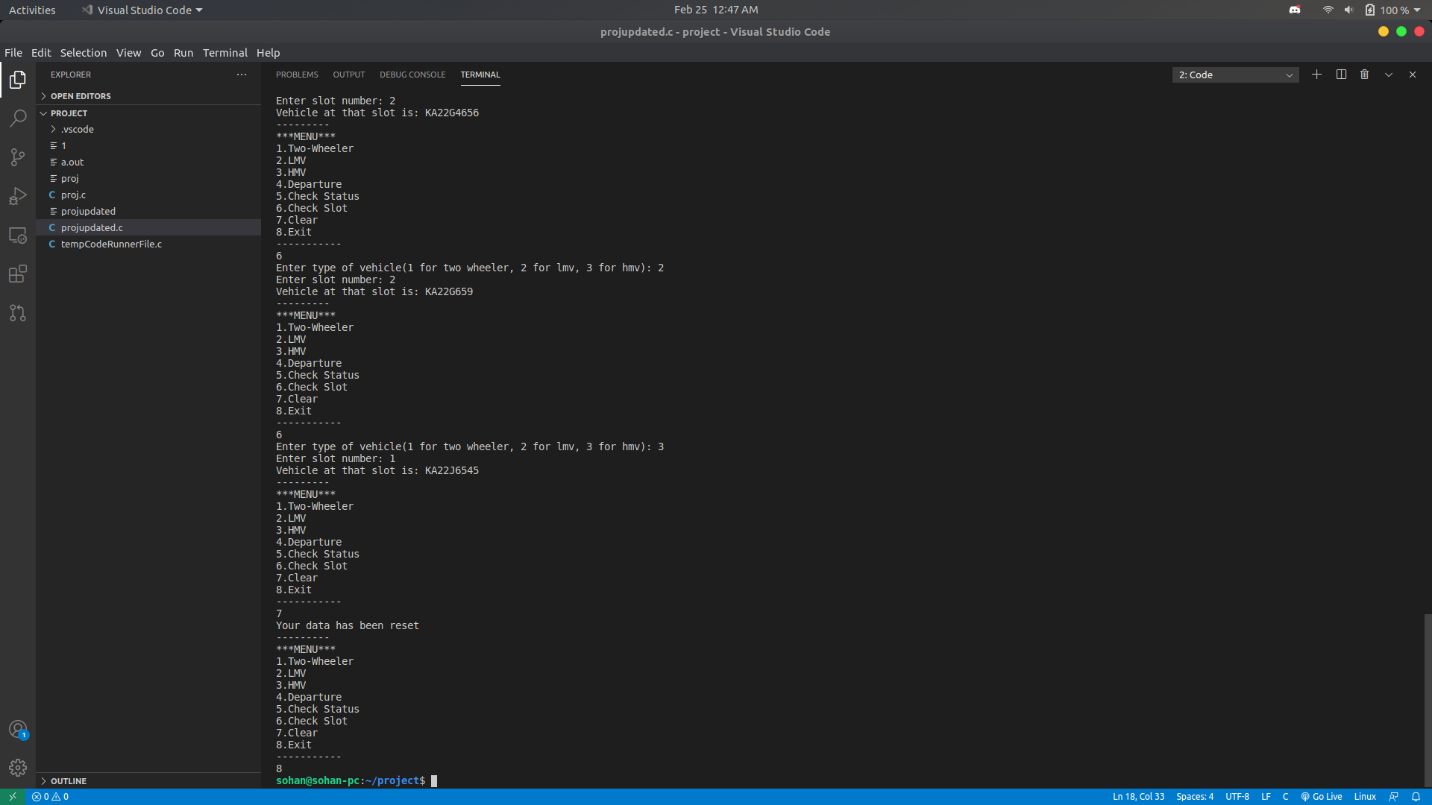
**Status check:**

****

**Slot check:**

****

**Clearing data:**

****

**THANK YOU!**