RAILWAY ONLINE BOOKING SYSTEM DESIGN AND IMPLEMENTATION

GROUP MEMBERS-

-M S Sharath Chandra

19BCE0316

L11+L12

-A Madhughnea Sai

19BCE0314

L43+L44

-Aayush Anand

19BCE2260

L43+L44

ABSTRACT-

In this paper, we find out more about the usefulness of computerized passenger reservation system and introduce one approach to online transaction and usefulness in rough sets. Comparing with the method to get most interesting methods, ours is direct and objective. Interest must consider the predefined knowledge on what kind of information is interesting. Our method greatly reduces the rule numbers generated and provides a measure of its usefulness at the same time. Apart from the obvious advantage of being computerized over manual reservation and enquiry, this paper explores other advantage of implementing this system and its future.

If this project is successfully applied successfully by the INDIAN RAILWAY CATERING AND TOURISM CORPORATION then it will make ticket booking and reservation Simpler and Efficient.

AIM-

Railway Ticket Reservation using online ticketing system. This system helps in collecting passenger data, searching for destinations, fare display and adding new trains and editing old routes and payment methods.

OBJECTIVE-

Our objective is to Link passengers information to their desired seats and trains and make ticket booking and reservation a simpler task.

APPLICABILITY-

The railway lines and passengers have been increasing year by year in the country. With such a huge customer base, buying train tickets problem has been very prominent. The electronic commerce could solve the problem of railway ticketing. Introduced a new online ticketing system is not only technological innovation, but also will improve the railway services, to a certain extent, solve the difficult problem of railway ticketing.

INTRODUCTION-

The seat/ berth reservation on trains is pretty complex activity, not only because of the volumes involving seats/ berth reservations per day, but also because of several different categories of trains that are operating. This method of calculation of fare is also quite complex as charges are based on distance, comfort level and transit time. Also, there were many infirmities with the manual system like the current status did not get updated, it was slow and time consuming, inadvertent errors and malpractices in reservation were there. Because of the complexity and sheer volumes involved, there was a need for development of computerized reservation.

LITERATURE REVIEW-

The Indian railway website, IRCTC is the most traffic-filled site. The tickets are usually booked through this official site. If you are planning to book tickets the day before your departure date, tatkal tickets are available from 10:00 AM and ends as soon as the tickets are booked. The tickets are limited and the demand will be sky-high. If you are able to reserve a ticket through tatkal, you can get berth seats. Cancellation of tatkal ticket will not be refunded. Anyone can reserve tickets through IRCTC. You can pay through debit card, credit card, international card and others.

On the other hand, you can choose tour portals to reserve tickets. The sites will be faster and user-friendly. Not all trains will be displayed in such portals and a small fee will be levied as service charge. A foreigner can also book tickets through an account that will be verified through and Indian mobile number or by mailing passport details to IRCTC. Foreigners can easily book train tickets through tour portals.

LIST OF MODULES-

1. PASSENGER DATA COLLECTION AND STORAGE
2. SEARCHING FOR DESTINATIONS AND DISPLAY OF FARE
3. ADDING NEW TRAINS AND EDITING OLD ROUTES
4. SEARCHING BOOKING AND CANCELLATION
5. DISPLAYING BILLS, PAYMENT OPTIONS AND FINAL OUTPUT

Implementation-

The project is divided into 5 modules and each module is significantly important for the successful application of the project. The project has two parts one for the user and other for the admin.

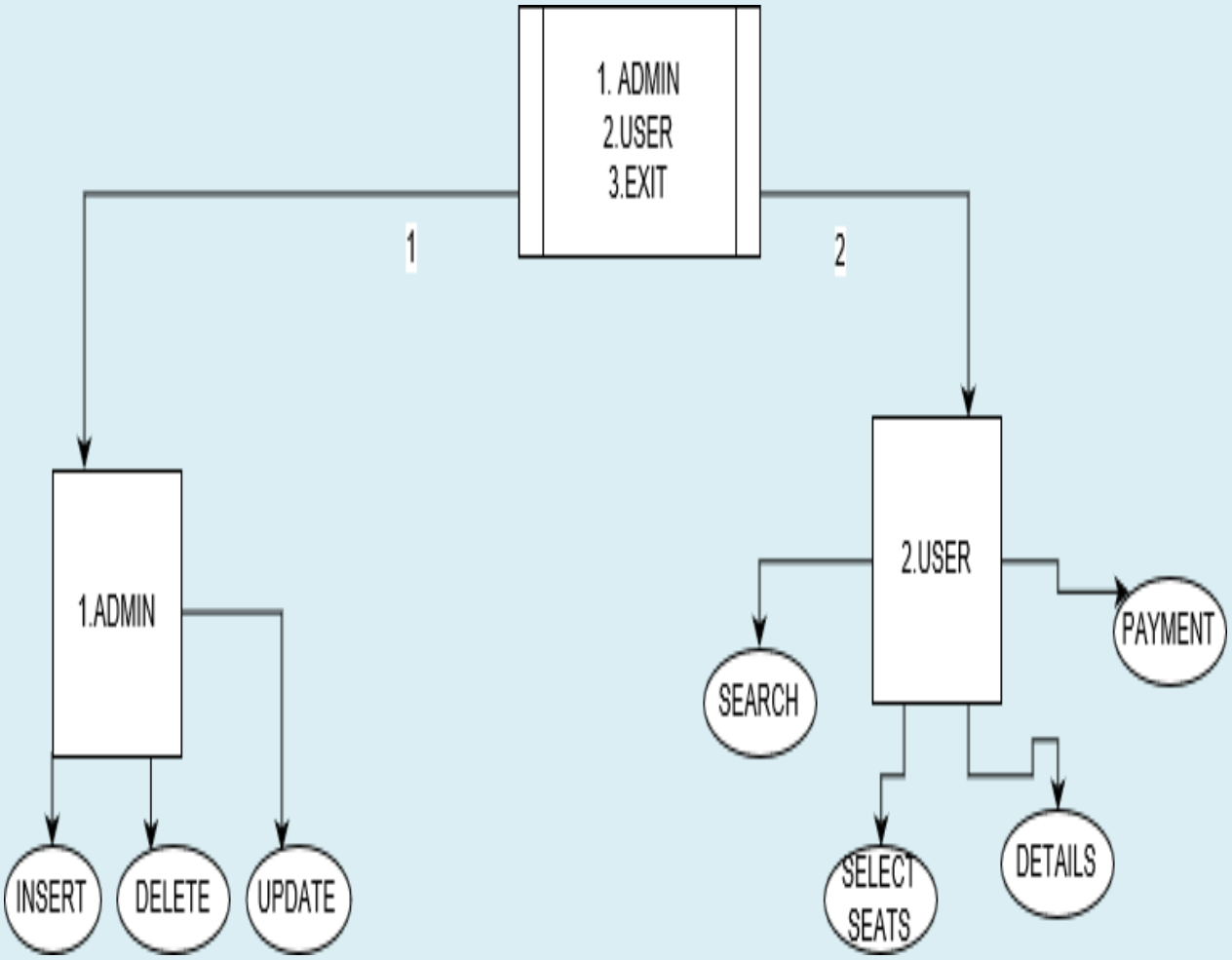
The user part allows the passenger to book seats via the use of all the modules and complete the reservation process by doing the payment.

The admin part allows the admin to add and delete trains. It allows addition of new trains, editing routes and discarding old trains. But the admin part can only be accessed by the authorities by entering the password.

The user part begins by allowing the user to search for a train, after searching the train the available trains are displayed and the user can select any one of the trains to check for availability of seats in that train. After the user selects the preferred train then he is directed to the payments page where he can choose the payment method and complete the method by providing the details of payment. Then a message for completion of booking is displayed.

The admin part contains the addition and deletion of trains. The admin can search for a train and as per his requirements he can delete the train or update its route. He can also add new trains to the directory.

ARCHITECTURE DIAGRAM-



MODULE DESCRIPTION-

Module 1: Passenger data collection and storage

This module deals with the collection and storage of Passenger information. To do this we are using a structure named as ‘passenger’, in that struct we are initializing name, age, date of journey, starting and destination stations with their respective data types. We also made a stack data structure by defining a push function which initializes a variable in structure and takes the respective inputs i.e. the passengers name, age, starting and destination stations and adds the address of that variable to the stack. To view the previous data entry the user can use display command to view the data. The next modules will be able to use the address of the current passenger by using the pop function on the stack. Each passenger’s information is stored and can be retrieved any time by just entering its reservation number or by searching the name of passenger by using the search module. The use of pointers in the code of the module helps other functions in different modules to easily access the passenger info for further use.

Module 2: Searching for destinations and display of fare

My module is based on searching for the destinations on which it will show us the distance and number of trains available. A stack is created which is to search the names of cities. Another two stacks store the coordinates of the cities, one for the x coordinate and one for the y coordinate. A function is created which retains the indices for accessing the stacks. Another function calls the previous function and stores the returned value. The value inputted by the user for departure and arrival location is taken. Available trains will also be displayed with the date and time. After displaying the distance of the trains from the user's departure till arrival will be displayed so that they can choose from it. Example,

Departure location – Mumbai

Arrival location - Delhi

First the function mentioned previously is called in a loop that executes till the city inputted for departure matches the cities in the stack. This is done in the method mentioned later as the method called only returns the index. The x and y coordinates for the city found is accessed from the returned index and stored in the respective variable. The same is done for the arrival location. The distance between the two cities is found by using the coordinates stored in the location by the distance formula.

Distance = Square root ((x1 – x2) ^2 +(y1-y2) ^2)

Module 3: Adding new trains and editing old routes

My module can only be used by the authorities. This module will give the railway authority to add new trains to the system, it will also allow the authority to change the routes of old trains and even change the details regarding the train like number of seats, timings and prices, and it will also allow the authority to remove the trains that are no more functioning so that the passengers are not confused due to availability of wrong trains. The authority will have a fixed password that will only allow them to access this module and make the required changes so that no other person can disrupt the system. Basically, the work of this module is to give the authority the power to update the railway system with the latest changes and keep the system up to date for the convenience of the users.

Module 4: Searching booking and cancelation

The types of functions that are done are as follows: customer register function, customer cancellation function, searching function, booking function. Functional blocks of the system Customer register function is Customers could order tickets after they login on the system. The registration information includes user's name, user's telephone, user's address. According to the customer's cancellation request, the system could complete the cancelling operation, delete the user's registration information from the table in the database. Searching function: Customers can use the train number or site to site to query. When they type the information, the system will return the relevant information to the customers,

including the type of train, the type of the tickets, the number of remaining tickets, the price of tickets. Booking function: Customers could order tickets after they login on the system. They could select the Train Number, sites interval of trips, the number of train tickets. Refunding function: Customers may refund their tickets for some reasons. The system checks tickets information to determine whether to allow refunding tickets, then returns the process result.

Module 5: Displaying bill, payment options and final output.

In my module, I have displayed the bill and given an option for cancellation. The module which takes the input for the details is called and the passenger information is displayed as a bill e.g.: name, age, phone number which is stored in a structure and is easily accessible. Then the module that selects the train info is called and the train details such as arrival, date, departure etc are displayed by using the display function to display the elements of the stack. The duration of the journey along with the cost of the ticket will be displayed. An option will be given to the customer to confirm the ticket, or the ticket can be cancelled by the customer if he wants to. If he cancels the program will end and the bill will be displayed as void ad the program will end. If the customer confirms his ticket, the bill will be displayed and the payment options will be displayed, after the user chooses an option, the booking will be done, if the payment is successful and the program ends.

TIME COMPLEXITY-

Time Complexity of Linked lists (singly-linked)

A linked list can typically only be accessed via its head node. From there you can only traverse from node to node until you reach the node you seek. Thus **access is O(n)**.
Searching for a given value in a linked list similarly requires traversing all the elements until you find that value.
Thus **search is O(n)**.

```
currNode = head

forever:

if currNode == NULL
     break

cur = cur->Next
```

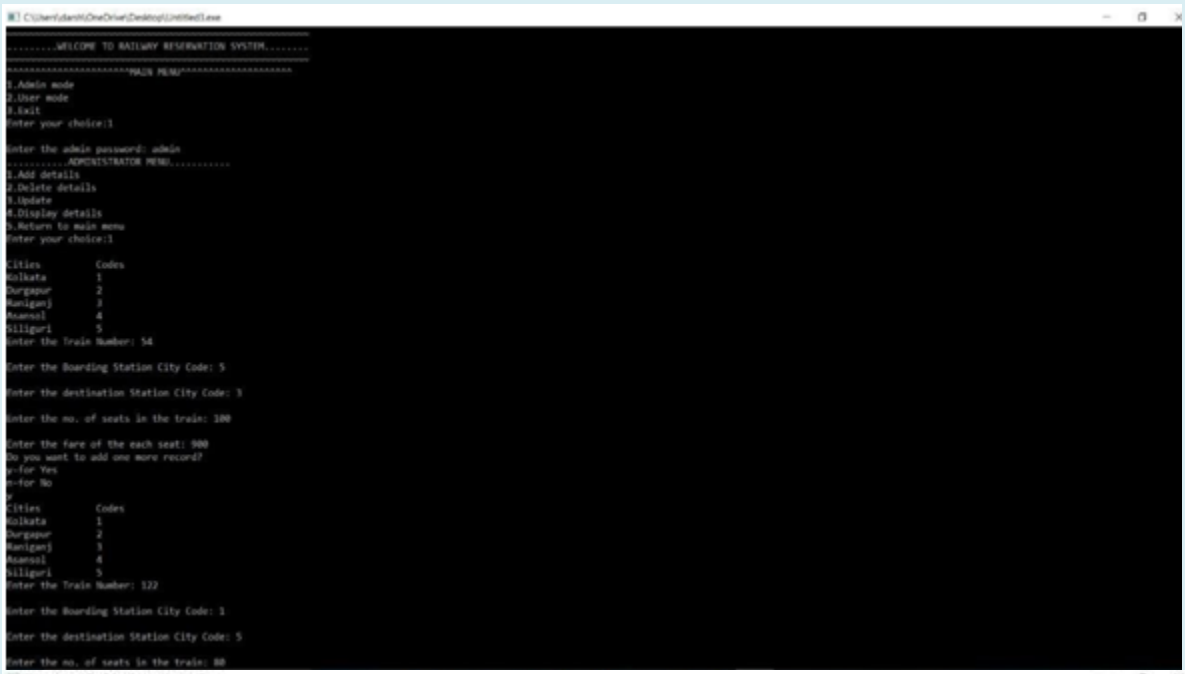
Inserting into a linked list requires re-pointing the previous node (the node before the insertion point) to the inserted node, and pointing the newly-inserted node to the next node. Thus **insertion is O(1)**.

```
firstNode = Head->Next

newNode->Next = firstNode

Head->Next = newNode
```

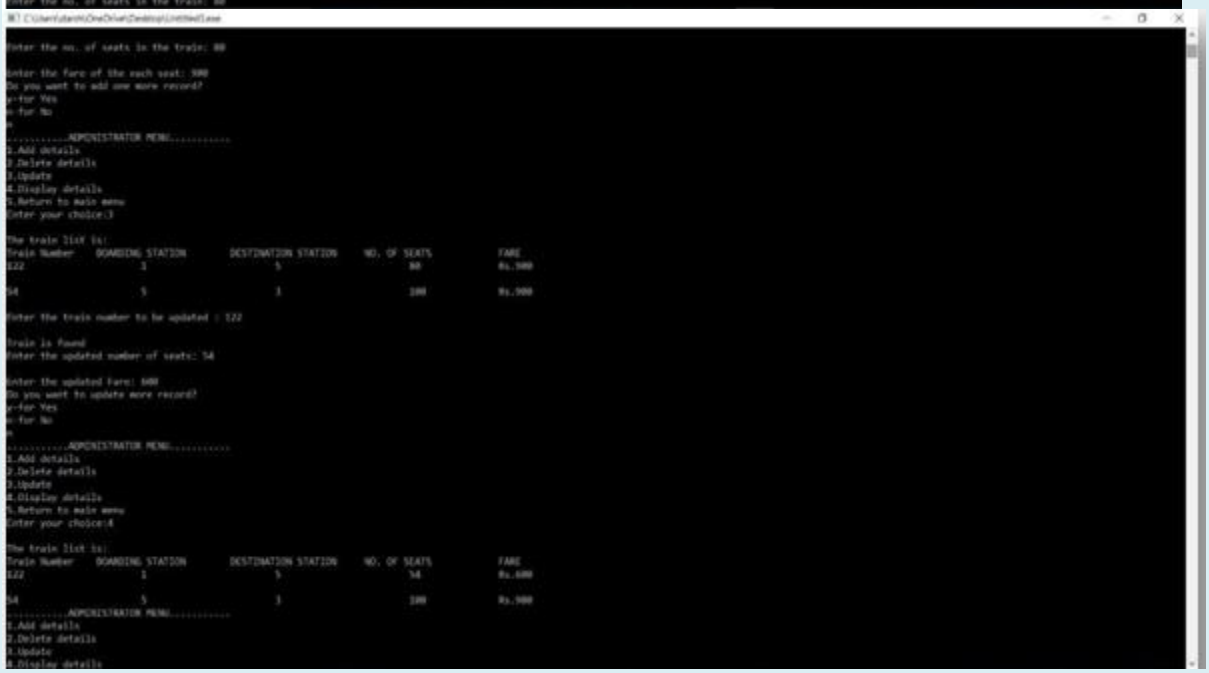
Deleting from a linked list requires re-pointing the previous node (the node before the deleted node) to the next node (the node after the deleted node). Thus **deletion is O(1)**.

```
if (currNode->data = deletionData)

previousNode->Next = currNode->next;

free(currNode)
```

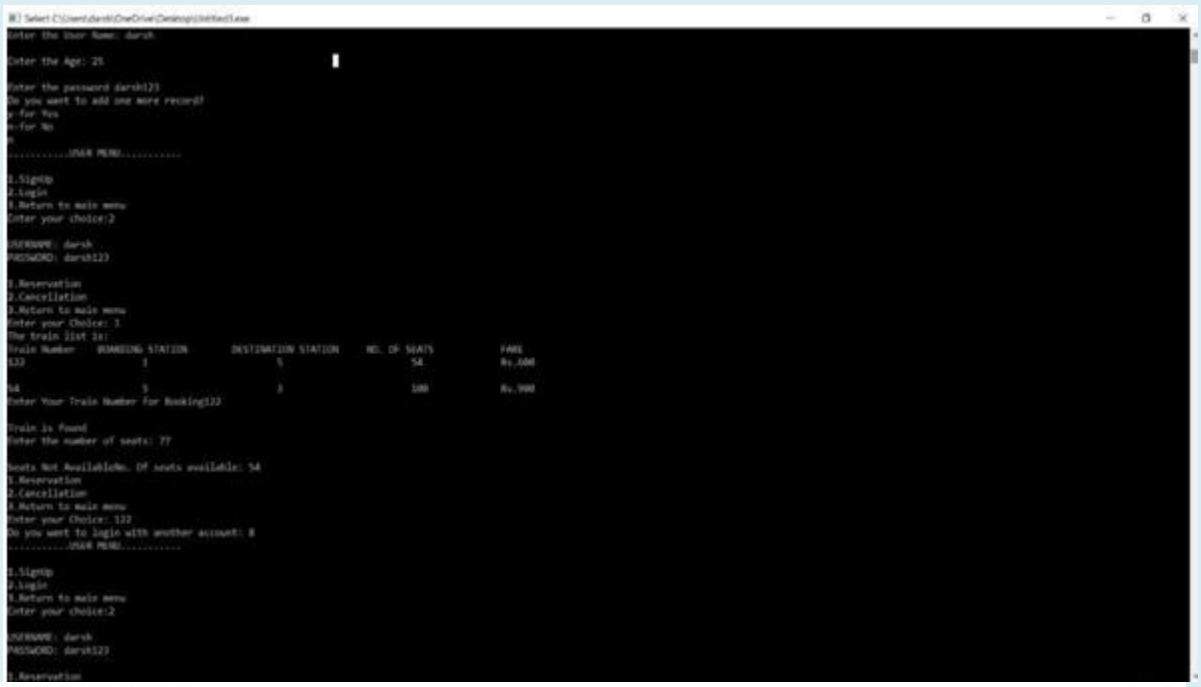
RESULTS-

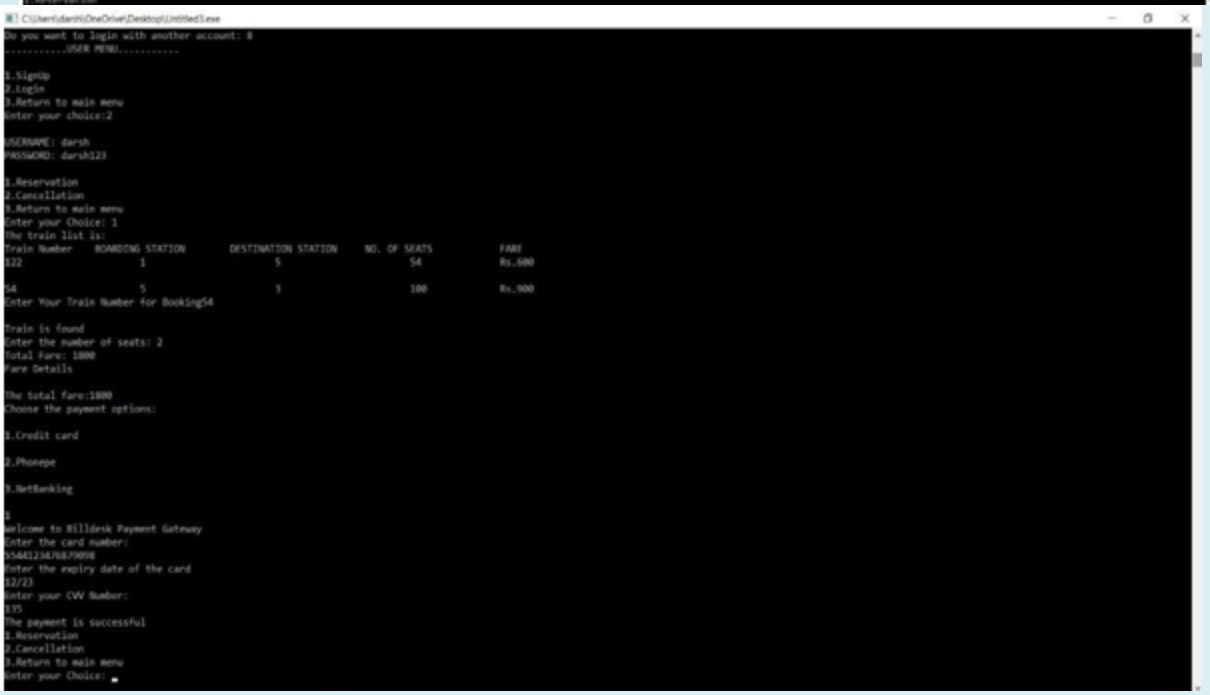


```
.....WELCOME TO RAILWAY RESERVATION SYSTEM.....

**********MAIN MENU**********
1.Admin mode
2.User mode
3.Exit
Enter your choice:1

Enter the admin password: admin
..........ADMINISTRATOR MENU..........
1.Add details
2.Delete details
3.Update
4.Display details
5.Return to main menu
Enter your choice:1

Cities          Codes
Kolkata         1
Durgapur        2
Maniganj        3
Asansol         4
Siliguri        5
Enter the Train Number: 54

Enter the Boarding Station City Code: 5

Enter the destination Station City Code: 3

Enter the no. of seats in the train: 100

Enter the fare of the each seat: 900
Do you want to add one more record?
y-for Yes
n-for No
y
Cities          Codes
Kolkata         1
Durgapur        2
Maniganj        3
Asansol         4
Siliguri        5
Enter the Train Number: 122

Enter the Boarding Station City Code: 1

Enter the destination Station City Code: 5

Enter the no. of seats in the train: 80
```



```
Enter the no. of seats in the train: 80

Enter the fare of the each seat: 500
Do you want to add one more record?
y-for Yes
n-for No
n
..........ADMINISTRATOR MENU..........
1.Add details
2.Delete details
3.Update
4.Display details
5.Return to main menu
Enter your choice:3

The train list is:
Train Number    BOARDING STATION        DESTINATION STATION     NO. OF SEATS            FARE
122                     1                       5                       80              Rs.500

54                      5                       3                       100             Rs.900

Enter the train number to be updated : 122

Train is found
Enter the updated number of seats: 54

Enter the updated fare: 600
Do you want to update more record?
y-for Yes
n-for No
n
..........ADMINISTRATOR MENU..........
1.Add details
2.Delete details
3.Update
4.Display details
5.Return to main menu
Enter your choice:4

The train list is:
Train Number    BOARDING STATION        DESTINATION STATION     NO. OF SEATS            FARE
122                     1                       5                       54              Rs.600

54                      5                       3                       100             Rs.900
..........ADMINISTRATOR MENU..........
1.Add details
2.Delete details
3.Update
4.Display details
```


Enter the User Name: darsh
Enter the Age: 25
Enter the password darsh123
Do you want to add one more record?
y-for Yes
n-for No
n
.....USER MENU.....
1.SignUp
2.Login
3.Return to main menu
Enter your choice:2
USERNAME: darsh
PASSWORD: darsh123
1.Reservation
2.Cancellation
3.Return to main menu
Enter your Choice: 1
The train list is:
Train Number BOARDING STATION DESTINATION STATION NO. OF SEATS FARE
122 1 5 54 Rs.600
54 5 3 100 Rs.900
Enter Your Train Number for Booking122
Train is found
Enter the number of seats: 77
Seats Not AvailableNo. Of seats available: 54
1.Reservation
2.Cancellation
3.Return to main menu
Enter your Choice: 122
Do you want to login with another account: 8
.....USER MENU.....
1.SignUp
2.Login
3.Return to main menu
Enter your choice:2
USERNAME: darsh
PASSWORD: darsh123
1.Reservation


Do you want to login with another account: 8
.....USER MENU.....
1.SignUp
2.Login
3.Return to main menu
Enter your choice:2
USERNAME: darsh
PASSWORD: darsh123
1.Reservation
2.Cancellation
3.Return to main menu
Enter your Choice: 1
The train list is:
Train Number BOARDING STATION DESTINATION STATION NO. OF SEATS FARE
122 1 5 54 Rs.600
54 5 3 100 Rs.900
Enter Your Train Number for Booking54
Train is found
Enter the number of seats: 2
Total Fare: 1800
Fare Details
The total fare:1800
Choose the payment options:
1.Credit card
2.Phonepe
3.NetBanking
1
Welcome to Billdesk Payment Gateway
Enter the card number:
9564123678679098
Enter the expiry date of the card
12/23
Enter your CVV Number:
835
The payment is successful
1.Reservation
2.Cancellation
3.Return to main menu
Enter your Choice:

REFERENCES-

• Nitesh, Manbir Singh, Rahul Yadav: RESEARCH PAPER ON STACK AND QUEUE© 2014 IJIRT | Volume 1 Issue 7 | ISSN: 2349-6002 www.ijirt.org/master/publishedpaper/IJIRT101357_PAPER.pdf

• Anoop, Sunil Rai: A COMPARATIVE STUDY OF STACK AND QUEUES IN DATA STRUCTUR © 2014 IJIRT | Volume 1 Issue 6 | ISSN : 2349-6002 http://ijirt.org/master/publishedpaper/IJIRT101022_PAPER.pdf

• Seema Agarwal, COMPUTERIZED PASSENGER RESERVATION SYSTEM FOR INDIAN RAILWAYS – ITS DEVELOPMENT AND SYSTEM ARCHITECTURE Journal of Engineering, Computers & Applied Sciences (JEC&AS) ISSN No: 23195606 Volume 2, No.6, June 2013

• Wang Zongjiang Weifang University,Weifang China , 2012 International Conference on Medical Physics and Biomedical Engineering

• Wang Lei. ARIS-based modeling of enterprise and application service [D] . Nanjing Nanjing University of Science 2006] • International Journal Of Engineering Research & Management Technology-Dynamic Implementation Using Linked List

Karuna and Garima Gupta
http://www.ijermt.org/publication/11/IJERMT%20V-1-5-6.pdf

• IMPLEMENTATION OF ENHANCED SINGLY LINKED LIST EQUIPPED WITH DLL OPERATIONS: AN APPROACH TOWARDS ENORMOUS MEMORY SAVING Devishree Naidu and Abhishek Prasad Jr., Member, IACSIT
http://www.ijfcc.org/papers/276-E1045.pdf

• A STUDY ON THE USAGE OF DATA STRUCTURES IN INFORMATION RETRIEVAL
https://arxiv.org/ftp/arxiv/papers/1602/1602.07799.pdf • Nitesh, Manbir Singh, Rahul Yadav Department of Electronics and Communication Dronacharya College Of Engineering Farruknagar, Khetawas, Gurgaon
http://www.ijirt.org/master/publishedpaper/IJIRT101357_PAPER.pdf

• N.M. GIRINIVAS*, P. HEMANAND*, K.P. CHETAN*, S.R. JANANI UG Student, Assistant Professor Department of Information Technology, R.M.K Engineering College-601206, India
https://www.ijcsmc.com/docs/papers/March2015/V4I3201594.pdf

• Suan Hsi Yong and Susan Horwitz, Computer Sciences Department, University of WisconsinMadison, POINTER-RANGE ANALYSIS
https://pdfs.semanticscholar.org/b27d/585a6a6c7733bf210f791c365a9cedd29a31.pdf

• Nick Parlante Copyright ©1998-2000, Nick Parlante, Pointers and Memory http://cslibrary.stanford.edu/102/PointersAndMemory.pdf

• David Koes, Mihai Budiu, Girish Venkataramani , CS Department Carnegie Mellon University PROGRAMMER SPECIFIED POINTER INDEPENDENCE http://www.cs.cmu.edu/~mihaib/research/msp04.pdf

• Hongxia Peng , Xianhao Xu , Wen Chen , TOURIST BEHAVIORS IN ONLINE BOOKING June 2013, Vol. 3 Iss. 6, PP. 280-285 http://www.academicpub.org/downloadpaper.aspx?paperid=13505

• Walter Murray and Margaret h. Wright, EFFICIENT LINEAR SEARCH ALGORITHMS FOR THE LOGARITHMIC BARRIER FUNCTION, technical report sol. 76-18 September 1976 https://apps.dtic.mil/dtic/tr/fulltext/u2/a033432.pdf

• Morrice Joseph , Palak Keshwani , COMPARISON BETWEEN LINEAR SEARCH AND BINARY SEARCH ALGORITHMS - IJARIIT(ISSN: 2454-132X) https://www.ijariit.com/conference-proceedings/15%20CSE%20107.pdf

CODE:

```
#include<stdio.h>
#include<iostream>
#include<stdlib.h>
#include<malloc.h>
#include<string.h>
using namespace std;
struct train
{
        int bs;
        int ds;
        int ns;
        int fare;
        int trno;
        struct train *next;
}*start=NULL,*q,*t;
struct user
{
        char name[100];
        char pass[20];
   int age;
   int utno;
   int uns;
   int ufare;
        struct user *next1;
}*start1,*q1,*t1;
class tra
{
        public:
                void insertrain()
                {
                        int bss,dss,nss,fares,trnos;
                        cout<<"\nEnter the Train Number: ";
```

```
            cin>>trnos;
            cout<<"\nEnter the Boarding Station City Code: ";
            cin>>bss;
            cout<<"\nEnter the destination Station City Code: ";
            cin>>dss;
            cout<<"\nEnter the no. of seats in the train: ";
            cin>>nss;
            cout<<"\nEnter the fare of the each seat: ";
            cin>>fares;

            int num;
t=(struct train*)malloc(sizeof(struct train));
t->bs=bss;
t->ds=dss;
t->ns=nss;
t->fare=fares;
t->trno=trnos;

   if(start==NULL)
{
t->next=NULL;
start=t;
}
else
{
t->next=start;
start=t;
}

}
void deletetrain()
{
            int bss,dss;
```

```
        cout<<"Enter the boarding station city code: ";
        cin>>bss;
        cout<<"Enter the destination station city code: ";
        cin>>dss;

struct train* temp = start, *prev;

if (temp != NULL && temp->bs == bss && temp->ds == dss)
{
start = temp->next;
free(temp);
return;
}
while (temp != NULL && temp->bs != bss && temp->ds != dss)
{
prev = temp;
temp = temp->next;
}
if (temp == NULL) return;

prev->next = temp->next;
        free(temp);

}
void displaytrain()
{
        if(start==NULL)
{
printf("List is empty!!");
}
else
{
q=start;
```

```
		cout<<"The train list is:\n";
		cout<<"Train Number"<<"\t"<<"BOARDING
STATION"<<"\t"<<"DESTINATION STATION"<<"\t"<<"NO. OF SEATS"<<"\t\t"<<"FARE";
		while(q!=NULL)
		{
		cout<<"\n"<<q->trno<<"\t\t\t"<<q->bs<<"\t\t\t"<<q->ds<<"\t\t\t"<<q-
>ns<<"\t\t"<<"Rs."<<q->fare<<"\n";
		q=q->next;
		}
		}
		}

		void updatetrain()
		{
			int trnos,nss,fares;
			displaytrain();
			cout<<"\nEnter the train number to be updated : ";
			cin>>trnos;

			struct train *w,*tmp;
			w=start;
			while(w!=NULL)
			{
				if(w->trno==trnos)
				{
					cout<<"\nTrain is found";
					cout<<"\nEnter the updated number of seats: ";
					cin>>nss;
					cout<<"\nEnter the updated Fare: ";
					cin>>fares;
					w->ns=nss;
					w->fare=fares;
					break;
				}
```

```
                        else
                        {
                                w=w->next;
                        }
                }
                if(w==NULL)
                {
                        printf("Train not found");
                }

        }
};

class passenger
{
        public:
                tra tt;
                void signup()
                {
                        char names[100],passs[20];
                        int ages;
                        cout<<"\nEnter the User Name: ";
                        cin>>names;
                        cout<<"\nEnter the Age: ";
                        cin>>ages;
                        cout<<"\nEnter the password ";
                        cin>>passs;

                        int num;
                t1=(struct user*)malloc(sizeof(struct user));
                strcpy(t1->name,names);
                t1->age=ages;
                strcpy(t1->pass,passs);
```

```
    if(start==NULL)
{
t1->next1=NULL;
start1=t1;
}
else
{
t1->next1=start1;
start1=t1;
}

}

void login()
{
        char names[100],passs[20];
        cout<<"USERNAME: ";
        cin>>names;
        cout<<"PASSWORD:  ";
        cin>>passs;

        struct user* temp1 = start1, *prev1;
        int ch11;
if (temp1 != NULL && (strcmp(temp1->name,names)==0)&& (strcmp(temp1->pass,passs)==0))
{
do
{
                cout<<"\n1.Reservation\n2.Cancellation\n3.Return to main menu";
                cout<<"\nEnter your Choice: ";
                cin>>ch11;
                switch(ch11)
```

```
                        {
                        case 1:
                        {
                                reservation();
                                break;
                        }
                        case 2:
                        {
                                cancel();
                                break;
                        }
                        }
                }
                        while(ch11<=2);

        }
        while (temp1 != NULL && (strcmp(temp1->name,names)!=0)&&
(strcmp(temp1->pass,passs)!=0))
        {
        prev1 = temp1;
        temp1 = temp1->next1;
        }
        if (temp1 == NULL) return;

        }
        void reservation()
        {
                tt.displaytrain();
                int trno11,nss1,tf;
                cout<<"Enter Your Train Number for Booking";
                cin>>trno11;

                struct train *w1,*tmp1;
                w1=start;
```

```
while(w1!=NULL)
{
        if(w1->trno==trno11)
        {
                cout<<"\nTrain is found";
                cout<<"\nEnter the number of seats: ";
                cin>>nss1;
                if(nss1<=w1->ns)
                {
                        w1->ns=w1->ns-nss1;
                        cout<<"Total Fare: "<<nss1*w1->fare;
                        tf=nss1*w1->fare;
                        payment(tf);


                }
                else
                {
                        cout<<"\nSeats Not Available";
                        cout<<"No. Of seats available: ";
                        cout<<w1->ns;
                }

                break;
        }
        else
        {
                w1=w1->next;
        }
}
if(w1==NULL)
{
        printf("Train not found");
```

```
		}


		return;
	}


	void cancel()
	{
		char username1[100],passs11[20];
		cout<<"\nUsername: ";
		cin>>username1;
		cout<<"\nPassword: ";
		cin>>passs11;
		struct user* temp12 = start1, *prev1;
		int ch112;
	if (temp12 != NULL && (strcmp(temp12->name,username1)==0)&&
(strcmp(temp12->pass,passs11)==0))
	{
		cout<<"Enter the number of tickets to be cancelled: ";
		cin>>ch112;
		temp12->uns=temp12->uns-ch112;
		cout<<"\nCancellation Done.";



	}


	while (temp12 != NULL && (strcmp(temp12->name,username1)!=0)&&
(strcmp(temp12->pass,passs11)!=0))
	{
	prev1 = temp12;
	temp12 = temp12->next1;
	}
```

```
        if (temp12 == NULL) return;




                return;
        }

        void payment(int s)
        {
                cout<<"\nFare Details"<<endl;
                cout<<"\nThe total fare:"<<s;
                cout<<"\nChoose the payment options:"<<endl;
                cout<<"\n1.Credit
card"<<endl<<"\n2.Phonepe"<<endl<<"\n3.NetBanking\n"<<endl;
                int ch;
                scanf("%d",&ch);
                switch(ch)
                {

                        case 1:
                                creditcard();
                                break;
                        case 2:
                                Phonepe();
                                break;
                        case 3:
                                NetBanking();
                                break;
        }
    }

        char cdno[20];
        int cvv;
        char expiry[8];
```

```
void creditcard()
{
        cout<<"Welcome to Billdesk Payment Gateway"<<endl;
        cout<<"Enter the card number:"<<endl;
        cin>>cdno;
        cout<<"Enter the expiry date of the card"<<endl;
        cin>>expiry;
        cout<<"Enter your CVV Number:"<<endl;
        cin>>cvv;
        cout<<"The payment is successful";
}
char upi[5]="6837";
char upi1[5];
void Phonepe()
{
        cout<<"Enter the Upi pin"<<endl;
        cin>>upi1;
        if(strcmp(upi1,upi)==0)
        {
                cout<<"Payment successful"<<endl;
        }
else
        {
                cout<<"Payment unsuccessful"<<endl;
                Phonepe();
        }

}
char userid[10];
char password[10];
char mall[40]="Vellore";
char mall1[40];
```

```
        void NetBanking()
        {
                cout<<"Netbanking Login:"<<endl;
                cout<<"Enter the User-id:"<<endl;
                cin>>userid;
                cout<<"Enter the password"<<endl;
                cin>>password;
                cout<<"What is the mall closest from your house:"<<endl;
                cin>>mall1;
                if(strcmp(mall1,mall)==0)
                {
                        cout<<"Payment successful"<<endl;
                }
                else
                {
                        cout<<"Payment unsuccessful"<<endl;
                        NetBanking();
                }
        }


};

void user1()
{
        int ch;

        passenger p;

        do
    {
       cout<<".....USER MENU. ..... \n";
       cout<<"\n1.SignUp\n2.Login\n";
```

```
        cout<<"3.Return to main menu\n";
        cout<<"Enter your choice:";
        cin>>ch;

                        char c;
        cout<<endl;
        switch(ch)
        {
        case 1:
          do
           {
                  p.signup();
              cout<<"Do you want to add one more record?\n";
              cout<<"y-for Yes\nn-for No\n";
              cin>>c;
           }
           while(c=='y');
           break;
        case 2:
          do
          {
                  p.login();
                  cout<<"Do you want to login with another account: ";
                  cin>>c;
                                  }
                                  while(c=='y');
                                  break;
        }
      }
      while(ch<=2);

}
void admin()
```

```
{
          tra t;
          char password[10];
          char pass[]="admin";
          cout<<"Enter the admin password: ";
          cin>>password;
   int ch;
   char c;
   if(strcmp(pass,password)!=0)
   {
      cout<<"Enter the password correctly \n";
      cout<<"You are not permitted to logon this mode\n";
   }
   if(strcmp(pass,password)==0)
   {
         char c;
      do
      {
         cout<<".....ADMINISTRATOR MENU. ..... \n";
         cout<<"1.Add details\n2.Delete details\n3.Update\n";
         cout<<"4.Display details\n";
         cout<<"5.Return to main menu\n";
         cout<<"Enter your choice:";
         cin>>ch;


         cout<<endl;
         switch(ch)
         {
         case 1:
          do
           {
                    cout<<"Cities"<<"\t\t"<<"Codes";
```

```
            cout<<"\nKolkata"<<"\t\t"<<1;
            cout<<"\nDurgapur"<<"\t"<<2;
            cout<<"\nRaniganj"<<"\t"<<3;
            cout<<"\nAsansol"<<"\t\t"<<4;
            cout<<"\nSiliguri"<<"\t"<<5;
        t.insertrain();
        cout<<"Do you want to add one more record?\n";
        cout<<"y-for Yes\nn-for No\n";
        cin>>c;
      }
      while(c=='y');
      break;
case 2:
  do
  {
            t.deletetrain();
            cout<<"Do you want to delete one more record?\n";
        cout<<"y-for Yes\nn-for No\n";
        cin>>c;
                                }
                                while(c=='y');
                                break;
                    case 3:
                                do
  {
             t.updatetrain();
            cout<<"Do you want to update more record?\n";
        cout<<"y-for Yes\nn-for No\n";
        cin>>c;
                                }
                                while(c=='y');
                                break;
                    case 4:
```

```
                                    t.displaytrain();
          break;
        }
      }
      while(ch<=4);

        }
}

int main()
{
   int ch;
   cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n";
   cout<<".....WELCOME TO RAILWAY RESERVATION SYSTEM ..... \n";
   cout<<"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n";
   do
   {
      cout<<"^^^^^^^^^^^^^^^^^^^^^^MAIN MENU^^^^^^^^^^^^^^^^^^^^\n";
      cout<<"1.Admin mode\n2.User mode\n3.Exit\n";
      cout<<"Enter your choice:";
      cin>>ch;
      cout<<endl;
      switch(ch)

      {
      case 1:
         admin();
         break;
      case 2:
          user1();
          break;
      case 3:
         exit(0);
```

```
        }
    }
    while(ch<=3);
    return 0;
}
```