PPS MINI PROJECT – BUS RESERVATION SYSTEM

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>
#include <ctype.h>

#define MAX_YR  9999
#define MIN_YR  1900
#define MAX_SIZE_USER_NAME 30
#define MAX_SIZE_PASSWORD  20
#define FILE_NAME  "PassengerRecordSystem.bin"

#define MAX_PASSENGER_NAME 50
#define MAX_PASSENGER_ADDRESS 300
#define MAX_PASSENGER_MOB_NUM 20
#define FILE_HEADER_SIZE  sizeof(sFileHeader)

typedef struct
{
   int yyyy;
   int mm;
   int dd;
} Date;
typedef struct
{
   char username[MAX_SIZE_USER_NAME];
   char password[MAX_SIZE_PASSWORD];
} sFileHeader;
```

```
typedef struct
{
   unsigned int passengerId;
   float ticketPrice;
   unsigned int passengerSeatNum;
   Date passengerTravelingDate;
   char passengerName[MAX_PASSENGER_NAME];
   char passengerMobNum[MAX_PASSENGER_MOB_NUM];
   char passengerAddr[MAX_PASSENGER_ADDRESS];
} s_PassengerInfo;

void fgetsRemovedNewLine(char * restrict buf, int n,FILE * restrict stream)
{
   if (fgets(buf, n, stream) == NULL)
   {
      printf("Fail to read the input stream");
   }
   else
   {
      buf[strcspn(buf, "\n")] = '\0';
   }
}

void printMessageCenter(const char* message)
{
   int len =0;
   int pos = 0;

   len = (78 - strlen(message))/2;
   printf("\t\t\t");
   for(pos =0 ; pos < len ; pos++)
   {


       printf(" ");
   }


   printf("%s",message);
}

void headMessage(const char *message)
{
   system("cls");
printf("\t\t\t#####################################################################
###");
   printf("\n\t\t\t############                                        ############");
   printf("\n\t\t\t############         Bus Ticket Booking System in C         ############");
   printf("\n\t\t\t############                                        ############");
printf("\n\t\t\t###################################################################
#####");
   printf("\n\t\t\t---------------------------------------------------------------------------------\n");
   printfMessageCenter(message);
   printf("\n\t\t\t-----------------------------------------------------------------------------------");
}

void welcomeMessage()
{

   printf("\n\n\n\n\n");
   printf("\n\t\t\t  -------------------\n");
   printf("\n\t\t\t        =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=");
   printf("\n\t\t\t        =                 WELCOME                  =");
```

```
    printf("\n\t\t\t       =                 TO                  =");
    printf("\n\t\t\t       =              Bus Ticket             =");
    printf("\n\t\t\t       =             Booking  SYSTEM         =");
    printf("\n\t\t\t       =                                     =");
    printf("\n\t\t\t       =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=");
    printf("\n\t\t\t  ------------------\n");
    printf("\n\n\n\t\t\t Enter any key to continue.....");
    getchar();
}

int isNameValid(const char *name)
{
    int validName = 1;
    int len = 0;
    int index = 0;
    len = strlen(name);
    for(index =0; index <len ; ++index)
    {
        if(!(isalpha(name[index])) && (name[index] != '\n') && (name[index] != ' '))
        {
            validName = 0;
            break;
        }
    }
    return validName;
}

int isValidMobNumber(const char *name)
{
    int validName = 1;
    int len = 0;
    int index = 0;
    len = strlen(name);
    for(index =0; index <len ; ++index)
    {
        if(!(isdigit(name[index])) && (name[index] != '\n') && (name[index] != ' '))
        {
            validName = 0;
            break;
        }
    }
    return validName;
}

int  IsLeapYear(int year)
{
    return (((year % 4 == 0) &&
             (year % 100 != 0)) ||
            (year % 400 == 0));
}

int isValidDate(Date *validDate)
{

    if (validDate->yyyy > MAX_YR ||
            validDate->yyyy < MIN_YR)
        return 0;
    if (validDate->mm < 1 || validDate->mm > 12)
        return 0;
    if (validDate->dd < 1 || validDate->dd > 31)
        return 0;
    if (validDate->mm == 2)
    {
        if (IsLeapYear(validDate->yyyy))
            return (validDate->dd <= 29);
        else
            return (validDate->dd <= 28);
    }

    if (validDate->mm == 4 || validDate->mm == 6 ||
            validDate->mm == 9 || validDate->mm == 11)
        return (validDate->dd <= 30);
    return 1;
}

void addPassengerInDataBase()
{
    s_PassengerInfo addPassengerInfoInDataBase = {0};
    FILE *fp = NULL;
    int status = 0;
    fp = fopen(FILE_NAME,"ab+");
    if(fp == NULL)
    {
        printf("File is not opened\n");
        exit(1);
    }
    headMessage("ADD NEW PASSENGER");
    printf("\n\n\t\t\tENTER YOUR DETAILS BELOW:");
    printf("\n\t\t\t---------------------------------------------------------------------------\n");
    printf("\n\t\t\tPassenger ID  = ");
    fflush(stdin);
    scanf("%u",&addPassengerInfoInDataBase.passengerId);
```

```
 do
  {


     fflush(stdin);

fgetsRemovedNewLine(addPassengerInfoInDataBase.passengerName,MAX_PASSENGER_NAME,stdi
n);
     status = isNameValid(addPassengerInfoInDataBase.passengerName);
     if (!status)
     {
       printf("\n\t\t\tName contain invalid character. Please enter again.");
     }
  }


  while(!status);
  do
  {
     printf("\n\t\t\tPassenger Mob: = ");
     fflush(stdin);

fgetsRemovedNewLine(addPassengerInfoInDataBase.passengerMobNum,MAX_PASSENGER_MOB_N
UM,stdin);
     status = isValidMobNumber(addPassengerInfoInDataBase.passengerMobNum);
     if (!status)
     {
       printf("\n\t\t\tName contain invalid character. Please enter again.");
     }
  }
  while(!status);
  do
  {
     printf("\n\t\t\tPassenger Address  = ");
```

```
        fflush(stdin);

fgetsRemovedNewLine(addPassengerInfoInDataBase.passengerAddr,MAX_PASSENGER_ADDRESS,st
din);
        status = isNameValid(addPassengerInfoInDataBase.passengerAddr);
        if (!status)
        {
            printf("\n\t\t\tName contain invalid character. Please enter again.");
        }
    }
    while(!status);
    printf("\n\t\t\tPassenger Ticket Price = ");
    fflush(stdin);
    scanf("%f",&addPassengerInfoInDataBase.ticketPrice);
    do
    {
        printf("\n\t\t\tPassenger Traveling Date:- ");


        printf("\n\t\t\tEnter date in format (dd/mm/yyyy): ");

scanf("%d/%d/%d",&addPassengerInfoInDataBase.passengerTravelingDate.dd,&addPassengerInfoIn
DataBase.passengerTravelingDate.mm,&addPassengerInfoInDataBase.passengerTravelingDate.yyyy)
;


        status = isValidDate(&addPassengerInfoInDataBase.passengerTravelingDate);
        if (!status)
        {
            printf("\n\t\t\tPlease enter a valid date.\n");
        }
    }
    while(!status);
    do
    {
        unsigned int tempSeatNumber = 0;
        printf("\n\t\t\tPassenger Seat number = ");
        fflush(stdin);
        scanf("%u",&tempSeatNumber);
        status = (tempSeatNumber != addPassengerInfoInDataBase.passengerSeatNum);
        if(!status)
        {
            printf("\n\t\t\tAlready allocate Seat, Choose another Seat. \n");
        }
    }
    while(!status);
    fwrite(&addPassengerInfoInDataBase,sizeof(addPassengerInfoInDataBase), 1, fp);
    fclose(fp);
}

void searchPassenger()
{
    int found = 0;
    int passengerId =0;
    s_PassengerInfo addPassengerInfoInDataBase = {0};
    FILE *fp = NULL;
    fp = fopen(FILE_NAME,"rb");
    if(fp == NULL)
    {
        printf("\n\t\t\tFile is not opened\n");
        exit(1);
    }
    headMessage("SEARCH PASSENGER");

    if (fseek(fp,FILE_HEADER_SIZE,SEEK_SET) != 0)
    {
        fclose(fp);
        printf("\n\t\t\tFacing issue while reading file\n");
        exit(1);
    }
    printf("\n\n\t\t\tEnter passenger  ID NO to search:");
    fflush(stdin);
    scanf("%u",&passengerId);
    while (fread (&addPassengerInfoInDataBase, sizeof(addPassengerInfoInDataBase), 1, fp))
    {
        if(addPassengerInfoInDataBase.passengerId == passengerId)
        {
            found = 1;
            break;
        }
    }
    if(found)
    {
        printf("\n\t\t\tPassenger id = %d\n",addPassengerInfoInDataBase.passengerId);


        printf("\n\t\t\tPassenger Mob = %s\n",addPassengerInfoInDataBase.passengerMobNum);


        printf("\n\t\t\tPassenger Ticket Price = %f\n",addPassengerInfoInDataBase.ticketPrice);
        printf("\n\t\t\tPassenger Address = %s\n",addPassengerInfoInDataBase.passengerAddr);
        printf("\n\t\t\tPassenger Admited Date(day/month/year) =
(%d/%d/%d)\n",addPassengerInfoInDataBase.passengerTravelingDate.dd,
               addPassengerInfoInDataBase.passengerTravelingDate.mm,
addPassengerInfoInDataBase.passengerTravelingDate.yyyy);
    }
    else
    {
        printf("\n\t\t\tNo Record");
    }
    fclose(fp);
    printf("\n\n\n\t\t\tPress any key to go to main menu.....");
    fflush(stdin);
    getchar();
}

void viewPassenger()
{
    int found = 0;
    s_PassengerInfo addPassengerInfoInDataBase = {0};
    FILE *fp = NULL;
    unsigned int countPassenger = 1;
    headMessage("VIEW PASSENGER DETAILS");
    fp = fopen(FILE_NAME,"rb");
    if(fp == NULL)
    {
        printf("File is not opened\n");
        exit(1);
    }
    if (fseek(fp,FILE_HEADER_SIZE,SEEK_SET) != 0)
    {
        fclose(fp);
        printf("Facing issue while reading file\n");
        exit(1);
    }

    printf("\n\t\t\tPassenger Count = %d\n\n",countPassenger);
    while (fread (&addPassengerInfoInDataBase, sizeof(addPassengerInfoInDataBase), 1, fp))
    {
        printf("\n\t\t\tPassenger id = %d\n",addPassengerInfoInDataBase.passengerId);
        printf("\n\t\t\tPassenger Mob = %s\n",addPassengerInfoInDataBase.passengerMobNum);


        printf("\n\t\t\tPassenger Ticket Price = %f\n",addPassengerInfoInDataBase.ticketPrice);

        printf("\n\t\t\tPassenger Address = %s\n",addPassengerInfoInDataBase.passengerAddr);

        printf("\n\t\t\tPassenger Admited Date(day/month/year) =
(%d/%d/%d)\n",addPassengerInfoInDataBase.passengerTravelingDate.dd,

            addPassengerInfoInDataBase.passengerTravelingDate.mm,
addPassengerInfoInDataBase.passengerTravelingDate.yyyy);

        found = 1;

        ++countPassenger;

    }

    fclose(fp);

    if(!found)

    {

        printf("\n\t\t\tNo Record");

    }

    printf("\n\n\t\t\tPress any key to go to main menu.....");

    fflush(stdin);

    getchar();

}


void deletePassenger()

{

    int found = 0;

    int passengerDelete = 0;

    sFileHeader fileHeaderInfo = {0};

    s_PassengerInfo addPassengerInfoInDataBase = {0};

    FILE *fp = NULL;

    FILE *tmpFp = NULL;

    headMessage("Delete passenger Record Details");

    fp = fopen(FILE_NAME,"rb");

    if(fp == NULL)
```

```
{
   printf("File is not opened\n");
   exit(1);
}
tmpFp = fopen("tmp.bin","wb");
if(tmpFp == NULL)
{
   fclose(fp);
   printf("File is not opened\n");
   exit(1);
}
fread (&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
fwrite(&fileHeaderInfo,FILE_HEADER_SIZE, 1, tmpFp);
printf("\n\t\t\tEnter passenger ID NO. for delete:");
scanf("%d",&passengerDelete);
while (fread (&addPassengerInfoInDataBase, sizeof(addPassengerInfoInDataBase), 1, fp))
{
   if(addPassengerInfoInDataBase.passengerId != passengerDelete)
   {
      fwrite(&addPassengerInfoInDataBase,sizeof(addPassengerInfoInDataBase), 1, tmpFp);
   }
   else
   {
      found = 1;
   }
}
(found)? printf("\n\t\t\tRecord deleted successfully....."):printf("\n\t\t\tRecord not found");
fclose(fp);
fclose(tmpFp);
remove(FILE_NAME);
rename("tmp.bin",FILE_NAME);
```

```
}

void updateCredential(void)
{
    sFileHeader fileHeaderInfo = {0};
    FILE *fp = NULL;
    char userName[MAX_SIZE_USER_NAME] = {0};
    char password[MAX_SIZE_PASSWORD] = {0};
    headMessage("Update Credential");
    fp = fopen(FILE_NAME,"rb+");
    if(fp == NULL)
    {
        printf("File is not opened\n");
        exit(1);
    }
    fread (&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
    if (fseek(fp,0,SEEK_SET) != 0)
    {
        fclose(fp);
        printf("\n\t\t\tFacing issue while updating password\n");
        exit(1);
    }

    fflush(stdin);
    fgetsRemovedNewLine(userName,MAX_SIZE_USER_NAME,stdin);
    printf("\n\n\t\t\tNew Password:");
    fflush(stdin);
    fgetsRemovedNewLine(password,MAX_SIZE_PASSWORD,stdin);
    strncpy(fileHeaderInfo.username,userName,sizeof(userName));
    strncpy(fileHeaderInfo.password,password,sizeof(password));
    fwrite(&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
    fclose(fp);
    printf("\n\t\t\tYour Password has been changed successfully");
    printf("\n\t\t\tRe-Run Application and Login with new Credential:");
    fflush(stdin);
    getchar();
    exit(1);
}

void menu()
{
    int choice = 0;
    do
    {
        headMessage("MAIN MENU");
        printf("\n\n\n\t\t\t1.Add New passenger Record");
        printf("\n\t\t\t2.Search passenger Record");
        printf("\n\t\t\t3.View passenger Record");
        printf("\n\t\t\t4.Delete passenger Record");
        printf("\n\t\t\t5.Update Password");
        printf("\n\t\t\t0.Exit");
        printf("\n\n\n\t\t\tEnter choice => ");
        scanf("%d",&choice);
        switch(choice)
        {
        case 1:
            addPassengerInDataBase();
            break;
        case 2:
            searchPassenger();
            break;
        case 3:
            viewPassenger();
            break;
        case 4:
            deletePassenger();
            break;
        case 5:
            updateCredential();
            break;
        case 0:
            printf("\n\n\n\t\t\t\tThank you!!!\n\n\n\n\n");
            exit(1);
            break;
        default:
            printf("\n\n\n\t\t\tINVALID INPUT!!! Try again...");
        }
    }
    while(choice!=0);
}

void login()
{
    char userName[MAX_SIZE_USER_NAME] = {0};
    char password[MAX_SIZE_PASSWORD] = {0};
    int L=0;
    sFileHeader fileHeaderInfo = {0};
    FILE *fp = NULL;
    headMessage("Login");
    fp = fopen(FILE_NAME,"rb");
    if(fp == NULL)
    {
        printf("Data base is not opened\n");
        exit(1);
    }
    fread (&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
    fclose(fp);
    do
    {
        printf("\n\n\n\t\t\t\tUsername:");
        fgetsRemovedNewLine(userName,MAX_SIZE_USER_NAME,stdin);
        printf("\n\t\t\t\tPassword:");
        fgetsRemovedNewLine(password,MAX_SIZE_PASSWORD,stdin);
        if((!strcmp(userName,fileHeaderInfo.username)) &&
(!strcmp(password,fileHeaderInfo.password)))
        {
            menu();
        }
        else
        {
            printf("\t\t\t\tLogin Failed Enter Again Username & Password\n\n");
            L++;
        }
    }
    while(L<=3);
    if(L>3)
    {
        headMessage("Login Failed");
        printf("\t\t\t\tSorry,Unknown User.");
        getchar();
        system("cls");
    }
}
```

```
int isFileExists(const char *path)
{

   FILE *fp = fopen(path, "rb");
   int status = 0;

   if (fp != NULL)
   {
      status = 1;

      fclose(fp);
   }
   return status;
}
void init()
{
   FILE *fp = NULL;
   int status = 0;
   const char defaultUsername[] ="sree";
   const char defaultPassword[] ="kamya";
   sFileHeader fileHeaderInfo = {0};
   status = isFileExists(FILE_NAME);
   if(!status)
   {

      fp = fopen(FILE_NAME,"wb");
      if(fp != NULL)
      {

         strncpy(fileHeaderInfo.password,defaultPassword,sizeof(defaultPassword));
         strncpy(fileHeaderInfo.username,defaultUsername,sizeof(defaultUsername));
```

```
            fwrite(&fileHeaderInfo,FILE_HEADER_SIZE, 1, fp);
            fclose(fp);
        }
    }
}
int main()
{
    init();
    welcomeMessage();
    login();
    return 0;
}
```