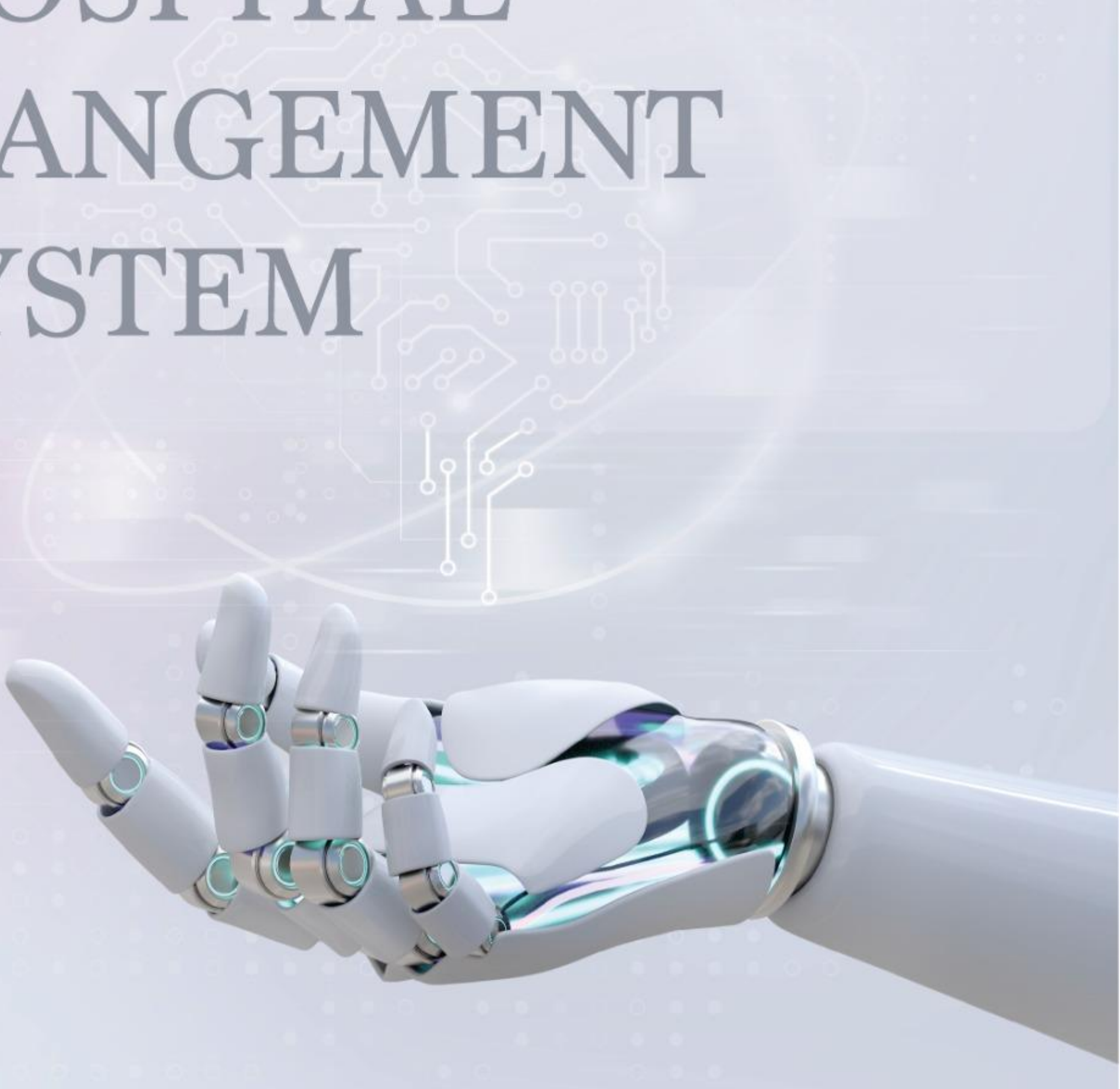
# HOSPITAL MANGEMENT SYSTEM



| NAME | ID |
|---|---|
| ATAHAR HOSSAIN | 2231877642 |
| TANVIR AHMED | 2231047642 |

Date of submission : 15 /11/2023

GROUP NO : 7

PROJECT NAME:

HOSPITAL MANAGEMENT SYSTEM

| | |
|---|---|
| COURSE | : CSE215L |
| SECTION | : 10 |
| FACULTY INITIAL | : MUO |
| LAB INSTRUSTOR | : MD. ABDULLAH AL SAYED |

# Introduction:

This project is created by Tanvir Ahmed and Atahar Hossain. The following project is a mini "Hospital Management System" called "HT HOSPITAL MANAGEMENT SYSTEM" where we tried to create a short version of a management system in a real hospital. We tried to keep it as simple as possible and didn't include GUI or JAVAFX. Here we kept 6 variables or categories of management. Here 3 of them are taken as human-based and 3 are non-human-based objects. They are –

- Human Based
  1. Doctor
  2. Patient
  3. Staff
- Non-Human Based
  1. Lab
  2. Medicine
  3. Facilities

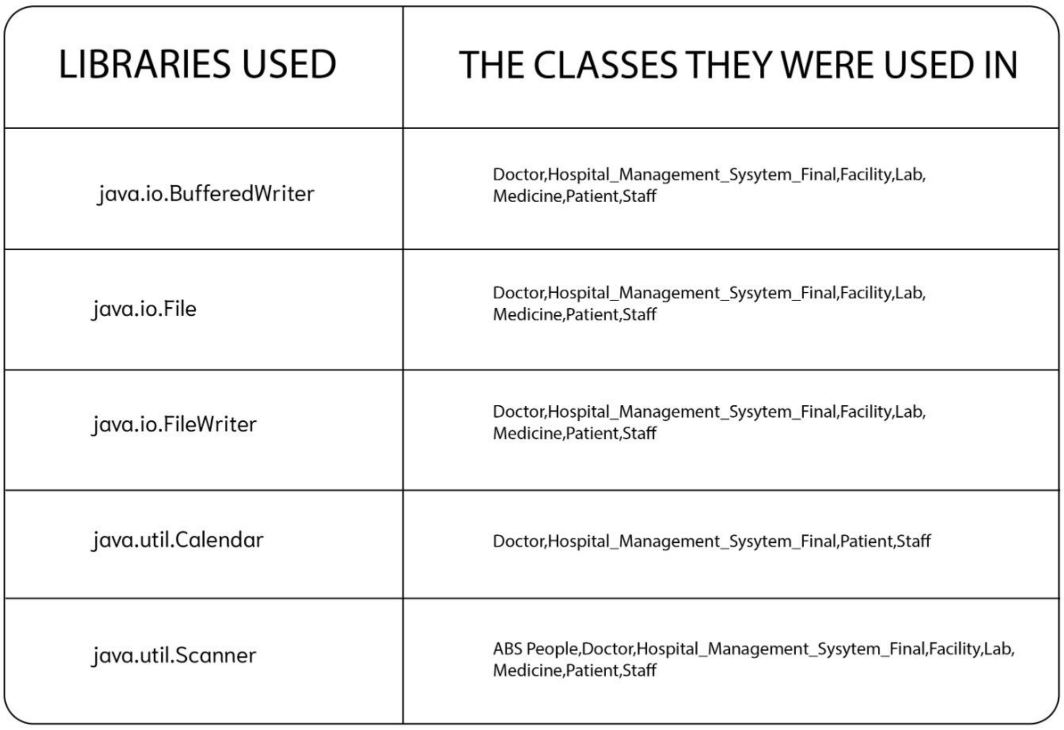
We have divided the whole system in 2 divisions which are "Admin" and "Receptionist". We created these to divide the categories or objects and their methods to use in main method in a systematic way. In admin we kept all the functions of Doctor, Staff, Lab, Medicine and Facilities free to use but in receptionist we only kept the view option. We also decided that Patient part should be available in receptionist thus kept it's functions only in receptionist.

We have kept methods to add, edit and view all existing objects for these 6 of these categories. We used files to save their data and created and used necessary methods to accomplish our tasks. We used OOP(Object Oriented Programme) while adding, viewing and editing these objects. We have used inheritance, interface, polymorphism, abstraction in the classes where needed and used exception handling in file and other methods where it was necessary. In patient we also added a simple appointment system.

If given proper utilization this project can be a great source of resource in future. A Hospital Management System, featuring add, edit, and view functions, is crucial for healthcare efficiency. It organizes patient information, saving time for staff and fostering better teamwork. With a user-friendly interface, it enhances patient care through quick data access while ensuring privacy. Adaptable and rule-compliant, it serves as a foundation for future growth and skill development.

## Libraries used:

As we have not used GUI or JAVAFX in our project, we only needed a few java libraries in our project's classes. The below chart gives the information about the used java libraries and the classes where they have been used.



| LIBRARIES USED | THE CLASSES THEY WERE USED IN |
|---|---|
| java.io.BufferedWriter | Doctor,Hospital_Management_Sysytem_Final,Facility,Lab, Medicine,Patient,Staff |
| java.io.File | Doctor,Hospital_Management_Sysytem_Final,Facility,Lab, Medicine,Patient,Staff |
| java.io.FileWriter | Doctor,Hospital_Management_Sysytem_Final,Facility,Lab, Medicine,Patient,Staff |
| java.util.Calendar | Doctor,Hospital_Management_Sysytem_Final,Patient,Staff |
| java.util.Scanner | ABS People,Doctor,Hospital_Management_Sysytem_Final,Facility,Lab, Medicine,Patient,Staff |

Picture: Used libraries and classes there used in

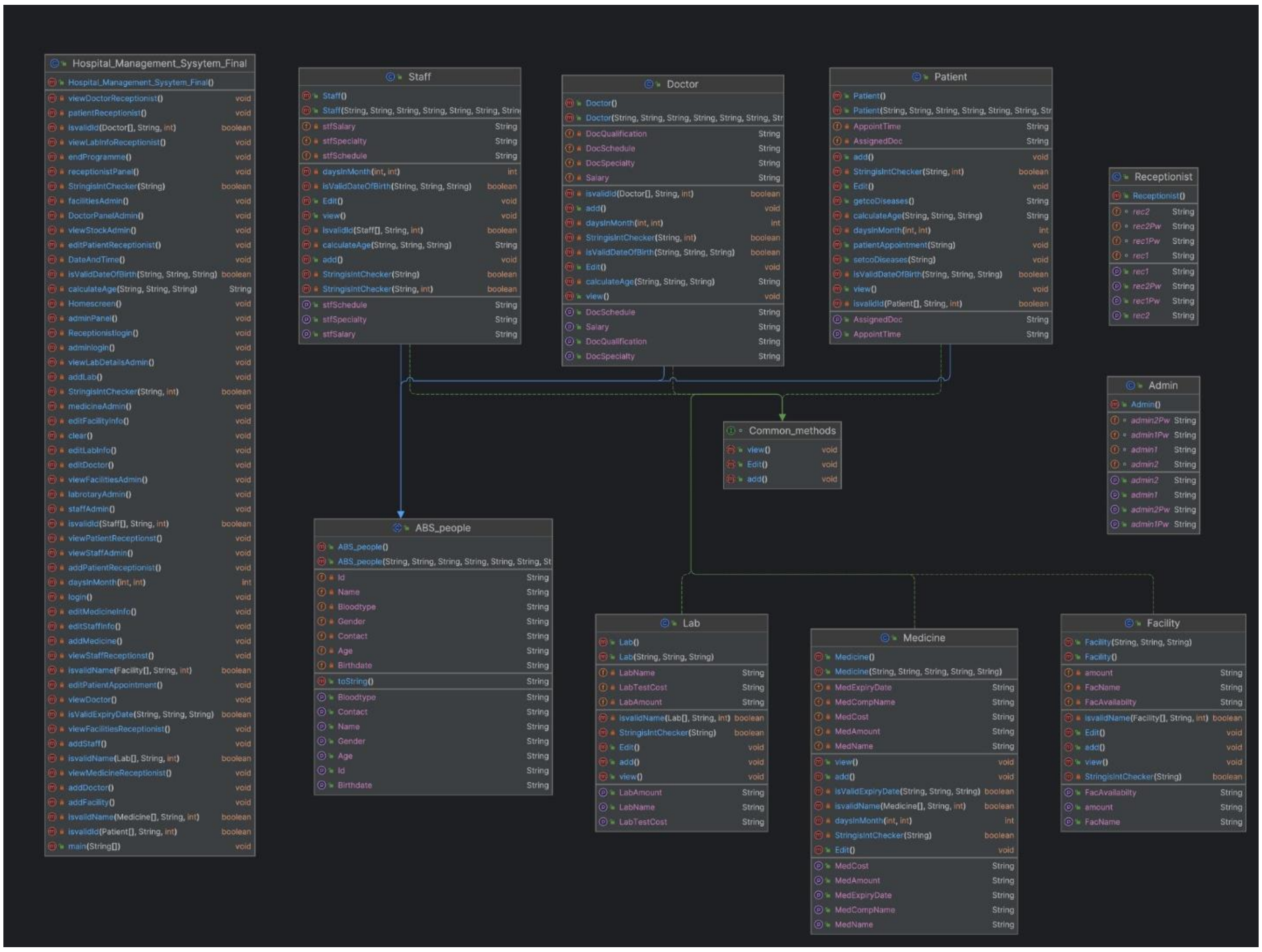
# Project Feature Table and Work Distribution:

| Class and Interface Names | Atahar Hossain | Tanvir Ahmed |
|---|---|---|
| abstract class ABS_people | NULL | Full class and it's methods |
| Admin class | Full class and it's methods | NULL |
| Common_methods Interface | NULL | Full interface and it's methods |
| Doctor class | Full class and its methods | NULL |
| Facility class | -boolean StringisIntChecker(String str)<br>-boolean isvalidName(Facility[] p, String name, int count ) | Class, constructors, accessors and mutators, edit, view and add methods |
| Hospital_Management_<br>System_Final class | -void DateAndTime()<br>-void DoctorPanelAdmin() and the edit, add, view methods inside it .<br>-void staffAdmin() and the edit, add, view methods inside it .<br>-void patientReceptionist() and the edit, add, view methods inside it .<br>-void editPatientAppointment()<br>- static boolean StringisIntChecker(String str)<br>- static boolean StringisIntChecker(String str,int num)<br>- static boolean isValidExpiryDate(String day, String month, String year)<br>- static int daysInMonth(int month, int year)<br>- static boolean isValidDateOfBirth(String day, String month, String year)<br>- static String calculateAge(String day, String month, String year)<br>- static boolean isvalidName(Medicine[] p, String name, int count ) | Rest of the methods |

| | | |
|---|---|---|
| | - static boolean isvalidName(Lab[] p, String name, int count )<br>- boolean isvalidName(Facility[] p, String name, int count )<br>-static boolean isvalidId(Doctor[] p, String id, int count )<br>- static boolean isvalidId(Staff[] p, String id, int count )<br>- static Boolean isvalidId(Patient[] p, String id, int count ) | |
| Lab class | -boolean isvalidName(Lab[] p, String name, int count )<br>-boolean StringisIntChecker(String str) | Class, constructors, accessors and mutators, edit, view and add methods |
| Medicine class | -boolean StringisIntChecker(String str)<br>-boolean isValidExpiryDate(String day, String month, String year)<br>-int daysInMonth(int month, int year)<br>-boolean isvalidName(Medicine[] p, String name, int count ) | Class, constructors, accessors and mutators, edit, view and add methods |
| Patient class | Full class and its methods | NULL |
| Receptionist class | NULL | Full class and it's methods |
| Staff class | -boolean StringisIntChecker(String str)<br>-boolean isValidDateOfBirth(String day, String month, String year)<br>-int daysInMonth(int month, int year)<br>-String calculateAge(String day, String month, String year)<br>-boolean isvalidId(Staff[] p, String id, int count ) | Class, constructors, accessors and mutators, edit, view and add methods |

Disclaimer: There are nearly 52 methods in Hospital_Management_System_Final class and we made the structure of the programme based on one method calling another.

# UML Diagram of Project and It's Classes:


Hospital_Management_Sysytem_Final
Hospital_Management_Sysytem_Final()
viewDoctorReceptionist() void
patientReceptionist() void
isvalidId(Doctor[], String, int) boolean
viewLabInfoReceptionist() void
endProgramme() void
receptionistPanel() void
StringisIntChecker(String) boolean
facilitiesAdmin() void
DoctorPanelAdmin() void
viewStockAdmin() void
editPatientReceptionist() void
DateAndTime() void
isValidDateOfBirth(String, String, String) boolean
calculateAge(String, String, String) String
Homescreen() void
adminPanel() void
Receptionistlogin() void
adminlogin() void
viewLabDetailsAdmin() void
addLab() void
StringisIntChecker(String, int) boolean
medicineAdmin() void
editFacilityinfo() void
clear() void
editLabInfo() void
editDoctor() void
viewFacilitiesAdmin() void
labrotaryAdmin() void
staffAdmin() void
isvalidId(Staff[], String, int) boolean
viewPatientReceptionst() void
viewStaffAdmin() void
addPatientReceptionist() void
daysInMonth(int, int) int
login() void
editMedicineInfo() void
editStaffInfo() void
addMedicine() void
viewStaffReceptionst() void
isvalidName(Facility[], String, int) boolean
editPatientAppointment() void
viewDoctor() void
isValidExpiryDate(String, String, String) boolean
viewFacilitiesReceptionist() void
addStaff() void
isvalidName(Lab[], String, int) boolean
viewMedicineReceptionist() void
addDoctor() void
addFacility() void
isvalidName(Medicine[], String, int) boolean
isvalidId(Patient[], String, int) boolean
main(String[]) void
Staff
Staff()
Staff(String, String, String, String, String, String, Strin
stfSalary String
stfSpecialty String
stfSchedule String
daysInMonth(int, int) int
isValidDateOfBirth(String, String, String) boolean
Edit() void
view() void
isvalidId(Staff[], String, int) boolean
calculateAge(String, String, String) String
add() void
StringisIntChecker(String) boolean
StringisIntChecker(String, int) boolean
stfSchedule String
stfSpecialty String
stfSalary String
Doctor
Doctor()
Doctor(String, String, String, String, String, String, Str
DocQualification String
DocSchedule String
DocSpecialty String
Salary String
isvalidId(Doctor[], String, int) boolean
add() void
daysInMonth(int, int) int
StringisIntChecker(String, int) boolean
isValidDateOfBirth(String, String, String) boolean
Edit() void
calculateAge(String, String, String) String
view() void
DocSchedule String
Salary String
DocQualification String
DocSpecialty String
Patient
Patient()
Patient(String, String, String, String, String, String, Str
AppointTime String
AssignedDoc String
add() void
StringisIntChecker(String, int) boolean
Edit() void
getcoDiseases() String
calculateAge(String, String, String) String
daysInMonth(int, int) int
patientAppointment(String) void
setcoDiseases(String) void
isValidDateOfBirth(String, String, String) boolean
view() void
isvalidId(Patient[], String, int) boolean
AssignedDoc String
AppointTime String
Receptionist
Receptionist()
rec2 String
rec2Pw String
rec1Pw String
rec1 String
rec1 String
rec2Pw String
rec1Pw String
rec2 String
Admin
Admin()
admin2Pw String
admin1Pw String
admin1 String
admin2 String
admin2 String
admin1 String
admin2Pw String
admin1Pw String
Common_methods
view() void
Edit() void
add() void
ABS_people
ABS_people()
ABS_people(String, String, String, String, String, String, St
Id String
Name String
Bloodtype String
Gender String
Contact String
Age String
Birthdate String
toString() String
Bloodtype String
Contact String
Name String
Gender String
Age String
Id String
Birthdate String
Lab
Lab()
Lab(String, String, String)
LabName String
LabTestCost String
LabAmount String
isvalidName(Lab[], String, int) boolean
StringisIntChecker(String) boolean
Edit() void
add() void
view() void
LabAmount String
LabName String
LabTestCost String
Medicine
Medicine()
Medicine(String, String, String, String, String)
MedExpiryDate String
MedCompName String
MedCost String
MedAmount String
MedName String
view() void
add() void
isValidExpiryDate(String, String, String) boolean
isvalidName(Medicine[], String, int) boolean
daysInMonth(int, int) int
StringisIntChecker(String) boolean
Edit() void
MedCost String
MedAmount String
MedExpiryDate String
MedCompName String
MedName String
Facility
Facility(String, String, String)
Facility()
amount String
FacName String
FacAvailability String
isvalidName(Facility[], String, int) boolean
Edit() void
add() void
view() void
StringisIntChecker(String) boolean
FacAvailability String
amount String
FacName String

Picture: UML Diagram for Hospital Management system

**Use of abstraction, polymorphism, interface and other OOP concepts:**

As per the diagram we can see that, there is **abstract class ABS_people** which is extended by **Staff class, Patient class** and **Doctor class**.

These classes including **Medicine class**, **Lab class** and **Facility class** implements the **Interface Common_methods.**

In various classes including **Hospital_Management_System_Final class** we used **polymorphism** in **isvalidName()** and **isvalidId()** and also in **StringisIntChecker()** methods.

We kept the necessary methods public and kept the methods that are only needed in their respective classes private and ensured **encapsulation**.

# Description:

First, we would like start with the classes and interface which were needed beside the main method:

## Public Abstract Class ABS_people:

The following class is an abstract class created for the benefit to extend it to other human based classes which are “Staff”, “Doctor” and “Patient”. It’s variables, accessors and mutators are also variables of these “Human Type” classes as they extend the abstract people class. Every extended class uses a system of “3 Digit Id” which is basically a unique id for each object and proper methods are used in the classes to ensure that. The variables in this class are ID, Name, Age, Contact, Birthdate, Bloodtype and Gender with necessary constructors, accessors and mutators.

## Interface Common_methods:

The interface “Common_methods” has 3 methods declared in them. They are-

1: public void add();

2: public void view();

3: public void Edit();

The interface is implemented by all six classes which are “Doctor”, “Patient”, “Staff”, “Lab”, “Medicine” and “Facility” which ensures the classes are forced to use these feature in their own classes.

## Public Class Admin and Public Class Receptionist:

These 2 classes are made only to divide features of the project in 2 separate groups. Admin class and Receptionist class has hard coded passwords and names for 2 admins and 2 receptionists. Means only people with these passwords can access the admin or receptionist panel. In the main method, if you are logging in as admin then you will be given the right to access all the features that have been allowed for an admin to use and the same goes for receptionist as the receptionist has been granted a few yet some unique feature like viewing all the objects and able to edit and select appointment time for patients.

## Public Class Doctor:

Doctor class extends abstract class ABS_people and implements the Common_methods Interface. Doctor class has all the variables of  ABS_people and has 4 more variables which contains a doctors salary, qualification, specialty and schedule.

It has the needed constructors, accessors and mutators.

The other methods are –

1. add(): Appends a new doctor's details, including ID, name, age, contact, birthdate, blood type, gender, salary, qualification, specialty, and schedule, to the "Doctor.txt" file for record-keeping. The input is taken in the main function.

2. view(): Reads and prints all existing doctor records from the "Doctor.txt" file, displaying comprehensive information, including doctor ID, name, age, contact, birthdate, blood type, gender, salary, qualification, specialty, and schedule. This method provides an overview of the doctor database.

3. Edit(): First it reads the info from the "Doctor.txt" file and puts all the info in an array of Doctor objects . Then searches for a specific doctor by ID, allowing the user to selectively modify details such as ID, name, contact, birthdate, blood type, gender, salary, qualification, specialty, and schedule. Then, it rewrites the whole file with the current info from the Doctor array into the "Doctor.txt" file. This method ensures data accuracy, uniqueness of IDs, and includes input validation for numeric entries and date of birth.

4. StringisIntChecker(String str, int num): Checks if a given string is a valid integer with a specified number of digits. This is used to make sure the String variable given by user during editing phone number is valid and is in 11 digits and if the id given by user is in 3 digits. The id of each user must be consisting of a unique 3 digit string or number.

5. isValidDateOfBirth(String day, String month, String year):Checks the input for a date of birth, ensuring it falls within a reasonable range and conforms to a valid date format.

6. daysInMonth(int month, int year): Calculates the number of days in a given month and year for date validation. This is used in “isValidDateOfBirth(String day, String month, String year)” method to see if the month has 28,29,30 or 31 days in it.

7. calculateAge(String day, String month, String year): Calculates the age of a doctor based on the given date of birth.

8. isvalidId(Doctor[] p, String id, int count): Checks if a given doctor ID is valid and unique among existing doctor records. During editing a doctor’s id, the given id can’t be same as any existing doctors.

## Public Class Patient:

Patient class extends abstract class ABS_people and implements the Common_methods Interface. Patient class has all the variables of  ABS_people and has 3 more variables which contains a patient’s category of diseases, appointment time and assigned doctor’s name for appointment.

It has the needed constructors, accessors and mutators.

The other methods are –

1. add(): Appends a new patient's details, including ID, name, age, contact, birthdate, blood type, gender, category of diseases, assigned doctor, and appointment time, to the "Patient.txt" file for record-keeping and scheduling. The input is taken in the main function.

2. view(): Reads and prints all existing patient records from the "Patient.txt" file, displaying comprehensive information, including patient ID, name, age, contact, birthdate, blood type, gender, category of diseases, assigned doctor, and appointment time. This method provides an overview of the patient database.

3. Edit():First it reads the info from the "Patient.txt" file and puts all the info in a array of Patient objects. Then searches for a specific patient by ID, allowing the user to selectively modify details such as ID, name, contact, birthdate, blood type, gender, and category of diseases. Then, it rewrites the whole file with the updated info from the Patient array into the "Patient.txt" file. This method ensures data accuracy, uniqueness of IDs, and includes input validation for numeric entries and date of birth.

4. patientAppointment(String ID): Facilitates the appointment process for a patient by assigning a doctor based on the patient's category of diseases and available doctors. The patient's appointment time and assigned doctor are updated in the "Patient.txt" file. Both files of "Patient.txt" and "Doctor.txt" are read and put in their respective array of objects. Then after concluding its work it rewrites the updated info of the Patient array in the “Patient.txt" file.

5. *StringisIntChecker(String str, int num):* Checks if a given string is a valid integer with a specified number of digits. This is used to make sure the String variable given by user during editing phone number is valid and is in 11 digits and if the id given by user is in 3 digits. The id of each user must be consisting of a unique 3 digit string or number.

6. isValidDateOfBirth(String day, String month, String year): Checks  the input for a date of birth, ensuring it falls within a reasonable range and conforms to a valid date format.

7. daysInMonth(int month, int year): Calculates the number of days in a given month and year for date validation. This is used in “isValidDateOfBirth(String day, String month, String year)” method to see if the month has 28,29,30 or 31 days in it.

8. calculateAge(String day, String month, String year): Calculates the age of a patient based on the given date of birth.

9. isvalidId(Patient[] p, String id, int count): Checks if a given patient ID is valid and unique among existing patient records. During editing a patient’s id, the given id can’t be same as any existing patients.

## Public Class Staff:

Staff class extends abstract class ABS_people and implements the Common_methods Interface. Staff class has all the variables of  ABS_people and has 3 more variables which contains a staff's salary, specialty or working post and work schedule.

It has the needed constructors, accessors and mutators.

The other methods are –

1. add(): Appends a new staff member's details, including ID, name, age, contact, birthdate, blood type, gender, salary, position, and schedule, to the "Staff.txt" file for record-keeping. The input is taken in the main function.

2. view(): Reads and prints all existing staff records from the "Staff.txt" file, displaying comprehensive information, including staff ID, name, age, contact, birthdate, blood type, gender, salary, position, and schedule. This method provides an overview of the staff database.

3. Edit():First it reads the info from the "Staff.txt" file and puts all the info in an  array of Staff objects. Then searches for a specific staff member by ID, allowing the user to selectively modify details such as ID, name, contact, birthdate, blood type, gender, position, and schedule. Then, it rewrites the whole file with the updated info from the Staff array into the "Staff.txt" file. This method ensures data accuracy, uniqueness of IDs, and includes input validation for numeric entries and date of birth.

4. StringisIntChecker(String str, int num): Checks if a given string is a valid integer with a specified number of digits. This is used to make sure the String variable given by user during editing phone number is valid and is in 11 digits and if the id given by user is in 3 digits. The id of each user must be consisting of a unique 3 digit string or number.

5. isValidDateOfBirth(String day, String month, String year): Validates the input for a date of birth, ensuring it falls within a reasonable range and conforms to a valid date format.

6. daysInMonth(int month, int year): Calculates the number of days in a given month and year for date validation. This is used in "isValidDateOfBirth(String day, String month, String year)" method to see if the month has 28,29,30 or 31 days in it.

7. calculateAge(String day, String month, String year): Calculates the age of a staff member based on the given date of birth.

8. isvalidId(Staff[] p, String id, int count): Checks if a given staff ID is valid and unique among existing staff records. During editing a staff's id, the given id can't be same as any existing staff's.

## Public Class Facility:

Facility class implements the Common_methods Interface. Facility class has variables  regarding name, it's availability and how many there exists .

It has the needed constructors, accessors and mutators.

The other methods are –

1. add(): Appends a new facility's details, including name, availability status, and quantity, to the "Facility.txt" file for record-keeping. The input is taken in the main function.

2. view(): Reads and prints all existing facility records from the "Facility.txt" file, displaying information such as facility name, availability status, and quantity. This method provides an overview of the available facilities.

3. Edit():First it reads the info from the "Facility.txt" file and puts all the info in an  array of Facility objects. Then searches searches for a specific facility by name, allowing the user to selectively modify details such as name, quantity, and availability status. Then, it rewrites the whole file with the updated info from the Facility array into the "Facility.txt" file. This method ensures data accuracy, includes input validation for numeric entries, and checks for the uniqueness of facility names.

4. StringisIntChecker(String str): Checks if a given string is a valid integer.

5. isvalidName(Facility[] p, String name, int count): Checks if a given facility name is valid and unique among existing facility records.

## Public Class Lab:

Lab class implements the Common_methods Interface. Lab class has variables  regarding  name, it's cost per usage and how many there exists .

It has the needed constructors, accessors and mutators.

The other methods are –

1. add(): Appends a new lab test's details, including name, quantity, and cost, to the "Lab.txt" file for record-keeping. The input is taken in the main function.

2. view(): Reads and prints all existing lab test records from the "Lab.txt" file, displaying information such as lab test name, total quantity, and cost. This method provides an overview of available lab tests.

3. Edit(): First it reads the info from the "Lab.txt" file and puts all the info in an  array of Lab objects. Then searches for a specific lab test by name, allowing the user to selectively modify details such as name, quantity, and cost. Then, it rewrites the whole file with the updated info from the Lab array into the "Lab.txt" file. This method ensures data accuracy, includes input validation for numeric entries, and checks for the uniqueness of lab test names.

4. *StringisIntChecker(String str):* Checks if a given string is a valid integer.

5. *isvalidName(Lab[] p, String name, int count):* Checks if a given lab test name is valid and unique among existing lab test records.

## Public Class Medicine:

Medicine class implements the Common_methods Interface. Medicine class has variables  regarding medicine name, company name, expiry date, quantity, and cost.

It has the needed constructors, accessors and mutators.

The other methods are –

1. add(): Appends a new medicine's details, including name, company name, expiry date, quantity, and cost, to the "Medicine.txt" file for record-keeping. The input is taken in the main function.

2. view(): Reads and prints all existing medicine records from the "Medicine.txt" file, displaying information such as medicine name, company name, expiry date, quantity, and cost. This method provides an overview of the medicine inventory.

3. *Edit():First it reads the info from the "Medicine.txt" file and puts all the info in an  array of Medicine objects. Then searches for a specific medicine by name, allowing the user to selectively modify details such as name, company name, expiry date, quantity, and cost. Then, it rewrites the whole file with the updated info from the Medicine array into the "Medicine.txt" file. This method ensures data accuracy, includes input validation for numeric entries and expiry dates, and checks for the uniqueness of medicine names.

4. *StringisIntChecker(String str):* Checks if a given string is a valid integer.

5. *isValidExpiryDate(String day, String month, String year):* Validates the input for an expiry date, ensuring it falls within a reasonable range and conforms to a valid date format.

6. *daysInMonth(int month, int year):* Calculates the number of days in a given month and year for date validation. This is used in “isValidExpiryDate(String day, String month, String year)”method to see if the month has 28,29,30 or 31 days in it.

7. *isvalidName(Medicine[] p, String name, int count):* Checks if a given medicine name is valid and unique among existing medicine records.

## Main Class:

We used “Hospital_Management_System_Final” as our main class where we conducted our programme. In this class we only called the “Homescreen();” method in the main function and by the use of method calling and recursion we created the main skeleton of the programme. We tried our best to be careful regarding method calling and using recursions as if not careful enough this type of structure may throw us in an infinite loop and create various opportunities for other logical errors.

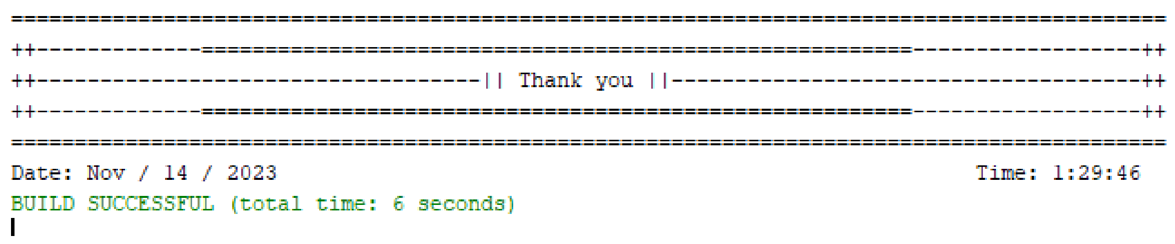
main():In the main method only Homescreen() method is called .

### All methods before Admin panel and Receptionist panel:

DateAndTime(): Prints the current date and time using the `Calendar` class, displaying it in a formatted manner. It contributes to presenting a timestamp in various parts of the program.

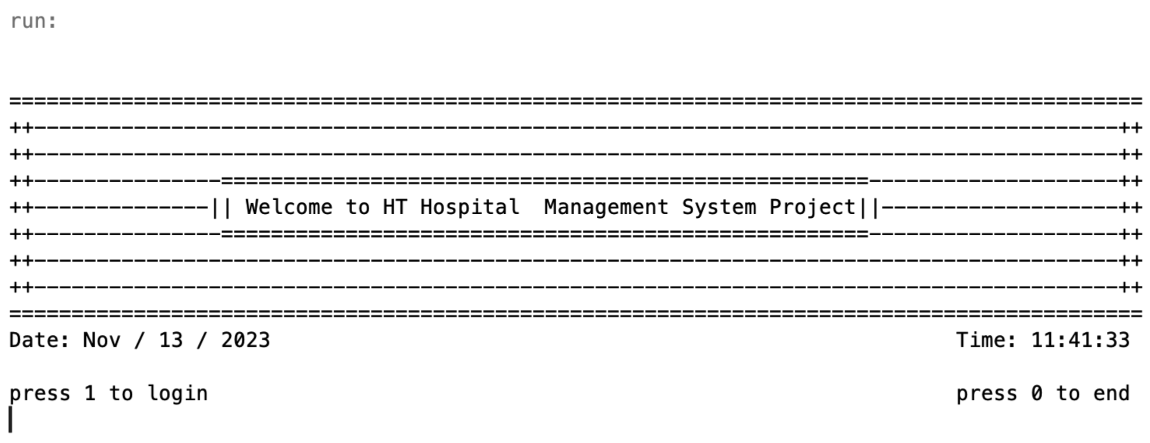
clear(): Clears the console screen by printing a specific number of empty lines, providing a cleaner and more organized display.

endProgramme(): Displays a farewell message along with the current date and time with the DateAndTime() method , offering a structured conclusion to the program.



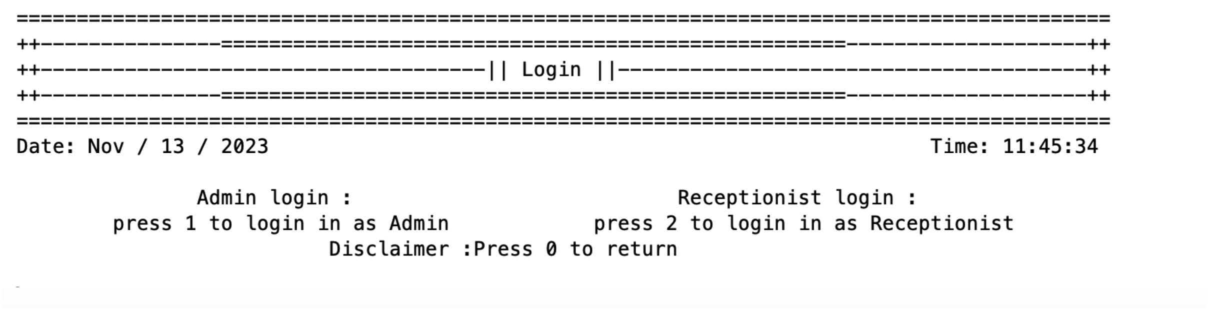
```
===========================================================================================
++-------------========================================================------------------++
++-----------------------------------|| Thank you ||------------------------------------++
++-------------========================================================------------------++
===========================================================================================
Date: Nov / 14 / 2023                                                          Time: 1:29:46
BUILD SUCCESSFUL (total time: 6 seconds)
```

Homescreen(): Displays the initial home screen, prompting the user to either log in as an admin, log in as a receptionist, or exit the program. Handles user input accordingly and through the use of "if else" statements user is given the chance to move forward or end the programme.If wrong input is given then still it will end the programme.



```
run:


==========================================================================================
++------------------------------------------------------------------------------------++
++------------------------------------------------------------------------------------++
++---------------===================================================--------------------++
++--------------|| Welcome to HT Hospital  Management System Project||-------------------++
++---------------===================================================--------------------++
++------------------------------------------------------------------------------------++
++------------------------------------------------------------------------------------++
==========================================================================================
Date: Nov / 13 / 2023                                                          Time: 11:41:33

press 1 to login                                                               press 0 to end
```

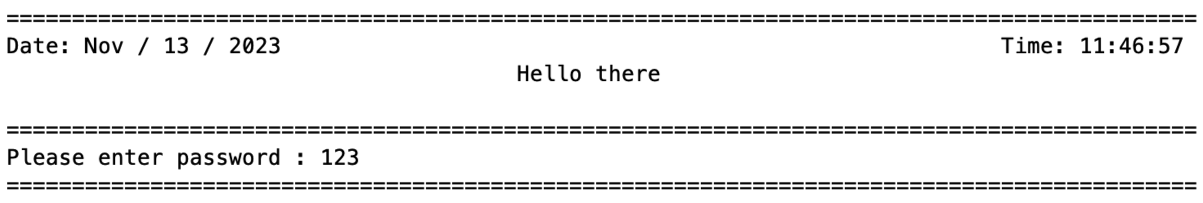
login(): Guides the user through the login process, allowing access for either an admin or a receptionist. Handles invalid inputs and provides the option to return to the home screen.



```
=========================================================================================
++---------------==================================================--------------------++
++------------------------------------|| Login ||--------------------------------------++
++---------------==================================================--------------------++
=========================================================================================
Date: Nov / 13 / 2023                                                      Time: 11:45:34

              Admin login :                                 Receptionist login :
        press 1 to login in as Admin            press 2 to login in as Receptionist
                         Disclaimer :Press 0 to return
```

adminlogin(): Manages the admin login process, verifying the entered password and granting access to the admin panel. Handles incorrect passwords and provides the option to try again or return to the home screen.

Receptionistlogin(): Manages the receptionist login process, verifying the entered password and granting access to the receptionist panel. Handles incorrect passwords and provides the option to try again or return to the home screen.

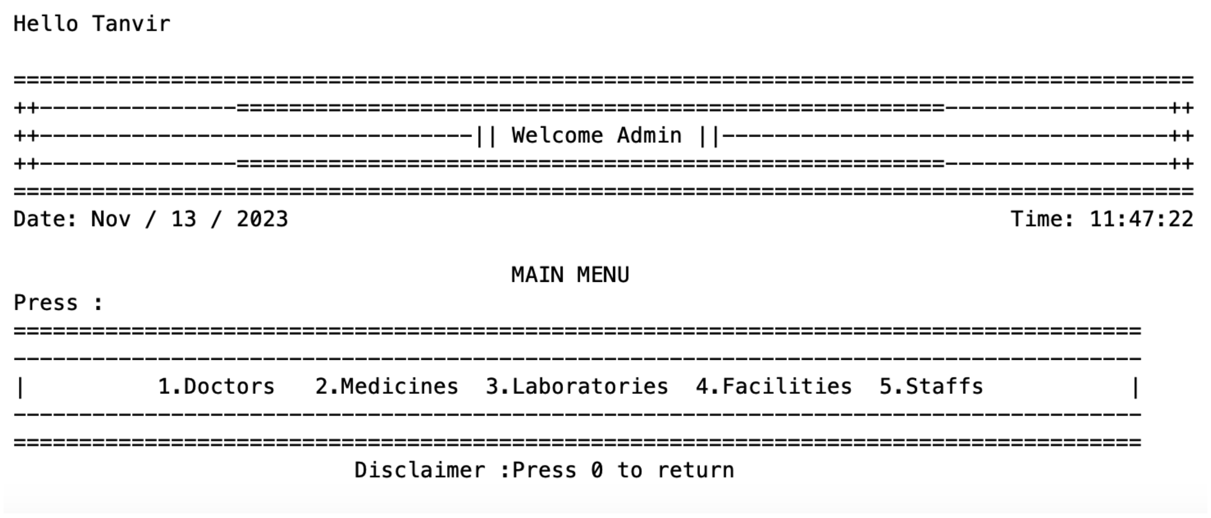


```
=========================================================================================
Date: Nov / 13 / 2023                                                      Time: 11:46:57
                                        Hello there

=========================================================================================
Please enter password : 123
=========================================================================================
```

**Admin panel :**

adminPanel():

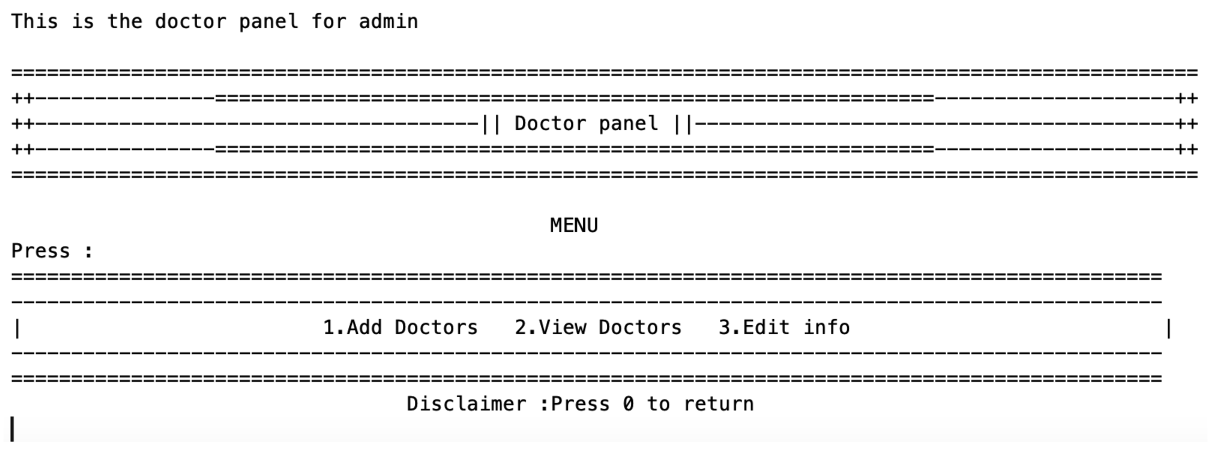
- Displays the admin panel with various options.
- Allows the admin to perform actions like managing doctors, staff, facilities, etc.
- Redirects to the home screen or logs out based on user input.



```
Hello Tanvir

==========================================================================================
++-------------=======================================================---------------++
++-------------------------------|| Welcome Admin ||--------------------------------++
++-------------=======================================================---------------++
==========================================================================================
Date: Nov / 13 / 2023                                                     Time: 11:47:22

                                      MAIN MENU
Press :
=====================================================================================
-------------------------------------------------------------------------------------
|          1.Doctors   2.Medicines  3.Laboratories  4.Facilities  5.Staffs           |
-------------------------------------------------------------------------------------
=====================================================================================
                          Disclaimer :Press 0 to return
```

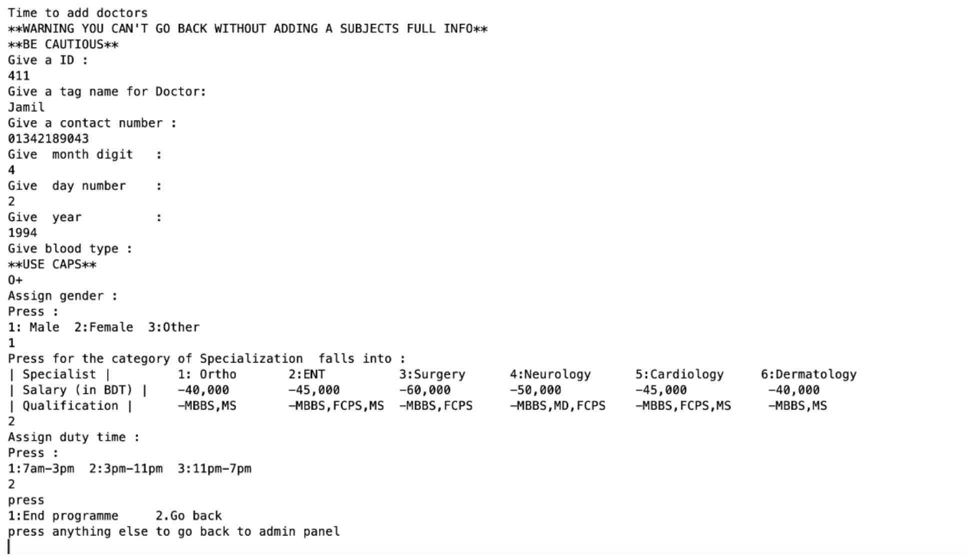
**Doctor part in admin panel**

DoctorPanelAdmin(): This method displays a menu for admin tasks in the doctor panel, allowing the admin to add doctors, view doctor information, edit doctor details, or return to the admin panel based on user input.



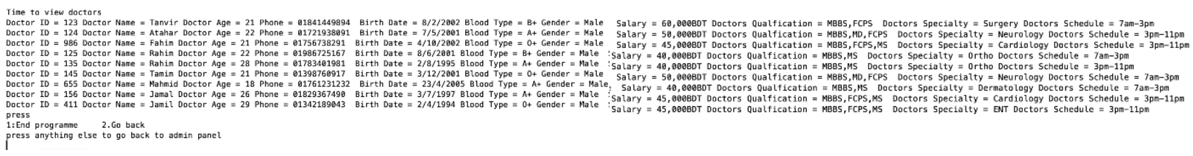
```
This is the doctor panel for admin

==========================================================================================
++-------------=======================================================---------------++
++-------------------------------|| Doctor panel ||---------------------------------++
++-------------=======================================================---------------++
==========================================================================================

                                         MENU
Press :
=====================================================================================
-------------------------------------------------------------------------------------
|                         1.Add Doctors   2.View Doctors   3.Edit info                |
-------------------------------------------------------------------------------------
=====================================================================================
                              Disclaimer :Press 0 to return
|
```

addDoctor(): This method collects information from the admin to add a new doctor to the system. It reads existing doctor data from a file, prompts the admin for new doctor details, checks if the input is valid and updates the file with the new doctor's information.



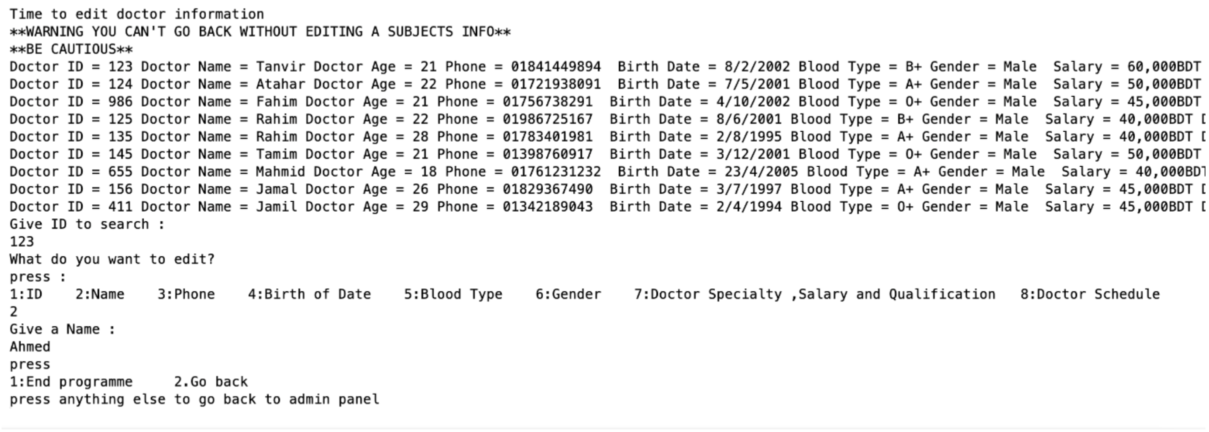
```
Time to add doctors
**WARNING YOU CAN'T GO BACK WITHOUT ADDING A SUBJECTS FULL INFO**
**BE CAUTIOUS**
Give a ID :
411
Give a tag name for Doctor:
Jamil
Give a contact number :
01342189043
Give  month digit   :
4
Give  day number    :
2
Give  year          :
1994
Give blood type :
**USE CAPS**
O+
Assign gender :
Press :
1: Male  2:Female  3:Other
1
Press for the category of Specialization  falls into :
| Specialist |         1: Ortho       2:ENT          3:Surgery      4:Neurology      5:Cardiology     6:Dermatology
| Salary (in BDT) |    -40,000        -45,000        -60,000        -50,000          -45,000           -40,000
| Qualification |      -MBBS,MS       -MBBS,FCPS,MS  -MBBS,FCPS     -MBBS,MD,FCPS    -MBBS,FCPS,MS     -MBBS,MS
2
Assign duty time :
Press :
1:7am-3pm  2:3pm-11pm  3:11pm-7pm
2
press
1:End programme     2.Go back
press anything else to go back to admin panel
|
```

viewDoctor(): This method displays the information of all doctors in the system. It retrieves and prints doctor details from a file, providing options for the admin to either end the program, go back to the doctor panel, or return to the admin panel.



```
Time to view doctors
Doctor ID = 123 Doctor Name = Tanvir Doctor Age = 21 Phone = 01841449894  Birth Date = 8/2/2002 Blood Type = B+ Gender = Male   Salary = 60,000BDT Doctors Qualfication = MBBS,FCPS  Doctors Specialty = Surgery Doctors Schedule = 7am-3pm
Doctor ID = 124 Doctor Name = Atahar Doctor Age = 22 Phone = 01721938091  Birth Date = 7/5/2001 Blood Type = A+ Gender = Male   Salary = 50,000BDT Doctors Qualfication = MBBS,MD,FCPS  Doctors Specialty = Neurology Doctors Schedule = 3pm-11pm
Doctor ID = 986 Doctor Name = Fahim Doctor Age = 21 Phone = 01756738291  Birth Date = 4/10/2002 Blood Type = O+ Gender = Male   Salary = 45,000BDT Doctors Qualfication = MBBS,FCPS,MS  Doctors Specialty = Cardiology Doctors Schedule = 3pm-11pm
Doctor ID = 125 Doctor Name = Rahim Doctor Age = 22 Phone = 01986725167  Birth Date = 8/6/2001 Blood Type = B+ Gender = Male  :Salary = 40,000BDT Doctors Qualfication = MBBS,MS  Doctors Specialty = Ortho Doctors Schedule = 7am-3pm
Doctor ID = 135 Doctor Name = Rahim Doctor Age = 28 Phone = 01783401981  Birth Date = 2/8/1995 Blood Type = A+ Gender = Male  :Salary = 40,000BDT Doctors Qualfication = MBBS,MS  Doctors Specialty = Ortho Doctors Schedule = 3pm-11pm
Doctor ID = 145 Doctor Name = Tamim Doctor Age = 21 Phone = 01398760917  Birth Date = 3/12/2001 Blood Type = O+ Gender = Male   Salary = 50,000BDT Doctors Qualfication = MBBS,MD,FCPS  Doctors Specialty = Neurology Doctors Schedule = 7am-3pm
Doctor ID = 655 Doctor Name = Mahmid Doctor Age = 18 Phone = 01761231232  Birth Date = 23/4/2005 Blood Type = A+ Gender = Male,  Salary = 40,000BDT Doctors Qualfication = MBBS,MS  Doctors Specialty = Dermatology Doctors Schedule = 7am-3pm
Doctor ID = 156 Doctor Name = Jamal Doctor Age = 26 Phone = 01829367490  Birth Date = 3/7/1997 Blood Type = A+ Gender = Male  :Salary = 45,000BDT Doctors Qualfication = MBBS,FCPS,MS  Doctors Specialty = Cardiology Doctors Schedule = 3pm-11pm
Doctor ID = 411 Doctor Name = Jamil Doctor Age = 29 Phone = 01342189043  Birth Date = 2/4/1994 Blood Type = O+ Gender = Male  :Salary = 45,000BDT Doctors Qualfication = MBBS,FCPS,MS  Doctors Specialty = ENT Doctors Schedule = 3pm-11pm
press
1:End programme     2.Go back
press anything else to go back to admin panel
|
```

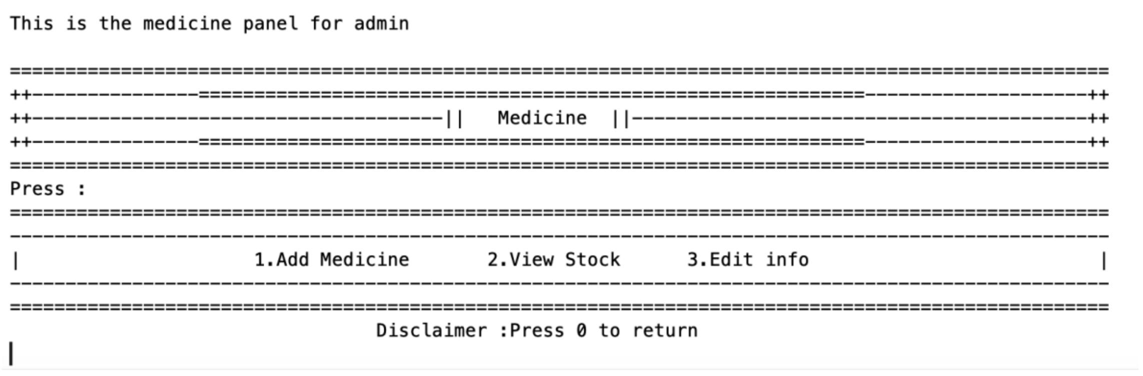
editDoctor(): This method allows the admin to edit the information of a specific doctor. It first displays the current doctor details, prompts the admin for modifications, and updates the doctor's information in the file. The admin can choose to end the program, go back to the doctor panel, or return to the admin panel after editing.



```
Time to edit doctor information
**WARNING YOU CAN'T GO BACK WITHOUT EDITING A SUBJECTS INFO**
**BE CAUTIOUS**
Doctor ID = 123 Doctor Name = Tanvir Doctor Age = 21 Phone = 01841449894  Birth Date = 8/2/2002 Blood Type = B+ Gender = Male  Salary = 60,000BDT
Doctor ID = 124 Doctor Name = Atahar Doctor Age = 22 Phone = 01721938091  Birth Date = 7/5/2001 Blood Type = A+ Gender = Male  Salary = 50,000BDT
Doctor ID = 986 Doctor Name = Fahim Doctor Age = 21 Phone = 01756738291  Birth Date = 4/10/2002 Blood Type = O+ Gender = Male  Salary = 45,000BDT
Doctor ID = 125 Doctor Name = Rahim Doctor Age = 22 Phone = 01986725167  Birth Date = 8/6/2001 Blood Type = B+ Gender = Male  Salary = 40,000BDT
Doctor ID = 135 Doctor Name = Rahim Doctor Age = 28 Phone = 01783401981  Birth Date = 2/8/1995 Blood Type = A+ Gender = Male  Salary = 40,000BDT
Doctor ID = 145 Doctor Name = Tamim Doctor Age = 21 Phone = 01398760917  Birth Date = 3/12/2001 Blood Type = O+ Gender = Male  Salary = 50,000BDT
Doctor ID = 655 Doctor Name = Mahmid Doctor Age = 18 Phone = 01761231232  Birth Date = 23/4/2005 Blood Type = A+ Gender = Male  Salary = 40,000BD
Doctor ID = 156 Doctor Name = Jamal Doctor Age = 26 Phone = 01829367490  Birth Date = 3/7/1997 Blood Type = A+ Gender = Male  Salary = 45,000BDT
Doctor ID = 411 Doctor Name = Jamil Doctor Age = 29 Phone = 01342189043  Birth Date = 2/4/1994 Blood Type = O+ Gender = Male  Salary = 45,000BDT
Give ID to search :
123
What do you want to edit?
press :
1:ID    2:Name    3:Phone    4:Birth of Date    5:Blood Type    6:Gender    7:Doctor Specialty ,Salary and Qualification   8:Doctor Schedule
2
Give a Name :
Ahmed
press
1:End programme     2.Go back
press anything else to go back to admin panel
```

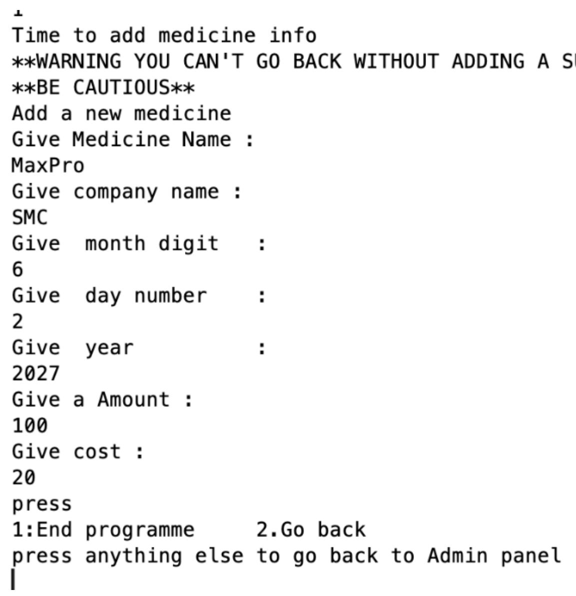
**Medicine part in Admin**

medicineAdmin(): Displays a menu for admin tasks in the medicine panel, allowing the admin to add new medicines, view current stock, edit medicine information, or return to the admin panel based on user input.



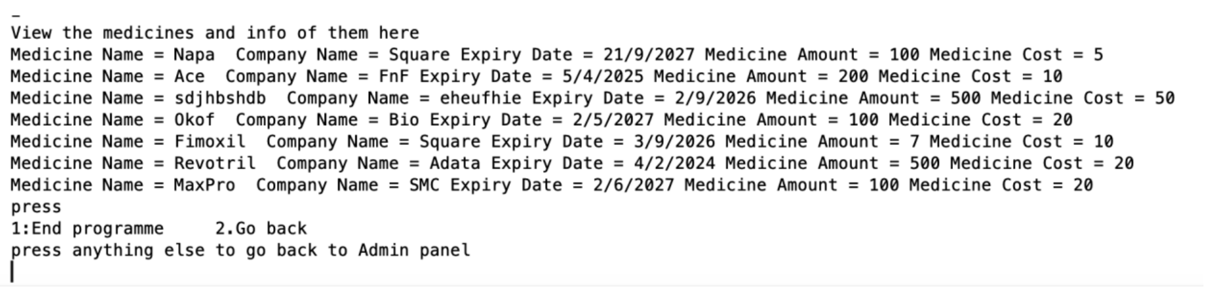
```
This is the medicine panel for admin

===================================================================================================
++---------------============================================================--------------------++
++-------------------------------------||   Medicine  ||-----------------------------------------++
++---------------============================================================--------------------++
===================================================================================================
Press :
===================================================================================================
---------------------------------------------------------------------------------------------------
|                     1.Add Medicine       2.View Stock      3.Edit info                          |
---------------------------------------------------------------------------------------------------
===================================================================================================
                                 Disclaimer :Press 0 to return
|
```

addMedicine(): Collects information from the admin to add a new medicine to the system. It reads existing medicine data from a file, prompts the admin for new medicine details, checks the input for validation and updates the file with the new medicine's information.



```
1
Time to add medicine info
**WARNING YOU CAN'T GO BACK WITHOUT ADDING A SUBJECTS FULL INFO**
**BE CAUTIOUS**
Add a new medicine
Give Medicine Name :
MaxPro
Give company name :
SMC
Give  month digit   :
6
Give  day number    :
2
Give  year          :
2027
Give a Amount :
100
Give cost :
20
press
1:End programme     2.Go back
press anything else to go back to Admin panel
|
```

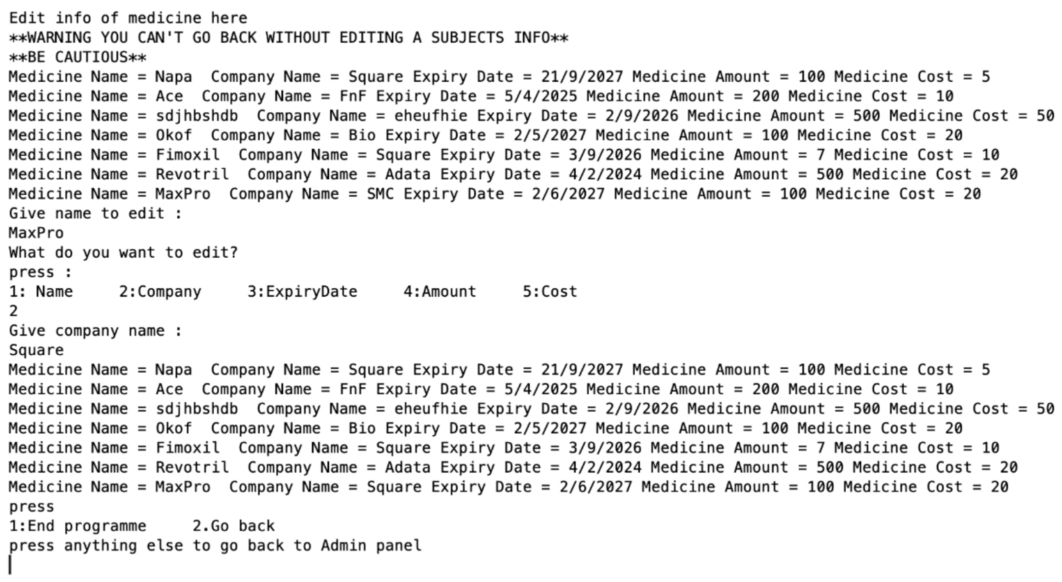
3. viewStockAdmin(): Displays the information of all medicines in the system, retrieved from a file. Provides options for the admin to either end the program, go back to the medicine panel, or return to the admin panel.



```
View the medicines and info of them here
Medicine Name = Napa  Company Name = Square Expiry Date = 21/9/2027 Medicine Amount = 100 Medicine Cost = 5
Medicine Name = Ace  Company Name = FnF Expiry Date = 5/4/2025 Medicine Amount = 200 Medicine Cost = 10
Medicine Name = sdjhbshdb  Company Name = eheufhie Expiry Date = 2/9/2026 Medicine Amount = 500 Medicine Cost = 50
Medicine Name = Okof  Company Name = Bio Expiry Date = 2/5/2027 Medicine Amount = 100 Medicine Cost = 20
Medicine Name = Fimoxil  Company Name = Square Expiry Date = 3/9/2026 Medicine Amount = 7 Medicine Cost = 10
Medicine Name = Revotril  Company Name = Adata Expiry Date = 4/2/2024 Medicine Amount = 500 Medicine Cost = 20
Medicine Name = MaxPro  Company Name = SMC Expiry Date = 2/6/2027 Medicine Amount = 100 Medicine Cost = 20
press
1:End programme     2.Go back
press anything else to go back to Admin panel
|
```

4. *editMedicineInfo():* Allows the admin to edit the information of a specific medicine. It first displays the current medicine details, prompts the admin for modifications, updates the medicine's

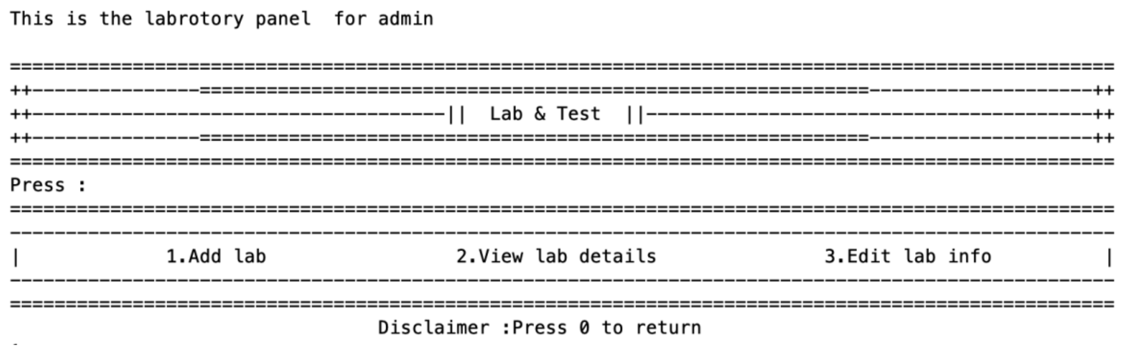
information in the file, and then provides options to end the program, go back to the medicine panel, or return to the admin panel after editing.



```
Edit info of medicine here
**WARNING YOU CAN'T GO BACK WITHOUT EDITING A SUBJECTS INFO**
**BE CAUTIOUS**
Medicine Name = Napa  Company Name = Square Expiry Date = 21/9/2027 Medicine Amount = 100 Medicine Cost = 5
Medicine Name = Ace  Company Name = FnF Expiry Date = 5/4/2025 Medicine Amount = 200 Medicine Cost = 10
Medicine Name = sdjhbshdb  Company Name = eheufhie Expiry Date = 2/9/2026 Medicine Amount = 500 Medicine Cost = 50
Medicine Name = Okof  Company Name = Bio Expiry Date = 2/5/2027 Medicine Amount = 100 Medicine Cost = 20
Medicine Name = Fimoxil  Company Name = Square Expiry Date = 3/9/2026 Medicine Amount = 7 Medicine Cost = 10
Medicine Name = Revotril  Company Name = Adata Expiry Date = 4/2/2024 Medicine Amount = 500 Medicine Cost = 20
Medicine Name = MaxPro  Company Name = SMC Expiry Date = 2/6/2027 Medicine Amount = 100 Medicine Cost = 20
Give name to edit :
MaxPro
What do you want to edit?
press :
1: Name     2:Company     3:ExpiryDate     4:Amount     5:Cost
2
Give company name :
Square
Medicine Name = Napa  Company Name = Square Expiry Date = 21/9/2027 Medicine Amount = 100 Medicine Cost = 5
Medicine Name = Ace  Company Name = FnF Expiry Date = 5/4/2025 Medicine Amount = 200 Medicine Cost = 10
Medicine Name = sdjhbshdb  Company Name = eheufhie Expiry Date = 2/9/2026 Medicine Amount = 500 Medicine Cost = 50
Medicine Name = Okof  Company Name = Bio Expiry Date = 2/5/2027 Medicine Amount = 100 Medicine Cost = 20
Medicine Name = Fimoxil  Company Name = Square Expiry Date = 3/9/2026 Medicine Amount = 7 Medicine Cost = 10
Medicine Name = Revotril  Company Name = Adata Expiry Date = 4/2/2024 Medicine Amount = 500 Medicine Cost = 20
Medicine Name = MaxPro  Company Name = Square Expiry Date = 2/6/2027 Medicine Amount = 100 Medicine Cost = 20
press
1:End programme     2.Go back
press anything else to go back to Admin panel
|
```

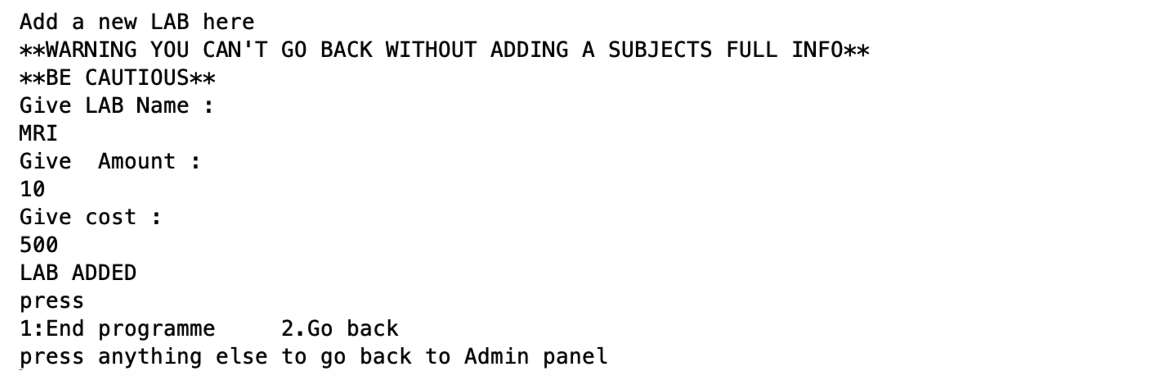
**Lab Part in Admin**

labrotaryAdmin(): Displays a menu for admin tasks in the laboratory panel, providing options to add a new lab, view lab details, edit lab information, or return to the admin panel based on user input.



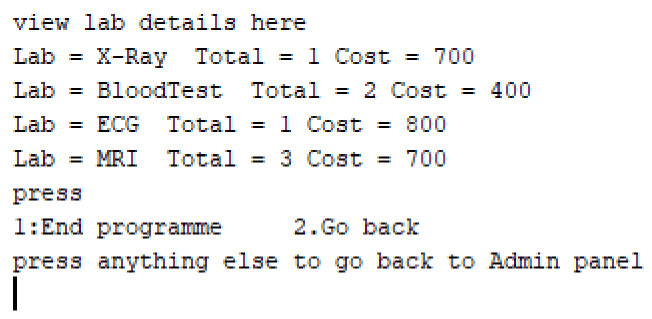
```
This is the labrotory panel  for admin

==========================================================================================
++---------------======================================================---------------------++
++------------------------------------|| Lab & Test ||-----------------------------------------++
++---------------======================================================---------------------++
==========================================================================================
Press :
==========================================================================================
------------------------------------------------------------------------------------------
|            1.Add lab               2.View lab details             3.Edit lab info          |
------------------------------------------------------------------------------------------
==========================================================================================
                              Disclaimer :Press 0 to return
```

addLab(): Gathers information from the admin to add a new lab to the system. It reads existing lab data from a file, prompts the admin for new lab details, validates input, and updates the file with the new lab's information.



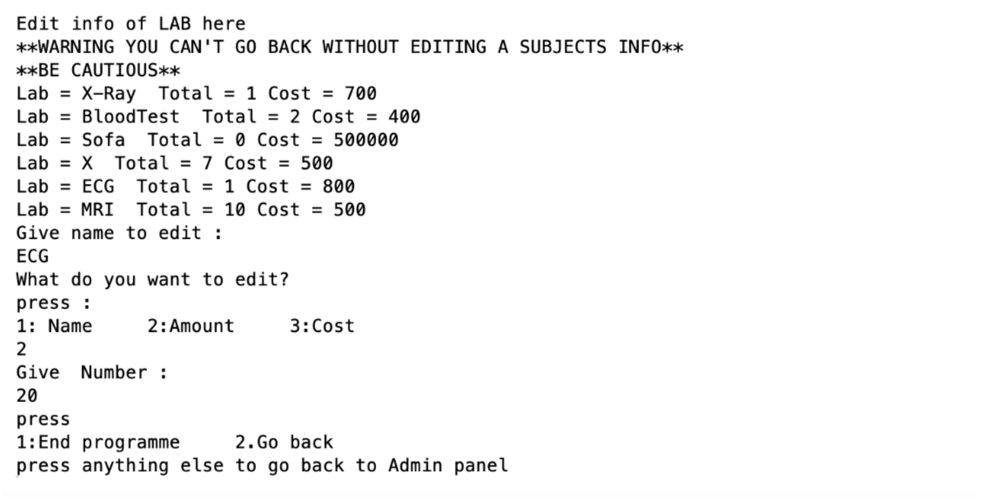
```
Add a new LAB here
**WARNING YOU CAN'T GO BACK WITHOUT ADDING A SUBJECTS FULL INFO**
**BE CAUTIOUS**
Give LAB Name :
MRI
Give  Amount :
10
Give cost :
500
LAB ADDED
press
1:End programme     2.Go back
press anything else to go back to Admin panel
```

viewLabDetailsAdmin(): Displays information about all labs in the system, retrieved from a file. Provides options for the admin to either end the program, go back to the laboratory panel, or return to the admin panel.



```
view lab details here
Lab = X-Ray  Total = 1 Cost = 700
Lab = BloodTest  Total = 2 Cost = 400
Lab = ECG  Total = 1 Cost = 800
Lab = MRI  Total = 3 Cost = 700
press
1:End programme     2.Go back
press anything else to go back to Admin panel
```

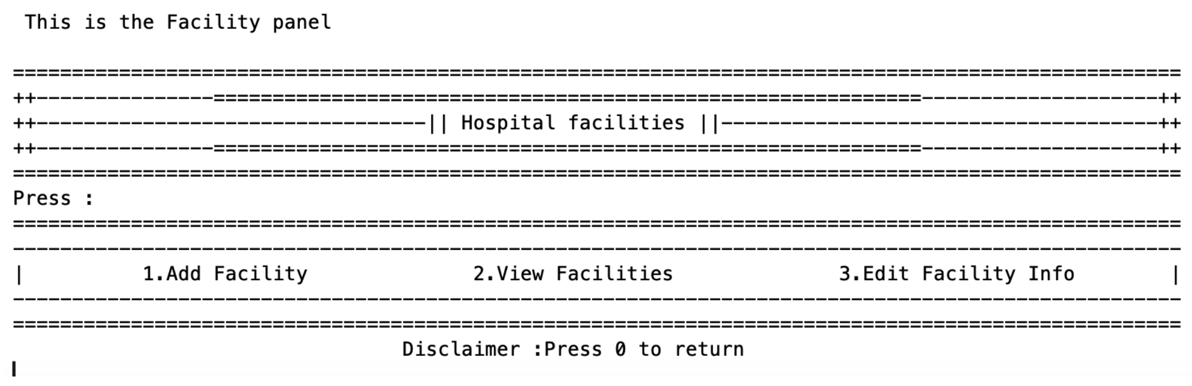
editLabInfo(): Allows the admin to edit the information of a specific lab. It first displays the current lab details, prompts the admin for modifications, updates the lab's information in the file, and then provides options to end the program, go back to the laboratory panel, or return to the admin panel after editing.



```
Edit info of LAB here
**WARNING YOU CAN'T GO BACK WITHOUT EDITING A SUBJECTS INFO**
**BE CAUTIOUS**
Lab = X-Ray  Total = 1 Cost = 700
Lab = BloodTest  Total = 2 Cost = 400
Lab = Sofa  Total = 0 Cost = 500000
Lab = X  Total = 7 Cost = 500
Lab = ECG  Total = 1 Cost = 800
Lab = MRI  Total = 10 Cost = 500
Give name to edit :
ECG
What do you want to edit?
press :
1: Name     2:Amount     3:Cost
2
Give  Number :
20
press
1:End programme     2.Go back
press anything else to go back to Admin panel
```

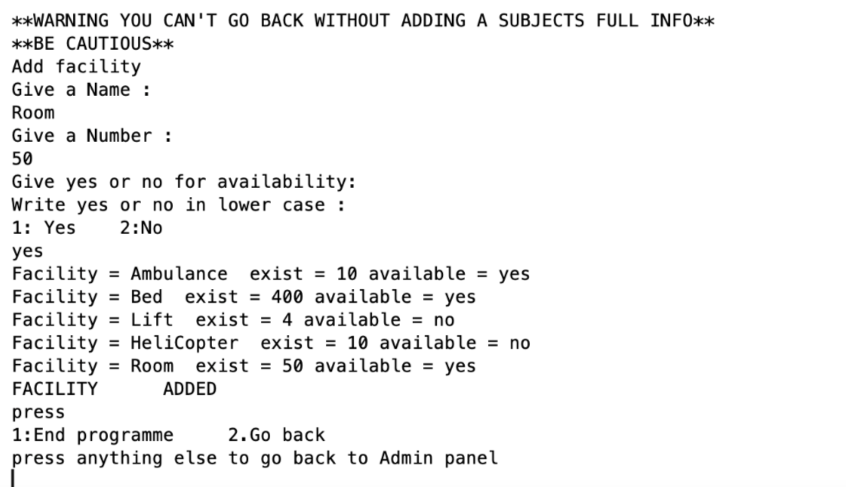
## Facilities part in admin

facilitiesAdmin(): Displays a menu for admin tasks in the facilities panel, allowing the admin to add new facilities, view facility details, edit facility information, or return to the admin panel based on user input.

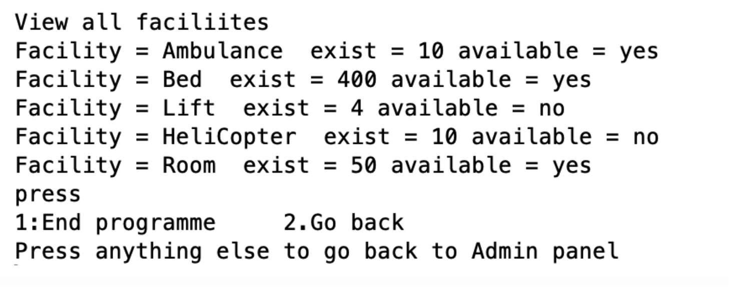


```
 This is the Facility panel

==============================================================================================================
++------------------=================================================================---------------------++
++-------------------------------------|| Hospital facilities ||----------------------------------------++
++------------------=================================================================---------------------++
==============================================================================================================
Press :
==============================================================================================================
--------------------------------------------------------------------------------------------------------------
|          1.Add Facility              2.View Facilities              3.Edit Facility Info        |
--------------------------------------------------------------------------------------------------------------
==============================================================================================================
                                    Disclaimer :Press 0 to return
```

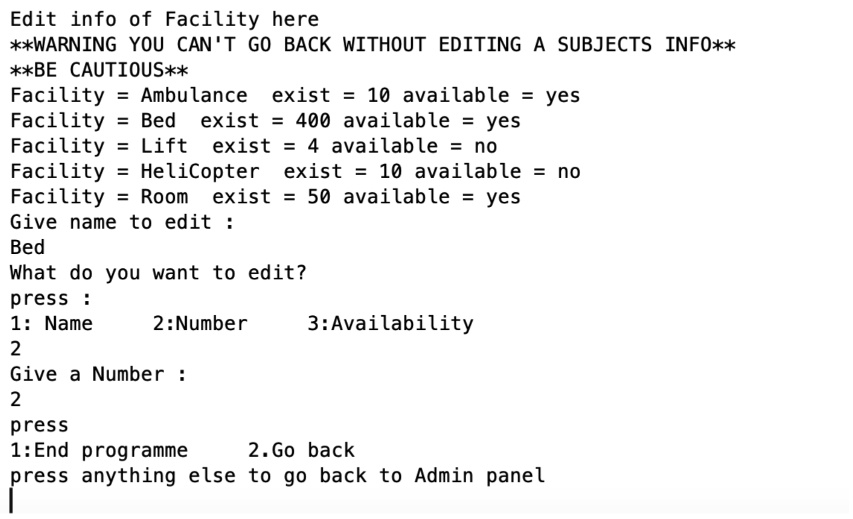
addFacility(): Gathers information from the admin to add a new facility to the system. It reads existing facility data from a file, prompts the admin for new facility details, validates input, updates the file with the new facility's information, and provides options to end the program, go back to the facilities panel, or return to the admin panel.


```
**WARNING YOU CAN'T GO BACK WITHOUT ADDING A SUBJECTS FULL INFO**
**BE CAUTIOUS**
Add facility
Give a Name :
Room
Give a Number :
50
Give yes or no for availability:
Write yes or no in lower case :
1: Yes    2:No
yes
Facility = Ambulance  exist = 10 available = yes
Facility = Bed  exist = 400 available = yes
Facility = Lift  exist = 4 available = no
Facility = HeliCopter  exist = 10 available = no
Facility = Room  exist = 50 available = yes
FACILITY      ADDED
press
1:End programme     2.Go back
press anything else to go back to Admin panel
|
```

viewFacilitiesAdmin(): Displays information about all facilities in the system, retrieved from a file. Provides options for the admin to either end the program, go back to the facilities panel, or return to the admin panel.

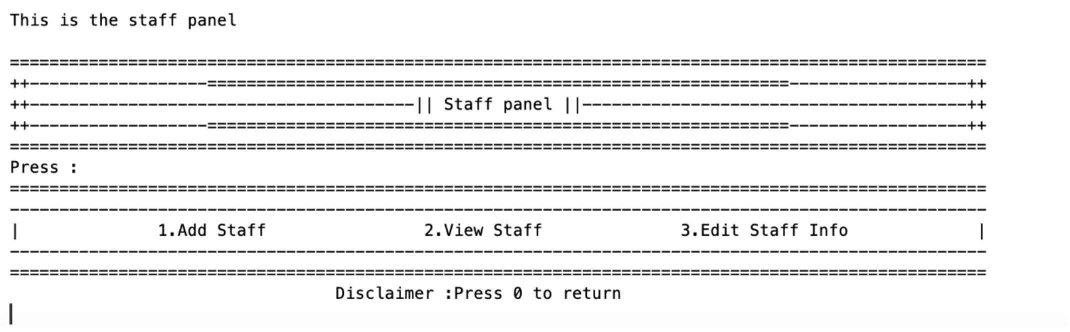

```
View all faciliites
Facility = Ambulance  exist = 10 available = yes
Facility = Bed  exist = 400 available = yes
Facility = Lift  exist = 4 available = no
Facility = HeliCopter  exist = 10 available = no
Facility = Room  exist = 50 available = yes
press
1:End programme     2.Go back
Press anything else to go back to Admin panel
```

editFacilityInfo(): Allows the admin to edit the information of a specific facility. It first displays the current facility details, prompts the admin for modifications, updates the facility's information in the file, and then provides options to end the program, go back to the facilities panel, or return to the admin panel after editing.


```
Edit info of Facility here
**WARNING YOU CAN'T GO BACK WITHOUT EDITING A SUBJECTS INFO**
**BE CAUTIOUS**
Facility = Ambulance  exist = 10 available = yes
Facility = Bed  exist = 400 available = yes
Facility = Lift  exist = 4 available = no
Facility = HeliCopter  exist = 10 available = no
Facility = Room  exist = 50 available = yes
Give name to edit :
Bed
What do you want to edit?
press :
1: Name     2:Number     3:Availability
2
Give a Number :
2
press
1:End programme     2.Go back
press anything else to go back to Admin panel
|
```

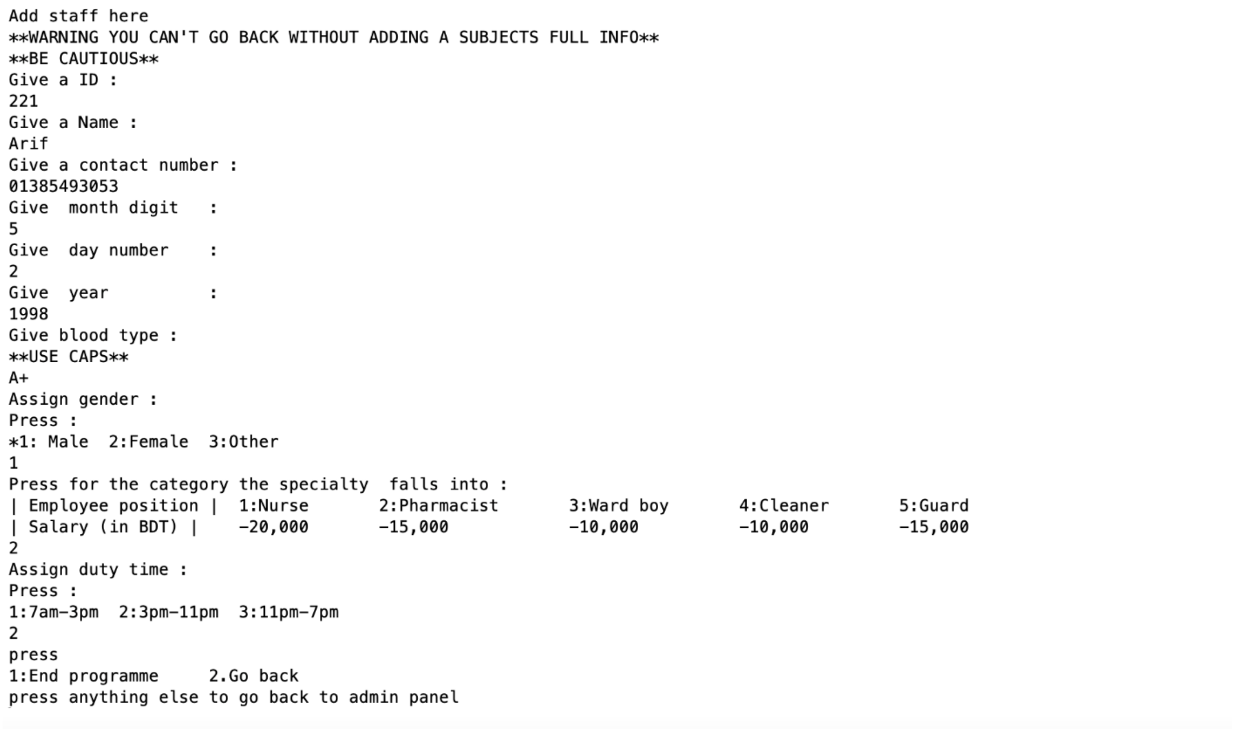
**Staff part in admin**

staffAdmin(): Displays a menu for admin tasks in the staff panel, allowing the admin to add new staff, view staff details, edit staff information, or return to the admin panel based on user input.



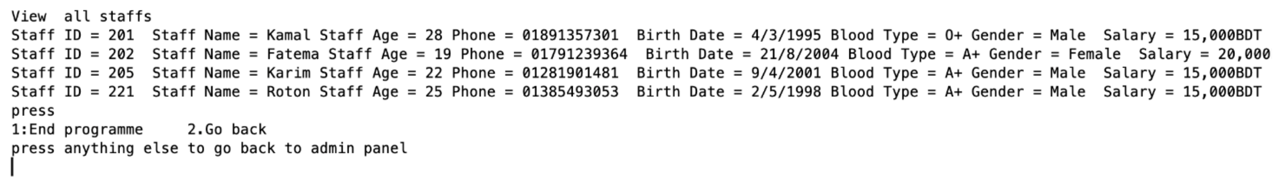
```
This is the staff panel

====================================================================================================
++------------------============================================================------------------++
++----------------------------------------|| Staff panel ||---------------------------------------++
++------------------============================================================------------------++
====================================================================================================
Press :
====================================================================================================
----------------------------------------------------------------------------------------------------
|              1.Add Staff                2.View Staff              3.Edit Staff Info             |
----------------------------------------------------------------------------------------------------
====================================================================================================
                                 Disclaimer :Press 0 to return
|
```

addStaff(): Gathers information from the admin to add a new staff member to the system. Reads existing staff data from a file, prompts the admin for new staff details, validates input, updates the file with the new staff's information, and provides options to end the program, go back to the staff panel, or return to the admin panel.



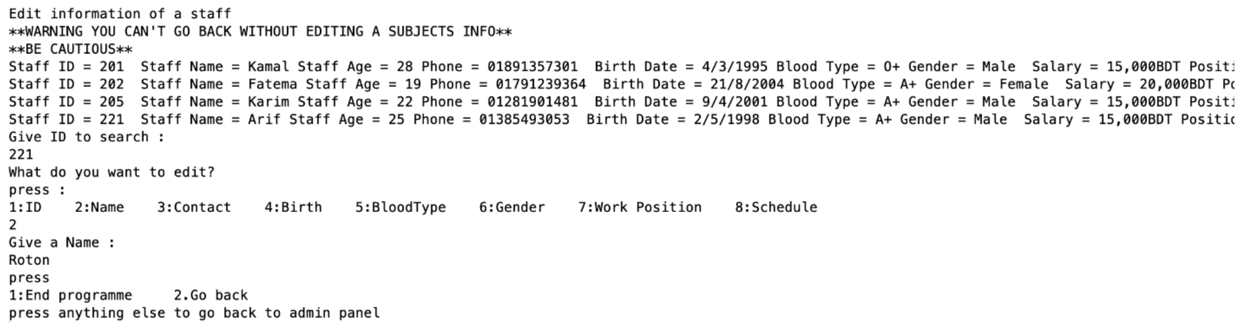
```
Add staff here
**WARNING YOU CAN'T GO BACK WITHOUT ADDING A SUBJECTS FULL INFO**
**BE CAUTIOUS**
Give a ID :
221
Give a Name :
Arif
Give a contact number :
01385493053
Give  month digit   :
5
Give  day number    :
2
Give  year          :
1998
Give blood type :
**USE CAPS**
A+
Assign gender :
Press :
*1: Male  2:Female  3:Other
1
Press for the category the specialty  falls into :
| Employee position |  1:Nurse       2:Pharmacist       3:Ward boy       4:Cleaner       5:Guard
| Salary (in BDT) |    -20,000       -15,000            -10,000          -10,000         -15,000
2
Assign duty time :
Press :
1:7am-3pm  2:3pm-11pm  3:11pm-7pm
2
press
1:End programme     2.Go back
press anything else to go back to admin panel
```

viewStaffAdmin(): Displays information about all staff members in the system, retrieved from a file. Provides options for the admin to either end the program, go back to the staff panel, or return to the admin panel.



```
View  all staffs
Staff ID = 201  Staff Name = Kamal Staff Age = 28 Phone = 01891357301  Birth Date = 4/3/1995 Blood Type = O+ Gender = Male  Salary = 15,000BDT
Staff ID = 202  Staff Name = Fatema Staff Age = 19 Phone = 01791239364  Birth Date = 21/8/2004 Blood Type = A+ Gender = Female  Salary = 20,000
Staff ID = 205  Staff Name = Karim Staff Age = 22 Phone = 01281901481  Birth Date = 9/4/2001 Blood Type = A+ Gender = Male  Salary = 15,000BDT
Staff ID = 221  Staff Name = Roton Staff Age = 25 Phone = 01385493053  Birth Date = 2/5/1998 Blood Type = A+ Gender = Male  Salary = 15,000BDT
press
1:End programme     2.Go back
press anything else to go back to admin panel
|
```

editStaffInfo(): Allows the admin to edit the information of a specific staff member. Displays the current staff details, prompts the admin for modifications, updates the staff's information in the file, and then provides options to end the program, go back to the staff panel, or return to the admin panel after editing.

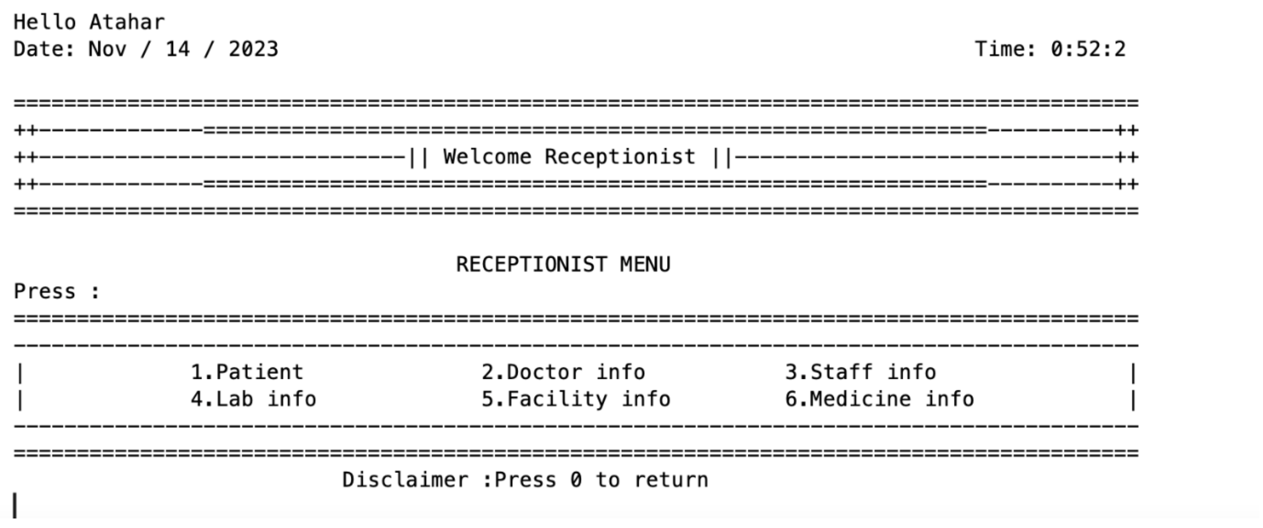


```
Edit information of a staff
**WARNING YOU CAN'T GO BACK WITHOUT EDITING A SUBJECTS INFO**
**BE CAUTIOUS**
Staff ID = 201  Staff Name = Kamal Staff Age = 28 Phone = 01891357301  Birth Date = 4/3/1995 Blood Type = O+ Gender = Male  Salary = 15,000BDT Positi
Staff ID = 202  Staff Name = Fatema Staff Age = 19 Phone = 01791239364  Birth Date = 21/8/2004 Blood Type = A+ Gender = Female  Salary = 20,000BDT Po
Staff ID = 205  Staff Name = Karim Staff Age = 22 Phone = 01281901481  Birth Date = 9/4/2001 Blood Type = A+ Gender = Male  Salary = 15,000BDT Positi
Staff ID = 221  Staff Name = Arif Staff Age = 25 Phone = 01385493053  Birth Date = 2/5/1998 Blood Type = A+ Gender = Male  Salary = 15,000BDT Positi
Give ID to search :
221
What do you want to edit?
press :
1:ID    2:Name    3:Contact    4:Birth    5:BloodType    6:Gender    7:Work Position    8:Schedule
2
Give a Name :
Roton
press
1:End programme     2.Go back
press anything else to go back to admin panel
```

**Receptionist panel:**

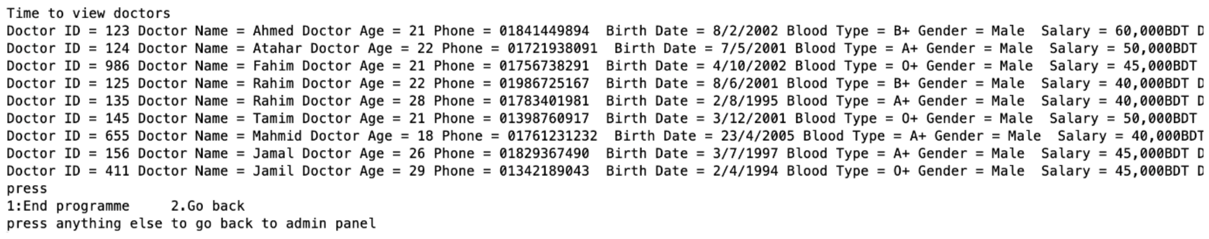
---

receptionistPanel(): Displays a menu for the receptionist, allowing them to access information about patients, doctors, staff, lab details, facility information, and medicine details. Based on user input, it redirects to specific methods to handle each category or returns to the login screen.



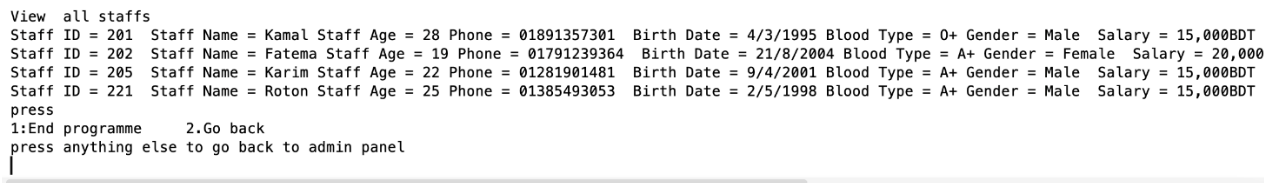
```
Hello Atahar
Date: Nov / 14 / 2023                                                      Time: 0:52:2

=========================================================================================
++-------------==============================================================----------++
++-----------------------------|| Welcome Receptionist ||------------------------------++
++-------------==============================================================----------++
=========================================================================================

                                    RECEPTIONIST MENU
Press :
=========================================================================================
-----------------------------------------------------------------------------------------
|             1.Patient              2.Doctor info           3.Staff info               |
|             4.Lab info             5.Facility info         6.Medicine info            |
-----------------------------------------------------------------------------------------
=========================================================================================
                          Disclaimer :Press 0 to return
|
```

viewDoctorReceptionist():Displays information about all doctors in the system, retrieved from a file. Provides options for the receptionist to either end the program, go back to the receptionist panel, or return to the login screen after viewing doctor details.



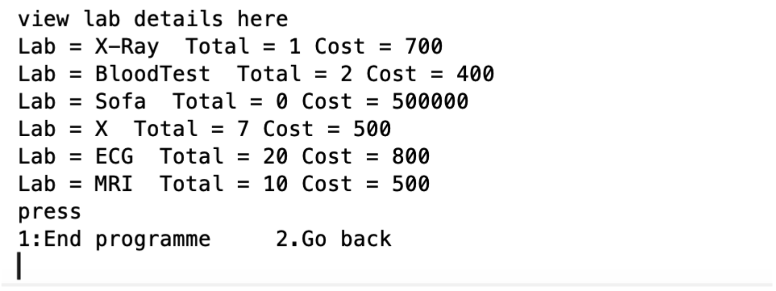
```
Time to view doctors
Doctor ID = 123 Doctor Name = Ahmed Doctor Age = 21 Phone = 01841449894  Birth Date = 8/2/2002 Blood Type = B+ Gender = Male  Salary = 60,000BDT D
Doctor ID = 124 Doctor Name = Atahar Doctor Age = 22 Phone = 01721938091  Birth Date = 7/5/2001 Blood Type = A+ Gender = Male  Salary = 50,000BDT
Doctor ID = 986 Doctor Name = Fahim Doctor Age = 21 Phone = 01756738291  Birth Date = 4/10/2002 Blood Type = O+ Gender = Male  Salary = 45,000BDT
Doctor ID = 125 Doctor Name = Rahim Doctor Age = 22 Phone = 01986725167  Birth Date = 8/6/2001 Blood Type = B+ Gender = Male  Salary = 40,000BDT D
Doctor ID = 135 Doctor Name = Rahim Doctor Age = 28 Phone = 01783401981  Birth Date = 2/8/1995 Blood Type = A+ Gender = Male  Salary = 40,000BDT D
Doctor ID = 145 Doctor Name = Tamim Doctor Age = 21 Phone = 01398760917  Birth Date = 3/12/2001 Blood Type = O+ Gender = Male  Salary = 50,000BDT
Doctor ID = 655 Doctor Name = Mahmid Doctor Age = 18 Phone = 01761231232  Birth Date = 23/4/2005 Blood Type = A+ Gender = Male  Salary = 40,000BDT
Doctor ID = 156 Doctor Name = Jamal Doctor Age = 26 Phone = 01829367490  Birth Date = 3/7/1997 Blood Type = A+ Gender = Male  Salary = 45,000BDT D
Doctor ID = 411 Doctor Name = Jamil Doctor Age = 29 Phone = 01342189043  Birth Date = 2/4/1994 Blood Type = O+ Gender = Male  Salary = 45,000BDT D
press
1:End programme     2.Go back
press anything else to go back to admin panel
```

viewStaffReceptionst(): Displays detailed information about all staff members in the system, retrieved from a file. Provides options for the receptionist to either end the program, go back to the receptionist panel, or return to the admin panel after viewing staff details.



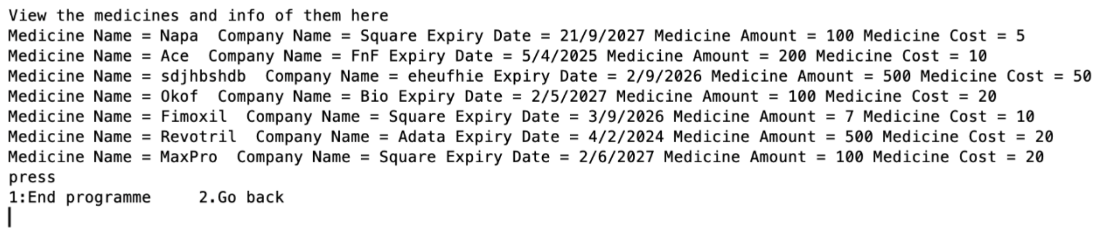
```
View  all staffs
Staff ID = 201  Staff Name = Kamal Staff Age = 28 Phone = 01891357301  Birth Date = 4/3/1995 Blood Type = O+ Gender = Male  Salary = 15,000BDT
Staff ID = 202  Staff Name = Fatema Staff Age = 19 Phone = 01791239364  Birth Date = 21/8/2004 Blood Type = A+ Gender = Female  Salary = 20,000
Staff ID = 205  Staff Name = Karim Staff Age = 22 Phone = 01281901481  Birth Date = 9/4/2001 Blood Type = A+ Gender = Male  Salary = 15,000BDT
Staff ID = 221  Staff Name = Roton Staff Age = 25 Phone = 01385493053  Birth Date = 2/5/1998 Blood Type = A+ Gender = Male  Salary = 15,000BDT
press
1:End programme     2.Go back
press anything else to go back to admin panel
|
```

viewLabInfoReceptionist(): Allows the receptionist to view details of the laboratory, including available tests and their prices. The receptionist can choose to end the program or go back to the receptionist panel after viewing lab information.



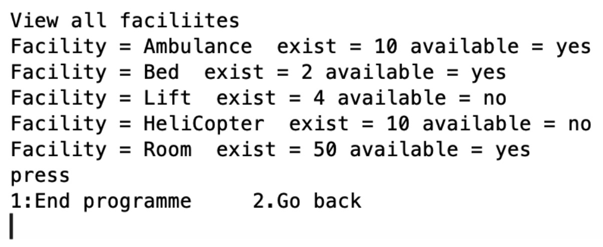
```
view lab details here
Lab = X-Ray  Total = 1 Cost = 700
Lab = BloodTest  Total = 2 Cost = 400
Lab = Sofa  Total = 0 Cost = 500000
Lab = X  Total = 7 Cost = 500
Lab = ECG  Total = 20 Cost = 800
Lab = MRI  Total = 10 Cost = 500
press
1:End programme     2.Go back
|
```

viewMedicineReceptionist(): Displays information about all available medicines in the system, retrieved from a file. The receptionist can choose to end the program or go back to the receptionist panel after viewing medicine details.



```
View the medicines and info of them here
Medicine Name = Napa  Company Name = Square Expiry Date = 21/9/2027 Medicine Amount = 100 Medicine Cost = 5
Medicine Name = Ace  Company Name = FnF Expiry Date = 5/4/2025 Medicine Amount = 200 Medicine Cost = 10
Medicine Name = sdjhbshdb  Company Name = eheufhie Expiry Date = 2/9/2026 Medicine Amount = 500 Medicine Cost = 50
Medicine Name = Okof  Company Name = Bio Expiry Date = 2/5/2027 Medicine Amount = 100 Medicine Cost = 20
Medicine Name = Fimoxil  Company Name = Square Expiry Date = 3/9/2026 Medicine Amount = 7 Medicine Cost = 10
Medicine Name = Revotril  Company Name = Adata Expiry Date = 4/2/2024 Medicine Amount = 500 Medicine Cost = 20
Medicine Name = MaxPro  Company Name = Square Expiry Date = 2/6/2027 Medicine Amount = 100 Medicine Cost = 20
press
1:End programme     2.Go back
|
```

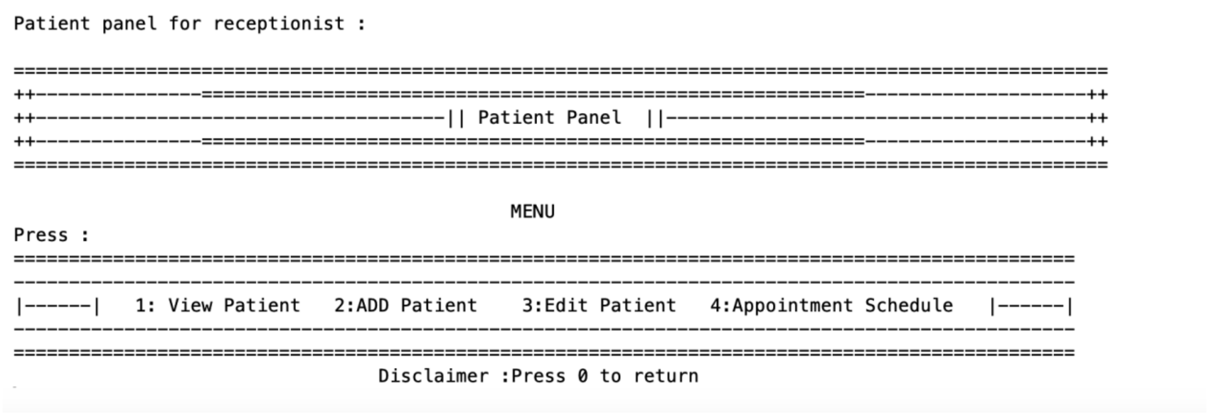
viewFacilitiesReceptionist(): Shows details of all facilities available in the hospital, retrieved from a file. The receptionist can choose to end the program or go back to the receptionist panel after viewing facility information.



```
View all faciliites
Facility = Ambulance  exist = 10 available = yes
Facility = Bed  exist = 2 available = yes
Facility = Lift  exist = 4 available = no
Facility = HeliCopter  exist = 10 available = no
Facility = Room  exist = 50 available = yes
press
1:End programme     2.Go back
|
```

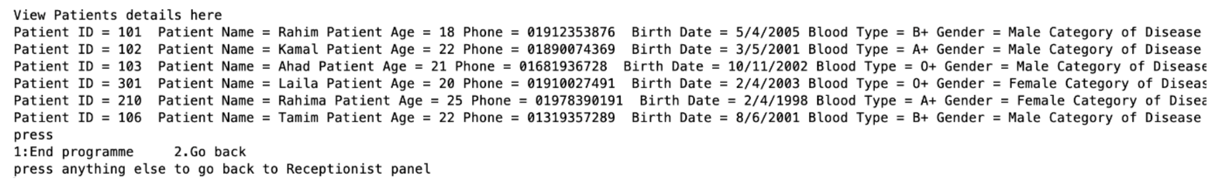
**Patient part in receptionist**

patientReceptionist(): Displays a menu for the receptionist related to patient management, allowing them to view, add, edit patient information, and manage appointment schedules. The method then redirects to specific functions based on the receptionist's input.



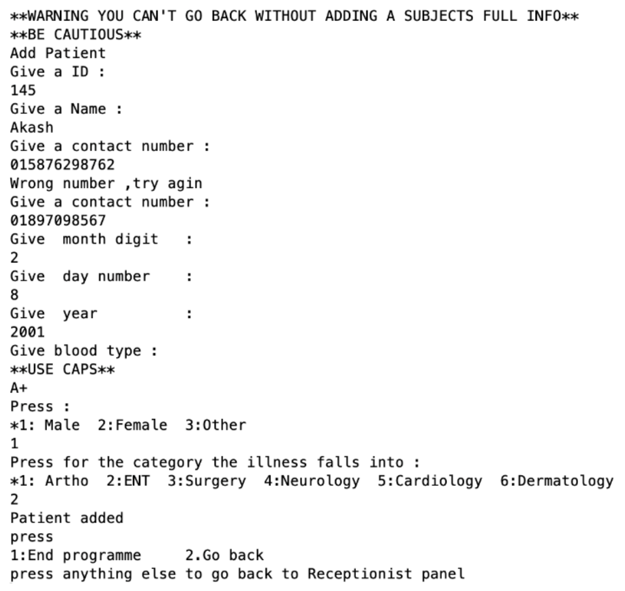
```
Patient panel for receptionist :

=========================================================================================
++---------------=====================================================-------------------++
++------------------------------------|| Patient Panel  ||-------------------------------++
++---------------=====================================================-------------------++
=========================================================================================

                                         MENU
Press :
======================================================================================
--------------------------------------------------------------------------------------
|------|   1: View Patient   2:ADD Patient    3:Edit Patient   4:Appointment Schedule   |------|
--------------------------------------------------------------------------------------
======================================================================================
                              Disclaimer :Press 0 to return
```

viewPatientReceptionst(): Displays detailed information about all patients in the system, retrieved from a file. Provides options for the receptionist to either end the program, go back to the patient receptionist panel, or return to the receptionist panel after viewing patient details.



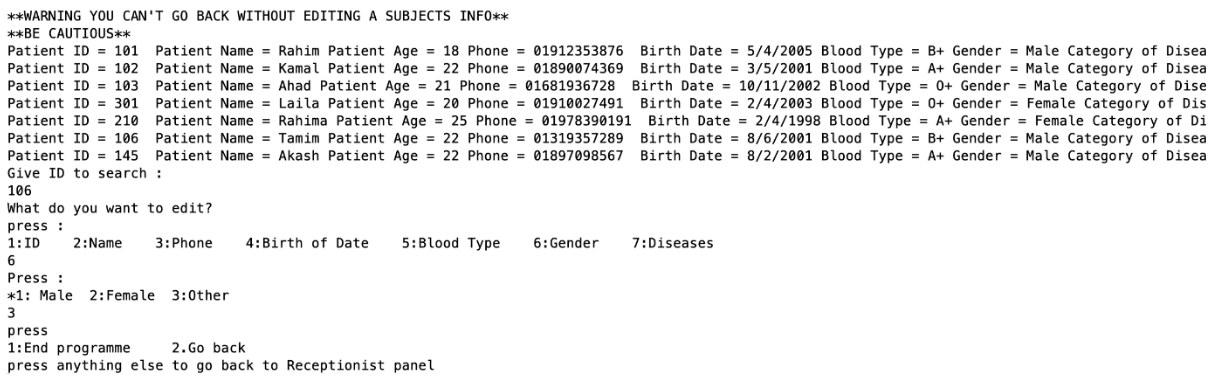
```
View Patients details here
Patient ID = 101  Patient Name = Rahim Patient Age = 18 Phone = 01912353876  Birth Date = 5/4/2005 Blood Type = B+ Gender = Male Category of Disease
Patient ID = 102  Patient Name = Kamal Patient Age = 22 Phone = 01890074369  Birth Date = 3/5/2001 Blood Type = A+ Gender = Male Category of Disease
Patient ID = 103  Patient Name = Ahad Patient Age = 21 Phone = 01681936728  Birth Date = 10/11/2002 Blood Type = O+ Gender = Male Category of Disease
Patient ID = 301  Patient Name = Laila Patient Age = 20 Phone = 01910027491  Birth Date = 2/4/2003 Blood Type = O+ Gender = Female Category of Diseas
Patient ID = 210  Patient Name = Rahima Patient Age = 25 Phone = 01978390191  Birth Date = 2/4/1998 Blood Type = A+ Gender = Female Category of Disea
Patient ID = 106  Patient Name = Tamim Patient Age = 22 Phone = 01319357289  Birth Date = 8/6/2001 Blood Type = B+ Gender = Male Category of Disease
press
1:End programme     2.Go back
press anything else to go back to Receptionist panel
```

addPatientReceptionist(): Takes user inputs to add a new patient, including ID, name, contact, birthdate, blood type, gender, and associated illness category. The method adds the patient to the system, and the receptionist can choose to end the program, go back to the patient receptionist panel, or return to the receptionist panel.



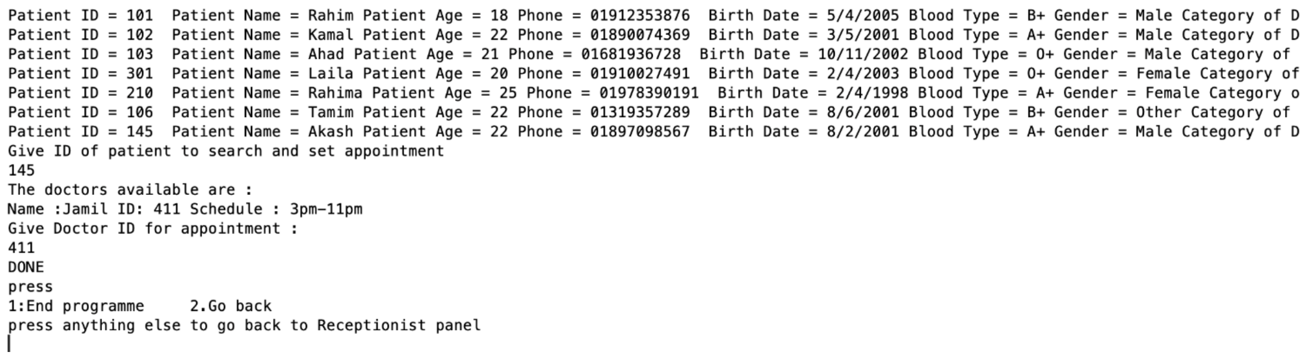
```
**WARNING YOU CAN'T GO BACK WITHOUT ADDING A SUBJECTS FULL INFO**
**BE CAUTIOUS**
Add Patient
Give a ID :
145
Give a Name :
Akash
Give a contact number :
015876298762
Wrong number ,try agin
Give a contact number :
01897098567
Give  month digit   :
2
Give  day number    :
8
Give  year          :
2001
Give blood type :
**USE CAPS**
A+
Press :
*1: Male  2:Female  3:Other
1
Press for the category the illness falls into :
*1: Artho  2:ENT  3:Surgery  4:Neurology  5:Cardiology  6:Dermatology
2
Patient added
press
1:End programme     2.Go back
press anything else to go back to Receptionist panel
```

editPatientReceptionist(): Allows the receptionist to edit patient information by displaying details, prompting for edits, and saving the changes. The receptionist can choose to end the program, go back to the patient receptionist panel, or return to the receptionist panel.



```
**WARNING YOU CAN'T GO BACK WITHOUT EDITING A SUBJECTS INFO**
**BE CAUTIOUS**
Patient ID = 101  Patient Name = Rahim Patient Age = 18 Phone = 01912353876  Birth Date = 5/4/2005 Blood Type = B+ Gender = Male Category of Disea
Patient ID = 102  Patient Name = Kamal Patient Age = 22 Phone = 01890074369  Birth Date = 3/5/2001 Blood Type = A+ Gender = Male Category of Disea
Patient ID = 103  Patient Name = Ahad Patient Age = 21 Phone = 01681936728  Birth Date = 10/11/2002 Blood Type = O+ Gender = Male Category of Dise
Patient ID = 301  Patient Name = Laila Patient Age = 20 Phone = 01910027491  Birth Date = 2/4/2003 Blood Type = O+ Gender = Female Category of Dis
Patient ID = 210  Patient Name = Rahima Patient Age = 25 Phone = 01978390191  Birth Date = 2/4/1998 Blood Type = A+ Gender = Female Category of Di
Patient ID = 106  Patient Name = Tamim Patient Age = 22 Phone = 01319357289  Birth Date = 8/6/2001 Blood Type = B+ Gender = Male Category of Disea
Patient ID = 145  Patient Name = Akash Patient Age = 22 Phone = 01897098567  Birth Date = 8/2/2001 Blood Type = A+ Gender = Male Category of Disea
Give ID to search :
106
What do you want to edit?
press :
1:ID    2:Name    3:Phone    4:Birth of Date    5:Blood Type    6:Gender    7:Diseases
6
Press :
*1: Male  2:Female  3:Other
3
press
1:End programme     2.Go back
press anything else to go back to Receptionist panel
```

editPatientAppointment(): Prompts the receptionist to enter a patient ID to search and set a new appointment schedule. After setting the appointment, the receptionist can choose to end the program, go back to the patient receptionist panel, or return to the receptionist panel.



```
Patient ID = 101  Patient Name = Rahim Patient Age = 18 Phone = 01912353876  Birth Date = 5/4/2005 Blood Type = B+ Gender = Male Category of D
Patient ID = 102  Patient Name = Kamal Patient Age = 22 Phone = 01890074369  Birth Date = 3/5/2001 Blood Type = A+ Gender = Male Category of D
Patient ID = 103  Patient Name = Ahad Patient Age = 21 Phone = 01681936728  Birth Date = 10/11/2002 Blood Type = O+ Gender = Male Category of
Patient ID = 301  Patient Name = Laila Patient Age = 20 Phone = 01910027491  Birth Date = 2/4/2003 Blood Type = O+ Gender = Female Category of
Patient ID = 210  Patient Name = Rahima Patient Age = 25 Phone = 01978390191  Birth Date = 2/4/1998 Blood Type = A+ Gender = Female Category o
Patient ID = 106  Patient Name = Tamim Patient Age = 22 Phone = 01319357289  Birth Date = 8/6/2001 Blood Type = B+ Gender = Other Category of
Patient ID = 145  Patient Name = Akash Patient Age = 22 Phone = 01897098567  Birth Date = 8/2/2001 Blood Type = A+ Gender = Male Category of D
Give ID of patient to search and set appointment
145
The doctors available are :
Name :Jamil ID: 411 Schedule : 3pm-11pm
Give Doctor ID for appointment :
411
DONE
press
1:End programme     2.Go back
press anything else to go back to Receptionist panel
```

**All extra methods that were needed during validating inputs**

**Integer checkers:**

StringisIntChecker(String str): Checks if the given string consists entirely of digits. Returns `true` if all characters are digits, `false` otherwise. It is used to validate numerical values.

StringisIntChecker(String str, int num):* Checks if the given string consists entirely of digits and has a specified length . Returns `true` if the string is numeric and of the specified length, `false` otherwise.It is used to validate contact number and id of an object

## Date checkers:

isValidDateOfBirth(String day, String month, String year): Validates the format of a date of birth by checking if the day, month, and year components form a valid date within a reasonable range for birth years. Returns `true` if the date of birth is valid, `false` otherwise. Uses daysInMonth(int month, int year) method to find the limit of days for given month

isValidExpiryDate(String day, String month, String year): Validates the format of a expiry date by checking if the day, month, and year components form a valid date. Returns `true` if the date is valid, `false` otherwise. It checks if the

daysInMonth(int month, int year): Returns the number of days in a given month of a specified year, accounting for leap years in the case of February.

calculateAge(String day, String month, String year): Calculates the age based on the provided day, month, and year of birth. It uses the Calendar class to handle date calculations, determining the age by considering the birthdate and the current date, adjusting for cases where the birthday hasn't occurred yet in the current year. The age is returned as a String.

These methods are used in add methods to check their respective functions.

## Valid name check:

isvalidName(Medicine[] p, String name, int count): Checks if the provided name is valid for a Medicine by comparing it with existing Medicine objects in the array. Returns true if the name is unique, false otherwise.

isvalidName(Lab[] p, String name, int count): Checks if the provided name is valid for a Lab by comparing it with existing Lab objects in the array. Returns true if the name is unique, false otherwise.

isvalidName(Facility[] p, String name, int count): Checks if the provided name is valid for a Facility by comparing it with existing Facility objects in the array. Returns true if the name is unique, false otherwise.

These methods are used in add methods to check if the unique or tag names taken as input matches with any existing names in the database. We used **polymorphism** concept here for our advantage.

## Valid ID check:

isvalidId(Doctor[] p, String id, int count):Checks if the provided ID is valid for a Doctor by comparing it with existing Doctor objects in the array. Returns true if the ID is unique, false otherwise.

isvalidId(Staff[] p, String id, int count):Checks if the provided ID is valid for a Staff member by comparing it with existing Staff objects in the array. Returns true if the ID is unique, false otherwise.

isvalidId(Patient[] p, String id, int count): Checks if the provided ID is valid for a Patient by comparing it with existing Patient objects in the array. Returns true if the ID is unique, false otherwise.

These methods are used in add methods to check if the unique ID taken as input matches with any existing names in the database. We used **polymorphism** concept here for our advantage

## Possible Improvements and Limitations:

-Due to time constraints, we didn’t use GUI or JAVAFX. If used, a lot of our work could have become easier and our programme would have been more appealing.

-We wanted to add more features but realized the concept of a hospital management is vast. Thus, we had to narrow down our point of focus in certain features only.

-Using recursion call has led to problems like facing an infinite loop or improper execution of the progaramme.

-The writing in a file could have been set in a more efficient way for access.

-More features could be added to make it more dynamic.

## Conclusion:

In summary, our Java-based Hospital Management System is a user-friendly tool designed to make running a healthcare facility smoother with features like Edit, View, and Add, it simplifies managing Doctors, Patients, Staff, Facilities, Medicines, and Labs.

You can easily update information, check details, and add new data, ensuring accurate and current records. The system helps staff work efficiently and ensures that only authorized personnel access sensitive information.

It's not just about managing data; it's about making the lives of doctors, patients, and staff easier. Doctors can keep track of their schedules and patient interactions, while patients benefit from streamlined appointments and better record-keeping. Staff, facility, medicine, and lab management are all simplified for a more effective healthcare system.

In a nutshell, our system aims to improve overall hospital management, contributing to better patient care and smoother operations.