Qn.

Create the methods to take input in each class

Override it in the inherited class so that it takes the appropriate details

Hint : Keep the common ones in the base

Override the tostring method so that the details printed are complete

public class User

{

public int Id { get; set; }

public string Name { get; set; }

public string Password { get; set; }

public int Age { get; set; }

public string Type { get; set; }

public void TakeLoginDetailsFromUserAndAuthenticate(User user, User[] arr)

{

Boolean isLoginSuccess = false;

while (isLoginSuccess == false) {

Console.WriteLine("Please enter the user ID");

Id = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Please enter the password");

Password = Console.ReadLine();

var fooItem = Array.Find(arr, item => (item.Id == Id) && (item.Type == user.Type)) ;

int arrNo = Array.IndexOf(arr, fooItem);

if (arrNo != -1)

{

//user exists

Console.WriteLine("Login successful! Welcome "+fooItem.Name+".");

isLoginSuccess = true;

}

else

{

//user does not exist

Console.WriteLine("Login unsuccessful!");

}

}

}

public override string ToString() //override normal tostring

{

return "User ID " + Id + " Name " + Name + " Age " + Age + " User Type "+ Type;

}

}

public class Doctor : User

{

public int Experience { get; set; } //yrs

public string Speciality { get; set; } //choose one category

public Doctor(int id, string name, int age, string password, int experience, string speciality)

{

Id = id;

Name = name;

Age = age;

Password = password;

Type = "Doctor";

Experience = experience;

Speciality = speciality;

}

public Doctor()

{

Type = "Doctor";

}

}

public class Patient : User

{

public string Remarks { get; set; }

public string Status { get; set; }

public Patient(int id, string name, int age, string password, string status, string remarks)

{

Id = id;

Name = name;

Age = age;

Password = password;

Type = "Patient";

Status = status;

Remarks = remarks;

}

public Patient()

{

Type = "Patient";

}

//methods related to patient

}

public class Appointment

{

public Appointment()

{

}

public Appointment(int id, string status, string remarks, string details, int doctorId, int patientId, DateTime appDate, string category, double paymentAmount, string paymentStatus)

{

Id = id;

Status = status;

Remarks = remarks;

Details = details;

DoctorId = doctorId;

PatientId = patientId;

DateCreated = DateTime.Now;

AppDate = appDate;

Category = category;

PaymentAmount = paymentAmount;

PaymentStatus = paymentStatus; //new, paid, unpaid(time limit)

}

public int Id { get; set; }

public string Status { get; set; } //ongoing, new, old

public string Remarks { get; set; } //

public string Details { get; set; } //need to separate into smaller parts?

public int DoctorId { get; set; }

public int PatientId { get; set; }

public DateTime DateCreated { get; set; }

public DateTime AppDate { get; set; }

public string Category { get; set; }

public double PaymentAmount { get; set; }

public string PaymentStatus { get; set; }

public void BookAppointment(User u, User[] users)

{

//for doctor and patient

Appointment appointment = new Appointment();

Console.WriteLine("You have chosen book appointment.");

if (u.Type == "Doctor")

{

appointment.DoctorId = u.Id;

appointment.PatientId = FindPatientIDByName(users);

}

else

{

appointment.PatientId = u.Id;

//patient cannot choose doctor, to be set later

}

//any time slot for now

Console.WriteLine("Please enter a date and time in format dd-mm-yyyy hh:mm tt (eg. 20-08-22 12:30 AM)");

AppDate = Convert.ToDateTime(Console.ReadLine());

Console.WriteLine("Please enter a reason for the appointment.");

Category = Console.ReadLine();

DateCreated = DateTime.Now;

Status = "New";

}

public void ViewAppointmentDetails(User u, User[] users, Appointment[] appointments)

{

//for doctor and patient

Console.WriteLine("You have chosen to view appointment.");

//get those details n priint out line by line

int appIdIndex = FindAppoinmentByAppDateAndPatientId(appointments, users);

//Console.WriteLine("appid: "+ appIdIndex);

//Console.WriteLine("doc id: "+appointments[appIdIndex].DoctorId);

Console.WriteLine("Your selected appointment details are:");

Console.WriteLine("Doctor name: "+users[appointments[appIdIndex].DoctorId-1].Name);

Console.WriteLine("Patient name: " + users[appointments[appIdIndex].PatientId - 1].Name);

Console.WriteLine("Details: "+ appointments[appIdIndex].Details);

Console.WriteLine("Remarks: "+ appointments[appIdIndex].Remarks);

Console.WriteLine("Status: "+appointments[appIdIndex].Status);

Console.WriteLine("Appointment Date: "+ appointments[appIdIndex].AppDate);

Console.WriteLine("Category: "+ appointments[appIdIndex].Category);

Console.WriteLine("Payment amount: "+ appointments[appIdIndex].PaymentAmount);

Console.WriteLine("Payment Status: "+ appointments[appIdIndex].PaymentStatus);

}

public void UpdateAppointment(User u, Appointment[] appointments, User[] users)

{

//for doctor(mostly actually) and patient

Console.WriteLine("You have chosen to update appointment.");

//find appointment

int appArr = FindAppoinmentByAppDateAndPatientId(appointments, users);

//ask which field to edit

Console.WriteLine("Please enter which field do you want to edit.");

Console.WriteLine("(1) Details");

Console.WriteLine("(2) Remarks" );

Console.WriteLine("(3) Status");

Console.WriteLine("(4) Appointment Date in format dd-mm-yyyy hh:mm tt (eg. 20-08-22 12:30 AM)");

Console.WriteLine("(5) Category");

Console.WriteLine("(6) Payment amount");

Console.WriteLine("(7) Payment Status");

int choice = Convert.ToInt32(Console.ReadLine());

//edit field into

Console.WriteLine("Please enter what values you want to change to:");

string editTo = Console.ReadLine();

//set

Appointment app = new Appointment();

app = appointments[appArr];

switch (choice)

{

case 1:

app.Details = editTo ;

break;

case 2:

app.Remarks = editTo;

break;

case 3:

app.Status = editTo;

break;

case 4:

app.AppDate = Convert.ToDateTime(editTo);

break;

case 5:

app.Category = editTo;

break;

case 6:

app.PaymentAmount = Convert.ToDouble(editTo);

break;

case 7:

app.PaymentStatus = editTo;

break;

default:

Console.WriteLine("Invalid Entry.");

break;

}

Console.WriteLine("The appointment has been changed.");

ViewAppointmentDetails(u, users, appointments);

//Console.WriteLine("status= "+app.Status);

//Console.WriteLine("take from arr print: "+appointments[0].Status);

}

public void CreatePayment(User u)

{

//for doctor

Console.WriteLine("Please enter payment amount");

PaymentAmount = Convert.ToDouble(Console.ReadLine());

PaymentStatus = "Created";

//appointment status has to be updated to complete

Console.WriteLine("Payment is created");

}

public void MakePayment(User u)

{

//for patient

PaymentStatus = "Successful";

Console.WriteLine("Payment is successful");

}

public int FindPatientIDByName(User[] users)

{

string patientName="";

Boolean userFound = false;

int arrNo = 0;

while (userFound == false)

{

Console.WriteLine("Please enter patient's full name");

patientName = Console.ReadLine();

arrNo = Array.IndexOf(users, Array.Find(users, item => item.Name == patientName));

if (arrNo != -1)

{

//patient exists

Console.WriteLine("Patient found!");

userFound = true;

}

else

{

//patient does not exist

Console.WriteLine("Patient not found!");

}

}

return arrNo;

}

public int FindAppoinmentByAppDateAndPatientId(Appointment[] apps, User[] users)

{

DateTime appDate;

int patientId = 0;

int arrNoApp = 0;

Boolean appFound = false;

while (appFound == false)

{

patientId = users[FindPatientIDByName(users)].Id;

Console.WriteLine("Please enter your appointment date in format dd-mm-yyyy hh:mm tt (eg. 20-08-22 12:30 AM)");

appDate = Convert.ToDateTime(Console.ReadLine());

arrNoApp = Array.IndexOf(apps, Array.Find(apps, item => (item.PatientId == patientId) && (item.AppDate == appDate)));

if (arrNoApp != -1)

{

//Appointment exists

Console.WriteLine("Appointment found!");

appFound = true;

}

else

{

//Appointment does not exist

Console.WriteLine("Appointment not found!");

}

}

return arrNoApp;

}

}