Faculty of Engineering and Information Technology
School of Computer Science

31927 – Application Development with .NET
32998 - .NET Application Development

SPRING 2023
ASSIGNMENT 1 – SPECIFICATION

Due date	Monday 25 September 2023, 11:00am
Demonstration	Required in the lab/tutorial session
Weight	35%
Groupwork	Individual
Submission	Complete project folder zip
Submit to	Canvas

Summary

In this assignment, you are required to model the dotnet Hospital Management System by developing a console application using C#.

The application should have appropriate data structures to distinguish between a Doctor, Administrator and Patient as well as store the necessary links between them (such as an appointment, between a Doctor and a Patient). You are given complete control in how you create these data structures, but guidance has been given overpage.

Objects should be stored in the system and also written to .txt files such that the system can read them in and regenerate existing objects on load. This is mandatory.

Students will need to submit the complete project folder in zip format, which will have the complete C# code, solution file, data file, etc. required to run/test the program. Any special instructions required to run the code has to be provided in a text file. Submitting a lone ".exe" is not acceptable.

Assignment Objectives:

The purpose of this assignment is to demonstrate competence in the following skills:

- Ensure firm understanding of the .NET framework, C# basics and C# syntax
- Understand how the .NET framework implements OO concepts and the implications this has for new language design
- Array and string manipulation
- Creating custom classes and methods in C#
- File reading, writing and manipulation in C#
- Creating interactive console applications
- Creating good OO design.

Program and data structures:

- How you structure the classes in your program is your choice. One thing you are not allowed to do is make your program fully contained inside Program.cs, or any single class.
- In the dotnet Hospital Management System, a user can log in as either a Patient, a Doctor, or an administrator. These are different roles, which would store different information, and would have a different menu. Your code structure should reflect this.

- Additionally, your code will need to generate Appointments. An appointment would need a reference to a single Doctor and a single Patient; your code structure should reflect this. You do not need to manage Dates/times for appointments (or anywhere, for that matter) in your code. This may result in appointments being difficult to distinguish and sort; you will not be marked down for this.
- Every patient, doctor and admin, has a unique ID. This ID should be an integer of reasonable length (5-8 digits). This can be randomly generated, or incremental, but it should be generated by the system on object creation, not chosen/inputted by the user.

Further recommendations:

- Objects will need to be printed out one by one, it's recommended that each data structure has a toString() function which compresses the notable data of the class into a succinct line.
- Each role should be its own class and it should have its own MainMenu method, don't try and create separate versions of the menu in Program.cs.
- Don't get confused by Administrators. Doctors and Patients can't at any stage have "admin privileges"; the administrator is a completely separate entity.
- You may find it useful to abstract common functionality into its own class, such as a FileManager class with static read and write methods or a Utils class which contains the methods to generate ID's and filter lists.

Marking Guide

Below is the marking guide for this assessment. It is designed to allow you to get a Pass grade with minimal effort while still demonstrating that you understand the core principles of .NET development, to get a Distinction with reasonable effort, and to get a High Distinction with solid effort, and 100% with considerable effort. It is recommended that you pay attention to the grade distribution and work towards your own skill level.

In the demos in the lab, your code needs to be compiled in Visual Studio Community edition 2022 and then the tutor will test for normal functionality as described in the descriptions above. If your code does not compile you will receive zero marks (no exceptions).

Task	Items	Max Point	Total
Console Design	- Appropriate Headings for menus - Basic console design	2	9
	- Helpful comments - Appropriate indentening and whitespacing - Consistent and appropriate C# naming convention used	2	
	Strong OOP principles used	3	
	Low coupling, high cohesion, general code quality	2	
Login Menu	Functionality including cross checking credentials with .txt file	1	2
	Password input masked in console	1	
Patient Menu	List Patient Details	1	6
	List My Doctor Details	1	
	List All Appointments	1	
	Book Appointment, functionality including creating Appointment object and writing to txt file	3	
Doctor Menu	List Doctor Details	1	7
	List Patients	1	
	List Appointments	1	
	Check Particular Patient	2	
	List Appointments With Patient	2	
Admin Menu	List All Doctors	1	10
	Check Doctor Details	2	
	List All Patients	1	
	Check Patient Details	2	
	Add Doctor	3	
	Add Patient	3	
Logout/Exit	Logout functionality and Exit functionality for all 3 menus	1	1
	Maximum Full Marks		35
Bonus Marks	- Meaningful use of inheritance AND - Meaningful use of a custom interface AND - A useful abstraction	2	5
	Email functionality (e.g. confirmation of patient registration, or confirmation of booking)	1	
	Additional user role (e.g. receptionist)	2	

Specific expected tasks and examples

In this assignment, you are required to develop a console-based, menu driven hospital management system using C#. Necessary data should be stored in ".txt" files. There are 3 core roles in the system.

1. Patient
2. Doctor
3. Administrator

Each of these roles have their own version of the menu with it's own subset of menu options.

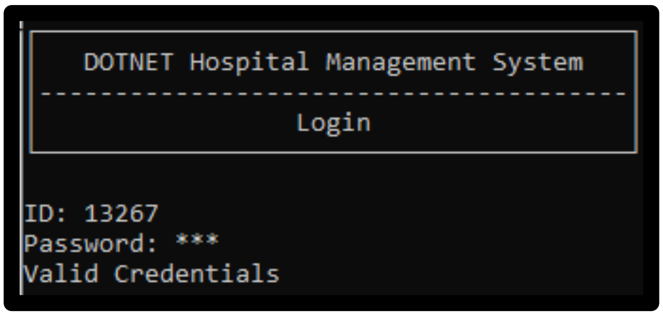
Login Menu

The user should be presented with this menu on program startup and cannot proceed further without entering valid login details.

Input fields:

- ID: Display the typed characters as is
- Password: Display "*" instead of the actual characters

ID and password should be checked against valid credentials inside your files. How you do this depends on how you've structured your classes; but any patient/doctor or admin should have a unique ID and a password. These values should be, in some way, stored in a .txt file, which your code should cross check at this screen. If entered values are not valid, display appropriate error message and allow the user to retry.



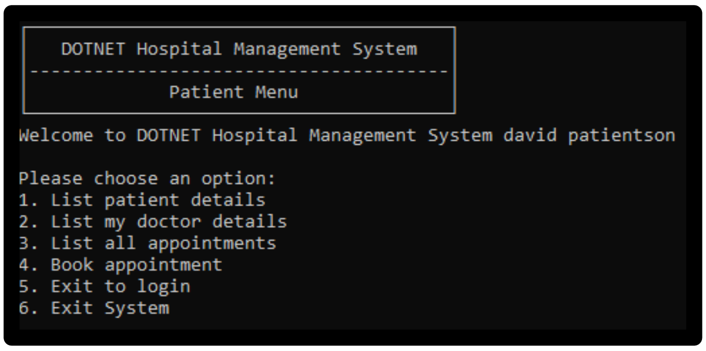
```
 DOTNET Hospital Management System
------------------------------------
              Login


ID: 13267
Password: ***
Valid Credentials
```

Patient Menu

The patient menu should have all of the following menu options (see right).

All menu options MUST share this functionality (see example below):

- Console is emptied to display new data
- Appropriate heading displayed before data printed
- Error handling for bad user input
- Menu should never return itself, the user should be able to press a single key to return.

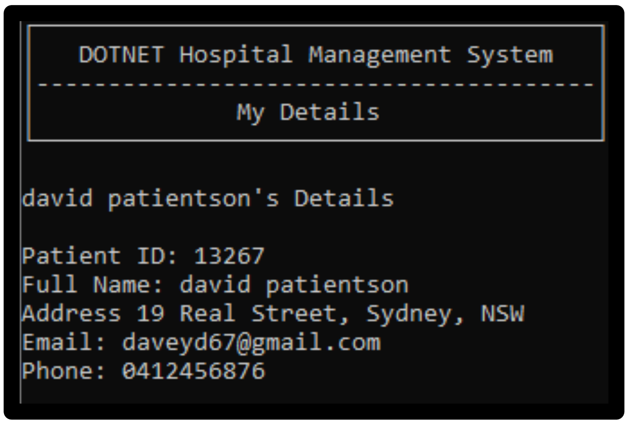


```
 DOTNET Hospital Management System
------------------------------------
           Patient Menu
Welcome to DOTNET Hospital Management System david patientson

Please choose an option:
1. List patient details
2. List my doctor details
3. List all appointments
4. Book appointment
5. Exit to login
6. Exit System
```

List Patient Details

Functionality: Lists all the fields of the currently logged in patient to the console.

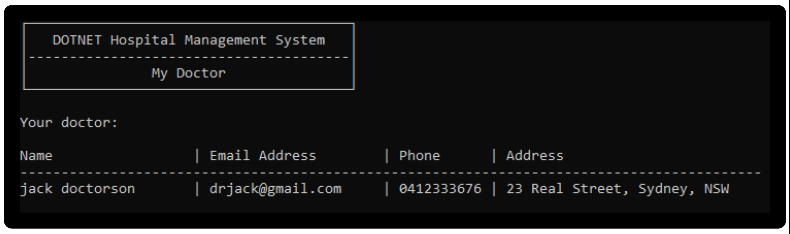
Input: Key press to return to menu.



```
 DOTNET Hospital Management System
------------------------------------
            My Details

david patientson's Details

Patient ID: 13267
Full Name: david patientson
Address 19 Real Street, Sydney, NSW
Email: daveyd67@gmail.com
Phone: 0412456876
```

List My Doctor Details

Functionality: Lists all the fields of the doctor that is registered with the currently logged in patient, to the console. It is recommended that the Doctor data structure has a toString function which compresses the necessary details of the patient into a short line.

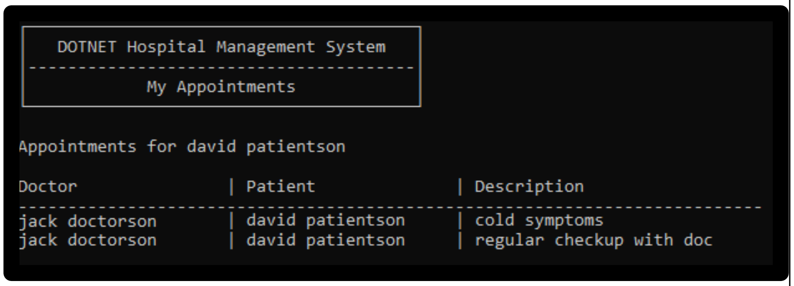


```
 ┌──────────────────────────────────────┐
 │   DOTNET Hospital Management System   │
 │ ------------------------------------- │
 │              My Doctor                │
 └──────────────────────────────────────┘

Your doctor:

Name                 | Email Address        | Phone      | Address
------------------------------------------------------------------------------------------
jack doctorson       | drjack@gmail.com     | 0412333676 | 23 Real Street, Sydney, NSW
```

List All Appointments

Functionality: Lists the details of all past appointments involving the currently logged in patient.

Input: Key press to return to menu.



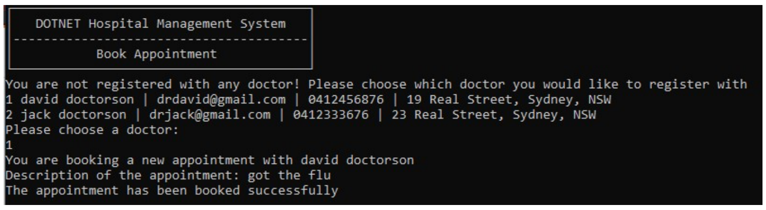
```
 ┌──────────────────────────────────────┐
 │   DOTNET Hospital Management System   │
 │ ------------------------------------- │
 │           My Appointments             │
 └──────────────────────────────────────┘

Appointments for david patientson

Doctor               | Patient              | Description
------------------------------------------------------------------------------------------
jack doctorson       | david patientson     | cold symptoms
jack doctorson       | david patientson     | regular checkup with doc
```

Book Appointment

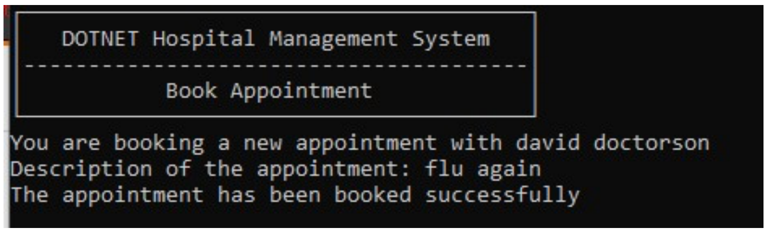
Functionality: Prompts the user for all the necessary information to generate a new appointment. This must be done with the doctor registered to the user. If the user has not registered to a doctor, they should be prompted to choose from a list of all doctors and register.

Input:

- Integer to select doctor (if applicable)
- All necessary information to generate your Doctor data structure
- Key press to return to menu.



```
 ┌──────────────────────────────────────┐
 │   DOTNET Hospital Management System   │
 │ ------------------------------------- │
 │           Book Appointment            │
 └──────────────────────────────────────┘
You are not registered with any doctor! Please choose which doctor you would like to register with
1 david doctorson | drdavid@gmail.com | 0412456876 | 19 Real Street, Sydney, NSW
2 jack doctorson | drjack@gmail.com | 0412333676 | 23 Real Street, Sydney, NSW
Please choose a doctor:
1
You are booking a new appointment with david doctorson
Description of the appointment: got the flu
The appointment has been booked successfully
```



```
 ┌──────────────────────────────────────┐
 │   DOTNET Hospital Management System   │
 │ ------------------------------------- │
 │           Book Appointment            │
 └──────────────────────────────────────┘

You are booking a new appointment with david doctorson
Description of the appointment: flu again
The appointment has been booked successfully
```

Logout

Functionality: Returns to the login menu

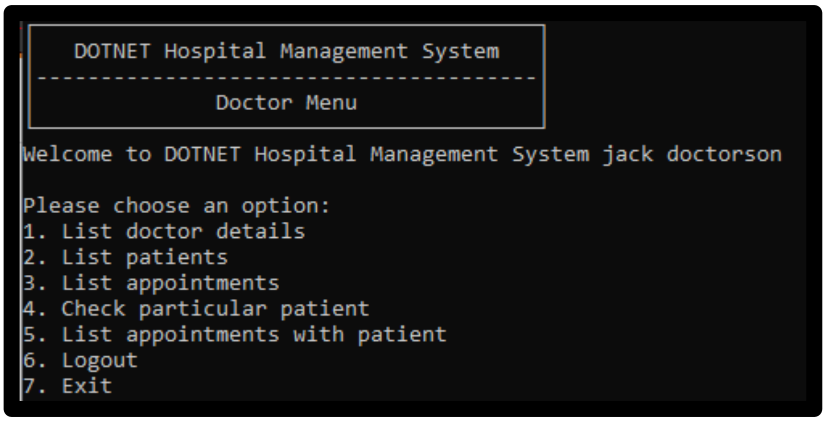
Exit

Functionality: Exits the application

Doctor Menu

The doctor menu should have all of the following menu options (see right)
All menu options MUST share this functionality:

- Console is emptied to display new data
- Appropriate heading displayed before data printed
- Error handling for bad user input
- Menu should never return itself, the user should be able to press a single key to return.

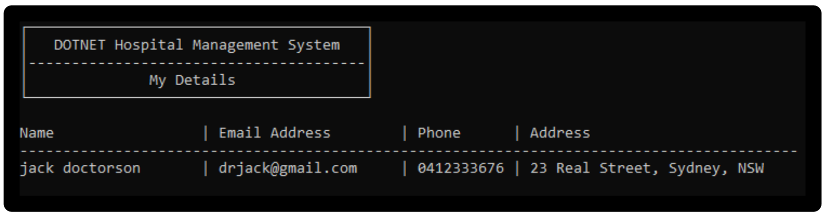


```
 DOTNET Hospital Management System
 ---------------------------------------
            Doctor Menu

Welcome to DOTNET Hospital Management System jack doctorson

Please choose an option:
1. List doctor details
2. List patients
3. List appointments
4. Check particular patient
5. List appointments with patient
6. Logout
7. Exit
```

List Doctor Details

Functionality: Lists the fields of the currently logged in doctor to the console.

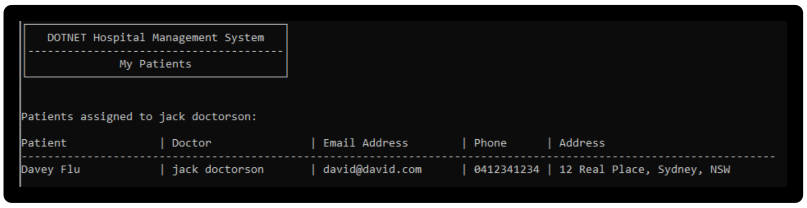
Input: Key press to return to menu.



```
 DOTNET Hospital Management System
 ---------------------------------------
            My Details

Name                 | Email Address        | Phone      | Address
-------------------------------------------------------------------------------
jack doctorson       | drjack@gmail.com     | 0412333676 | 23 Real Street, Sydney, NSW
```

List Patients

Functionality: Lists a shorthand outline of every patient that is registered with the currently logged in doctor to the console, line by line. It is recommended that the Patient data structure has a toString function which compresses the necessary details of the patient into a short line.

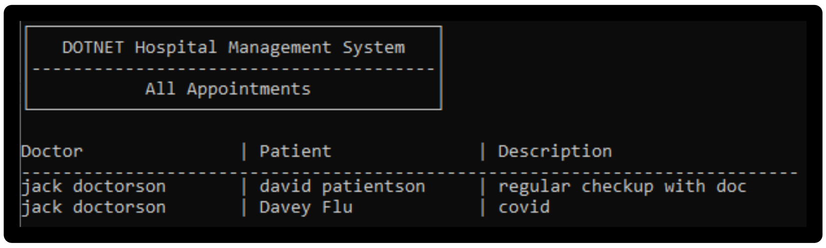


```
 DOTNET Hospital Management System
 ---------------------------------------
            My Patients

Patients assigned to jack doctorson:

Patient              | Doctor               | Email Address        | Phone      | Address
-------------------------------------------------------------------------------------------
Davey Flu            | jack doctorson       | david@david.com      | 0412341234 | 12 Real Place, Sydney, NSW
```

List Appointments

Functionality: Lists every appointment involving the currently logged in doctor, regardless of which patient is involved, to the console, line by line. It is recommended that that the Appointment data structure has a toString function which compresses the necessary details of the appointment into a short line.

Input: Key press to return to menu.

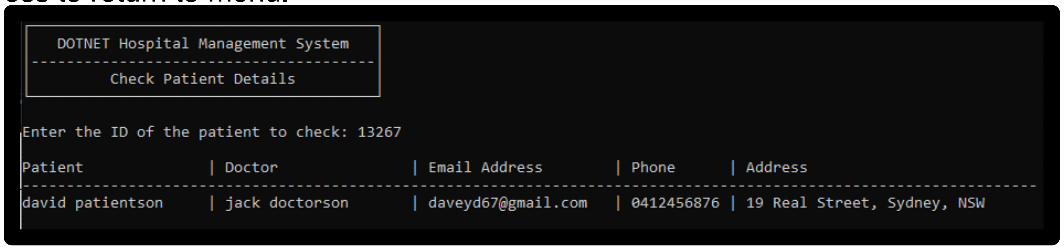


```
 DOTNET Hospital Management System
-----------------------------------------
         All Appointments

Doctor               | Patient              | Description
-------------------------------------------------------------------
jack doctorson       | david patientson     | regular checkup with doc
jack doctorson       | Davey Flu            | covid
```

Check Particular Patient

Functionality: Prompts the user for an ID and prints the details of the patient whose ID it belongs to, to the console line by line. If there is no patient with that ID, your code should handle the error and print an appropriate error message. It is your choice whether the user is prompted to re-enter the ID or if the program returns to the Patient menu.

Input: Key press to return to menu.

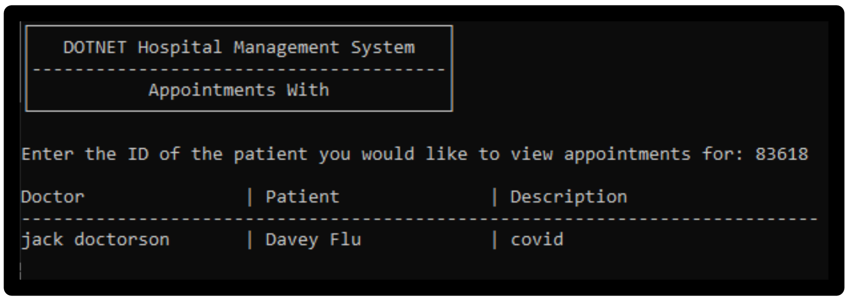


```
 DOTNET Hospital Management System
-----------------------------------------
       Check Patient Details

Enter the ID of the patient to check: 13267
Patient              | Doctor               | Email Address        | Phone      | Address
------------------------------------------------------------------------------------------------
david patientson     | jack doctorson       | daveyd67@gmail.com   | 0412456876 | 19 Real Street, Sydney, NSW
```

List Appointments With Patient

Functionality: Prompts the user for an ID and finds the patient with that ID. Your code should then print every appointment between that patient and the currently logged in doctor. (You should not need to filter this list, as the patient should have been forced to be registered to only 1 doctor). If there is no patient with that ID, your code should handle the error and print an appropriate error message. It is your choice whether the user is prompted to re-enter the ID or if the program returns to the Patient menu.

Input: Key press to return to menu.



```
 DOTNET Hospital Management System
-----------------------------------------
         Appointments With

Enter the ID of the patient you would like to view appointments for: 83618
Doctor               | Patient              | Description
-------------------------------------------------------------------
jack doctorson       | Davey Flu            | covid
```

Logout

Functionality: Returns to the login menu

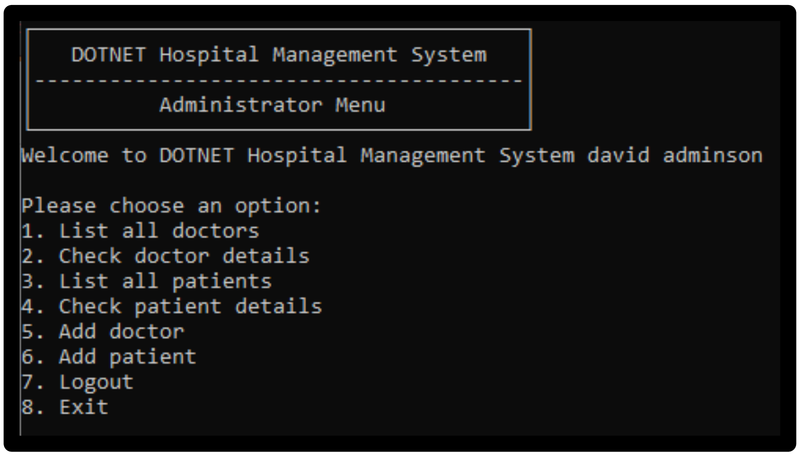
Exit

Functionality: Exits the application

Administrator menu

The administrator menu should have all of the following menu options (see right)
All menu options MUST share this functionality:

- Console is emptied to display new data
- Appropriate heading displayed before data printed
- Error handling for bad user input
- Menu should never return itself, the user should be able to press a single key to return.

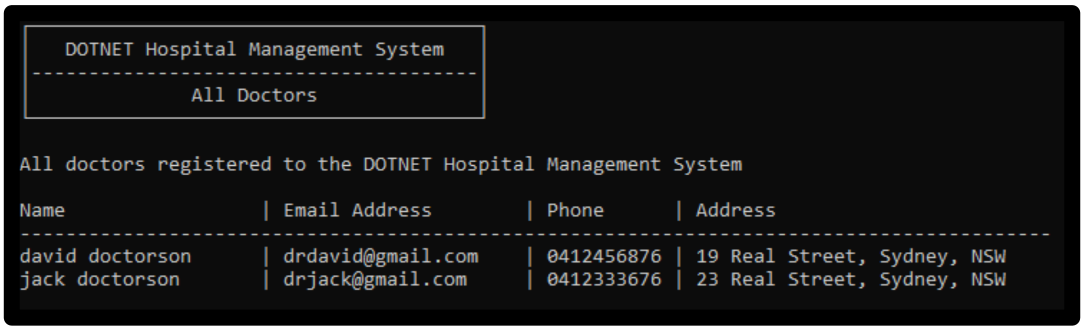


```
 ┌───────────────────────────────────────┐
 │   DOTNET Hospital Management System    │
 │ -------------------------------------- │
 │          Administrator Menu            │
 └───────────────────────────────────────┘
Welcome to DOTNET Hospital Management System david adminson

Please choose an option:
1. List all doctors
2. Check doctor details
3. List all patients
4. Check patient details
5. Add doctor
6. Add patient
7. Logout
8. Exit
```

List All Doctors

Functionality: Lists a shorthand version of every doctor contained in the system to the console, line by line. It is recommended that the Doctor data structure has a toString function which compresses the necessary details of the doctor into a short line.

Input: Key press to return to menu.

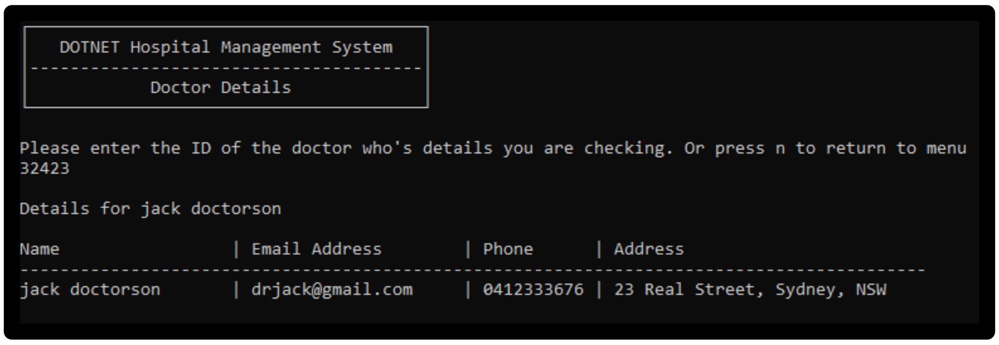


```
 ┌───────────────────────────────────────┐
 │   DOTNET Hospital Management System    │
 │ -------------------------------------- │
 │              All Doctors               │
 └───────────────────────────────────────┘

All doctors registered to the DOTNET Hospital Management System

Name                 | Email Address       | Phone      | Address
---------------------------------------------------------------------------------------------
david doctorson      | drdavid@gmail.com   | 0412456876 | 19 Real Street, Sydney, NSW
jack doctorson       | drjack@gmail.com    | 0412333676 | 23 Real Street, Sydney, NSW
```

Check Doctor Details

Functionality: Prompts the user for an ID and prints the details of the doctor whose ID it belongs to, to the console line by line. If there is no doctor with that ID, your code should handle the error and print an appropriate error message. It is your choice whether the user is prompted to re-enter the ID or if the program returns to the Administrator menu.

Input: Key press to return to menu.

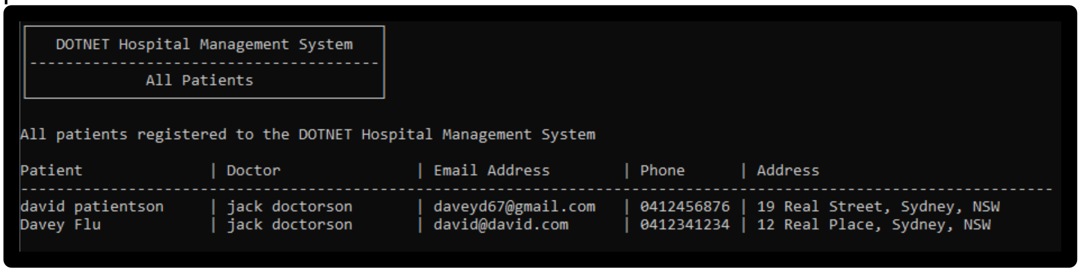


```
 ┌──────────────────────────────────────┐
 │  DOTNET Hospital Management System   │
 │--------------------------------------│
 │            Doctor Details            │
 └──────────────────────────────────────┘

Please enter the ID of the doctor who's details you are checking. Or press n to return to menu
32423

Details for jack doctorson

Name                 | Email Address        | Phone      | Address
-------------------------------------------------------------------------------------------
jack doctorson       | drjack@gmail.com     | 0412333676 | 23 Real Street, Sydney, NSW
```

List All Patients

Functionality: Lists the shorthand version of every patient in the system to the console, line by line. It is recommended that the Patient data structure has a toString function which compresses the necessary details of the patient into a short line.

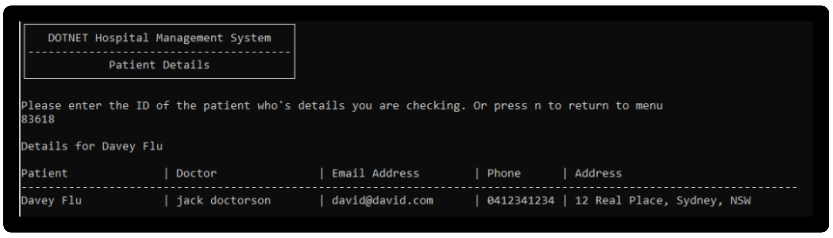
Input: Key press to return to menu.



```
 ┌──────────────────────────────────────┐
 │  DOTNET Hospital Management System   │
 │--------------------------------------│
 │             All Patients             │
 └──────────────────────────────────────┘

All patients registered to the DOTNET Hospital Management System

Patient              | Doctor               | Email Address        | Phone      | Address
-------------------------------------------------------------------------------------------------------------
david patientson     | jack doctorson       | daveyd67@gmail.com   | 0412456876 | 19 Real Street, Sydney, NSW
Davey Flu            | jack doctorson       | david@david.com      | 0412341234 | 12 Real Place, Sydney, NSW
```

Check Patient Details

Functionality: Prompts the user for an ID and prints the details of the doctor whose ID it belongs to, to the console line by line. If there is no doctor with that ID, your code should handle the error and print an appropriate error message. It is your choice whether the user is prompted to re-enter the ID or if the program returns to the Administrator menu.



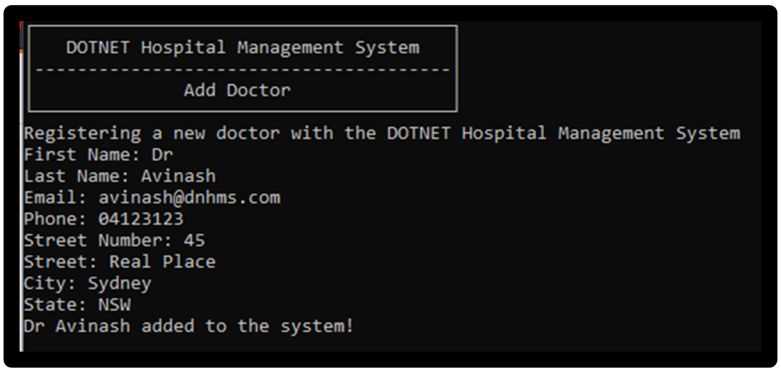
```
 ┌──────────────────────────────────────┐
 │  DOTNET Hospital Management System   │
 │--------------------------------------│
 │            Patient Details           │
 └──────────────────────────────────────┘

Please enter the ID of the patient who's details you are checking. Or press n to return to menu
83618

Details for Davey Flu

Patient              | Doctor               | Email Address        | Phone      | Address
-------------------------------------------------------------------------------------------------------------
Davey Flu            | jack doctorson       | david@david.com      | 0412341234 | 12 Real Place, Sydney, NSW
```

Input: Key press to return to menu.

Add Doctor

Functionality: Prompts the user for all the necessary data needed to generate a new Doctor and add it to the system.

Input: Key press to return to menu.

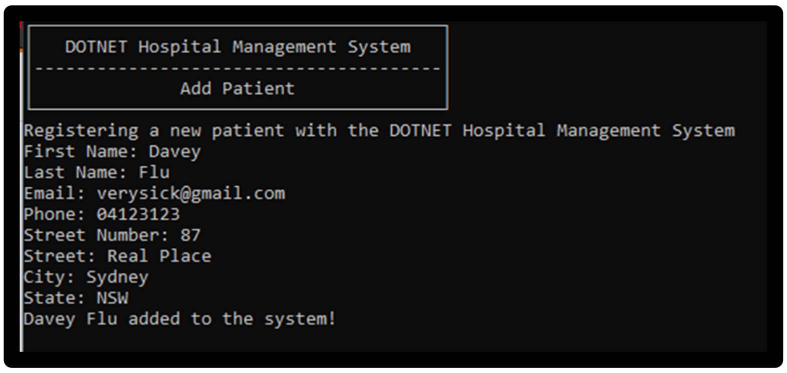


```
 DOTNET Hospital Management System
--------------------------------------
            Add Doctor
Registering a new doctor with the DOTNET Hospital Management System
First Name: Dr
Last Name: Avinash
Email: avinash@dnhms.com
Phone: 04123123
Street Number: 45
Street: Real Place
City: Sydney
State: NSW
Dr Avinash added to the system!
```

Add Patient

Functionality: Prompts the user for all the necessary data needed to generate a new Patient and add it to the system.

Input: Key press to return to menu.



```
 DOTNET Hospital Management System
--------------------------------------
            Add Patient
Registering a new patient with the DOTNET Hospital Management System
First Name: Davey
Last Name: Flu
Email: verysick@gmail.com
Phone: 04123123
Street Number: 87
Street: Real Place
City: Sydney
State: NSW
Davey Flu added to the system!
```

Logout

Functionality: Returns to the login menu

Exit

Functionality: Exits the application

Additional Information:

Assignment Submission

You must upload a zip file of the C# solution to Canvas. This must be done by the Due Date. You may submit as many times as you like until the due date. The final submission you make is the one that will be marked. If you have not uploaded your zip file within 7 days of the Due Date, or it cannot be compiled and run in the lab, then your assignment will receive a zero mark

- NOTE 1: It is your responsibility to make sure you have thoroughly tested your program to make sure it is working correctly.
- NOTE 2: Your final submission to Canvas is the one that is marked. It does not matter if earlier submissions were working; they will be ignored. Download your submission from Canvas and test it thoroughly.

Queries

If you have a problem such as illness which will affect your assignment submission contact the subject coordinator or lab tutor as soon as possible.

Dr. Avinash Singh
Room: CB11.07.108
Phone: 9514 4426
Email: avinash.singh@uts.edu.au

If you have a question about the assignment, please post it to the Canvas discussion board for this subject so that everyone can see the response.

If serious problems are discovered in assignment specification the class will be informed via an announcement on Canvas. It is your responsibility to make sure you frequently check Canvas.

PLEASE NOTE : If the answer to your questions can be found directly in any of the following
- Subject outline
- Assignment specification
- Canvas FAQ and addendum
- Canvas discussion board

You will be directed to these locations rather than given a direct answer.

Extensions and Special Consideration

Please refer to subject outline.

Academic Standards and Late Penalties

Please refer to subject outline.