



DUBLIN INSTITUTE
of TECHNOLOGY
Institiúid Teicneolaíochta Bhaile Átha Cliath

Project Report

A&E Department

DT8265
Higher Diploma in Computing

Michele Ercolino

School of Computing
Dublin Institute of Technology

09/12/2017

Introduction

The developed software aims to organize operational tasks in the new medical facility in Dublin for the accident and emergency department (A&E). The objective is to ensure that the patient with severe conditions has the priority to be visited by the doctor. Each patient who reaches the A&E department will observe the following procedures.

First of all, he/she will be asked to give the receptionist his/her personal details. After registration, the patient will be called by the nurses to determine his/her conditions. The nurses will set a priority level which will be using to decide which patient needs to be visited first by the doctor.

The last step will be the doctor's treatment. After the doctor concludes the visit, the patient will be released.

Approach taken

The software has been developed based on some major features to provide to achieve the requested requirements.

These features are:

- Storing persistent data into a database, either patient data or patient transactions.
- Providing a waiting list for the nurses and a priority queue for the doctors for easy access next patient's data.
- Providing a user-friendly interface for A&E department's personnel.

As a consequence of these three major points, the software implementation phase has been divided into three sections: database, data structures for queues and graphical user interface (GUI).

Database

A database must be chosen to permanently store patient data. SQLite was picked for this occasion. As well as an Object-Relational Mapping (ORM) has been built to map the domain classes to the database, converting the domain classes into database tables.

The system is composed of only two domain classes and thus the database will have only two tables.

The patient class is identified into the database by an ID which corresponds to the Individual Health Identification number. Whereas, the transaction class has an auto-increment ID which the database automatically applies to each transaction. The two entities have a one-to-many relationship, so therefore the transaction entity has a PatientID attribute as foreign key.

Data Structures for Queues

The A&E department would need two different queues, one for the nurses and a second one for the doctors.

The first waiting list is a typical First in - First out (FIFO) data structures, where the first patient registered will be the first to be served, i.e. to be visited by the nurses.

The second list is a priority queue, where the next patient to be visited is determined not only by his/her arrival, but also by the priority value associated to him/her.

The chosen data structure to tackle the above condition is a Doubly Linked List. Using a doubly linked list allows employing only one data structure to handle either the waiting list for nurses or the priority queue for doctors.

This is possible due to the feature of the doubly linked list to access the data structure either from the front or the back of it.

Basically, when each patient arrives at the reception, the receptionist will fill the patient details in and then add each one to the waiting list. After that, the patients will be called by the nurse based on the first come - first serve rule. So, in this case, the doubly linked list will behave as a typical queue. The last patient will be added to the back of the queue while the first patient arrived, will be the first to be visited by the nurses.

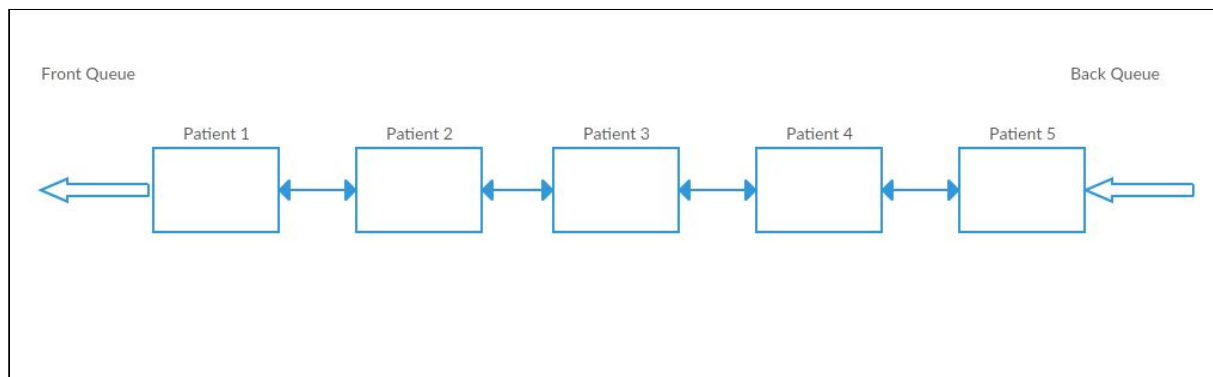


Fig 1. FIFO data structure using Doubly Linked List. The new patient is added to the back of the queue, and the first patient served is retrieved from the front. The patients are connected with a double arrow. This allows accessing the data structures in both directions.

Then, the nurses will set a priority level for each patient. This operation will modify the current queue. So the patient with priority will be pushed to the back of the queue.

If a new patient arrives at the reception, he/she will be added to the back of the queue, but as there is a patient with the priority set, the new patient will be pushed one position forward in the queue. When the nurses set a priority for the next patient, this will be pushed to the back of the queue up to the patient with higher priority than him/her. Then when doctors call the next patient, the patient will be retrieved from the back of the queue where the patient with the highest priority is placed.

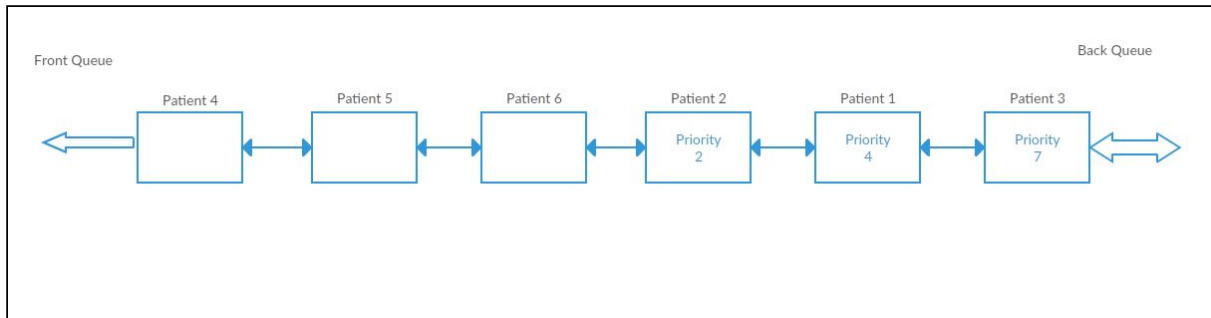


Fig 2. Two queues in one data structure. Typical FIFO from front to the back, up to the first patient with priority. Priority Queue from back to the front, up to the first patient with no priority set. So the nurses will get patients from the front whereas the doctors from the back of the list.

At the end, one single doubly linked list will have two queues in it. The waiting list for the nurses from the front to the back up to the first patient with priority set. The priority queue for doctors from the back to the front, up to the patient with priority set to zero.

GUI - Graphical User Interface

The third section involves the creation of a user-friendly interface to allow the A&E department's personnel to interact with the software. The interface is built using Swing. The interface consists of a single environment divided into six sub-windows. The sub-windows can be selected from a toolbar at the top.

The windows are:

Receptionist: window dedicated to the receptionists and it allows to register new patient (or update patient data if already exists) to the database and add it to the waiting list.

A&E Department

Receptionist Nurses Doctors Waiting List Patient History Search Patient

Individual Health ID

First Name

Last Name

Address

City

Nationality

Date of Birth yyyy-mm-dd

Phone Number

Comments

Save

Fig 3. Receptionist window. Save button will save the new patient to the database and add him/her to the waiting list.

Nurse: the window for nurses. The nurses can see the details of the next patient on the waiting list and set the priority (a value from 1 to 10) to them.

A&E Department

Receptionist Nurses Doctors Waiting List Patient History Search Patient

Individual Health ID

First Name

Last Name

Address

City

Nationality

Date of Birth

Comments

Set Priority

☒ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8 ☐ 9 ☐ 10

Save

Fig 4. Nurses window. Radio buttons to select the priority. Save button will reorder the waiting list based on the priority value.

Doctors: the software displays the next patient in the priority queue, then after the doctor finishes to visit the patient, the patient will be removed from the list (doubly linked list).

A&E Department

Receptionist Nurses Doctors Waiting List Patient History Search Patient

Individual Health ID

First Name Priority

Last Name

Address

City

Nationality

Date of Birth

Phone Number

Treatment Description

Terminate visit

Fig 5. Doctors window. Displays the next patient to be visited if any. The terminate visit button will remove the patient from any list.

In addition to the major windows, three more sub-windows are presented to the personnel.
Waiting List: this window displays the waiting list/priority queue of the patients.

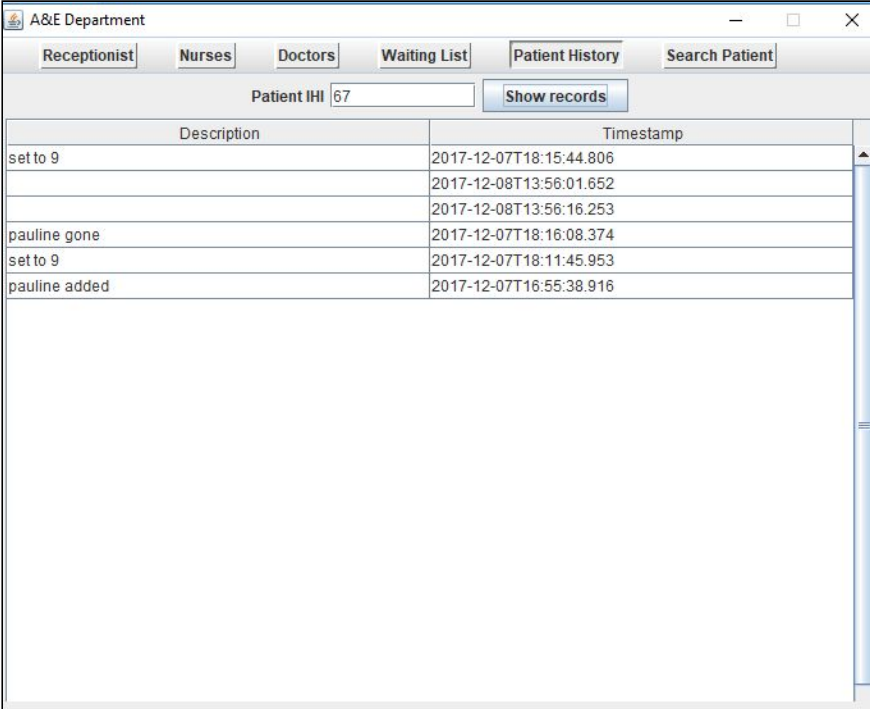
A&E Department

Receptionist Nurses Doctors Waiting List Patient History Search Patient

Patient Ihi	First Name	Last Name	Date of Birth	Priority
13	Mark	Mulling	1980-12-03	0
46	bill	pauli	1970-12-11	0
56	chery	kelly	1970-12-11	0

Fig 6. Waiting list window. Displays the current waiting list. The top one will be the next to be visited by the nurses.

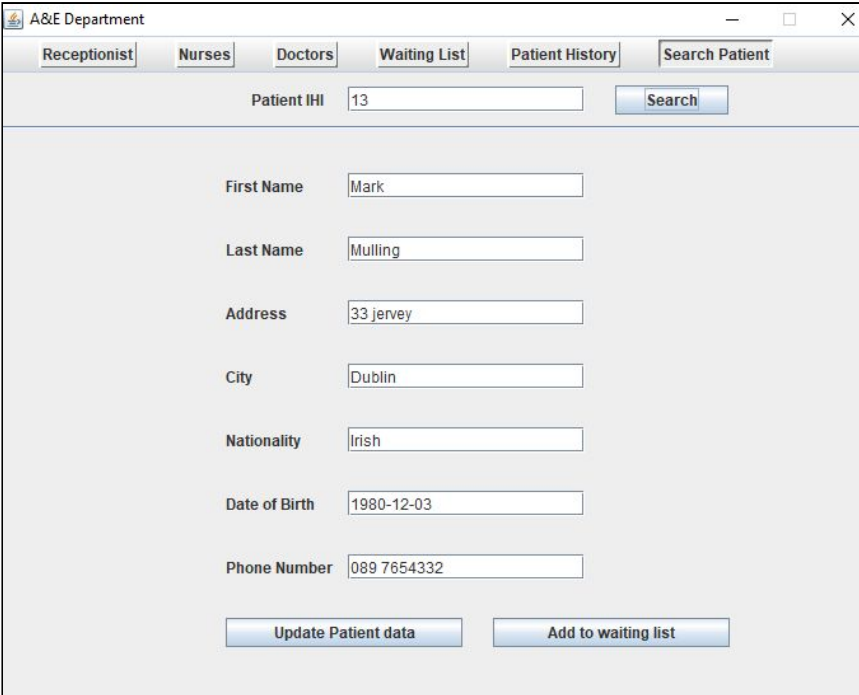
Patient History: in this window, it is possible to search for a patient by his/her ID and patient transactions will be displayed on the screen.



Description	Timestamp
set to 9	2017-12-07T18:15:44.806
	2017-12-08T13:56:01.652
	2017-12-08T13:56:16.253
pauline gone	2017-12-07T18:16:08.374
set to 9	2017-12-07T18:11:45.953
pauline added	2017-12-07T16:55:38.916

Fig 7. Patient History window. Displays the patient transactions.

Search Patient: a patient can be searched by ID. The stored details of the patient will be displayed if the patient has been already registered. After that, it will be possible to update the details if necessary as well as adding the patient to the waiting list.



Patient IHI: 13 [Search]

First Name: Mark

Last Name: Mulling

Address: 33 jervey

City: Dublin

Nationality: Irish

Date of Birth: 1980-12-03

Phone Number: 089 7654332

[Update Patient data] [Add to waiting list]

Fig 8. Search Patient window. Search patient by Individual Health ID. The patient data can be edited as well as the patient can be added to the waiting list.

Software Architecture

As mentioned earlier, the software can be divided into three main sections and thus the components of the software are divided into three areas that communicate with each other through interfaces.

Database Layer

The database area includes two interfaces: IDatabase and IOrm.

The IDatabase is an interface which declares three major behaviours that a database class must have: connecting to a physical database, and executing queries to the database itself either only for filtering and searching or for creating and modifying the rows into tables.

SQLiteDb class implements IDatabase, connecting to an SQLite database.

IOrm is an interface that allows the domain classes of the application to interact with the database, hiding the actual structure of the database.

IOrm declares the main functionalities to map the application objects to database tables, allowing to build tables based on the object's properties.

OrmMapper is a class that implements the IOrm interface. Creating tables, adding, removing and updating objects into the database are all operation that this class will handle.

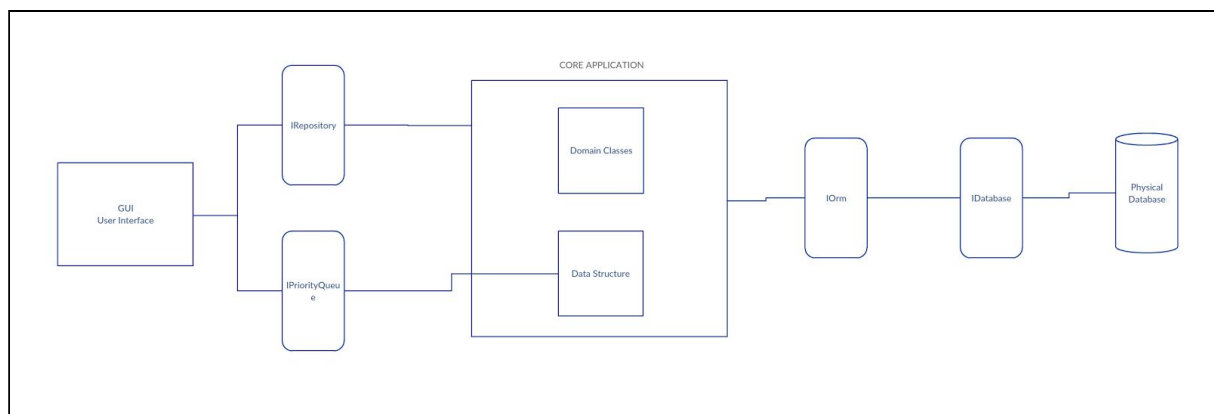


Fig 9. Overview of the application architecture.

Domain Classes

There are three domain classes into the application: Person, Patient, and Transaction.

Patient is a sub-class of Person and therefore inherits all the person properties.

IRepository is an interface which declares all the CRUD functionalities associated with domain classes such as adding, updating and removing objects from the storage. It works as

an interface between the ORM and the user interface. So the GUI will not interact directly to the database layer, but it will use the IRepository to achieve this. IRepository is implemented by BizLogic class.

Data Structures

A queue data structure has been built to handle the waiting list and priority queue.

The system includes two interfaces: IQueue and IPriorityQueue.

The latter is an extension of the IQueue as it is a special case FIFO data structure.

IQueue declares some main functionalities such as enqueue and dequeue to add and remove the new element to the queue.

IPriorityQueue adds two more functionalities to the IQueue interface which are strictly dependent on the priority condition.

DoublyLinkedList is a class which implements the IPriorityQueue. This class allows to handle the two lists (waiting list and priority queue) in one single data structure. This happens due to the specific features of the doubly linked list, i.e. communication between nodes in either way.

GUI

GUI class represents the user interface of the application and it is the class where all the previous components are interacted with each other to provide all the required functionalities to the users in a friendly way.