Report

The problem:

The recognized health provider has difficulties regarding the entry and egress of their patients in the clinical laboratory, when they check the history of the patient to decide to which unit they’re heading to (Hematology and general purpose), while also loading the info from the database.

Requirements (FR is Functional requirement and NFR is non-Functional Requirement):

FR1: The system allows the registration of a new patient to the laboratory, searching him in the database to see if he’s new.

FR2: The system allows the registration of the new patient to one of the units of the lab (Hematology or general purpose).

FR3: The system allows the egress of a patient from one of the units of the lab, indicating that he 's being attended.

FR4: The system allows the display of the actual status of the row of each instance of the laboratory.

FR5: The system must load the information of the already registered patients and also the sicknesses found on the database (txt file or csv).

FR6: The system allows the option to undo the action done before in order to get rid of a patient registration or checkout of a patient..

NFR1: The system has to implement generic structures done by the development team.

NFR2: The system has to be capable of continuing if the user does an error (error proof).

NFR3; The system has to be executed and managed via console.

Information supplied from our client and investigation

Hematology: Hematology is the medical specialty dedicated to the treatment, study and research of patients with blood diseases, in addition to its components.

Base diseases: there are many diseases that involve the blood, among the most recognized base cases are anemia, leukemia, hemophilia and cancer.

Generic structures used: The structures used in this project were the hashtable, the priority queue and stack.

Info to the client:

The priority is a number greater than or equal to 1 that the higher it is, the higher priority the person will have. This is calculated with respect to the information provided by the patient, such as age, being pregnant and, if they have an underlying disease, its severity. All the information can be typed so that misunderstandings do not occur.

Creative solutions:

* Use of arrays to store patients and to manage lab queues.
* Use of hashmap to store user information and priority stacks for lab queues.

Preparations and specifications:

| **Name** | FR1: Register new patient to the lab’s database | | |
| --- | --- | --- | --- |
| **Summary** | The user digits the required info of the patient, after doing this the patient’s data will be successfully added in the database | | |
| **Inputs** | **Name input** | **data Type** | **Condition of selection or repetition** |
| ID | String | N/A |
|  | name | String | N/A |
|  | sex | SEX | N/A |
|  | age | int | N/A |
|  | pregnant | boolean | N/A |
|  | underDeceases | String[] | If the patient has more than 1 decease |
| **Activities needed to accomplish the goal** | The user must type all the information about the patient correctly, following the program’s instructions | | |
| **Result or postcondition** | The info of the patient has been stored on the database and he’s been moved successfully to one of the units | | |
| **Output** | **Output name** | **Data type** | **Condition of selection or repetition** |
| patient | Patient | N/A |

| **Name** | FR2: Register new patient to a unit of the lab | | |
| --- | --- | --- | --- |
| **Summary** | After storing the patient’s info in the database the user decides to put him in one of 2 lab units: Hematology or general purpose | | |
| **Inputs** | **Name input** | **data Type** | **Condition of selection or repetition** |
| patient | Patient | N/A |
| **activities needed to accomplish the goal** | The user must select one of 2 units of the lab according to the patient’s needs. | | |
| **Result or postcondition** | The patient has been moved successfully to his assigned lab unit, a patient mustn’t be in a unit already in order to be able to be assigned to a unit. | | |
| **Output** | **Output name** | **Data type** | **Condition of selection or repetition** |
| hematología | PriorityQueue | N/A |
|  | general | PriorityQueue | N/A |

| **Name** | FR3: Egress a patient from the lab unit | | |
| --- | --- | --- | --- |
| **Summary** | The patient is removed from his unit’s queue | | |
| **Inputs** | **Name input** | Data type | Condition of selection or repetition |
|  | patient | Patient | N/A |
|  | hematologia | PriorityQueue | N/A |
|  | general | PriorityQueue | N/A |
| **Activities needed to accomplish the goal** | The user must look up the patient in the queue and extract him. | | |
| **Result or postcondition** | The patient has been successfully cleared from the unit’s queue | | |

| **Output** | **Output name** | **Data type** | **Condition of selection or repetition** |
| --- | --- | --- | --- |
|  | info | String | N/A |

| **Name** | FR4: Show unit status | | |
| --- | --- | --- | --- |
| **Summary** | The user is able to see the status of the unit’s queue | | |
| **Inputs** | **Name input** | **data Type** | **Condition of selection or repetition** |
| hematología | PriorityQueue | N/A |
|  | general | PriorityQueue | N/A |
| **Activities needed to accomplish the goal** | The user must select the option that allows the information display | | |
| **Result or postcondition** | The status of the unit is shown to the user. | | |
| **Output** | **Output name** | **Data type** | **Condition of selection or repetition** |
| info | String | N/A |

| **Name** | FR5: Load the patient’s information and sicknesses from the database | | |
| --- | --- | --- | --- |
| **Summary** | The user is able to load the information of all the patients and the sicknesses | | |
| **Inputs** | **Name input** | **data Type** | **Condition of selection or repetition** |
| database | HashTable | N/A |
| **Activities needed to accomplish the goal** | The user must select the option that implies the display of the information | | |
| **Result or postcondition** | There has to be at least info stored from 1 patient in order to work | | |
| **Output** | **Output name** | **Data type** | **Condition of selection or repetition** |
| info | String |  |

| **Name** | FR6: Undo the action done before | | |
| --- | --- | --- | --- |
| **Summary** | The user is able to unregister recently registered users with an undo option | | |
| **Inputs** | **Name input** | **data Type** | **Condition of selection or repetition** |
| hematología | PriorityQueue | N/A |
|  | general | PriorityQueue | N/A |
|  | node | NodeThreeValue | N/A |
| **Activities needed to accomplish the goal** | The user must select the option to undo the action | | |
| **Result or postcondition** | An action done beforehand has to be done in order to access the undo option | | |
| **Output** | **Output name** | **Data type** | **Condition of selection or repetition** |
| info | String | N/A |