|  |
| --- |
| AHS INFIRMARY SYSTEM |
| MIS 122  Ateneo de Manila University |
|  |
| **Chua, Milan |dela Cruz, Kelvin| Rivera, Nica| Valeña, Manolo| Zarate, Franco**  **March 18, 2013** |

Contents

[I. Introduction to the Study 1](#_Toc351379880)

[Methodology 1](#_Toc351379881)

[II. Current Systems Study Results 2](#_Toc351379882)

[III. Proposed System 3](#_Toc351379883)

[Overview 3](#_Toc351379884)

[Screens and User Interfaces 4](#_Toc351379885)

[Report 15](#_Toc351379886)

[IV. Implementation 18](#_Toc351379887)

[V. Maintenance Plan 19](#_Toc351379888)

[VI. Appendices 20](#_Toc351379889)

[Functional Requirements 20](#_Toc351379890)

[Proposed Physical DFD 21](#_Toc351379891)

[Use Case Description 23](#_Toc351379892)

[Final Database Design 31](#_Toc351379893)

[Final Database Dictionary 34](#_Toc351379894)

[Implementation 49](#_Toc351379895)

Executive Summary

Our team’s project involves the automation of the major processes of the Ateneo de Manila High School Health Services Office. The AHS Health services office is currently utilizing a paper-based system of record management. Handling records of roughly 2000 students provides the offices with difficulties such as records storage and retrieval, data integrity and the possibility of record damage or loss. Our system will aim to improve the infirmary processes by means of computerizing the office’s record management system. The processes we are targeting mainly involves the storage of records such as student immunization records and physical exams as well as daily infirmary visit tracking such as clinic visits and trauma cases. Lastly, the system will feature a report generation feature that currently includes the complaint tally report. Finally, to avoid having to manually input the records of all students, the system will feature an import function that would accept CSV files from obtained from the registrar (for student records) and the Human Resources Office (for the employee records).

Introduction to the Study

The Ateneo De Manila High School Infirmary has been manually inputting data into papers in order for them to store records and health files for each of their students, employees and others. In this regard, our client is looking for a way in order to computerize the majority of their medical records and the like. This system that they are requesting will resolve the dilemma of file keeping and storage due to the vastly growing number of student population. Also, an automatized system will improve and make it easier for them to tally complaints and generate reports about them.

The group has gathered the necessary information with regards to our client's dilemma through extensive observations and interviews. Seeing as their problem is their processes being paper-based, an automated system will ensure them of more efficiency and order when working with important medical files. Also, some of these forms contain similarities therefore they are prone to data redundancy. With having a system, it will lessen the chance or actually eliminate the redundancy. Due to the number of students that the school has, it is a hard task for anyone to look for pertinent medical background or information on a particular student.

Having said the preliminary requirements of having a more organized data storage and being able to print out reports, the objectives of the project will be to provide the client with the necessary tools that may enhance their work. The features of the system will include basics like standard Profile Management and Report Generation. As for the scope of the project, there will be features in the system like the addition of medical examination forms, clinic visits, immunization records, etc. In short, the system will encompass the different paper forms that the infirmary currently has. The group decided not to include certain parts of the overall examination like the dental records and the medical drugs given as such functionalities require a separate system altogether.

Methodology

The Ateneo de Manila High school Health Services Office is currently using a manual system of record keeping which is purely paper based. Improving the efficiency in the system is the aim of the project especially since the current system records are in danger of getting lost or destroyed. The large volume of records currently in the office would mean that searching for a particular record in the system would require an extensive amount of time to achieve. Dr. Cruz and Dr. Salcedo have provided the group with a thorough report on what they would want to see in an automated system. The report contains an outline of a system, and through this report, the group has gained more knowledge on the type of system that they currently need. With this, the group can further analyze the system and bring about new ideas that may be able to improve the current system. The report covers the current system the infirmary has, the different processes they perform, as well a baseline description on what they want for their system. To supplement our study of the current system we interviewed and collected data from the doctors in the infirmary. Our team also visited the grade infirmary on the suggestion of the high school health services in order to give us a walkthrough of what the accomplished system should be like.

The physicians in the infirmary also showed us existing documents and forms that they use on a regular basis. After which, they gave us copies of the form which the group plans to use for User Interface design and construction as well as for the process purposes

Current Systems Study Results

Our initial findings have yielded a few major problems for the AHS Infirmary system. First, the current system is purely paper-based with no duplicates to be kept as back up. This can be potentially problematic in cases such as files going missing or getting damaged. Records will be permanently destroyed and will require extensive efforts to be recreated. The amount of forms required to be filled out can also result in problems regarding storing and retrieving the records. Handling reams of forms for thousands of students require large volumes of physical storage that can cramp up the workspace. Retrieving different records would also entail having to go through the filing cabinets that could be time consuming. Lastly, from our findings we have seen similar fields present in several of the forms the infirmary requires. These similarities make the processes prone to work redundancy.

Having paper based systems would entail that the infirmary’s physicians would have to write data on paper forms.

The infirmary has a main clinic and a remote clinic. Keeping track of the details and happening from the remote clinic would be hard. Moreover, tallying and combining reports from both clinics would entail manual methods of putting the data together.

The group plans to solve these problems with the system that the group is proposing. With having an automated system, this would resolve having so many papers in the infirmary. Also, with having an automated system, it would be able to secure the necessary profiles and records from loss or damage. The system would also be able to provide less time in searching for such records needed, with the addition of profile system that the group will implement

The current system currently has 5 inputs; namely, First year medical clearance, annual physical examination, immunization, clinic visits and the trauma case. The current system output only includes the complaint tally.

Proposed System

Overview

Our system will begin with importing the data of students from the registrar. By doing so, our data will be consistent with the registrar’s data, which should secure the integrity of the data.

For the proposed system, since we are only doing automation to their system, the processes, inputs and outputs of their previous system will be retained. In doing so, we can increase the efficiency because we will be able to eliminate the additional time of going through all the paper based files when dealing with patients. The computerized system will already provide them all the needed information at their screen. Moreover, the system will improve their process of creating new records at the start of each school year, when a new batch of students comes in. They will have a more organized way of handling the incoming freshmen files, especially the medical clearance and immunization records each student will pass. In addition, using the system would also make the physicians be able to search for necessary data relevant to what they want to review.

The technology that the group's using are MySQL, MySQL Workbench and Visual Studio. These are the technology that was used by the group to create the AHS infirmary system.

MySQL

MySQL is one the most popular database services available for free and is the team’s top choice for the project. This is the database service which the developers are most familiar with.

MySQL Workbench

This technology is going to be used for specialized things like backing the system up. The various features of this program allows the group and/or the client to be able to schedule backups whenever they wish without having to go to the command prompt to navigate the database. This allows easier transition for both parties. This program also allows the user of the system to select where the program or data will be stored. Lastly, the system will be used to recover and import data whenever necessary.

Visual Studio

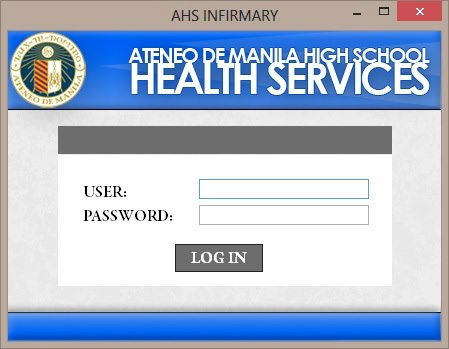
This is the primary IDE that the team used during development. It’s one of the most versatile IDEs supporting C++, C#, MS Visual Web developed as well as other programming languages. The team chose this IDE along with the C# programming language because it was the combination that the developers were most acquainted with along with the use of MySQL

**Database**

Our database is created using the MySQL language. It is made up of 19 tables namely: login, profile, contact number, student, medical certificate, employee, visitor, vaccine, immunization record, medicine, maintenance medication, clinic visit, complaint, diagnosis, treatment, physician, trauma case, affected areas and physical examination. Information that we will be getting from the registrar of Ateneo High School will be imported to the SQL database via .CSV file. This database will be used and connected with the system’s front end or user interface.

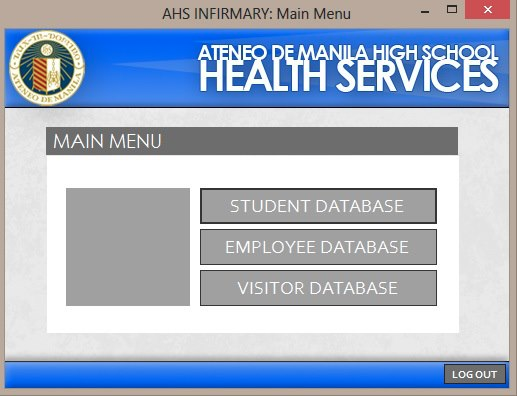
Screens and User Interfaces

**a. Log-in**



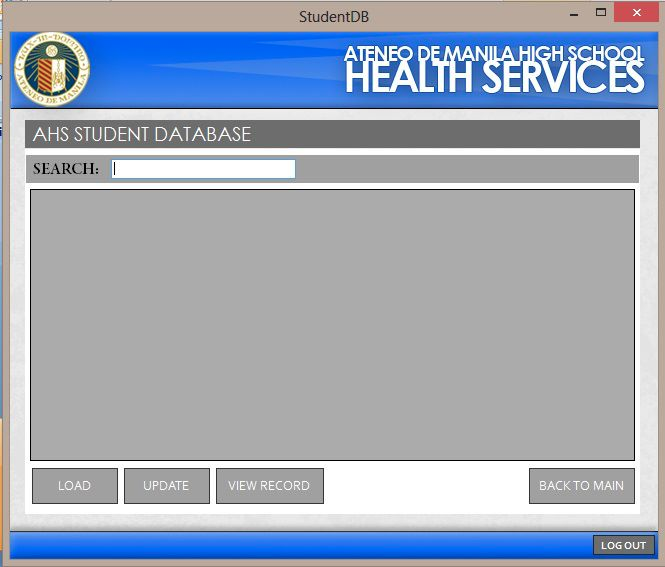
The Log-in screen will be the first screen that the user of the system will see upon opening the system. The user will log-in with his or her username and password. This screen provides initial security for the user’s access to the system.

**b. Main Menu**

****

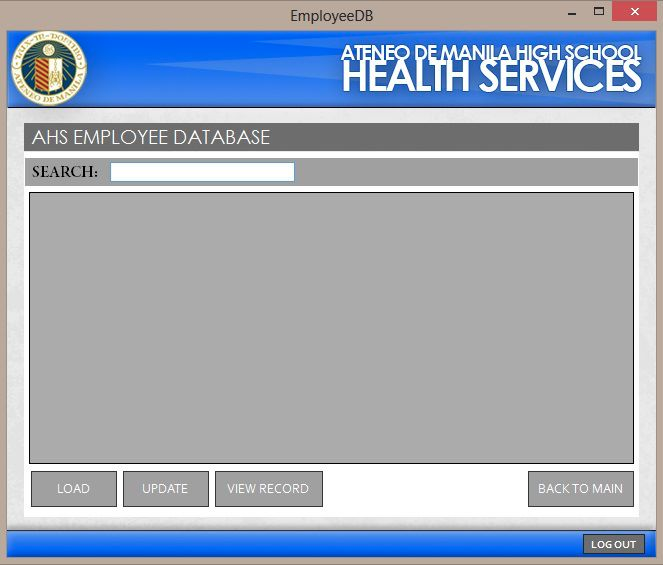
The main menu will be the 2nd screen the user will be able to see upon logging in. Here, the user can be able to select which database he or she would want to access. Our system is divided into 3 databases, namely student, employee and user. Selecting which database would lead to the specific records they need. A log-out function is also there to go back to the log-in screen.

**c. Student Database**

****

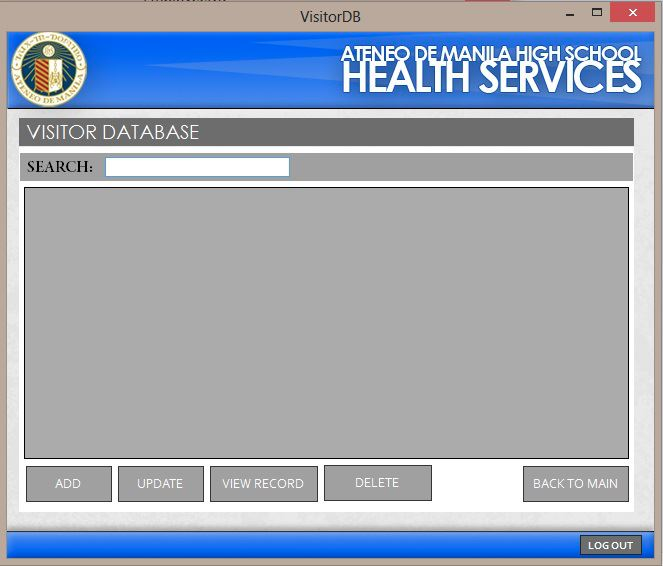
This screen will appear upon selecting student database in the main menu. Here, the database of students will appear, as well as the search function. There will be an update function for things the user would want to edit in the database. It also has a load function where the users can a load new batch of records in the form of CSV files. Selecting a patient and pressing view record will lead to the specific student’s records. The button “back to main” will redirect the user back to the main menu.

**d. Employee Database**

****

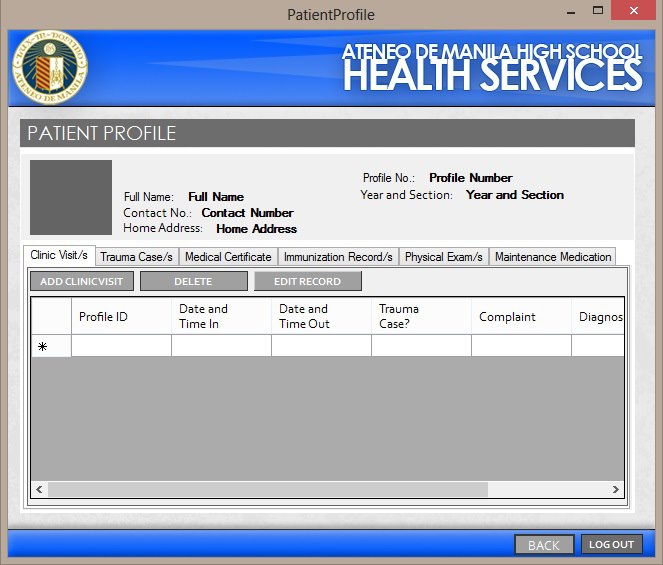
This screen will appear upon selecting employee database in the main menu. Here, the database of the school’s employees and staff will appear, as well as the search function. There will be an update function for things the user would want to edit in the database. It also has a load function where the user can load new records in the form of CSV files. Selecting a patient and pressing view record will lead to the specific employee’s records. The button “back to main” will redirect the user back to the main menu.

**e. Visitor Database**



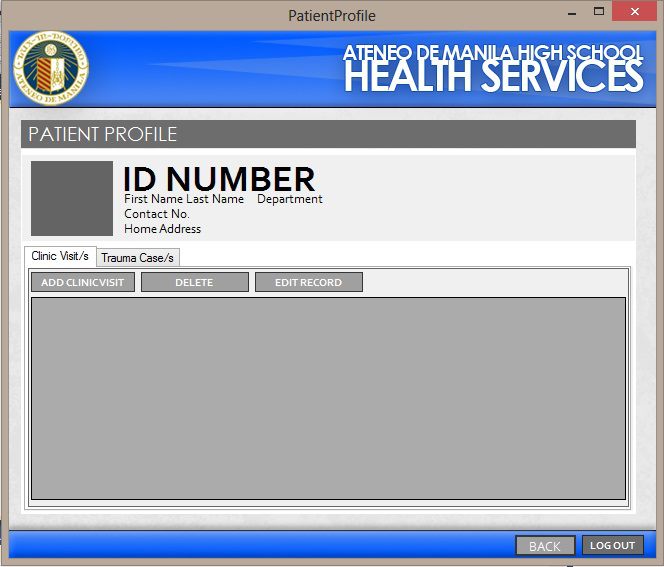
This screen will appear upon selecting the visitor database in the main menu. Here, the database of the school’s visitors (walk-in patients that are not students or employees) will appear, as well as the search function. This screen is slightly different from the student database and employee database since this screen has an add function to put in data since visitor’s data aren’t imported. There will be an update function for things the user would want to edit in the database. Selecting a patient and pressing view record will lead to the specific visitor’s records. There is also a delete function to delete these walk-in data in the case of errors. The button “back to main” will redirect the user back to the main menu.

**f. Patient Profile (Student)**

****

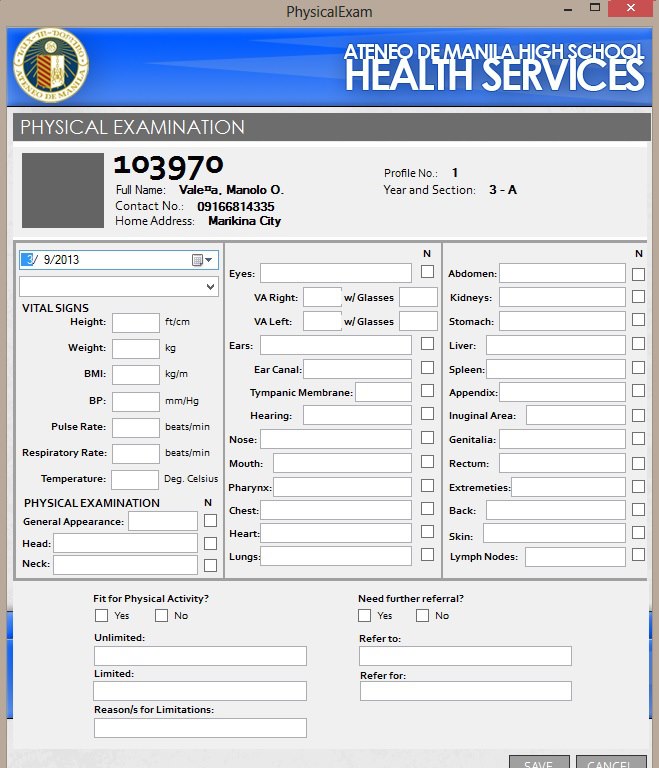
This screen appears upon selecting and viewing the record of the student from the student database screen. The screen initially shows the clinic visit menu where the user can add or delete a clinic visit. Going to the next tab, under the trauma case tab, the trauma cases of the student could be viewed. For the medical certificate tab, the user can upload an image of the patient’s medical certificate. For immunization records, there is also an option to add immunization records. For physical exam, there is an option to add a physical exam. The same goes for maintenance medications.

**g. Patient Profile (employee)**

****

Unlike the patient profile (student), the employee profile only has the clinic visit section where the user can add a clinic visit, and a trauma case section where the user can add a trauma case.

**h. Physical Exam**

****

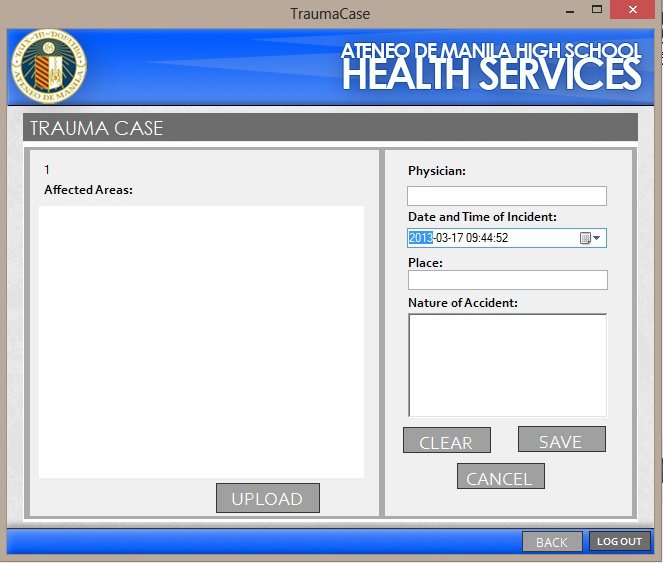
This screen appears when the add physical examination button is pressed under the physical examination tab of the patient profile of a student. Here, the data could be inputted in the fields as well as check radio buttons if findings are normal. The data in this form is consistent to the physical examination sheet of a student that is accomplished. There are also radio buttons for the summary below to tell if the student is fit for physical activities or needs referrals. The save button will save the inputted data for storage and record.

**i. Add Clinic Visit**

****

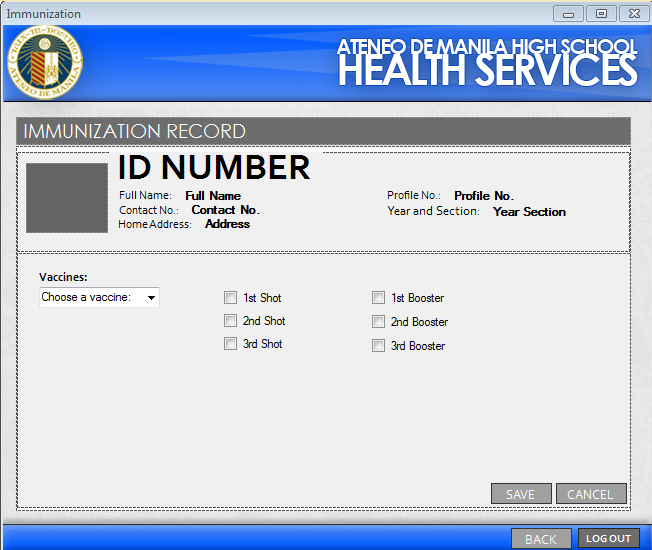
The add clinic visit screen will appear upon pressing the add clinic visit under the clinic visit tab of the patient profile. Here, there are input fields that could be accomplished and a radio button to signal a trauma case. The clear button will clear all inputs on the text fields and the save button will save all inputs.

**j. Trauma Case**

****

The trauma case screen will appear when the radio button in the add clinic visit is pressed. The user can upload an image of the affected area of the patient’s trauma case. Also, descriptions of the case can be inputted on the input fields. The clear button will clear all data inputted, and the save button will save all data inputted for record.

**k. Immunization Record**

****

The immunization record screen will appear upon selecting the immunization record tab on the patient profile (student) screen. This screen has a dropdown option to let the user select the type of vaccine, then a radio button to show the number of that vaccine.

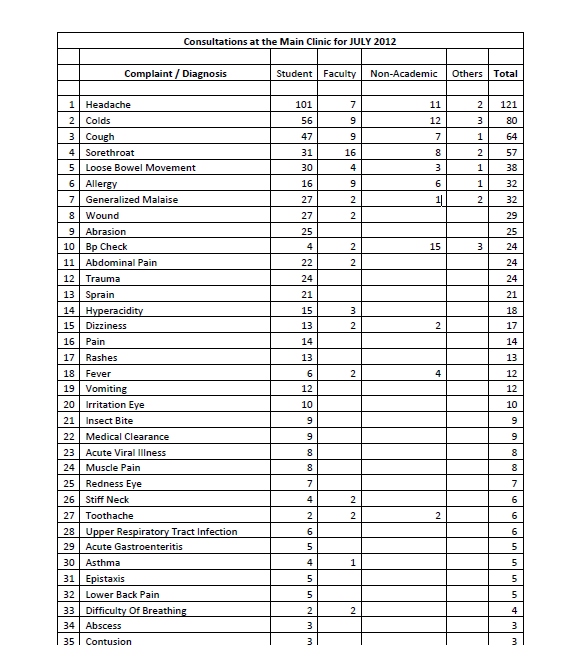
**l. Maintenance Medication**

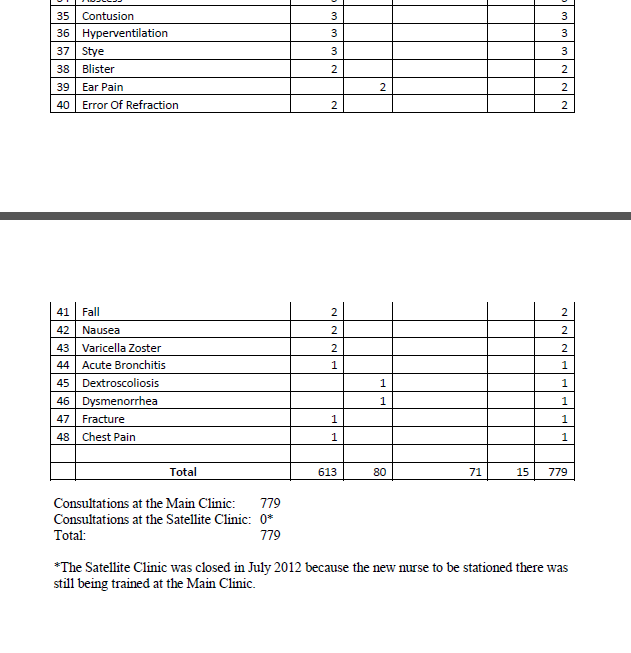


The maintenance medication screen will appear upon selecting the maintenance medication tab under the patient profile (student) screen. This screen will show a table of the patient’s maintenance medications. The data on this screen can also be edited like the immunization record. The data on this screen can be saved by pressing the save button.

Report

For the meantime, the system features a print function that allows the user to print out a tally of complaints based on the complaints recorded during clinic visits. Below is a copy of the report that the infirmary has provided us.





Implementation

For our implementation process as well, the group has already planned to install the necessary programs that may aid and/or are required to make the system work. These programs include the likes of MySQL, MySQL Workbench, .NET and others. Also after installing these programs and database systems, the group will configure the database in order for the client to use it accordingly. Correct versions or updated versions of the programs will be used in order to give them a system that has fully updated integrating software.

Before the client can actually use the system, the group plans to import the initial data like the list of users. This part of the process will also aid in the security of the system. Next, we would populate the tables using the records coming from both the Registrar's office with regards to the basic information of students and employees and the files that the infirmary currently holds with regards to the medical records and files of the students and employees. The records which will be coming from the Registrar's office of the high school will be requested upon, and that it be sent as an excel .CSV file. Using this .CSV file, it would be easier for the group to import such data into the database systems.

Lastly, the group will have one computer or laptop ready to have the same files and systems that will be installed on the client's computer. With having a laptop that has the same system, the group will be able to simulate and fix any errors.

The biggest change we made for the system involves the entity relations. We further divided the entities in our ERD in order to be correctly used and implemented in our database. We decomposed entities such as the “Clinic Visit” entity to correctly illustrate the decomposition from normalization.

As a result of the changes in the ERD, we have also revamped the Data Dictionary to reflect the changes. The changes include the new tables as well as the corrected values and data types for each table. Several changes in the User Interfaces have also been made based on the client’s feedback. The overall design has been modified as well as labels within the forms and additional fields not present during the last version to accommodate required information.

The group biggest hurdle was reviewing as well as researching new knowledge regarding the technology used particularly the programming language. Numerous changes and additions were also suggested by the client during progress reports which made the development more complex. The group had difficulty balancing the changes to be made and the limited time provided.

Maintenance Plan

Backups will be scheduled depending on how often data in the system is changed. For example, the clinic accepts an average of 3-4 patients a day, then weekly backing up will suffice. The client or any other user in the clinic should know how to back up the files for it may be necessary for each one of them in the clinic to operate and manage the data. With this in mind, the group will supervise a tutorial on backing up files. The group will specify where to put the files in order for the client to be able to recover data easily.

For the recovery process, given the group has already taught the client on where to put the backed up files, it would be necessary for the group to also teach and give detailed instructions to the client on how to recover them, may it be in the form of CD, inside Dropbox or any other storage device.

The probable technology that would be used in this project is the MySQL Workbench for it allows users to schedule their own backups, specify their own file directory, etc.

For the security requirements, the system will limit the access of people who can see the data. The group will enforce this by putting up accounts with usernames and passwords. For now, there will be generic accounts. This means that all accounts have the same privileges of editing and seeing data found in the system's databases. As for the passwords, the group will enable password hashing to add more security to the user accounts. In additional, the system would also allow the users to change their passwords whenever they see fit. The system will also feature access logs to be able to monitor those who logs in and when it was made.

The system that we will install is safer than those of web systems, because it will be installation based. It is also offline; therefore no other persons can access the data other than those who can use the computer that the clinic currently has.

For support, the group will provide the client with user manuals. The manuals will include the common issues while using the system. For example, the manual will have the things or inputs that will cause error for the system. Somehow, it may serve as the dos and don'ts as well. There will also be a part in the manual wherein the contact details of the group will be posted. This will allow the client to contact the developers whenever they need help in fixing bugs, etc. Along with this, the manual will contain the possible response time of the group whenever called. Inside the manual also would be the contract period of the support. Having a contract would engage the client and have agreements on what the group would be able to provide them during the support and maintenance phase. In addition, this would prevent dissatisfaction. Moreover, having a contract would be a formal agreement that would incorporate what can be done and what additions could be implemented. Lastly, the source code will also be placed in the manual. The group will teach the client the code, and if ever the contract expires, anyone they will hire next can easily understand the code that was already put up. From there, they can upgrade the system, etc. Additions made to the system such as the dentist forms with radio buttons that the client requested for would be added during this phase. It has been agreed upon with the client and decided upon the group that this would be the course of action in order to focus on the functional requirements of the system.

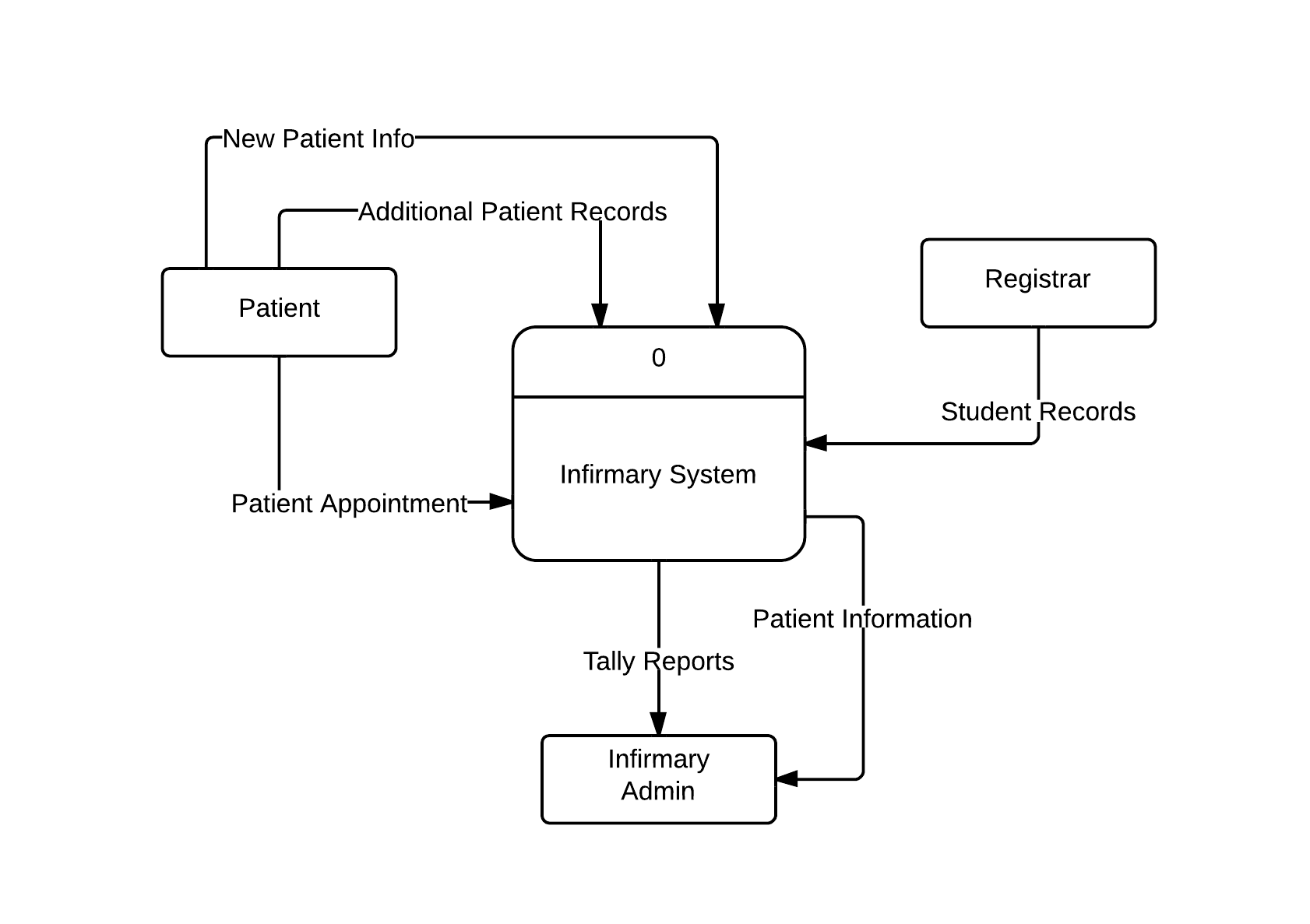
Appendices

**Functional Requirements**

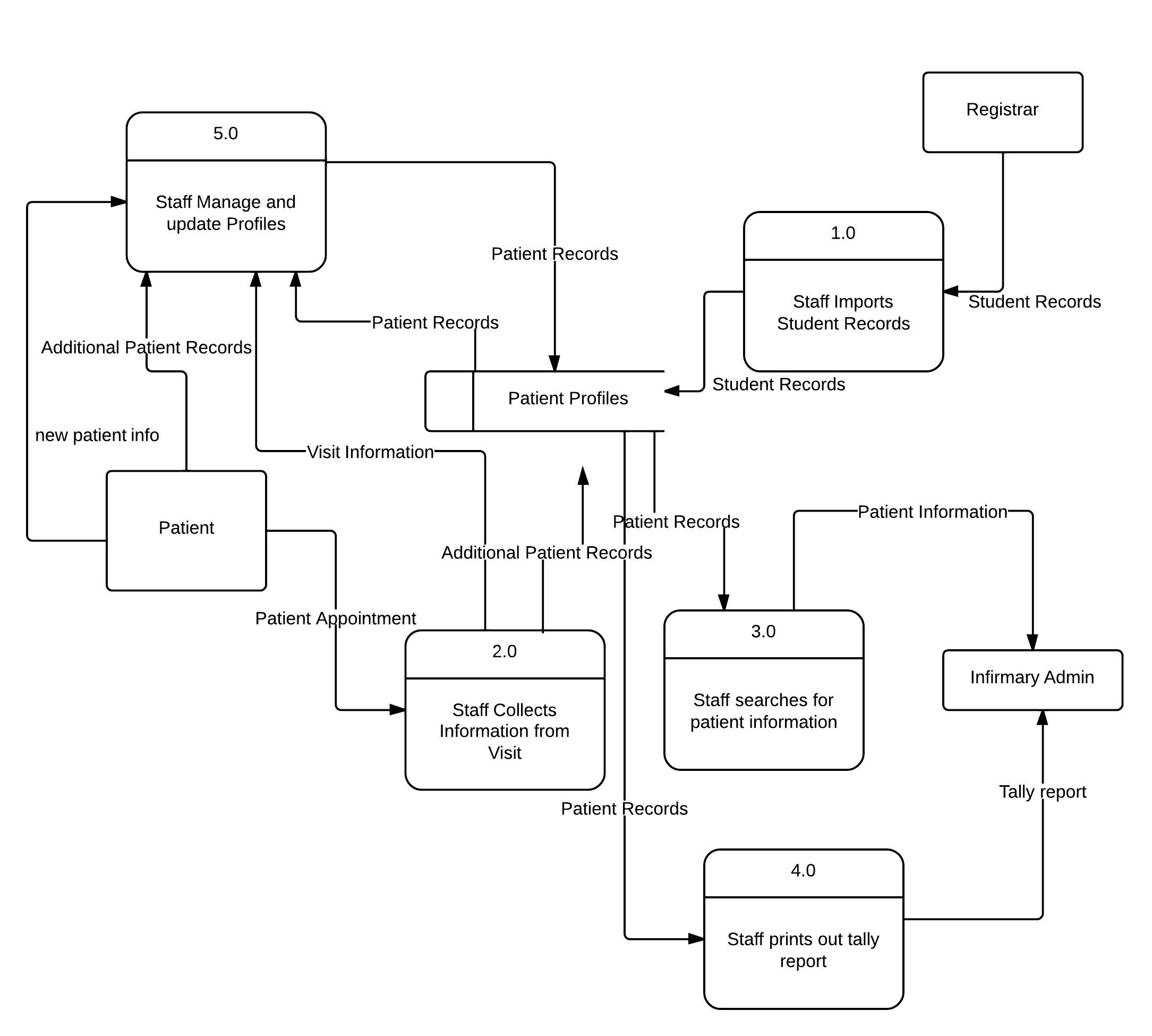
|  |  |
| --- | --- |
| **Functional Requirements (Description)** | **Priority** |
| Patient Profile The collection of records for all patients in the database | High |
| Tally Complains This will be for the use of the infirmary. The function aims to tally the complaints based on the records in the database | High |
| Update Profile This allows the user to update any existing profile as to be able to write new records for the patient profile. Users can update the following   * Clinic visits * Physical exam results * Trauma cases * Immunization records * Maintenance medication * Medical Certificate | High |
| Search Profile This will allow the user to search for the patient in the list of records given the user will type or give the user’s name or ID number. | Moderate |
| Import data Allows the user to import student database from the registrar | Moderate |
| Account Management This should allow the admin to create user accounts, update user accounts, delete user accounts as well as retrieve passwords for users who have forgotten their passwords | Moderate |
| Create  Profile Allows the user to create a profile patients that are not in the database | Low |
| **Nonfunctional Requirements** | **Priority** |
| Data Backup and recovery - Allows the user to schedule backups of the database as well as give the user the ability to recover them | High |
| User Login and Restrictions The system will implement a login system, because there are users that can create, search, view, and update profiles. There are users that can only view the profiles. | Low |

Proposed Physical DFD

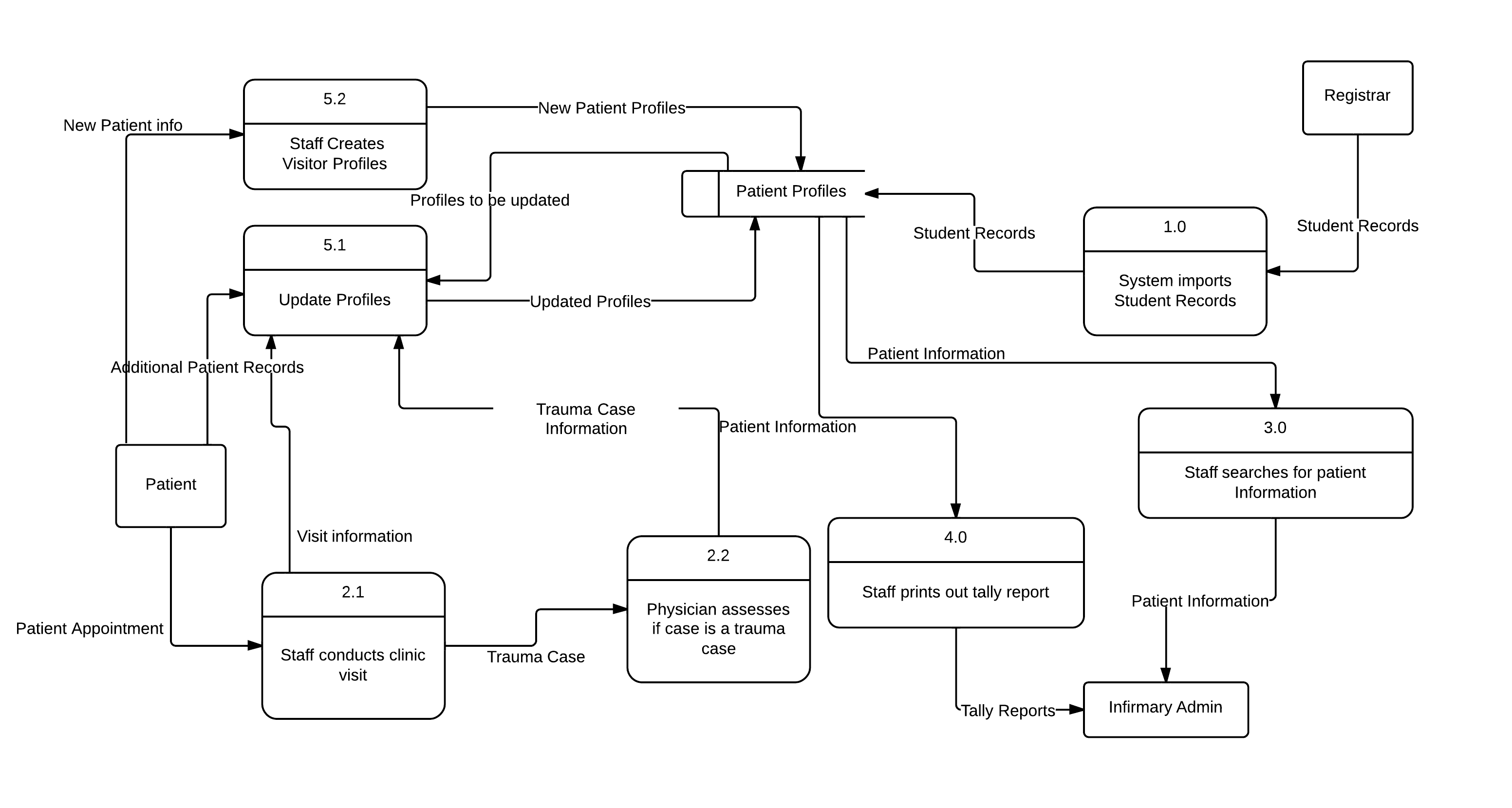
Context Level



Level 0



Level 1



Use Case Description

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-01 | | |
| **Use Case Name:** | Login | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | To allow users to log in the system and use it | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * none | | |
| **Post Condition:** | Users are logged in and are able to use the system | | |
| **Business Trigger:** | Users need access to the system features | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. User inputs the username and password 2. User clicks Login 3. User accesses the system | |  | 3.1 User reenters username and password if incorrect |
| **Business Rules:** | | | |
| Staff must have user accounts | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-02 | | |
| **Use Case Name:** | Import patient records | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | To import all records to the system | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff is logged in | | |
| **Post Condition:** | All patient records are imported to the system | | |
| **Business Trigger:** | * Staff needs to run system for the first time | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
|  | |  |  |
| **Business Rules:** | | | |
| Staff must have CSV files of records | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-03 | | |
| **Use Case Name:** | Select database | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | So select a database to use of the system | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff is logged in | | |
| **Post Condition:** | Staff gains access to the records in the selected database | | |
| **Business Trigger:** | * Staff needs access to the database and records | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. Select student database | |  | * 1. select employee database   2. select visitor database |
| **Business Rules:** | | | |
| Staff can only access one database at a time | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-04 | | |
| **Use Case Name:** | View Record | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | View a patient’s record | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff is logged in | | |
| **Post Condition:** | Staff is redirected to patient profile | | |
| **Business Trigger:** | * Staff needs to view or update a patient’s records | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. Staff selects a name in the data grid 2. User clicks “view record” button | | 2.1 Click Back to Main | * 1. Staff searches for a specific student using the search bar |
| **Business Rules:** | | | |
| Patient must have a profile in the system | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-05 | | |
| **Use Case Name:** | Add clinic visit | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | Add a clinic visit record to the profile | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff is logged in * Staff must be in the Patient profile screen | | |
| **Post Condition:** | User is redirected to the clinic visit screen | | |
| **Business Trigger:** | * Patient visits the clinic | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. Select the Clinic Visit tab on the patient profile 2. Click add clinic visit 3. Fill in the necessary fields 4. Click Save if student does immediately exits the infirmary | | 4.1 Click Clear to clear all fields  5. Click Cancel | 4.2 Click Save and Time out  5.1 Click Time out on the patient profile when the patient exits |
| **Business Rules:** | | | |
| Patient must have a profile in the system  Patient must provide necessary information to staff | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-06 | | |
| **Use Case Name:** | Add trauma case | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | Add a trauma case record in the patient’s profile | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff must be logged in * User must indicate in the clinic visit form that case is a trauma case | | |
| **Post Condition:** | New trauma case record is added | | |
| **Business Trigger:** | * Staff needs to create a trauma case record | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. Fill in fields with necessary information 2. Click save | | * 1. Click clear  1. Click Cancel |  |
| **Business Rules:** | | | |
| Patient must have a profile in the system  Patient must provide necessary information to staff | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-07 | | |
| **Use Case Name:** | Add physical exam | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | Add a physical exam record to the profile | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff must be in the patient profile screen * Staff must be logged in | | |
| **Post Condition:** | New trauma case record is added | | |
| **Business Trigger:** | * Student has a physical exam | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1.Fill out necessary fields  2.User clicks save | | 2.1 User clicks cancel |  |
| **Business Rules:** | | | |
| Patient must have a profile in the system  Patient must provide necessary information to staff | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-08 | | |
| **Use Case Name:** | Add immunization record | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | add an immunization record to the profile | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff must be logged in * Patient must have vaccination record * Staff must be in the student patient profile screen | | |
| **Post Condition:** | New vaccination record is added | | |
| **Business Trigger:** | * Staff needs to update vaccination record | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. User Selects vaccine to be updated 2. User Selects number update type (shots or boosters) 3. Click save | | 3.1 click cancel |  |
| **Business Rules:** | | | |
| Patient must have a profile in the system  Patient must provide necessary information to staff | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-09 | | |
| **Use Case Name:** | Add maintenance medication | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | Add new maintenance medication record to profile | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff must be logged in | | |
| **Post Condition:** | New maintenance medication added to profile | | |
| **Business Trigger:** | * Staff needs to update patient profile | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
|  | |  |  |
| **Business Rules:** | | | |
| Patient must have a profile in the system  Patient must provide necessary information to staff | | | |

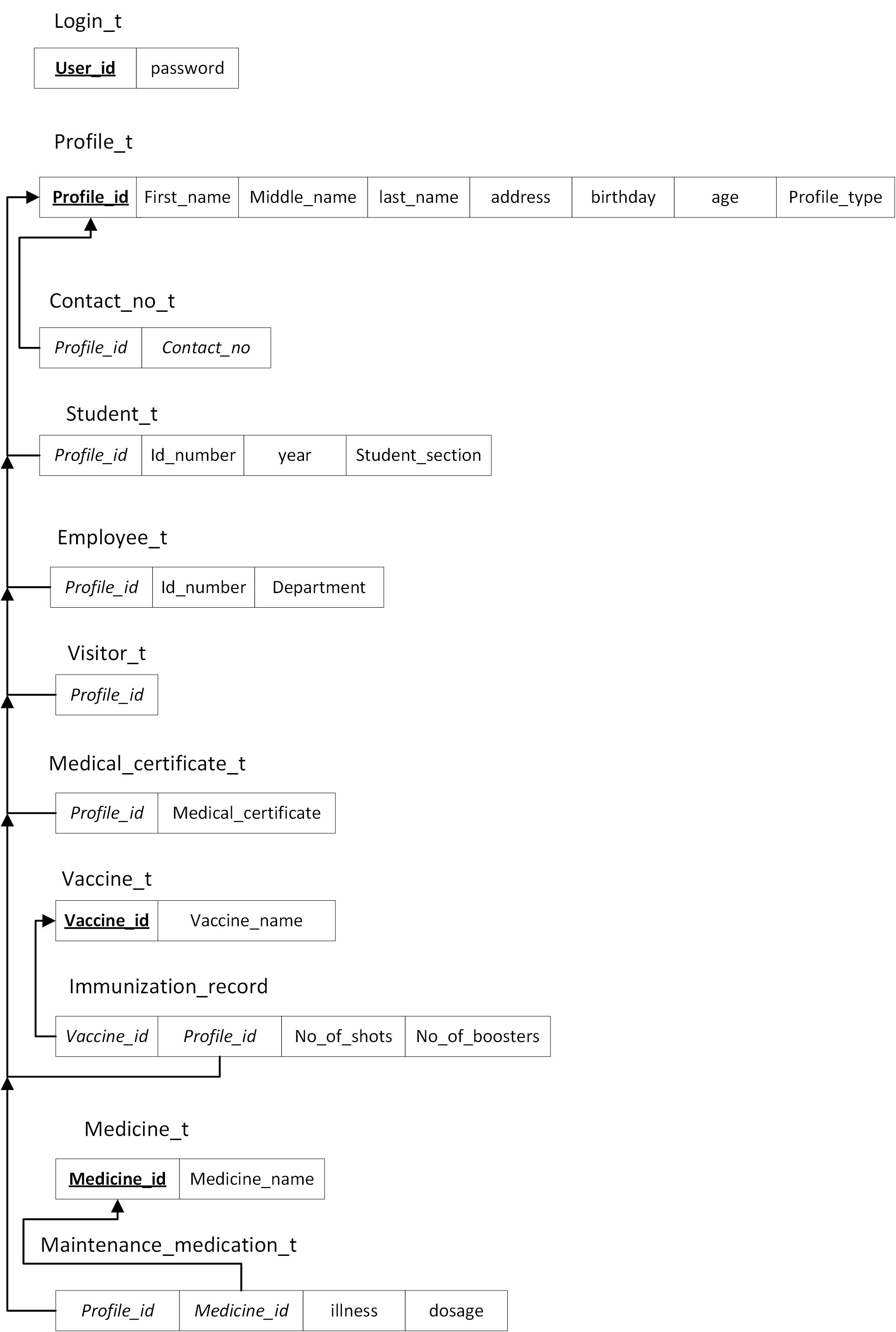
|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-10 | | |
| **Use Case Name:** | Print complaint report | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | Print a quarterly report of the complaints | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff must be logged in | | |
| **Post Condition:** | System generates a printout of quarterly complaints report | | |
| **Business Trigger:** | * Staff needs quarterly complaint report | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
|  | |  |  |
| **Business Rules:** | | | |
| System must have a tally of complaints | | | |

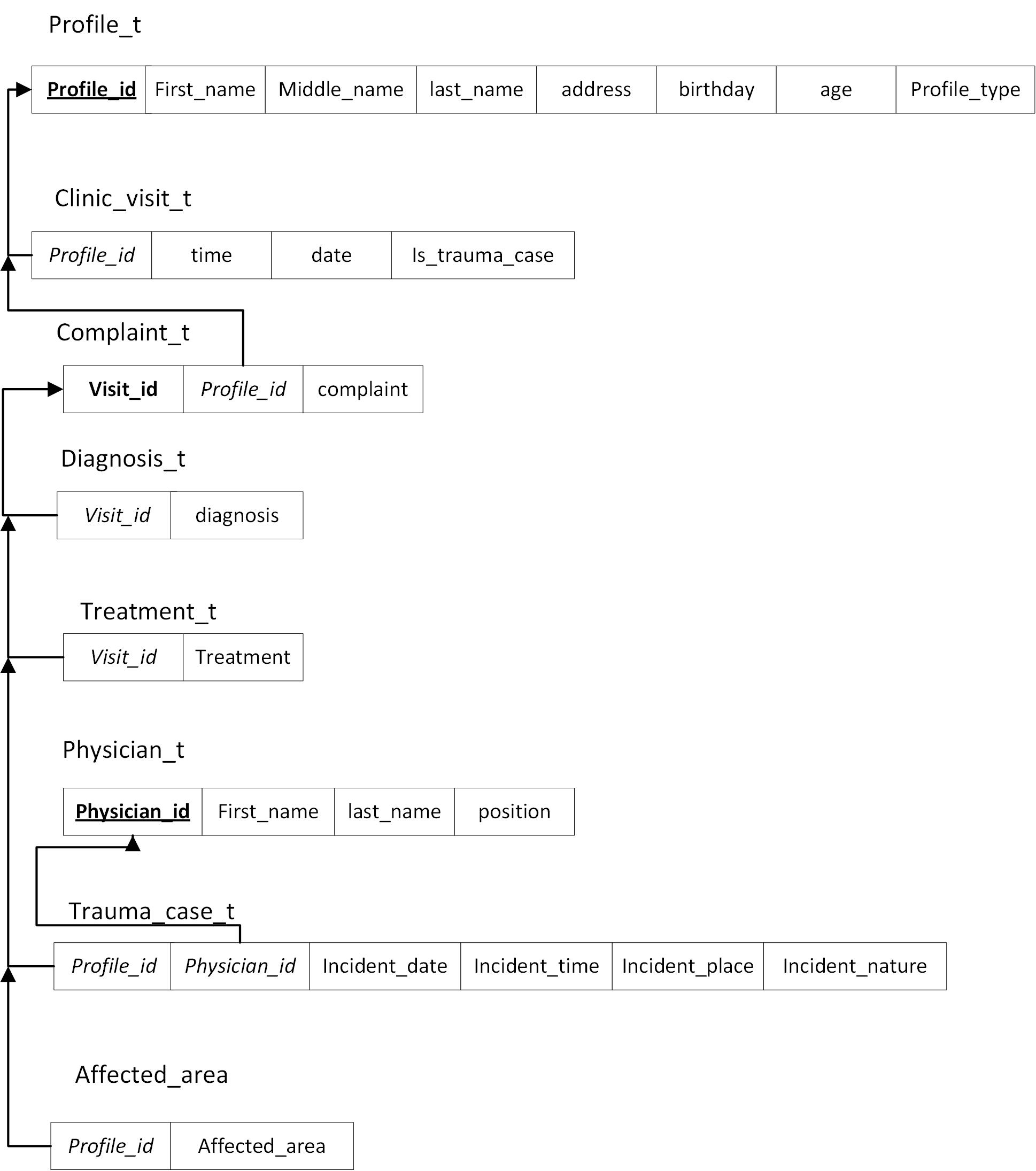
|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-11 | | |
| **Use Case Name:** | Log Out | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | To log out of the system | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff must be logged in * Staff must be in any screen with a log out button | | |
| **Post Condition:** | User is sent back to the Login screen | | |
| **Business Trigger:** | * User is done using the system and needs to log out | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. Select log out button | |  |  |
| **Business Rules:** | | | |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-12 | | |
| **Use Case Name:** | Delete profile | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | To delete a profile form the visitor database | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff must be logged in * Staff must be in the visitor database screen | | |
| **Post Condition:** | A record of fro the visitor database screen is deleted | | |
| **Business Trigger:** | * User is done using the system and needs to log out | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. Select a record from the visitor database 2. Select delete | |  | 1.1 search for a record from the visitor database |
| **Business Rules:** | | | |
| Patient must have a profile in the system  Patient must provide necessary information to staff | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case ID:** | UC-13 | | |
| **Use Case Name:** | Link trauma case | | |
| **Prepared By:** | Team Milan | | |
| **Last updated:** | March 10, 2013 | | |
| **Objective:** | To link a scan of the trauma case to the system | | |
| **Actor:** | Staff | | |
| **Preconditions:** | * Staff must be logged in * Staff must be in the trauma case screen | | |
| **Post Condition:** | A record of fro the visitor database screen is deleted | | |
| **Business Trigger:** | * User is done using the system and needs to log out | | |
| **Flow:** Identify the step by step actor-system interaction as the use case is being executed | | | |
| **Main Path** | | **Extension** | **Alternative Path** |
| 1. Select upload 2. Paste the link of the file to the screen 3. Select save | | 3.3 Select Cancel |  |
| **Business Rules:** | | | |
| Patient must have a profile in the system  Patient must provide necessary information to staff  Staff must have a link of the uploaded trauma case scan | | | |

Final Database Design







Final Database Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| **System Name:** | Ateneo Infirmary System | Date: | March 8, 2013 |
| **DBMS:** | MySql | Version: | 2.0 |
| **Analyzed By:** | Team Milan | | |

|  |  |
| --- | --- |
| **Table Name:** | Login\_t |
| **Table Description:** | Contains user information to access the system |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Username | Username of system user | VARCHAR (255) | Null | PK | No specific format | Alphanumeric characters | No |
| Password | Password for system user | VARCHAR(255) | Null | neither | No specific format | Alphanumeric characters | No |

|  |  |
| --- | --- |
| **Table Name:** | Profile\_t |
| **Table Description:** | Supertype of the employee,visitor and student type |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_ID | Record number of profile | Int | Auto-increment | PK | Single integer | 0-all positive integers | No |
| First\_Name | First name of patient | Varchar (12) | Null | neither | No specific format | Alphanumeric characters | No |
| Last\_Name | Last name of patient | Varchar(12) | null | neither | No specific format | Alphanumeric characters | No |
| Middle\_initial | Middle initial of patient | Varchar(1) | null | neither | Single character | Alphanumeric characters | No |
| Address | Address of patient | Varchar(40) | null | neither | No specific format | Alphanumeric characters | No |
| Birthday | Birthday of patient | Date | Null | Neither | Yyyy-mm-dd | Date | No |
| Age | Age of patient | Int(2) | Null | Neither |  | 00-99 | No |
| Profile\_type | Type of profile | Varchar(1) | null | neither |  | Alphanumeric characters | no |

|  |  |
| --- | --- |
| **Table Name:** | Employee |
| **Table Description:** | Subtype of the Profile supertype |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| ID\_Number | ID number of employee | Int(6) | null | Neither | Single integer | 0-all positive integers | No |
| Department | Department where employee works | Varchar(20) | null | neither | No specific format | Alphanumeric characters | no |

|  |  |
| --- | --- |
| **Table Name:** | Student\_t |
| **Table Description:** | Subtype of the Profile supertype |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type (Size) | Default Value | PK/ FK? | Format | Range/Allowable Values | Can be Null? |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| ID\_Number | Id number of student | Int(6) | Null | neither | Single integer | 000000-999999 | no |
| Student\_Section | Section of student | Varchar(2) | null | neither | All characters | Alphanumeric characters | No |
| year | Year level of student | Int(1) | null | neither | Single integer | 0-9 | no |

|  |  |
| --- | --- |
| **Table Name:** | Visitor\_t |
| **Table Description:** | Subtype of the Profile supertype |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |

|  |  |
| --- | --- |
| **Table Name:** | Contact\_no\_t |
| **Table Description:** | Contains the contact number of the patient |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| contact\_no | Contact number of patient | Varchar(11) | null | neither | Series of 11 numbers | 00000000000-99999999999 | Yes |

|  |  |
| --- | --- |
| **Table Name:** | Medical\_certificate\_t |
| **Table Description:** | Contains the link to the medical certificate of the patient |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| Medical\_certificate | URL of medical certificate image | Varchar(200) | null | neither | No specific format | Alphanumeric characters | yes |

|  |  |
| --- | --- |
| **Table Name:** | Vaccine\_t |
| **Table Description:** | Contains information of student vaccines |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Vaccine\_id | ID number of vaccine | Int(4) | null | PK | Single int | 0000-9999 | No |
| Vaccone\_name | Name of vaccine | Varchar(20) | null | neither | No specific format | Alphanumeric characters | no |

|  |  |
| --- | --- |
| **Table Name:** | Immunization\_record\_t |
| **Table Description:** | Contains information of student’s immunization record |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| Vaccine\_id | Id number of vaccine | Int(4) | Null | FK | Single int | 0-9 | Yes |
| No\_of\_shots | Number of shots given | Int(1) | Null | Neither | Singe int | 0-9 | no |
| No\_of\_boosters | Number of boosters given | Int(1) | null | neither | Single int | 0-9 | no |

|  |  |
| --- | --- |
| **Table Name:** | Medicine\_t |
| **Table Description:** | Contains information of medicine used |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Medicine\_id | Id number of medicine given | Int(4) | Null | PK | Single int | 0000-9999 | No |
| Medicine\_name | Name of medicine given | Varchar(20) | null | neither | No specific format | Alphanumeric characters | no |

|  |  |
| --- | --- |
| **Table Name:** | Maintenance\_Medication\_t |
| **Table Description:** | Contains information of maintenance medication |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| Id\_Number | Id number of patient | Int(6) | null | PK | Single integer | 0-all positive integers | No |
| Medicine\_id | Id number of medicine given | Int(4) | Null | PK | Single int | 0000-9999 | No |
| Illness | Patient’s illness | Varchar(50) | Null | neither | No specific format | Alphanumeric Characters | No |
| dosage | Dosage of medicine given to patient | Varchar(5) | null | neither | No specific format | Alphanumeric Characters | No |

|  |  |
| --- | --- |
| **Table Name:** | Clinic\_visit\_t |
| **Table Description:** | Contains information on patient’s clinic visits |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| Date | Date of visit | Date | Current date | neither | Yyyy-mm-dd | All dates | No |
| Timeout | Time of exit | time | Time of exit | neither | Hh:mm:ss | All time |  |
| is\_trauma\_case | Indicates whether case is a trauma case | Bit(1) | 0 | neither | Single bit | 0-1 | No |

|  |  |
| --- | --- |
| **Table Name:** | Complaint\_t |
| **Table Description:** | Contains information on patient’s complaints |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Visit\_id | Unique id for every visit | integer | Auto-increment | PK | Single integer | 0-all positive integers | no |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| complaint | Patient’s complain | VARCHAR(100) | null | neither | No specific format | Alphanumeric characters | no |

|  |  |
| --- | --- |
| **Table Name:** | Diagnosis\_t |
| **Table Description:** | Contains information on patient’s diagnosis |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Visit\_id | Unique id for every visit | integer | Auto-increment | FK | Single integer | 0-all positive integers | no |
| diagnosis | Diagnosis of patient | Varchar(200) | null | neither | No specific format | Alphanumeric characters | No |

|  |  |
| --- | --- |
| **Table Name:** | Treatment\_T |
| **Table Description:** | Contains information on patient’s treatment |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Visit\_id | Unique id for every visit | integer | Auto-increment | FK | Single integer | 0-all positive integers | no |
| Treatment | Patient’s treatment | VARCHAR(100) | null | Neither | No specific format | Alphanumeric characters | no |

|  |  |
| --- | --- |
| **Table Name:** | Physician\_t |
| **Table Description:** | Contains information on physician |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| physician\_id | Id of physician | Int(6) | null | PK | Single int | 000000-999999 | No |
| First\_name | First name of physician | Varchar(12) | Null | Neither | No specific format | Alphanumeric characters | No |
| Last\_name | Last name of physician | Varchar(12) | Null | Neither | No specific format | Alphanumeric characters | No |
| position | Position of staff | Varchar(12) | null | Neither | No specific format | Alphanumeric characters | no |

|  |  |
| --- | --- |
| **Table Name:** | Trauma\_case\_t |
| **Table Description:** | Contains information on physician |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| physician\_id | Id of physician | Int(6) | null | FK | No specific format | 000000-999999 | No |
| Incident\_time | Time of incident | Time | Null | Neither | Hh:mm:ss | All time | no |
| Incident\_date | Date of incident | Date | Null | Neither | Yyyy:mm:dd | All dates | No |
| Incident\_place | Place of incident | Varchar(12) | Null | Neither | No specific format | Alphanumeric characters | No |
| Incident\_nature | Nature of incident | Varchar(500) | null | Neither | No specific format | Alphanumeric characters | No |

|  |  |
| --- | --- |
| **Table Name:** | Affected\_Area\_t |
| **Table Description:** | Contains information on trauma case affected area |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| Affected\_Area | URL of uploaded image of affected area | Varchar (200) | null | neither | No specific characters | Alphanumeric characters | no |

|  |  |
| --- | --- |
| **Table Name:** | Physical\_examination\_t |
| **Table Description:** | Contains information on patient’s physical examinations |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | **Data Type (Size)** | **Default Value** | **PK/ FK?** | **Format** | **Range/Allowable Values** | **Can be Null?** |
| Profile\_id | Id number of profile | Integer | Auto-Increment | FK | Single integer | 0-all positive integers | No |
| Physician\_id | ID number of physicians | Int(6) | null | FK | Single int | 000000-999999 | No |
| Date\_time | Date and time of physical exam | DATETIME | Current date and time | Neither | yyyy-mm-dd hh:mm:ss | 0000-00-00 00:00:00 to 9999-12-31 12:59:59 | No |
| Height | Height of patient | Int(3) | null | Neither | Single int | 000-999 | No |
| Weight | Weight of patient | Int(3) | Null | Neither | Single int | 000-999 | No |
| BMI | Body mass index of patient | DOUBLE(4,2) | Null | Neither | 2 decimal places, 2 whole number places | 00.00-99.99 | No |
| Blood\_Pressure | Blood pressure of patient | Varchar(7) | Null | Neither | No specific format | Alphanumeric characters | No |
| Pulse\_rate | Pulse of patient | Int(3) | Null | Neither | Single int | 000-999 | No |
| Respiratory\_rate | Respiratory rate of patient | Int(3) | Null | Neither | Single int | 000-999 | No |
| temperature | Temperature of patient | Double(3,1) | Null | Neither | 1 decimal place, 2 whole number places | 00.0-99.9 | No |
| General\_appearance | General appearance of patient | Varchar(50) | Null | Neither | No specific characters | Alphanumeric characters | No |
| Head | Appearance of patient’s head | VARCHAR(100) | Null | Neither | No specific characters | Alphanumeric characters | No |
| eyes | Appearance of patient’s eyes | Varchar(100) | Null | Neither | No specific characters | Alphanumeric characters | No |
| Va\_right | Patient’s visual acuity on the right eye | Int(3) | Null | Neither | Single int | 000-999 | No |
| Va\_left | Patient’s visual acuity on the left eye | Int(3) | Null | Neither | Single int | 000-999 | No |
| Va\_right\_glasses | Patient’s visual acuity on the right eye (glasses) | Int(3) | Null | Neither | Single int | 000-999 | Yes |
| Va\_left\_glasses | Patient’s visual acuity on the left eye (glasses) | Int(3) | null | Neither | Single int | 000-999 | Yes |
| Ear\_canal | Appearance of patient’s ear canal | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Tympanic\_membrane | Appearance of patient’s tympanic membrane | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| hearing | Comment on patient’s  hearing | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| nose | Appearance of patient’s nose | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Mouth | Appearance of patient’s mouth | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Pharynx | Appearance of patient’s pharynx | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Neck | Appearance of patient’s neck | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Chest | Appearance of patient’s chest | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Heart | Appearance of patient’s heart | Varchar(100) | null | Neither | No specific format | Alphanumeric characters | No |
| Lungs | Appearance of patient’s lungs | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Abdomen | Appearance of patient’s abdomen | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Stomach | Appearance of patient’s stomach | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Liver | Appearance of patient’s liver | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Spleen | Appearance of patient’s spleen | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Appendix | Appearance of patient’s appendix | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Kidneys | Appearance of patient’s kidneys | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Inguinal\_area | Appearance of patient’s inguinal area | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Genitalia | Comment on patient’s genitalia condition | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Rectum | Comment on patient’s rectum condition | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Extremities | Appearance of patient’s extremities | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| back | Appearance of patient’s back | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Skin | Appearance of patient’s skin | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | No |
| Lymph\_nodes | Appearance of patient’s lymph nodes | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | no |
| Is\_fit\_for\_physical\_activity | Indicates whether patient is fit for physical activities | Bit(1) | 1 | Neither | Singe bit | 0-1 | No |
| Unlimited | Reason for unlimited physical fitness | Varchar(200) | Null | Neither | No specific format | Alphanumeric characters | Yes |
| Limited | Reasons for limited physical fitness | Varchar(200) | Null | Neither | No specific format | Alphanumeric characters | Yes |
| Reasons | Reason for patient’s limitation | Varchar(100) | Null | Neither | No specific format | Alphanumeric characters | Yes |
| Need | Need for further referral | BIT(1) | null | Neither | Single bit | 0-1 | Yes |
| Refer\_to | Physician which patient is being referred to | Varchar(30) | Null | Neither | No specific format | Alphanumeric characters | Yes |
| Refer\_for | Referring physician | Varchar(30) | Null | Neither | No specific format | Alphanumeric characters | Yes |
| General\_appearance\_normal | Indicates whether patient’s general appearance is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Head\_normal | Indicates whether patient’s head is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Neck\_normal | Indicates whether patient’s neck is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Eyes\_normal | Indicates whether patient’s eyes are normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Ears\_normal | Indicates whether patient’s ears are normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Tympanic\_membrane\_normal | Indicates whether patient’s tympanic membrane is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Hearing\_normal | Indicates whether patient’s hearing is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Nose\_normal | Indicates whether patient’s nose is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Mouth\_normal | Indicates whether patient’s mouth is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Pharynx\_normal | Indicates whether patient’s pharynx is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Chest\_normal | Indicates whether patient’s chest is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Heart\_normal | Indicates whether patient’s heart is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Lungs\_normal | Indicates whether patient’s lungs are normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Abdomen\_normal | Indicates whether patient’s abdomen is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Kidneys\_normal | Indicates whether patient’s kidneys are normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Stomach\_normal | Indicates whether patient’s stomach is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Liver\_normal | Indicates whether patient’s liver is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Spleen\_normal | Indicates whether patient’s spleen is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Appendix\_normal | Indicates whether patient’s appendix is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Inguinal\_area\_normal | Indicates whether patient’s inguinal area is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Genitalia\_normal | Indicates whether patient’s genitalia is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Rectum\_normal | Indicates whether patient’s rectum is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Extremities\_normal | Indicates whether patient’s extremities are normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Back\_normal | Indicates whether patient’s back is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Skin\_normal | Indicates whether patient’s skin is normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |
| Lymph\_nodes\_normal | Indicates whether patient’s lymph nodes are normal | Bit(1) | 0 | Neither | Single bit | 0-1 | Yes |

Implementation

Table and Integrity constraints

CREATE TABLE login\_t

(username VARCHAR(255) NOT NULL,

password VARCHAR(255) NOT NULL,

CONSTRAINT login\_pk PRIMARY KEY (username));

CREATE TABLE profile\_t

(profile\_id INTEGER AUTO\_INCREMENT,

first\_name VARCHAR(12) NOT NULL,

last\_name VARCHAR(12) NOT NULL,

middle\_initial VARCHAR(2) NOT NULL,

address VARCHAR(40) NOT NULL,

birthday DATE NOT NULL,

age INT(2) NOT NULL,

profile\_type VARCHAR(1) NOT NULL,

-- patient\_type CHAR(1) NOT NULL,

CONSTRAINT profile\_pk PRIMARY KEY (profile\_id));

CREATE TABLE contact\_no\_t

(profile\_id INTEGER AUTO\_INCREMENT,

contact\_no VARCHAR(11),

CONSTRAINT contact\_no\_fk FOREIGN KEY (profile\_id) REFERENCES profile\_t (profile\_id));

CREATE TABLE student\_t

(profile\_id INTEGER AUTO\_INCREMENT,

id\_number INTEGER(6) NOT NULL,

year INT(1) NOT NULL,

student\_section VARCHAR(2) NOT NULL,

CONSTRAINT student\_fk FOREIGN KEY (profile\_id) REFERENCES profile\_t (profile\_id));

CREATE TABLE medical\_certificate\_t

(profile\_id INTEGER AUTO\_INCREMENT,

medical\_certificate VARCHAR(20), -- url

CONSTRAINT medical\_certificate\_fk FOREIGN KEY (profile\_id) REFERENCES student\_t (profile\_id));

CREATE TABLE employee\_t

(profile\_id INTEGER AUTO\_INCREMENT,

id\_number INTEGER(6) NOT NULL,

department VARCHAR(20) NOT NULL,

CONSTRAINT employee\_fk FOREIGN KEY (profile\_id) REFERENCES profile\_t (profile\_id));

CREATE TABLE visitor\_t

(profile\_id INTEGER AUTO\_INCREMENT,

CONSTRAINT visitor\_fk FOREIGN KEY (profile\_id) REFERENCES profile\_t (profile\_id));

CREATE TABLE vaccine\_t

(vaccine\_id INTEGER(4) NOT NULL,

vaccine\_name VARCHAR(20) NOT NULL,

CONSTRAINT vaccine\_pk PRIMARY KEY (vaccine\_id));

CREATE TABLE immunization\_record\_t

(profile\_id INTEGER AUTO\_INCREMENT,

vaccine\_id INTEGER(4) NOT NULL,

No\_of\_shots INT(1) NOT NULL,

No\_of\_boosters INT(1) NOT NULL,

-- CONSTRAINT immunization\_record\_pk PRIMARY KEY (id\_number, vaccine\_id),

CONSTRAINT immunization\_record\_fk1 FOREIGN KEY (profile\_id) REFERENCES student\_t (profile\_id),

CONSTRAINT immunization\_record\_fk2 FOREIGN KEY (vaccine\_id) REFERENCES vaccine\_t (vaccine\_id));

CREATE TABLE medicine\_t

(medicine\_id INTEGER(4) NOT NULL,

medicine\_name VARCHAR(20) NOT NULL,

CONSTRAINT medicine\_pk PRIMARY KEY (medicine\_id));

CREATE TABLE maintenance\_medication\_t

(profile\_id INTEGER AUTO\_INCREMENT,

medicine\_id INTEGER(4) NOT NULL,

illness VARCHAR(50) NOT NULL,

dosage VARCHAR(5) NOT NULL,

-- CONSTRAINT maintenance\_medication\_pk PRIMARY KEY (id\_number, medicine\_id),

CONSTRAINT maintenance\_medication\_fk1 FOREIGN KEY (profile\_id) REFERENCES student\_t (profile\_id),

CONSTRAINT maintenance\_medication\_fk2 FOREIGN KEY (medicine\_id) REFERENCES medicine\_t (medicine\_id));

CREATE TABLE clinic\_visit\_t

(profile\_id INTEGER,

date datetime NOT NULL,

timeOut datetime,

is\_trauma\_case BIT(1) DEFAULT 0,

CONSTRAINT clinic\_visit\_fk FOREIGN KEY (profile\_id) REFERENCES profile\_t (profile\_id));

CREATE TABLE complaint\_t

(profile\_id INTEGER,

visit\_id INTEGER AUTO\_INCREMENT,

complaint VARCHAR(20) NOT NULL,

CONSTRAINT complaint\_pk1 PRIMARY KEY (visit\_id),

CONSTRAINT complaint\_fk FOREIGN KEY (profile\_id) REFERENCES clinic\_visit\_t (profile\_id));

CREATE TABLE diagnosis\_t

(visit\_id INTEGER,

diagnosis VARCHAR(200) NOT NULL,

CONSTRAINT diagnosis\_fk FOREIGN KEY (visit\_id) REFERENCES complaint\_t (visit\_id));

CREATE TABLE treatment\_t

(visit\_id INTEGER,

treatment VARCHAR(100) NOT NULL,

CONSTRAINT treatment\_fk FOREIGN KEY (visit\_id) REFERENCES complaint\_t (visit\_id));

CREATE TABLE physician\_t

(physician\_id INTEGER(6) NOT NULL,

first\_name VARCHAR(12) NOT NULL,

last\_name VARCHAR(12) NOT NULL,

position VARCHAR(12) NOT NULL,

CONSTRAINT physician\_pk PRIMARY KEY (physician\_id));

CREATE TABLE trauma\_case\_t

(profile\_id INTEGER AUTO\_INCREMENT,

physician\_id INTEGER(6) NOT NULL,

incident\_dateTime datetime NOT NULL,

incident\_place VARCHAR(12) NOT NULL,

incident\_nature VARCHAR(500) NOT NULL,

CONSTRAINT trauma\_case\_fk1 FOREIGN KEY (profile\_id) REFERENCES clinic\_visit\_t (profile\_id),

CONSTRAINT trauma\_case\_fk2 FOREIGN KEY (physician\_id) REFERENCES physician\_t (physician\_id));

CREATE TABLE affected\_area\_t

(profile\_id INTEGER AUTO\_INCREMENT,

affected\_area VARCHAR(50), -- url

CONSTRAINT affected\_area\_fk FOREIGN KEY (profile\_id) REFERENCES clinic\_visit\_t (profile\_id));

CREATE TABLE physical\_examination\_t

(profile\_id INTEGER AUTO\_INCREMENT,

physician\_id INTEGER(6) NOT NULL,

date DATE NOT NULL,

time TIME NOT NULL,

height INT(3) NOT NULL,

weight INT(3) NOT NULL,

BMI DOUBLE(4,2) NOT NULL,

blood\_pressure VARCHAR(7) NOT NULL,

pulse\_rate INT(3) NOT NULL,

respiratory\_rate INT(3) NOT NULL,

temperature DOUBLE(3,1) NOT NULL,

general\_appearance VARCHAR(50) NOT NULL,

head VARCHAR(100) NOT NULL,

eyes VARCHAR(100) NOT NULL,

ears VARCHAR(100) NOT NULL,

nose VARCHAR(100) NOT NULL,

mouth VARCHAR(100) NOT NULL,

pharynx VARCHAR(100) NOT NULL,

neck VARCHAR(100) NOT NULL,

chest VARCHAR(100) NOT NULL,

heart VARCHAR(100) NOT NULL,

lungs VARCHAR(100) NOT NULL,

abdomen VARCHAR(100) NOT NULL,

stomach VARCHAR(100) NOT NULL,

liver VARCHAR(100) NOT NULL,

spleen VARCHAR(100) NOT NULL,

appendix VARCHAR(100) NOT NULL,

kidneys VARCHAR(100) NOT NULL,

inguinal\_area VARCHAR(100) NOT NULL,

genitalia VARCHAR(100) NOT NULL,

rectum VARCHAR(100) NOT NULL,

extremities VARCHAR(100) NOT NULL,

back VARCHAR(100) NOT NULL,

skin VARCHAR(100) NOT NULL,

lymph\_nodes VARCHAR(100) NOT NULL,

is\_fit\_for\_physical\_activity BIT(1) NOT NULL DEFAULT 1,

unlimited VARCHAR(200),

limited VARCHAR(200),

reasons VARCHAR(100),

need BIT(1) DEFAULT 0,

refer\_to VARCHAR(30),

refer\_for VARCHAR(30),

-- CONSTRAINT physical\_examination\_pk PRIMARY KEY (id\_number),

CONSTRAINT physical\_examination\_fk1 FOREIGN KEY (profile\_id) REFERENCES profile\_t (profile\_id),

CONSTRAINT physical\_examination\_fk2 FOREIGN KEY (physician\_id) REFERENCES physician\_t (physician\_id));

Sample SQL Statements used

SELECT physician\_id, first\_name, last\_name, position FROM physician\_t;

SELECT profile\_id, vaccine\_id, no\_of\_shots, no\_of\_boosters FROM immunization\_record\_t;

"SELECT v.vaccine\_name, i.vaccine\_id, i.no\_of\_shots, i.no\_of\_boosters FROM immunization\_record\_t i JOIN vaccine\_t v ON i.vaccine\_id = v.vaccine\_id WHERE i.profile\_id = " + profile\_id + ";"

Select MAX(visit\_id) from complaint\_t;

SELECT s.profile\_id, s.id\_number, p.last\_name, p.first\_name, p.middle\_initial, s.year, s.student\_section, p.address, p.birthday, p.age, c.contact\_no FROM student\_t s JOIN profile\_t p ON s.profile\_id = p.profile\_id JOIN contact\_no\_t c ON c.profile\_id = p.profile\_id;

INSERT INTO treatment\_t VALUES (" + m.Visit\_id + ", " + m.Medicationn + ");

INSERT INTO diagnosis\_t VALUES ("+d.Visit\_id + ","+ d.Diagnose + ");

INSERT INTO complaint\_t (profile\_id, complaint) VALUES ('{0}',{1});", c.Profile\_id, c.Complain

INSERT INTO clinic\_visit\_t VALUES (" + cv.Profile\_id + ", '" + cv.DateTimeIn + "', " + cv.DateTimeOut + ", " + cv.TCase + ");

INSERT INTO trauma\_case\_t VALUES (" + t.Profile\_id + ", " + t.Physician\_id + ", " + t.Date + ", " + t.Place + ", " + t.Nature + ");

UPDATE immunization\_record\_t SET no\_of\_shots = 0, no\_of\_boosters = 0 WHERE profile\_id = " + profile\_id + " AND vaccine\_id = " + vaccine\_id + ";"

"UPDATE immunization\_record\_t SET no\_of\_shots = " + i.No\_of\_shots + ", no\_of\_boosters = " + i.No\_of\_boosters + " WHERE profile\_id = " + i.Profile\_id + " AND vaccine\_id = " + i.Vaccine\_id + ";"

Testing

The testing will primary focus on the making sure that every functionality of the system is working properly particularly the key features such as the import and the profile system. The general plan is to stress test the system, uploading different CSV files and checking to see if they were stored correctly on the database as well as manipulating the data within the database via the program itself.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | Login |
| Test Type: | Stress Testing, Security Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check whether login works properly. This means when a user logins, he is redirected to the correct screen * To be able to secure the system through checking the accounts that are able to use the system * Whenever there is an invalid login, the system notifies the current user about it | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test for valid username and password  INPUT:  Username = “physician”  Password = “encode”  Test if the accounts are available in the database or in the system. | Login | Milan Chua | Franco Zarate | The Main Menu containing the 3 databases should be shown. |  | H | Pass |
| 2 | Test for valid username and invalid password  INPUT:  Username = “encode”  Password = “physician”  Test an input error. | Login | Milan Chua | Franco Zarate | A pop-up should appear whenever a user inputs wrong account details.  Invalid login. Account not found. |  | H | pass |
| 3 | Test for valid username input using special characters  INPUT:  Username = !@#$%^  Password = +\_)(\*&  This case assumes that user !@#$%^ and its corresponding password is existing in the database | Login | Milan Chua | Franco Zarate | The Main Menu containing the 3 databases should be shown. |  | H | Pass |
| 4 | Test for incorrect username and password  INPUT:  Username = “12345”  Password = “@#$%^&”  This tests that the username is currently not in the system or database list of accounts. | Login | Milan Chua | Franco Zarate | A pop-up should appear whenever a user inputs wrong account details.  Invalid login. Account not found. |  | H | pass |
| 5 | Test for blank input  INPUT:  Username = “”  Password = “” | Login | Milan Chua | Franco Zarate | A pop-up should appear whenever a user inputs wrong account details.  Invalid login. Account not found. |  | H | Pass |
| 6 | Test for password text display  INPUT:  Username = “abc”  Password = “def”  The password should not be shown in original characters. | Login | Milan Chua | Franco Zarate | Password Textbox should display \*\*\*\*\*\*\*\*\* |  | H | pass |
| 7 | Test for minimum password length  INPUT: ?? | Login | Milan Chua | Franco Zarate | A pop-up should appear whenever a user inputs wrong account details.  Invalid login. Account not found. |  | L | Pass |
| 8 | Test for maximum password length  INPUT: ?? | Login | Milan Chua | Franco Zarate | A pop-up should appear whenever a user inputs wrong account details.  Invalid login. Account not found. |  | L | pass |

Franco Zarate

Franco Zarate

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | Main Menu |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To check whether the database buttons are clicked or chosen, they would proceed to the chosen database | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test the Student Database button | Main Menu | Milan Chua | Franco Zarate | The Student Database User Interface or screen should be shown. | When the student database button was clicked, the system loaded the student database screen. | H | Pass |
| 2 | Test the Employee Database button | Main Menu | Milan Chua | Franco Zarate | The Employee Database User Interface or screen should be shown. | When the employee database button was clicked, the system loaded the employee database screen. | H | Pass |
| 3 | Test the Visitor Database button | Main Menu | Milan Chua | Franco Zarate | The Visitor Database User Interface or screen should be shown. | When the visitor database button was clicked, the system loaded the visitor database screen. | H | Pass |
| 4 | Test the Logout button | Main Menu | Milan Chua | Franco Zarate | When the user clicks the logout button, the system should logout and return to the login screen. | The system returned to the login screen after closing the main menu screen. | H | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | Employee Database |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check whether the buttons of the screen redirects to the correct screen * To be able to check whether the search function works and other functions too * To be able to check whether update, load and view works correctly | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test to see if search button shows the necessary information that was inputted.  (Random ID Number)  Input: 100890 | Employee Database | Milan Chua | Franco Zarate | The screen should show the result of an employee having the ID number of 100890. | Untested | H |  |
| 2 | Test to see if the search button shows error messages whenever the said search query was not found in the database.  INPUT:  Username = “Milan Chua” | Employee Database | Milan Chua | Franco Zarate | A pop-up should appear or the screen should notify the user that there was no employee of that name or details in the database.  “No data found.” | Untested | H |  |
| 3 | Test and load the database into the screen. | Employee Database | Milan Chua | Franco Zarate | The screen should show the database of employees. | Untested | H |  |
| 4 | Test the update button. Whenever the user clicks on a name or a student in the data grid view, and clicks update, the user should be able to add or update existing information. | Employee Database | Milan Chua | Franco Zarate | The user should be redirected to the update screen. | Untested | H |  |
| 5 | Test the view record button. The user clicks an employee name on the data grid view, and then clicks the view record button. | Employee Database | Milan Chua | Franco Zarate | The user should be redirected to the employee profile. | Once the view record button was clicked, the system goes to the employee profile screen. | H | Pass |
| 6 | Test the back to main menu button. | Employee Database | Milan Chua | Franco Zarate | The user should be brought back to the screen where the 3 databases are shown. | The system closed the employee database screen and returned to the main menu screen. | H | Pass |
| 7 | Test the Logout button | Employee Database | Milan Chua | Franco Zarate | When the user clicks the logout button, the system should logout and return to the login screen. | The system returned to the login screen after closing the employee database screen. | L | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | Student Database |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check whether the buttons of the screen redirects to the correct screen * To be able to check whether the search function works and other functions too * To be able to check whether update, load and view works correctly | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test to see if search button shows the necessary information that was inputted.  (Random ID Number)  Input: 100890 | Student Database | Milan Chua | Franco Zarate | The screen should show the result of a student having the ID number of 100890. | untested | H |  |
| 2 | Test to see if the search button shows error messages whenever the said search query was not found in the database.  INPUT:  Username = “Milan Chua” | Student Database | Milan Chua | Franco Zarate | A pop-up should appear or the screen should notify the user that there was no student of that name or details in the database.  “No data found.” | untested | H |  |
| 3 | Test and load the database into the screen. | Student Database | Milan Chua | Franco Zarate | The screen should show the database of students. | Datagrid view showed the list of students currently in the in the database | H | Pass |
| 4 | Test the update button. Whenever the user clicks on a name or a student in the data grid view, and clicks update, the user should be able to add or update existing information. | Student Database | Milan Chua | Franco Zarate | The user should be redirected to another screen. | untested | H |  |
| 5 | Test the view record button. The user clicks a student name on the data grid view, and then clicks the view record button. | Student Database | Milan Chua | Franco Zarate | The user should be redirected to the student profile. | Once the view record button was clicked, the system goes to the student profile screen. | H | Pass |
| 6 | Test the back to main menu button. | Student Database | Milan Chua | Franco Zarate | The user should be brought back to the screen where the 3 databases are shown. | The system closed the student database screen and returned to the main menu screen. | H | Pass |
| 7 | Test the Logout button | Student Database | Milan Chua | Franco Zarate | When the user clicks the logout button, the system should logout and return to the login screen. | The system returned to the login screen after closing the student database screen. | L | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | Visitor Database |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check whether the buttons of the screen redirects to the correct screen * To be able to check whether the search function works and other functions too * To be able to check whether update, load and view works correctly | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test to see if search button shows the necessary information that was inputted.  (Random Name)  Input: Bogart | Visitor Database | Milan Chua | Franco Zarate | The screen should show the result of a student having the name of Bogart. | untested | H |  |
| 2 | Test to see if the search button shows error messages whenever the said search query was not found in the database.  INPUT:  Username = “Milan Chua” | Visitor Database | Milan Chua | Franco Zarate | A pop-up should appear or the screen should notify the user that there was no visitor of that name or details in the database.  “No data found.” | untested | H |  |
| 3 | Test and load the database into the screen. | Visitor Database | Milan Chua | Franco Zarate | The screen should show the database of visitors. | untested | H |  |
| 4 | Test the update button. Whenever the user clicks on a name or a student in the data grid view, and clicks update, the user should be able to add or update existing information. | Visitor Database | Milan Chua | Franco Zarate | The user should be redirected to the update screen. | untested | H |  |
| 5 | Test the view record button. The user clicks a student name on the data grid view, and then clicks the view record button. | Visitor Database | Milan Chua | Franco Zarate | The user should be redirected to the visitor profile. | untested | H |  |
| 6 | Test the back to main menu button. | Visitor Database | Milan Chua | Franco Zarate | The user should be brought back to the screen where the 3 databases are shown. | The system closed the visitor database screen and returned to the main menu screen. | H | Pass |
| 7 | Test the Logout button | Visitor Database | Milan Chua | Franco Zarate | When the user clicks the logout button, the system should logout and return to the login screen. | The visitor database screen closed and the system returned to the login screen. | L | Pass |
| 8 | Testing the delete button. The user clicks on a visitor name then clicks the delete button. | Visitor Database | Milan Chua | Franco Zarate | The name that was chosen should be deleted in the data grid view and in the database. | untested | L |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | (Student) Patient Profile |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check the different tabs that the screen has * To check whether the buttons of the screen work * To be able to check whether the button redirects the user to the correct screens afterwards | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test clinic visit tab. | Patient Profile | Milan Chua | Franco Zarate | The data grid view of the clinic visits should be shown. The data are based on the patient’s profile. | untested | H |  |
| 2 | Test trauma case tab. | Patient Profile | Milan Chua | Franco Zarate | The data grid view of the trauma cases should be shown. The data are based on the patient’s profile. | untested | M |  |
| 3 | Test medical certificate tab. | Patient Profile | Milan Chua | Franco Zarate | The scanned medical certificate should be shown. | untested | H |  |
| 4 | Test immunization records tab. | Patient Profile | Milan Chua | Franco Zarate | The data grid view of the immunization records should be shown. The data are based on patient’s profile. | Once the medical certificate tab was clicked, the data grid view showed all immunization records about the chosen patient profile. | H | Pass |
| 5 | Test physical examination tab. | Patient Profile | Milan Chua | Franco Zarate | The data grid view of the physical examination should be shown. The data are based on patient’s profile. | No data in data grid view. | H | Fail |
| 6 | Test maintenance medication tab. | Patient Profile | Milan Chua | Franco Zarate | The data grid view of the maintenance medication should be shown. The data are based on patient’s profile. | Data shown on datagrid | H | Pass |
| 7 | Testing the add clinic visit button. | Patient Profile | Milan Chua | Franco Zarate | The user should be redirected to the clinic visit form. | User redirected to proper screen | H | Pass |
| 8 | Testing the delete button. The user clicks on data on the data grid view. | Patient Profile | Milan Chua | Franco Zarate | The clicked data should be deleted in the data grid view and in the database for the patient’s profile. | In the immunization record tab, the selected record’s no. of shots and no. of boosters columns refreshed to 0 and the database updated the selected record accordingly. | L | Pass (for immunization record tab) |
| 9 | Testing the time out button. The user selects an entry to be timed out. | Patient Profile | Milan Chua | Franco Zarate | Time out field should be updated. | untested | L |  |
| 10 | Testing the back button. | Patient Profile | Milan Chua | Franco Zarate | When clicked, the screen should go back to the student database interface. | The patient profile screen closed and the student database screen was shown. | L | Pass |
| 11 | Test the Logout button | Patient Profile | Milan Chua | Franco Zarate | When the user clicks the logout button, the system should logout and return to the login screen. | The patient profile screen closed and the system returned to the login screen. | L | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | (Employee) Patient Profile |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check the different tabs that the screen has * To check whether the buttons of the screen work * To be able to check whether the button redirects the user to the correct screens afterwards | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test clinic visit tab. | Patient Profile | Milan Chua | Franco Zarate | The data grid view of the clinic visits should be shown. The data are based on the patient’s profile. | untested | H |  |
| 2 | Testing the add clinic visit button. | Patient Profile | Milan Chua | Franco Zarate | The user should be redirected to the clinic visit form. | User redirected to correct page | H | Pass |
| 3 | Testing the delete button. The user clicks on data on the data grid view. | Patient Profile | Milan Chua | Franco Zarate | The clicked data should be deleted in the data grid view and in the database for the patient’s profile. | untested | L |  |
| 4 | Testing the edit record button. The user clicks on data on the data grid view. | Patient Profile | Milan Chua | Franco Zarate | A screen should appear then the clicked data could be edited. | untested | L |  |
| 5 | Testing the back button. | Patient Profile | Milan Chua | Franco Zarate | When clicked, the screen should go back to the student database interface. | untested | L |  |
| 6 | Test the Logout button | Patient Profile | Milan Chua | Franco Zarate | When the user clicks the logout button, the system should logout and return to the login screen. | untested | L |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | Physical Examination |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check whether the inputs are valid or invalid * To check whether the buttons of the screen work * To be able to check whether the button redirects the user to the correct screens afterwards | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test all the fields or text boxes that require integer input. (Height, Weight, BMI, BP, Pulse rate, Respiratory rate, Temperature, etc.)  Input integers. | Physical Examination | Milan Chua | Franco Zarate | The inputted data should be stored in the database. | Inputted data to fields that require integers were stored in the database. | H | Pass |
| 2 | Test all the fields or text boxes that require integer input. (Height, Weight, BMI, BP, Pulse rate, Respiratory rate, Temperature, etc.)  Input characters. | Physical Examination | Milan Chua | Franco Zarate | A pop-up or screen should show saying that the inputs are invalid.  “Invalid input. Please try again.” | Error |  | Fail |
| 3 | Test all the fields or text boxes that require character input. (General appearance, head, eyes, etc.)  Input characters. | Physical Examination | Milan Chua | Franco Zarate | The inputted data should be stored in the database. | Inputted data to fields that require character input were stored in the database. | H | Pass |
| 4 | Test all the fields or text boxes that require character input. (General appearance, head, eyes, etc.)  Input integers. | Physical Examination | Milan Chua | Franco Zarate | A pop-up or screen should show saying that the inputs are invalid.  “Invalid input. Please try again.” | Error | H | Fail |
| 5 | Test all normal checkboxes. | Physical Examination | Milan Chua | Franco Zarate | The check states normal. This will also appear in the database. | Checked normal checkboxes were stored correctly in the database. A value of 1is shown in the database if the checkbox is checked and a value of 0 if the checkbox is not checked. | H | Pass |
| 6 | Test the save button. | Physical Examination | Milan Chua | Franco Zarate | All valid inputted data will be saved into the database and will be available for viewing in the patient’s profile. | All inputted data in each field of the physical examination screen is stored into the databases when the save button was clicked. | H | Pass |
| 7 | Testing the cancel button. | Physical Examination | Milan Chua | Franco Zarate | The cancel button will redirect the user back to the previous screen or the patient profile. | Physical examination screen closed and the previous screen, which is the patient profile screen, is shown. | L | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | Clinic Visit |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check whether the inputs are valid or invalid * To check whether the buttons of the screen work * To be able to check whether the button redirects the user to the correct screens afterwards | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test all the fields or text boxes that require user input. (Complaint, Diagnosis and Medication/Treatment)  Any character is acceptable. | Clinic Visit | Milan Chua | Franco Zarate | The inputted data should be stored in the database once saved. | Data stored correctly | H | Pass |
| 2 | Testing the date drop down list. | Clinic Visit | Milan Chua | Franco Zarate | The user should be able to click on the date on the day the test was held. Saving this, the date will also appear and be saved in the database. | Date properly saved |  | pass |
| 3 | Test the trauma case checkbox. | Clinic Visit | Milan Chua | Franco Zarate | Whether or not it is a trauma case, the visit will still be recorded. (Though the user will need to go to the trauma case screen to report the case) | Record recorded | H | Pass |
| 4 | Test time in and time out. Integers and characters are allowable as long as they are in the limit of 0-59, and AM or PM. | Clinic Visit | Milan Chua | Franco Zarate | The time will be recorded once saved. | untested | H |  |
| 5 | Test the clear button. Inputs will be typed. | Clinic Visit | Milan Chua | Franco Zarate | Once clicked, everything on the screen will be cleared. | All fields cleared | H | Pass |
| 6 | Test the save button. | Clinic Visit | Milan Chua | Franco Zarate | All valid inputted data will be saved into the database and will be available for viewing in the patient’s profile. | Data saved | H | Pass |
| 7 | Testing the upload button. Uploading a scanned trauma case picture. | Clinic Visit | Milan Chua | Franco Zarate | The user should be able to see the scanned photo on the free space in the screen. | untested | H |  |
| 8 | Testing the cancel button. | Clinic Visit | Milan Chua | Franco Zarate | The cancel button will redirect the user back to the previous screen or the patient profile. | User was redirected to proper page | L | Pass |
| 9 | Testing the back button. | Patient Profile | Milan Chua | Franco Zarate | When clicked, the screen should go back to the student database interface. | User was redirected to proper page | L | Pass |
| 10 | Test the Logout button | Patient Profile | Milan Chua | Franco Zarate | When the user clicks the logout button, the system should logout and return to the login screen. | User was redirected to proper page | L | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Team Milan | | | | | | |
| Test Specification No. | | 1 | System: | AHS Infirmary System | Module: | Trauma Case |
| Test Type: | Stress Testing | | | | | |
| Date Created: | March 16, 2013 | | | | | |
| Date Updated: | March 16, 2013 | | | | | |
| Created By: | Milan Chua | | | | | |
| Test Objectives: | * To be able to check whether the inputs are valid or invalid * To check whether the buttons of the screen work * To be able to check whether the button redirects the user to the correct screens afterwards | | | | | |

| Test Case # | Test Case Description | Screen/ Program Name | Prepared By | Admin. By | Expected Results | Actual Results | Criticality (H/M/L) | Pass/Fail |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Test all the fields or text boxes that require user input. (Place and Nature of Accident)  Any character is acceptable. | Clinic Visit | Milan Chua | Franco Zarate | The inputted data should be stored in the database once saved. | Data saved | H | Pass |
| 2 | Testing the date drop down list. | Clinic Visit | Milan Chua | Franco Zarate | The user should be able to click on the date on the day the test was held. Saving this, the date will also appear and be saved in the database. | Date properly saved | L | pass |
| 3 | Test time in and time out. Integers and characters are allowable as long as they are in the limit of 0-59, and AM or PM. | Clinic Visit | Milan Chua | Franco Zarate | The time will be recorded once saved. | Time properly saved | H | pass |
| 4 | Test the clear button. Inputs will be typed. | Clinic Visit | Milan Chua | Franco Zarate | Once clicked, everything on the screen will be cleared. | Input cleared | H | Pass |
| 5 | Test the save button. | Clinic Visit | Milan Chua | Franco Zarate | All valid inputted data will be saved into the database and will be available for viewing in the patient’s profile. | Input saved | H | Pass |
| 6 | Testing the upload button. Uploading a scanned trauma case picture. | Clinic Visit | Milan Chua | Franco Zarate | The user should be able to see the scanned photo on the free space in the screen. | untested | H |  |
| 7 | Testing the cancel button. | Clinic Visit | Milan Chua | Franco Zarate | The cancel button will redirect the user back to the previous screen or the patient profile. | User was redirected to proper page | L | Pass |
| 8 | Testing the back button. | Patient Profile | Milan Chua | Franco Zarate | When clicked, the screen should go back to the student database interface. | User was redirected to proper page | L | Pass |
| 9 | Test the Logout button | Patient Profile | Milan Chua | Franco Zarate | When the user clicks the logout button, the system should logout and return to the login screen. | untested | L |  |