

Jason Jensen and Ibra Cisse  CS 4318  03/04/2016

Database project for CS 4318

# Team Members:

Ibra Cisse

Jason Jensen

Project Name: Recommendation of medication

**Table of Content**

1. Abstract………………………………………………………………………3
2. Introduction…………………………………………………………………4
3. Database Description……………………….…………………………….5
4. Database Functionalities….……………….…………………………….5
5. Relationship…………………………………………………………………7
6. Use Cases……………………………………………………………………8
7. E/R Diagram………………………………………………………………11
8. Relational Model………………………………………………………….12
9. Database Prototype………………………………………………………13
10. Time Management Plan………………………………………………...14

**Abstract**

When sick, a trip to the drug store is an unpleasant endeavor. Even knowing your illness, choosing the best medication to alleviate your symptoms can be difficult due to the large number of options available. This project would make this choice, and therefore the trip, much simpler. By inputting a list of symptoms, the database will cross reference and recommend the best medication for your symptoms. Medication is ranked according to effectiveness based on an ever growing list of surveys. These symptoms would include: headaches, toothaches, stomachaches, allergies, etc.

**Introduction**

This project is meant to help regular people choose the best over the counter medication for minor illnesses. The concept is to develop a system where users can input the symptoms that they are experiencing and based on research and feedback from customers and medical documentation, we provide a recommendation of the best over the counter pills to take to alleviate the symptoms. The application will use several tables and databases which will have several indexed tables. The databases will be tied to each other with an index id to quicken queries. Once queried the application will search the databases that contain a medicines description and ranking, based on reviews from other users and scientific documentations, queries will then return the top sorted medicines for the specified symptoms along with their description. The results will be presented through a custom made user interface. Users will be able to select from drop down menus to view medicine descriptions and ranking. Users will also have the choice to log in and write reviews which will influence the ranking of medication.

**Database Description**

This database will allow most people to select the best medication to alleviate symptoms whilst sick. Our intent is to provide a recommendation for the best medication for minor illnesses that do not warrant a trip to a doctor or to simply have on hand as first aid. Within our ER diagram, we have included preliminary Primary keys and all references this project will have. This might change as we progress further into development and create a more consistent recommendation system based on ratings and reviews. Different users will have different views based on their level of access. For example, an administrator will be able to create, add, or delete rows based on the circumstances such as adding new medicine, archiving medicine that has poor reviews, or updating relevance of reviews and ratings. The typical user will be able to view medicines, rate medicine, and write a review for future users.

**Database Functionalities**

The following database will focus on 3 different type of aspect of view as of Administrator, user, and client. Each one of the following has specific roles and regulation on what they can do and access based on their rights set up by the database manager:

1. **Users**

Users are typical users who do not any rights but write reviews and increase the rate of a medicine. These are regular people that visit and can get their daily services.

1. **Client**

The client is a user register within the database. They have to be authenticated from the database and given the respect view. Their roles and right include being able to request data from the database such as most popular medicine for a sickness, average prices for medicine, highest or lowest rank medicine for a certain type of disease. Clients can also change the rank of medicines and delete reviews.

1. **Administrator role and functionalities**

The Administrator is the database manager and also controls everything within the database. Administrator has the right and roles to create new tables, update tables, and change table outcomes. The administrator can also delete tables or delete the existing database as well as initiate a backup of the database. Administrator can also change the rank of medicines and delete reviews.

1. **Database Functionalities**
2. Table for clients which will include Index ID, username, password, first name, last name, age, and sex.
3. Another table will hold medicine which will be linked to the clients’ table by IDs as primary keys, and the rest will have foreign keys. As for medicines, they will have several rows that holds the brand and generic names as well as the description and usage.
4. A third table will hold all of the symptoms that someone might have. These will have a description field where they will be linked to other tables such as medicine and which users have listed them before.
5. A rating and review table which will be linked to client, medicine, and sickness by indexed ids. The remaining fields will have foreign keys for faster access to information.
6. Administrators will be able to update all fields, create new fields, update rows, and change ratings.
7. Administrator will be able to delete rows based on relevant information that have been given or recorded on the database
8. Users will be able to register and log in to write reviews, rate medicine, and give suggestion.
9. Another table will be called Chemical Component, this table holds the chemical makeup of the medicine along with its known side effects. This table will be linked to both medicine and rating as well.
10. The last table will be medicine sales prices from different cities of the United States. We planned to use only 20 different states along with several cities in the USA. This will link to medicine, clients, administrators and as well as chemical contents.

**Relationships between Tables**

* The client and medicine tables will be linked based on IDs which will be the primary key. The rest will be foreign keys.
* The medicine table will be linked with the symptom table, each medicine will become a member of a symptom, which will hold a description of that symptom and the medicine most recommended to alleviate it.
* Ratings and reviews will be linked to the symptom and the medicine tables, it will be based on how many reviews have been received. Each review will be rated from 1 to 10.
* Updating will go to the targeted field based on customer ID, then the symptom table and the medicine will be given a review and rating. The average rating, along with number of reviews, will be display towards top bar of the pages
  + Two different type of client: regular user and administrator
  + Administrator can make changes towards core database
  + User can only update information and rating systems through reviews.
* The symptoms Table will contains all the symptoms for that specific illness. The symptoms will be linked to the client and Administrator table based on primary key with the same IDs field, names, and last names.
* Review will also be linked by primary keys to the client and Administrator table, from there, they will be linked to medicine, and symptoms.
* Medicine Sales values table will have the foreign keys to medicine table, and review.

**Use Cases and Scenarios for the database**

The following has several possible use cases for the databases and its entities.

**Case 1: User ask for average price in a city**

Input: search for average price

Output: database is query and returns the average price from specific table such as Sales Price, and also name of the medicine.

**Case 2: user asked for highest rated medicine for a specific sickness**

Input: user asked for the highest rated for headache

Output: query the select statement to show the highest rated medicine for headache, bring back which cities it is available and also their rating.

**Case 3: User writes a review for a medicine**

Input: writes a review on a textbox and submit it to the database.

Output: database executes the update statement, select a specific table and update the review and ranking based on the input. Also user is never recoded because it is a regular user.

**Case 4: user try to create a table**

Input: create a table for new attributes

Output: users cannot create new attributes or change attributes due to their restrictive rights.

**Case 5: User sings up to be a client**

Input: sign up process by providing username, first name, last name, password, repeated password, sex, age.

Output: insert statement is done and user can be query and added to a table as a client. Given the client rights, user can now alter tables but not create them, they can update rating and so forth….

**Case 6: Client signs in**

Input: input username and password

Output: select statement is queried and retrieved. Compare if the information are matching. If matching, then client can see their view. If the information is not matching, client will be redirected to entering username and password.

**Case 7: Client signs out**

Input: click the sign-out button

Output: closes the view and bring back the regular view which can only search and view basic information.

**Case 8: Client sings up**

Input: sign up process by providing username, first name, last name, password, repeated password, sex, age.

Output: insert into the Client table and made all right available to the client table. Client will be able to view certain things as well as update tables. Cannot create tables or alter changes on the database.

**Case 9: client updates review**

Input: writes a review in the provided textbox and then

Output: insert the review into the review table and change the rank if effected.

**Case 10: client updates rating for medicine**

Input: input the number of starts in the textbox

Output: goes to the review table and update the ranking either by higher or lower. This effects its status based on the number of starts it has

**Case 11: client update the table board for prices**

Input: input a new price for a medicine.

Output: database returns an error because client do not have the right to change this type of specification

**Case 12: Administrator log on**

Input: input username and password

Output: Administrator will be queried and see if his information matches. If matches, view will be changed and Admin can change anything due to all rights given.

**Case 13: Administrator logs out**

Input: clicks sign out

Output: Administrator view will be close, return back to sign in mode.

**Case 14: Administrator deletes a table**

Input: inputs delete a specific table

Output: database query the statement as "DELETE FROM medical Database WHERE table = “specific table”". Then the table will be deleted and return a successful answer.

**Case 15: Administrator creates a new table**

Input: create a new table by giving the entities of the table such as the type of information it can hold.

Output:

**Case 16: Administrator adds to the a table**

Input: uses the insert data and specified which table to insert it to

Output: inserts new record on the table and return successful

**Case 17: Administrator deletes a record from the database**

Input: uses the delete method by providing the specific table, which data to delete.

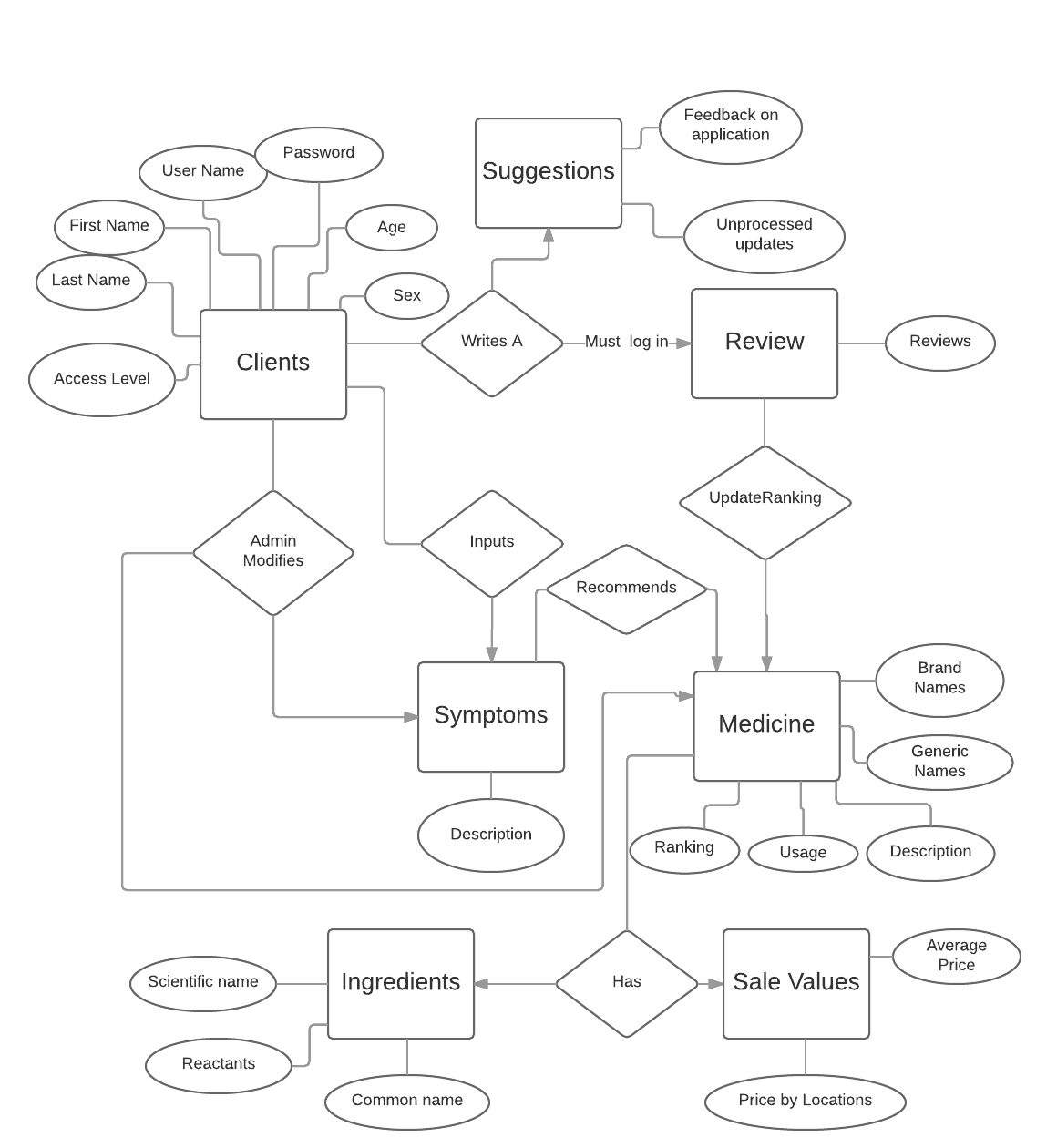
Output: data is deleted based on specification and also return successful

**Case 18: Administrator ask for average price of a drug from cities**

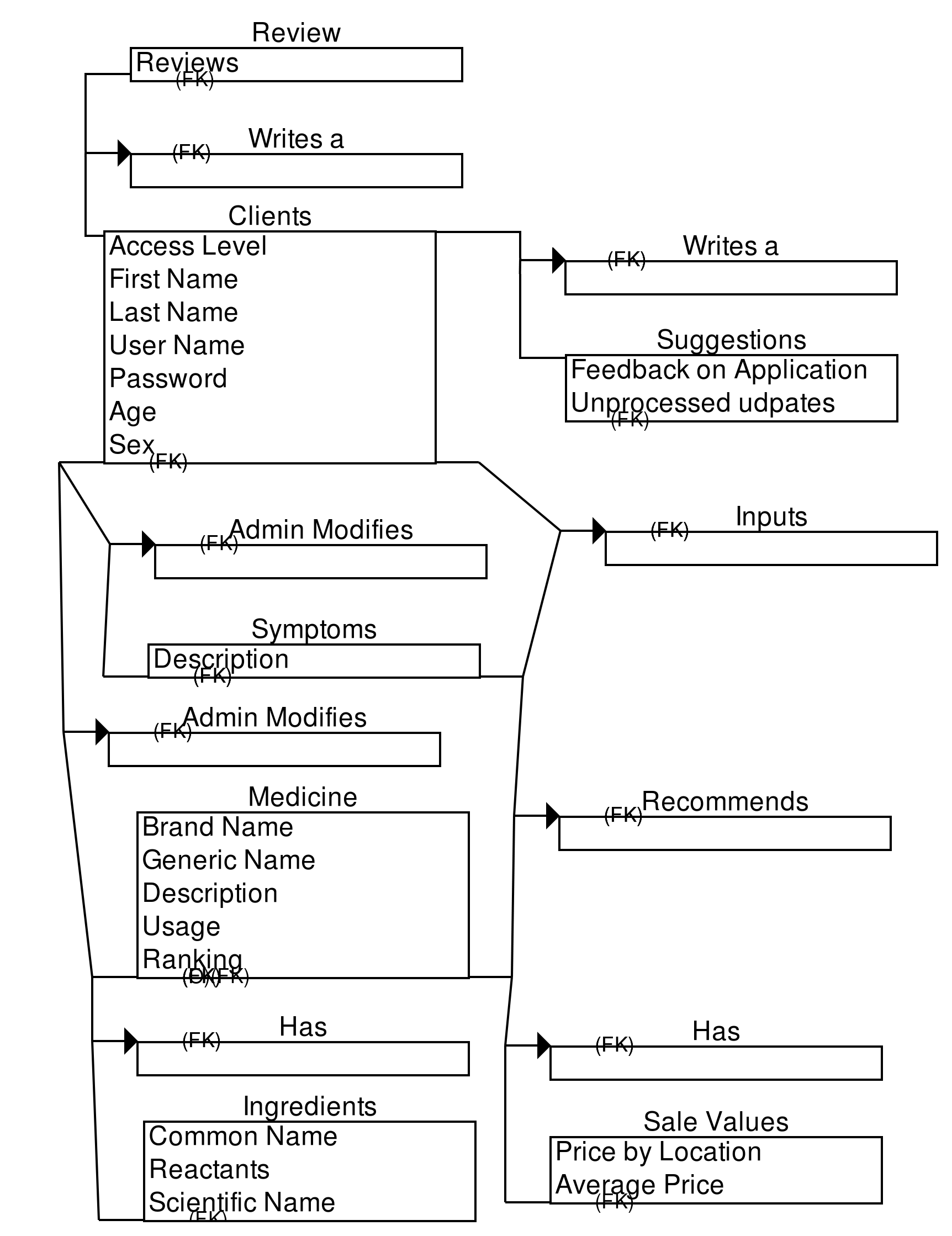
Input: ask for average price for certain city

Output: uses the sum to sum up prices for that city and bring back the average price on that city.

**E/R Diagram**



**Relational Model**



**Database Prototype**

For the database implementation, we are using MySQL to build all of our database. For plugins, we are using sublime editor to edit most of our codes. Sublime allow us to write multiple lines of code and import as a .sql file which can run all of the function from any database as long as it is uploaded. As for how we will supply information for the database, researching on Google and handing out surveys will give us the best medicines out currently most common sicknesses such as headache, toothache, stomachache, etc…. We might also implement a simple GUI interface in order to have a much friendlier environment.

**Time management Plan:**

The time management is how we plan to complete the project through mile stones. Each milestone has been reach to the most recent. More will be completed and also updated according to our progress as a team.

|  |  |  |
| --- | --- | --- |
| **Task** | **Status** | **Comments** |
| **Basic requirement of the database** | Completed on January 21st, 2016 | Completed by both team member Ibra and Jason |
| **Basic diagram for users** | Completed on January 21st, 2016 | Completed by Jason and reviewed by Ibra |
| **List of possible use cases** | Completed on March 3rd, 2016 | Completed by Ibra Cisse  Review by Jason |
| **E/R design of each functionality of the database** | Completed on March 17th, 2016 | Completed by Jason  Review by Ibra Cisse |
| **Database prototype** | In progress | Expect it to finish April 7th, 2016 |
| **Prototype Test** | In progress | Expected to finish by April 9th, 2016 |
| **Final documentation of the database** | In progress | Expected to finish at April 25th |
| **Final database design and implementation** | In progress | Expected to finish on May 1st, 2016 |
| **Final Presentation of the Database** | In progress | Expected to Finish on May 1st, 2016 |