Health Care Management System

Class: Software Requirements Engineering **Date**: 09/19/2013

Group: Mohammed Abdul Muqeet abdul@udel.edu , Pradnya Powar pradnya@udel.edu, Yi Zhu yizhu@udel.edu

1. Introduction:

This document goes over the requirements that must be fulfilled before Health Care Management System can be delivered to the customer. The purpose of this project is to store a patient's medical details and provide a medium through which these medical details can be shared with the doctors.

1.1 Purpose:

The purpose of this document is to present a detailed description of the Health Care Management System. It explains the purpose and features of the application itself, the web interface through which a user can access a personal account, the server that hosts the user accounts, and the constraints that must be satisfied for security purposes. This document is intended for both the customer and the project development team.

1.2 Scope:

Healthcare today is an integral part of our daily life. Every person would like his medical details to be recorded or stored at one place, so that even if he loses track of his history, it can be easily found. In this modern world of technology instead of hanging on to the medical papers it is better to have everything on computers. This application will be designed to provide a simple and reliable way to store and convey one's medical history. Moreover such records are vital in some case of medical diagnostic.

The application will contain extensive documentation of the patient's medical records, which can be authored by the owner or the owner's healthcare provider(s). This includes diagnoses, treatments, medications, allergies, and medical procedures. This would allow for uniform communication of information between the patient and their respective doctor(s). The software would label and organize information authored by health care providers separately from information authored by the hospital rep. to maintain authenticity.

Not only the patients but also the doctors or hospitals will benefit from such a system. If we provide a medium through which a patient’s medical history can be shared, it will be easier for the patients to search their medical data and learn potential health problems. At the same time, it will be a good way for the doctors to have the reports handy.

2. System Overview:

This system is a separate entity. It acts like a 3rd party between patient and hospital/clinic. The system provides credentials to patients, doctors and hospitals to log in to the system.

When the patient creates an account he needs to enter his demographic data viz. his name, address, age, sex, contact details, emergency contact.

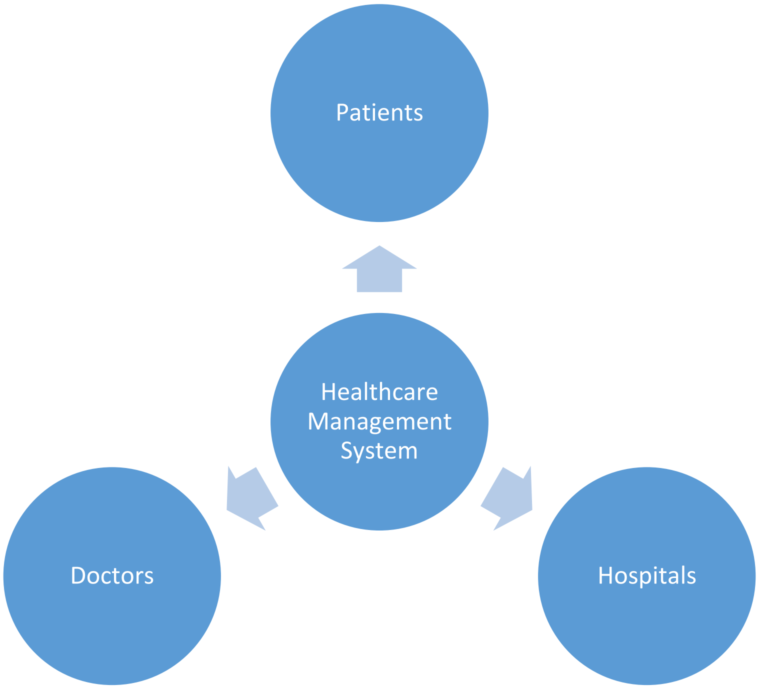
The hospitals account will have their name, address, list of all the doctors that are associated with the hospital, list of patients.

The doctor for his account needs to enter his name, address, for authenticity his unique registration number, contact details, specialization if any, the hospital he is associated with, the list of patients he is treating.

All these details need to be entered while signing up. Then every time for logging in to the system just the username and password is needed. For connecting the doctor and patients, the doctor sends a onetime request to the patient for accessing his records. In this way, all the patients that the doctor is treating will be linked to his account. Doctors and hospitals are able to insert or update the patient's medical information. Once any update is made to a patient’s information, he is notified electronically.

The information should be uploaded and updated immediately. Information like the X rays, scans and blood reports are uploaded to the patients profile as soon as the test results are available. To avoid any errors while inputting the information on the system we will be uploading the scanned copies of all the reports.

3. Context Diagram:



These are the various users that will be using this system

4. Assumptions and Dependencies:

- The patient agrees to store the data in system.
- All doctors associated with the hospital are under confidentiality clause of not revealing the information without the patient's authorization.
- This system will require network access at the hospital to connect to the servers on remote location.
- The system assumes that the users have adequate skill with using computers and computer software.
- The server assumes it will be installed with a high-speed Internet connection to communicate with the users of the Health Care Management System.
- The system assumes that the user has a computer with an Internet connection and a web browser to access the application.

5. Actors of the System:

The actors in this system are -

- **Patients** - The patients are the main users of this system. They have their medical data all stored securely at one place with doctors and hospitals having access to them only if the patient gives them the permission.
- **Hospitals** - Hospitals would have permission to search and synchronize information of the patients and the doctors. The purpose of the system is allow hospitals to exchange information between different hospitals if needed. They can access the patient's data only with the permission of the patient. They can upload/edit the patient's medical records only with the permission of the patient.
- **Doctors** - The main aim of this system is to provide a medium through which the patients' medical data can be stored all together and shared with other medical professionals if needed. The doctors can have all reports together so that he can refer to whatever he needs quickly. He can also write prescriptions for the patients associated with him.

6. Use Cases:

Login Use Case

Actors:	Patient, Doctor, Hospital rep.
Pre-Conditions:	The user is not already logged into the system. The user is authorized to use this system.
Basic functionality:	The login use case allows the User to authenticate him/herself to the system and obtain access to all features of the role he/she is associated with as defined in his/her user account. If the potential user does not enter a valid user name and password the system will not let the user proceed beyond the login screen.

Logout Use Case

Actors:	Patient, Doctor, Hospital rep.
Pre-Conditions:	The user is logged into the system.
Basic functionality:	The logout use case allows the User to disassociate him/herself from the system. The system removes any session information that it may be maintaining for the User. The User's account remains unchanged.

Registration Use Case

Actors:	Patient, Doctor, Hospital rep.
Pre-Conditions:	The user is not already registered in the system.
Basic functionality:	The user accesses the application and registers for an account.

Personal Details Use Case

Actors:	Patient, Doctor
Pre-Conditions:	The user is logged into the system.
Basic functionality:	The user accesses the Health Care Management System through a web application and updates or changes certain attributes of his/her personal details

Update Medical Records Use Case

Actors:	Hospital rep.
Pre-Conditions:	The user is logged into the system. The user has the permission of the patient to access his/her medical records.
Basic functionality:	The Hospital rep. accesses the Health Care Management System through a web application and updates or changes certain attributes of a medical record.

View Medical Records Use Case

Actors:	Patient, Doctor, Hospital rep.
Pre-Conditions:	The user is logged into the system.
	The user has the permission of the patient to access his/her medical records.
Basic functionality:	The user accesses the Health Care Management System through a web application and views the medical records of a certain patient.

Open Ticket Use Case

Actors:	Patient
Pre-Conditions:	The user is logged into the system.
	The user has found an issue or mistake in his medical records.
Basic functionality:	The user accesses the Health Care Management System through a web application and opens a ticket.

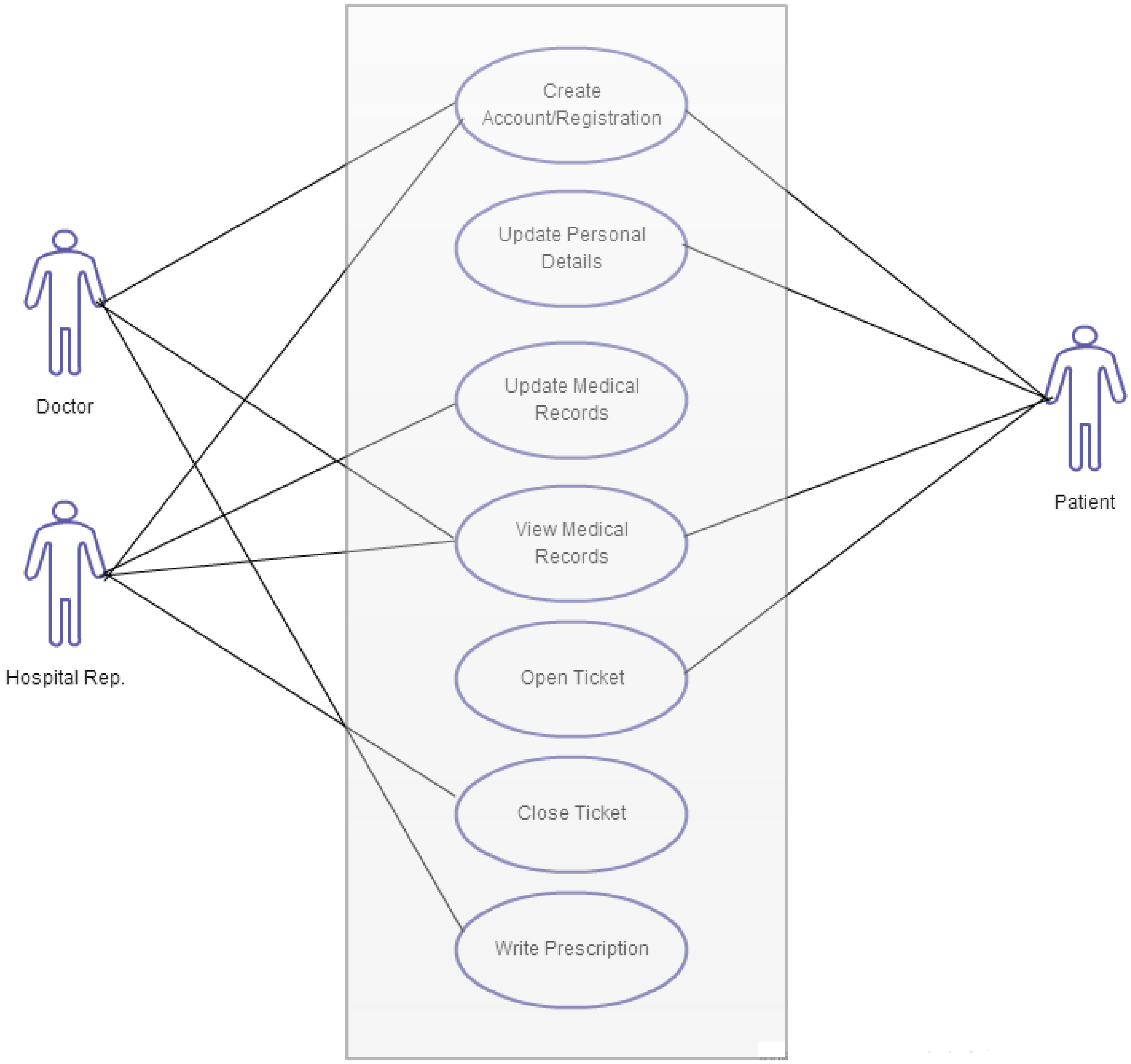
Close Ticket Use Case

Actors:	Hospital rep.
Pre-Conditions:	The user is logged into the system.
	The user has the permission of the patient to access his/her medical records.
Basic functionality:	The user accesses the Health Care Management System through a web application and closes the ticket once the issue is resolved.

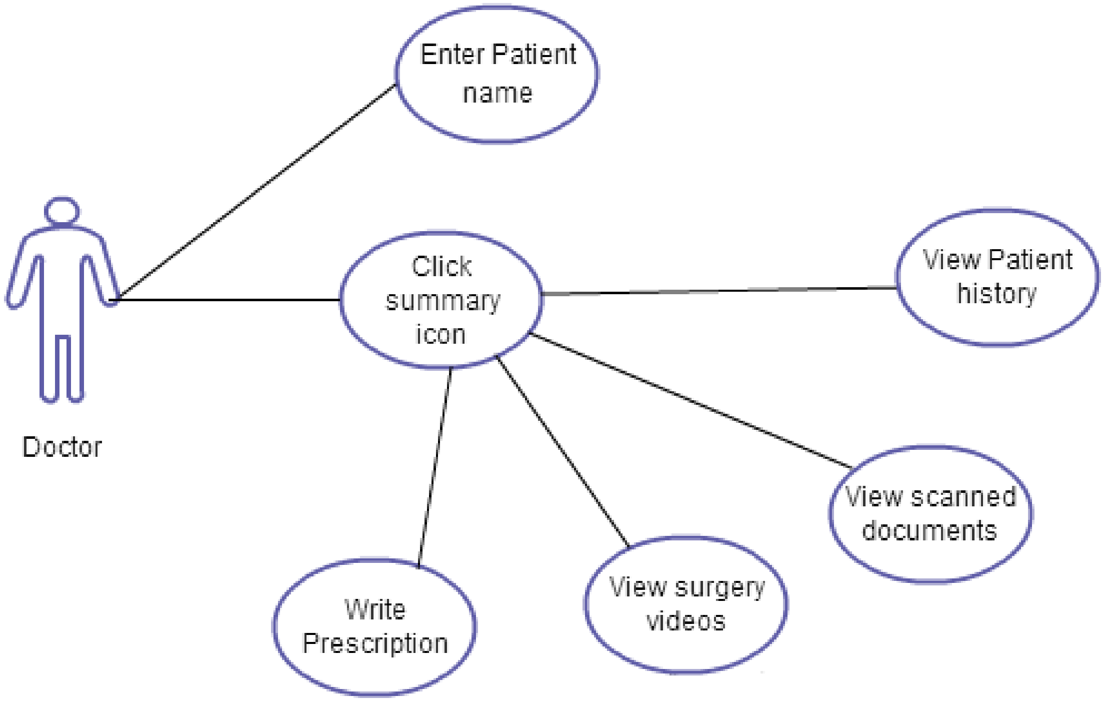
Write Prescription Use Case

Actors:	Doctor
Pre-Conditions:	The user is logged into the system.
	The user has the permission of the patient to access his/her medical records.
Basic functionality:	The user accesses the Health Care Management System through a web application and writes a prescription for a certain patient.

Use Case Diagram:



Search Patient Record Use Case:



7. Functional Requirements

- To Login into the system, the user needs to be registered in the system. The user provides a username and password.
- The user should logout after accessing the needed information. The session information is maintained until the user logs out and then the user is disassociated from the system.
- Each patient, during registration, enters his/her name, address, phone number, insurance type, sex, date of birth, age, medical condition, and a Social Security Number, which distinguishes him/her from other patients in the database.
- Any patient may access a list of physicians' names and a public subset of their contact information, including name, specialization, and phone number. In addition to their publicly accessible information, physicians have an address and a unique license number. They can care for any number of patients, and have the ability to add notes and prescriptions to their patients' records.
- The patient can grant any doctor or hospital rep. that exists in the system a permission to access his medical records.

- The software must allow input of patient medical data from a secured access at Hospital done by the Hospital rep.
- The medical records are stored in the form of images and videos.
- All medical test reports are scanned and then uploaded on the system by the hospital rep.
- The software must allow browsing by the doctor of the medical information of his/her patients only.
- The doctor can prescribe either medicine or lab tests to his patients.
- The software must allow patient to view their own medical record online allowing changes only to his personal demographics after initial input.
- The software must allow input of diagnostic imagery.
- The patient can open a ticket if he finds any issues or errors in his medical records.
- The ticket can only be closed by the hospital rep. upon resolving the issue.
- All users of the system need to exist in the system before they can log in and use the system. A user is assigned a user role and other details such as name, address, email, phone number, and a unique Social Security Number.

8. Non- Functional Requirements:

Usability:

- Since users of the Healthcare Management System will be of all sorts and types and not all of them will be very experienced with computers, it is essential that the application be as user-friendly as possible.
- The system should be intuitive enough for a new user to learn within 10 days of formal training.
- The software interface must follow design conventions which allow for familiar location of drop down menus, help etc.

Reliability:

- Reliability of this system is very crucial. The system should run efficiently all round the clock.
- The system saves sensitive medical data for thousands of patients. Hence the design must consider the integrity and security of the data.
- The data will be backed up every day.

- The system should display 100% accurate data.

Performance:

- The system should be very responsive.
- Every web page should correspond within the latency rate of three seconds.
- The system should be able to find a patient’s record in less than three seconds.
- The System must support 1000 people at a time.
- The user-interface screen shall respond within 5 seconds.

Security:

- The system requires the patient to identify himself /herself using SSN.
- Any user who uses the system shall have a Logon ID and Password.
- All user types such as patients, doctors and hospital rep. have distinct pages to which only they can access. The password safety will be provided by the encryption of the passwords saved in the database.
- More than three attempts at login and failure will produce a red flag to system administrator.

Maintainability:

- Routine maintenance is a must. Hence the system should be easily maintainable.
- The data needs to be backed up periodically so in case of a major catastrophe like flood, earthquake etc., the data should not be lost. And the site can be brought up quickly and data can be restored.
- The system shall keep a log of all the errors.
- The revision history of the code should be saved so that in case of a problem, the website can be reverted back to last good state.

Safety:

- Humans are error-prone, but the negative effects of common errors should be limited. E.g. users should realize that a given command will delete data, and be asked to confirm their intent or have the option to undo.

Upgradability:

- By having some knowledge of programming, some features of the system might be converted to a new version. According to the needs of upgrade, system requirements might change such as change in hardware or operating system or not.

Supportability:

- Supportability will be provided over the whole product life of the system. System will be quite easy to use but educational support will be given if needed.

Extensibility:

- Over time the system can be enhanced with small features, therefore the application should be extensible.

Robustness:

- It is expected that the system does not crash if a user inputs wrong data or makes mistakes or if too many people are trying to access the system at a time.
- Input errors will be returned in red with appropriate message box.
- The software must require high levels of error correction.

9. Environmental Requirements

9.1 System Hardware Requirements

- PC 1.6 GHz or higher.
- 512 Mb RAM or higher.
- 40 GB Disc Space or higher.
- Video-Graphic Card (800 x 600) 128 Mb or more.
- Internet Access.

9.2 System Software Requirements

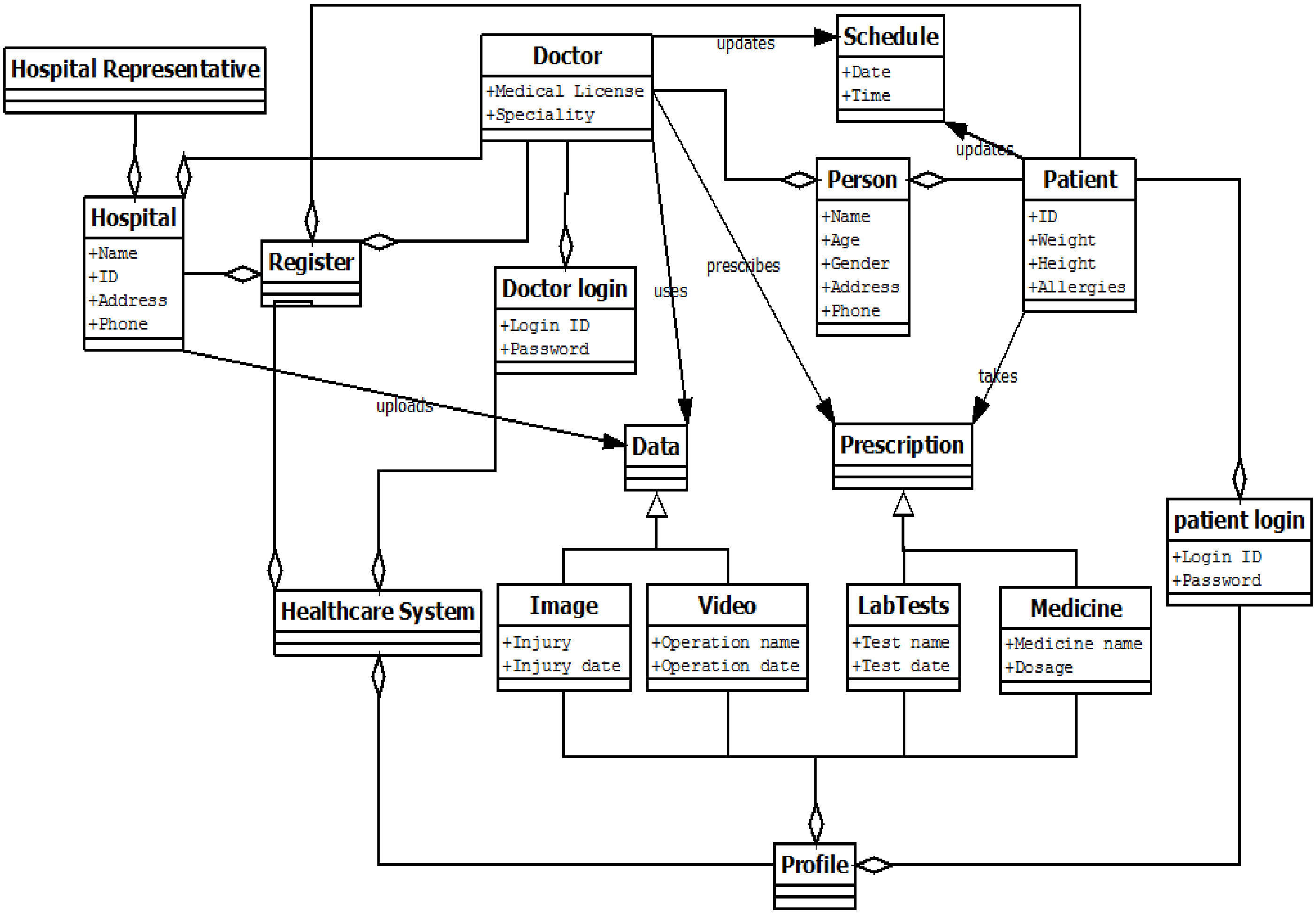
Software requirement to run the system:

- Microsoft Windows XP/Vista/7/8 or Linux or Mac OS X/Lion.

- Any version of Chrome, Internet Explorer, Mozilla Firefox, Opera etc.

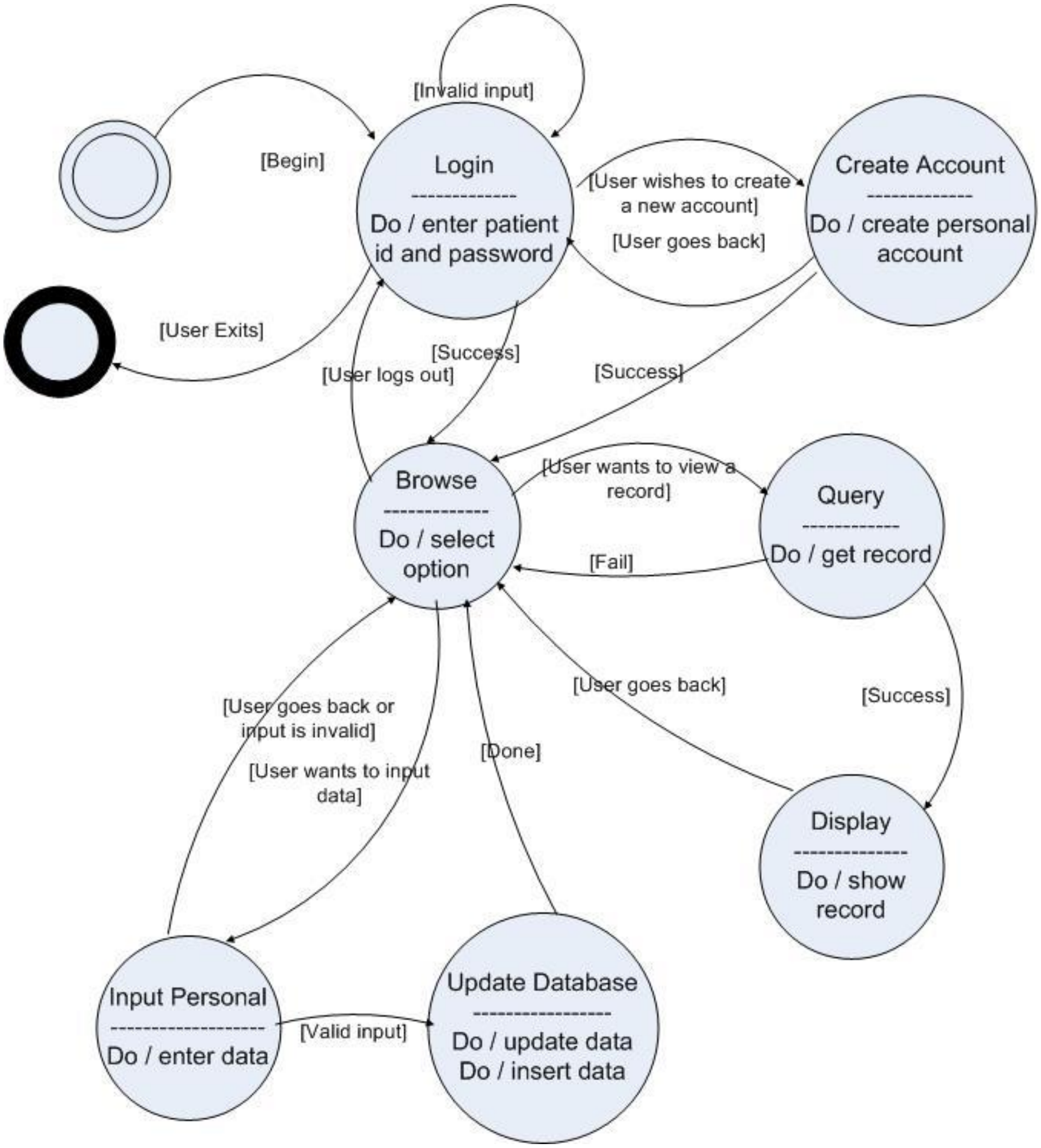
10. UML Diagrams:

10.1 Class Diagram:



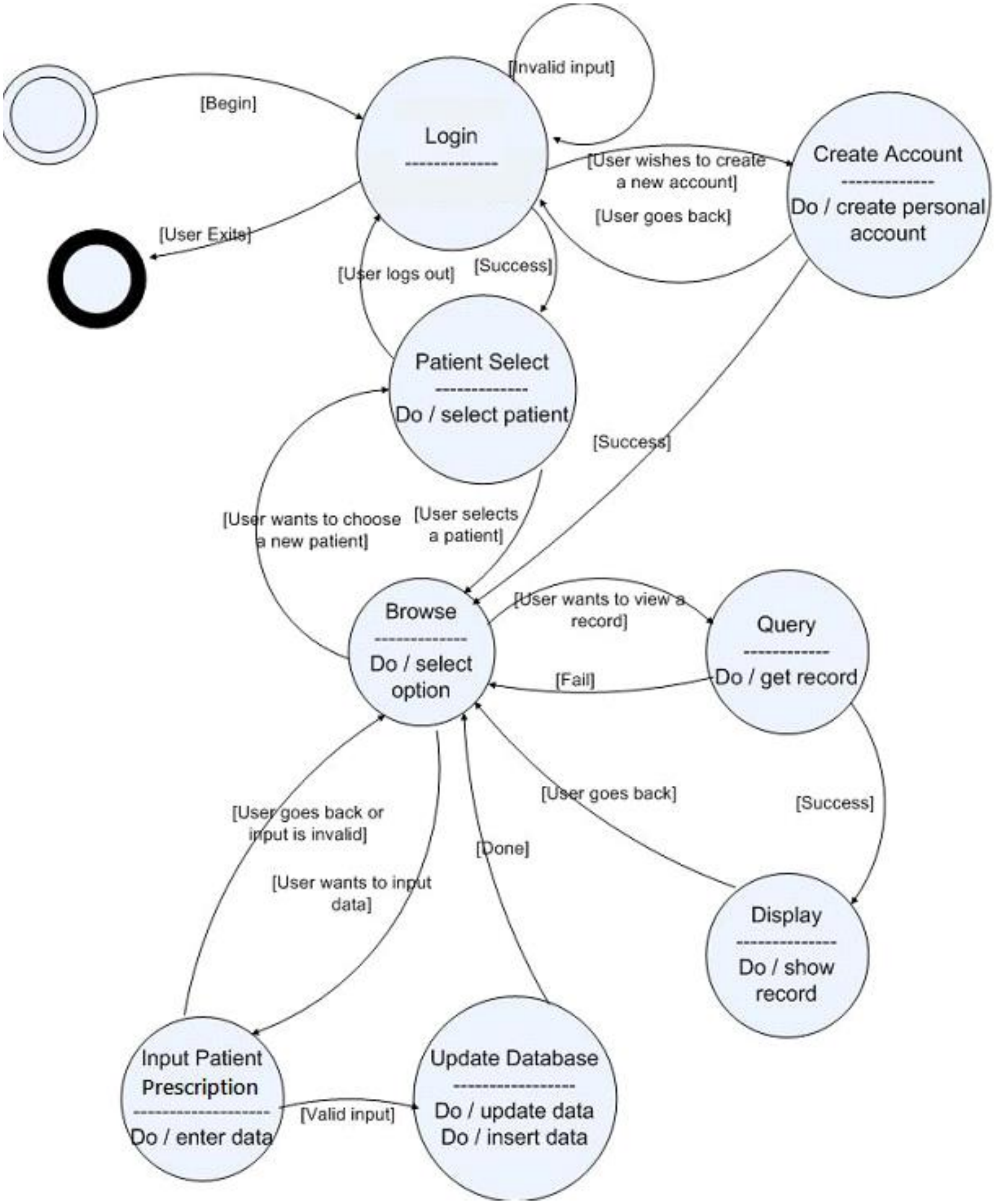
10.2 State Diagrams:

Patient:



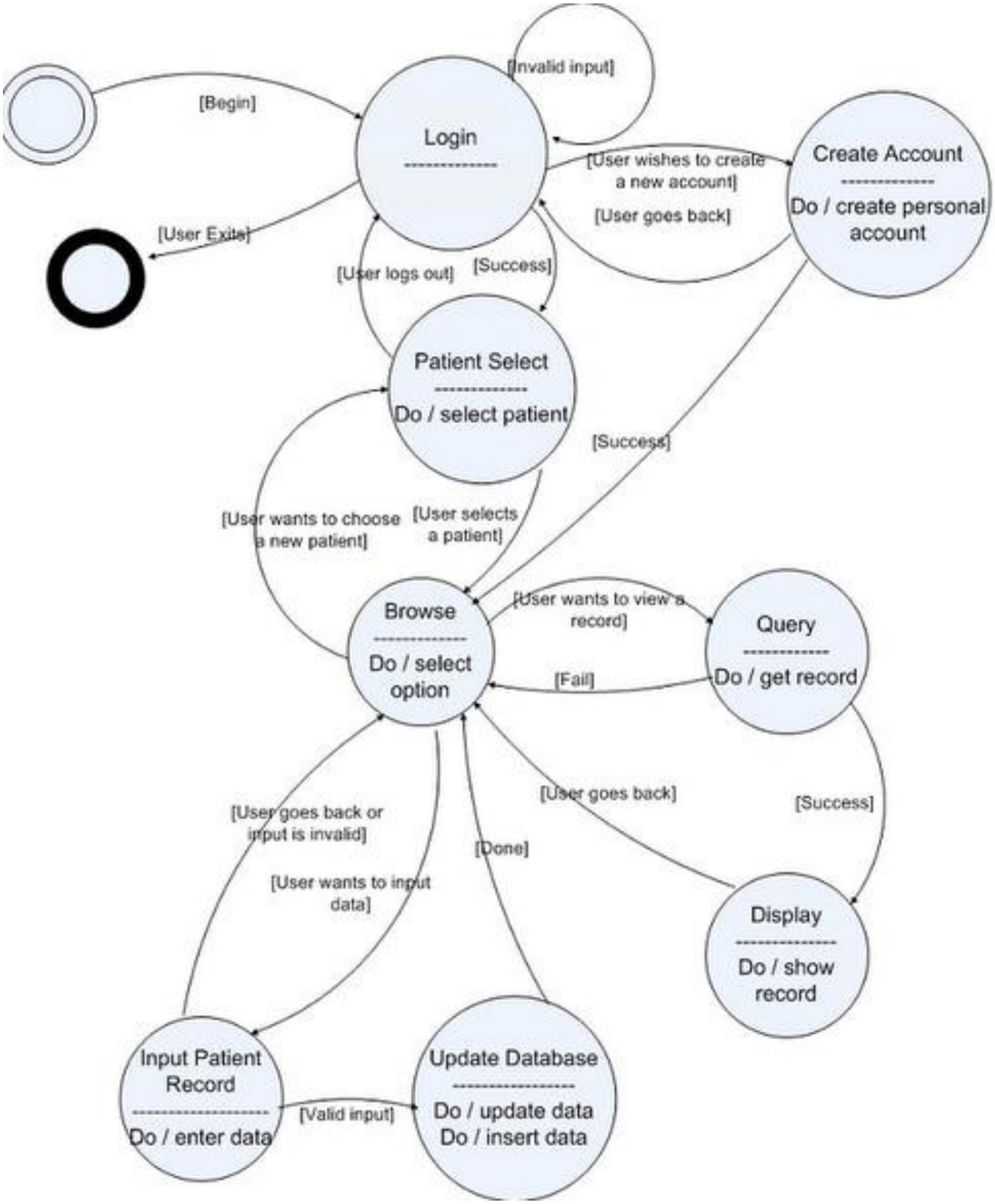
Once a patient enters the website, they are prompted to log in. If the patient enters incorrect information, that patient cannot proceed. If the input is valid, the patient is free to browse his/her personal medical records. The patient also has an option to create a new personal account. If he/she wishes to do so, the system prompts the patient for information, creates the account, and logs him/her in. The patient can choose to upload new personal data, or view an existing record. If the patient decides to input new personal data, he/she is taken to a page that allows him/her to do so. The patient is returned to the browsing page if he/she wishes to do so or inputs invalid data. If the patient inputs correct data, the database is updating accordingly and the patient is then returned to the browsing page. If the patient decides to view an existing record, the system will check to make sure the selected record exists. If so, the record is displayed. If the record does not exist, the patient is returned to the browsing page. The patient can log off at any time while in the browsing state.

Doctor:



Once a doctor enters the website, they are prompted to log in. If the doctor enters incorrect information, that doctor cannot proceed. If the input is valid, the doctor is free to select one of his/her patients. Once a patient is successfully chosen, the doctor is free to browse that patient's records. The doctor also has an option to create a new personal account. If he/she wishes to do so, the system prompts him/her for information, creates the account, and logs him/her in. The doctor can choose to upload new personal data, or view an existing record. If the doctor decides to input prescription for a patient, he/she is taken to a page that allows him/her to do so. The doctor is returned to the browsing page if he/she wishes to do so or inputs invalid data. If the doctor inputs correct data, the database is updating accordingly and the doctor is then returned to the browsing page. If the doctor decides to view an existing record, the system will check to make sure the selected record exists. If so, the record is displayed. If the record does not exist, the doctor is returned to the browsing page. The doctor can log off at any time while in the browsing state.

Hospital Rep.:



Once a hospital rep. enters the website, they are prompted to log in. If the hospital rep. enters incorrect information, that hospital rep. cannot proceed. If the input is valid, the hospital rep. is free to select one of his/her patients. Once a patient is successfully chosen, the hospital rep. is free to browse that patient's records. The hospital rep. also has an option to create a new account on behalf of the hospital. If he/she wishes to do so, the system prompts him/her for information, creates the account, and logs him/her in. The hospital rep. can choose to upload new medical records, or view an existing record. If the hospital rep. decides to input new data, he/she is taken to a page that allows him/her to do so. The hospital rep. is returned to the browsing page if he/she wishes to do so or inputs invalid data. If the hospital rep. inputs correct data, the database is updating accordingly and the hospital rep. is then returned to the browsing page. If the hospital rep. decides to view an existing record, the system will check to make sure the selected record exists. If so, the record is displayed. If the record does not exist, the hospital rep. is returned to the browsing page. The hospital rep. can log off at any time while in the browsing state.

11. Prolog Code:

```
/*Exist in the database*/
inDatabase(phil).
inDatabase(bell).
inDatabase(ryan).
inDatabase(nick).
inDatabase(dave).
inDatabase(brad).

/*Phil and Bell are doctors*/
doctor(phil).
doctor(bell).

/*Jake,Ryan and Nick are patients*/
patient(jake).
patient(ryan).
patient(nick).
```

```
/*Phil and Dave are logged in.*/
loggedIn(phil).
loggedIn(dave).

/*Dave and Brad are Hospital reps.*/
hospitalrep(dave).
hospitalrep(brad).

/*Dave has permission for Ryan's profile */
permission(dave,ryan).

/*Phil has permission for Ryan's profile */
permission(phil,ryan).

/*Ryan finds an issue in his medical records*/
issueInRecord(ryan).

/*Ryan's issue with his medical records is resolved*/
issueResolved(ryan).

/*Check if X can view Y's profile*/
canviewpatient(X,Y) :-  (doctor(X) ; hospitalrep(X)), patient(Y) , loggedIn(X) , permission(X,Y).

/*Check if X can upload data on to Y's profile */
canupload(X,Y) :- hospitalrep(X) , loggedIn(X) , patient(Y) , permission(X,Y).

/*Check if X can prescribe medicine for Y */
canprescribe(X,Y) :- doctor(X) , patient(Y) , loggedIn(X) , permission(X,Y).

/*Check if X can open ticket*/
openTicket(X) :- patient(X), loggedIn(X), issueInRecord(X).
```

```
/* Check if X can close ticket opened by Y */

closeTicket(X,Y) :-  hospitalrep(X) , loggedIn(X) , patient(Y) , permission(X,Y) , issueInRecord(Y), issueResolved(Y).


/*Check if X can update personal data*/

canUpdatePersonalData(X) :- patient(X) ; doctor(X) , loggedIn(X).


/*Check if X can register into the system*/

canRegister(X) :- doctor(X) ; hospitalrep(X) ; patient(X) , \+(inDatabase(X)).
```

12. Risks:

- The system is subject to some types of failure such as erroneous data, disk head crash, and so on. Keeping local copies or fragments of crucial data can be a reliable way to support the need for rapid access to data across the organizations.
- For the system to be robust, it must be able to detect a failure, and recover when a processor or link is repaired. Two functions are helpful: 1) Maintains a log of medical information and before and after database images 2) Maintains an appropriate concurrency control scheme to ensure data integrity during the execution of medical information at that system
- It may be difficult to upgrade the application due to the complexity of the system.

13. Future Scope:

- General Health Guide: A new module which informs the patients about what to and what not to do when they are ill can be added.
- News: A news module/page where the patients are informed about the ongoing flu shots camps, free eye care etc., can also be added.

14. Conclusion:

This document is used to give details regarding the Healthcare Management System. In this document all the functional and non-functional requirements are specified in order to get a clear idea on developing the project.