**Hospital Management System**

**Introduction:**

Hospital Management System is used to take the data from the patients and then store it for later use. The main goal of the Hospital Management System is to accurately treat as well as decrease overtime pay.

There are various features included in the HMS. Some of the system functions include Registration, Patient check out, Report generation, and more. In this documentation , let's check out the functional and non-functional requirements of the Hospital Management System in depth.

**1.a.Functional Requirements:**

There are a lot of software requirements specifications included in the functional requirements of the Hospital Management System, which contains various process, namely Registration, Check out, Report Generation, and Database.

**Sign In Process of SRS (Software Requirements Specification)**

● Login To System : The Hospital Management enables the user (doctor , patient , staff , admin ) enter into System and the user choose his dashboard.

**Admin Process of SRS (Software Requirements Specification)**

● Add User (Patient, Doctor) :The Hospital Management enables the Admin to include new patients/Doctors to the system.

● Display For Doctor / Patient : The Hospital Management enables the Admin to display all doctor/patient that included in the system file.

●Search Doctor / Patient: The Hospital Management enables the Admin to search for specific doctor/patient by id.

●Delete The Doctor: The Hospital Management enables the Admin to delete specific doctor from the system.

● Update Doctor / Patient: The Hospital Management enables the Admin to update specific doctor/patient.

**Doctor Process of SRS (Software Requirements Specification)**

●Make Report : The Hospital Management enables Doctor will write a report for his patient status.

●Show Message History of Visitor : The Hospital Management enables Doctor to show message history of visitor patient.

●Show Message History of Resident: The Hospital Management enables Doctor to show message history of Resident patient.

●Display All Patients: The Hospital Management enables doctor to show all patients with their information from file system.

**Staff Process of SRS (Software Requirements Specification)**

●Display All Appointment: The Hospital Management enables staff member to enable doctor/patient to see the appointments.

●Display All Time Table: The Hospital Management enables staff shows the available time to patient from doctor timetable.

●Filter Appointment: The Hospital Management enables staff to filter the doctor timetable by doctor id and time.

●Book Appointment for Visitor: The Hospital Management enables staff to book an appointment to the visitor patient.

●Book Appointment for Resident: The Hospital Management enables staff to book an appointment to the resident patient who set in the hospital for a long time.

**Patient Process of SRS (Software Requirements Specification)**

●Show All Time Table: The Hospital Management enables patient shows the available time from the system

●Show My Report: The Hospital Management enables patient to see his report after doctor appointment.

**Check Out of SRS:**

● Deleting Patient ID: The Admin can delete the patient ID from the system when the patient's checkout from the hospital.

● Adding new bed: The Admin can add bed.

**Report Generation of SRS:**

● Information of the Patient: The Hospital Management System generates a report on every patient regarding various information time table and his history and appointment.

● Availability of the Bed: The Hospital Management system also helps in generating reports on the availability of the bed regarding the information like bed number unoccupied or occupied, ward name, and more.

**File-Based-System of SRS:**

● Mandatory Patient Information: Every patient has some necessary data like doctor first and last name, 'doctor's ID number , message, message-history ,Time(the patient appointment ), UserName , Password , Age , etc.

● Updating information of the Patient: The hospital management system enables users to update the information of the patient as described in the mandatory information included.

**1.b.Non-Functional Requirements**

There are a lot of software requirements specifications included in the non-functional requirements of the Hospital Management System, which contains various process, namely Security, Performance, Maintainability, and Reliability.

**Security:**

● Patient Identification: The system needs the patient to recognize herself or himself using the phone.

● Logon ID: Any users who make use of the system need to hold a Logon ID and password.

● Modifications: Any modifications like insert, delete, update, etc. for the database can be synchronized quickly and executed only by the ward administrator.

● Front Desk Staff Rights: The staff in the front desk can view any data in the Hospital Management system, add new patients record to the HMS but they don't have any rights alter any data in it.

● Administrator rights: The administrator can view as well as alter any information in the Hospital Management System.

**Performance:**

● Response Time: The system provides acknowledgment in just one second once the 'patient's information is checked.

● Capacity: The system needs to support at least 1000 people at once.

● User-Interface: The user interface acknowledges within five seconds.

● Conformity: The system needs to ensure that the guidelines of the Microsoft accessibilities are followed.

**Maintainability:**

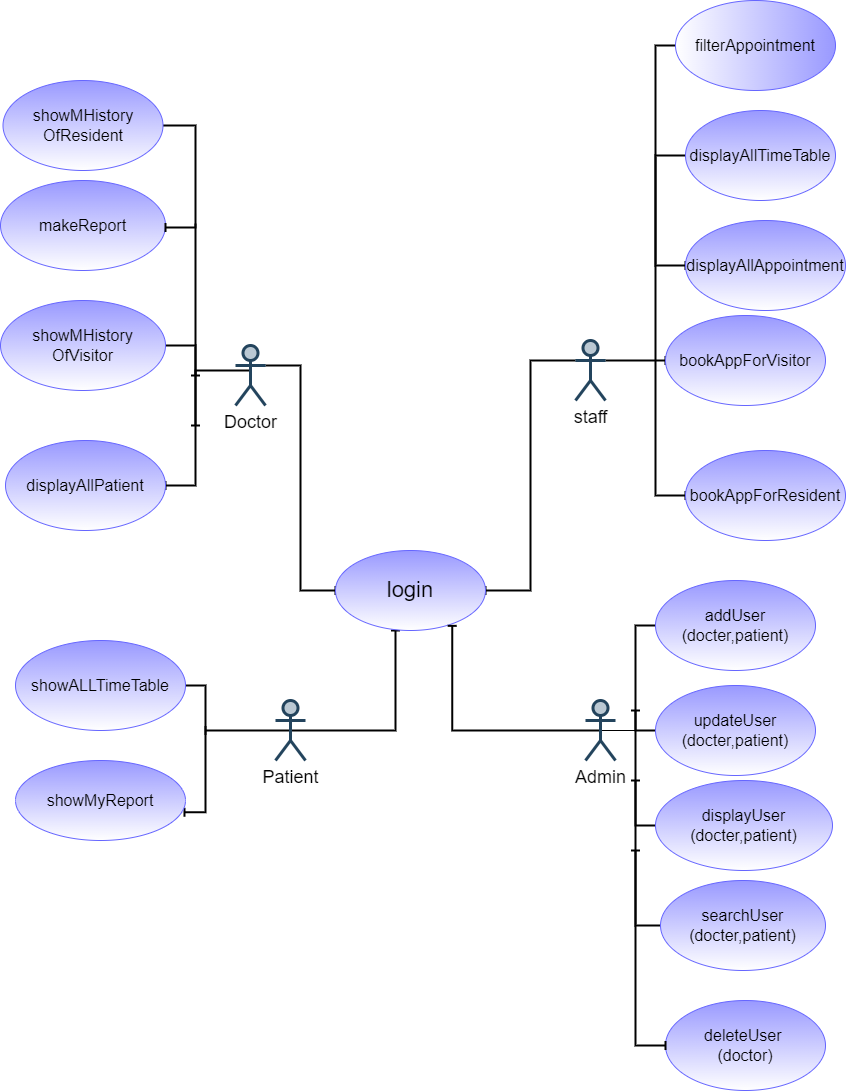
● Back-Up: The system offers the efficiency for data back up.

● Errors: The system will track every mistake as well as keep a log of it.

**Reliability:**

● Availability: The system is available all the time.



**2-usecase diagram**

**2-usecase scenarios**

|  |
| --- |
| Use case scenario for login to the system |
| Use case name : Login |
| Goal: System validates the user |
| Actor: Staff/Admin/Patient/Doctor |
| Pre condition: user must exist in file-based system and  Login page must be displayed |
| Post condition: user logged in successfully  (username and password of the user is Validated) |
| Main success scenario:  - user home page force him to make login not to choose.  -user Writes his email and password.  -user clicks on login.  -user logged in successfully.  Extensions:  1.a. his email or password isn’t correct  1.a.1. System alert him to write the right email and right  password and show message “invalid user …”  1.b. the user is an admin  1.b.1. the system opens admin dashboard  1.c. the user is a Staff  1.c.1. the system opens the system Staff dashboard  1.d. the user is a patient  1.d.1. the system opens the system home page  1.e. the user is a doctor  1.e.1 the system opens doctor page |

|  |
| --- |
| Use case scenario for show message history of resident patient to the system |
| Use case name : showMHistoryOfResident |
| Goal: System review message history of patient |
| Actor: Doctor |
| Pre condition: patient/doctor must exist in file based system and  Login page must be displayed to the doctor |
| Post condition: Doctor logged in successfully and choose the doctor  page  (username and password of the user is Validated) |
| Main success scenario:   * After login doctor will enter into his page. * Then click into resident patient history. * Then the doctor will write the patient id. * the system display the history of resident patient successfully.   Extensions:  1.a. the resident patient id that the doctor enter isn’t correct.  1.a.1. System alert him to write the right resident patient id and show message “invalid patient id …” |

|  |
| --- |
| Use case scenario for create patient report |
| Use case name : makeReport |
| Goal: System validates the patient report |
| Actor: Doctor |
| Pre condition: doctor must be logged in  and opened his home page |
| Post condition: report made successfully |
| Main success scenario:  - doctor log in successfully.  -doctor choose to click on my patient .  - doctor start to write the patient report.  -doctor click on save. |

|  |
| --- |
| Use case scenario for show message history of visitor patient to the system |
| Use case name : showMHistoryOfVisitor |
| Goal: System review message history of patient |
| Actor: Doctor |
| Pre condition: patient/doctor must exist in file based system and  Login page must be displayed to the doctor |
| Post condition: Doctor logged in successfully and choose the doctor  page  (username and password of the user is Validated) |
| Main success scenario:   * After login doctor will enter into his page. * Then click into visitor patient history. * Then the doctor will write the patient id. * the system display the history of resident patient successfully.   Extensions:  1.a. the visitor patient id that the doctor enter isn’t correct.  1.a.1. System alert him to write the right visitor patient id and show message “invalid patient id …” |

|  |
| --- |
| //Use case scenario for write prescribed test |
| Use case name : create prescribed test |
| Goal: patient take the prescribes test |
| Actor: Doctor |
| Pre condition: doctor must be logged in |
| Post condition: prescribed test made successfully |
| Main success scenario:  - doctor log in successfully.  -doctor choose to write prescribed test.  -doctor click on save. |

|  |
| --- |
| Use case scenario for adding new patient |
| Use case name : add new patient |
| Goal: patient added into system |
| Actor: staff |
| Pre condition: staff must be logged in |
| Post condition: adding new patient successfully |
| Main success scenario:  - staff log in successfully.  -staff choose to add new patient.  -staff click on save. |

|  |
| --- |
| Use case scenario for Create Patient Payment Info |
| Use case name : Create Patient Payment Info |
| Goal: making payment |
| Actor: staff |
| Pre condition: staff must be logged in  And check the patient report |
| Post condition: payment info made successfully |
| Flow of events:  - Staff log in successfully.  -staff check the patient report.  -staff choose to create the payment info .  -staff click save. |

|  |
| --- |
| Use case scenario for Create Available Beds Report |
| Use case name : Create Available Beds Report |
| Goal: write the available beds continuously into a report |
| Actor: staff |
| Pre condition: staff must be logged in  And check the deleted patients |
| Post condition: generate a report with available beds |
| Flow of events:  - Staff log in successfully.  -staff check the deleted patients.  -staff choose to create available beds report.  -staff click save. |

|  |
| --- |
| Use case scenario for Book New Appointment |
| Use case name : Book New Appointment |
| Goal: to book an appointment with doctor |
| Actor: staff / patient |
| Pre condition: staff/patient must be logged in |
| Post condition: new appointment added |
| Flow of events:  - Staff/patient log in successfully.  -staff /patient book new appointment.  -new appointment saved. |
| Use case scenario for Create User ID |
| Use case name : Create User ID |
| Goal: give each new user a unique id |
| Actor: Admin |
| Pre condition: Admin must be logged in  And check new user added |
| Post condition: generate a unique id for User (Doctor/Patient/Staff) |
| Flow of events:  - Admin log in successfully.  -Admin check new user added.  -Admin create a unique id for patient. |

|  |
| --- |
| Use case description for Add New User |
| Use case name : Add New User |
| Goal: adding new user(doctor/staff/patient) to the system |
| Actor: Admin |
| Pre condition: admin must be logged in  And want to add new user |
| Post condition: adding new user (doctor/staff/patient) successfully |
| Flow of events:  -Admin logged in successfully.  -Admin clicks on Add new user (doctor/staff/patient). |

|  |
| --- |
| Use case description for Update User |
| Use case name : Update User |
| Goal: make a specific update in user’s  (doctor/staff/patient) information |
| Actor: Admin |
| Pre condition: staff/patient must be logged in  And want to update user information |
| Post condition: user (doctor/staff/patient) details are updated in the  database |
| Flow of events:  - Admin log in successfully.  -select user.  -Admin clicks on update user (doctor/staff/patient) information.  -new updated information saved. |

|  |
| --- |
| Use case description for Delete User |
| Use case name : Delete User |
| Goal: delete a specific user (doctor/staff/patient) |
| Actor: Admin |
| Pre condition: Admin must be logged in |
| Post condition: the user is deleted from the system |
| Flow of events:  - Admin log in successfully.  -select user .  -Admin clicks on delete user (doctor/staff/patient) information.  -user information deleted successfully. |

**3-activity diagram**

**4-uml diagram**

**5-system architecture**

**6-sequence diagram**

**7-system sequence diagram**

**8-collaboration diagram**