

---

# **Software Requirements Specification**

for

## **Hospital Patient Management System**

Version 1.0 approved

**Prepared by Himanshu Shekhar**

**13 November, 2018**

## Table of Contents

<b>Revision History.....</b>	<b>0</b>
<b>1. Introduction.....</b>	<b>1</b>
1.1 Purpose.....	1
1.2 Scope.....	1
1.3 Definitions, Acronyms and Abbreviations.....	2
1.4 References.....	2
1.5 Overview.....	2
<b>2. Overall Description.....</b>	<b>2</b>
2.1 Product Perspective.....	3
2.1.1 System Interfaces.....	3
2.1.2 User Interfaces.....	3
2.1.3 Hardware Interfaces.....	3
2.1.4 Software Interfaces.....	3
2.1.5 Communications Interfaces.....	3
2.1.6 Memory Constraints.....	3
2.1.7 Operations.....	4
2.1.8 Site Adaptation Requirements.....	4
2.2 Product Functions.....	4
2.3 User Characteristics.....	4
2.4 Constraints.....	4
2.5 Assumptions and Dependencies.....	4
2.6 Apportioning of Requirements.....	4
<b>3. Specific Requirements.....</b>	<b>5</b>
3.1 External Interface Requirements.....	5
3.1.1 User Interfaces.....	5
3.1.2 Hardware Interfaces.....	10
3.1.3 Software Interfaces.....	10
3.1.4 Communication Interfaces.....	10
3.2 Functional Requirements.....	10
3.3 Performance Requirements.....	26
3.4 Logical Database Requirements.....	26
3.5 Design Constraints.....	26
3.6 Software System Attributes.....	26
3.7 Other Requirements.....	27

## Revision History

Name	Date	Reason For Changes	Version

# 1. Introduction

A hospital is organized in different departments and each department specializes in certain treatments. A hospital has an administration section that manages patient registration, appointments, treatments.

A software is to be developed for automating the manual hospital patient management system. The system should be stand-alone in nature. The system operator must obtain a login ID and password from the "System administrator". After registration, every patient is able to access medical facility from the hospital. The software gives an effective way to manage appointments, in-patient department needs, maintenance of patient and hospital resources.

## 1.1 Purpose

The proposed software product is the Hospital Patient Management System (HPMS). The system will be used by the hospital management for allocating beds and doctors to patients. Doctors can use the system to keep track of their patients. Nurses who are in direct contact with the patients will use the system to keep track of available beds, the patients in the different wards. The HPMS also maintains records of all the doctors, patients, nurses and wards in the hospital.

## 1.2 Scope

The name of the software is Hospital Patient Management System (HPMS). The system will be referred to as HPMS in rest of the SRS. The proposed HPMS must be able to perform the following functions:

### Do's

1. Register new patients by generating unique patient ID for further tracking visits.
2. Facilitates effective scheduling of appointments of patients for the doctors.
3. Manages all Inpatient department needs along with the provision to manage admissions, discharges, allocation of bed and wards.
4. Manage Inpatient billing with details of patient information and services provided on daily basis.
5. Issue separate login for doctors, nurses, and administrative department based on roles.
6. Allocate doctor and nurses to the admitted patient.
7. Allocate bed or wards to the required patient.
8. Manage doctor, nurse and patient details.
9. Manage wards or beds details allotted to patient.
10. Nurses should be able to keep track of direct patients who need special attention.
11. Generate graphical reports like:
  - Details of all registered patient.
  - Details of Appointments & Schedulings.
  - Status of visiting doctors/patients.
  - Details of allocated beds/rooms/wards on daily basis.

### Don't

1. Lab details is not covered.
2. Digital payment is not covered.

### Benefits

The HPMS provides the following benefits:

1. Easy allocation of hospital resources.
2. Efficient appointment and scheduling.
3. Generation of graphical reports.

### 1.3 Definitions, Acronyms and Abbreviations

**SRS:** Software requirement specification

**HPMS:** Hospital patient management system

**System operator:** System administrator, receptionist, doctor, nurse, data entry operator

**RAM:** Random access memory

**Patient ID:** It is a unique sequence number allocated to each registered patient in the hospital.

**Bill ID:** It is a unique sequence number allocated to each generated bill.

**Patient:** Any person registered with the hospital.

**Doctor:** Person who view and treats the patient.

**Nurse:** Person who take care of the admitted patients.

**Hospital:** Administrative unit that consist of various departments.

**Wards:** A unit that consist of various rooms and beds.

**Administrator:** User having all the privileges to operate the HPMS.

### 1.4 References

- (a) IEEE Standard for Software Test Documentation-IEEE Std. 829-1998.
- (b) IEEE Recommended Practice for Software Requirements Specifications-IEEE Std 830-1998.

### 1.5 Overview

The rest of the SRS document describes various system requirements, interfaces, features and functionalities.

## 2. Overall Description

The HPMS maintains records of registered patients, doctors, nurses, wards in the hospital. The system administrator will receive the list of doctors and nurses available, wards and beds details from the administrative section.

The HPMS register patients and allocate doctor, nurse to the registered patients. It also allocate wards or beds to the required patients. The doctor/nurse/receptionist can access HPMS with their respective privileges on the hospital's LAN in order to access patient details from the hospital.

The administrator will have to maintain the following information:

- Patients details
- Doctor details
- Nurse details
- Wards details

The receptionist will perform the following functions:

- Patient registration
- Allocate doctor
- Allocate nurse
- Allocate wards/beds
- Maintain patient details
- Generate bill
- Generate reports

The doctor will perform the following functions:

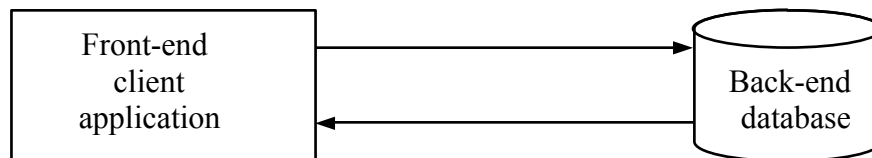
- View patients details
- Check patient reports
- Prescribe Medicine

The nurse will perform the following functions:

- View patients details
- View wards/beds details

## 2.1 Product Perspective

The HPMS shall be developed using client/server architecture and will be compatible with Microsoft Windows Operating System. The front-end of the system will be developed using Java 8 EE and the back-end will be developed using MySQL 6.3.8.



### 2.1.1 System Interfaces

System interfaces are addressed which are required by the product during its usage.

### 2.1.2 User Interfaces

The HPMS will have the following user-friendly and menu-driven interfaces:

- Login:** to allow the entry of only authorized users through valid login ID and password.
- Patient registration:** to allow receptionist to register patients to the hospital.
- Allocate doctor:** to allow receptionist to allocate doctor to the registered patient.
- Allocate nurse:** to allow receptionist to allocate nurse to the required patient.
- Allocate ward:** to allow receptionist to allocate allocate ward/beds to the required patient.
- Maintain patient details:** to maintain patient details.
- Maintain doctor details:** to maintain doctor details.
- Maintain nurse details:** to maintain nurse details.
- Maintain ward details:** to maintain ward details.
- Generate bill:** allow to generate bill.
- Generate reports:** allow to generate reports.

### 2.1.3 Hardware Interfaces

- Screen resolution of at least 640 x 480.
- Support for printer.
- Computer systems will be in the networked environment as it is a multi-user system.

### 2.1.4 Software Interfaces

- MS-Windows Operating System (XP/7/8/10)
- Java 8 EE for designing front-end
- MySQL 6.3.8 for back-end

### 2.1.5 Communications Interfaces

Communication is via local area network (LAN).

### 2.1.6 Memory Constraints

The minimum memory constraint required to run the HPMS software are:

- RAM - 512 MB
- Hard disk space - 500 MB

### 2.1.7 Operations

Various supported operations include:

- (a) Data processing support functions
- (a) Backup and recovery operations.

### 2.1.8 Site Adaptation Requirements

In the HPMS, the terminal at the client site will have to support the hardware and software interfaces specified in sections 2.1.3 and 2.1.4, respectively.

## 2.2 Product Functions

The HPMS will allow access only to the authorized users with specific roles (receptionist, doctor, nurse and administrator). Depending upon the user's role, he/she will be able to access only specific modules of the system.

A summary of major functions that the HPMS shall perform includes:

- A login facility for enabling only authorized access to the system.
- The system administrator will be able to maintain, add, modify, delete or view patient, doctor, nurse, wards and login information.
- The receptionist will be able to register patient and maintain patient details.
- The doctor will be able to view patient details.
- The nurse will be able to view patient and ward details.
- The receptionist will be able to generate bills.
- The receptionist will be able to generate various graphical reports from the HPMS.

## 2.3 User Characteristics

Qualification: At least matriculation and comfortable with English.

Experience: Should be well versed about the processes of the hospital management.

Technical experience: Elementary knowledge of computers.

## 2.4 Constraints

- There will be only one administrator.
- The delete operation is available to the administrator. To reduce the complexity of the system, there is no check on delete operation. Hence, the administrator should be very careful before deletion of any record and he/she will be responsible for data consistency.
- The user will not be allowed to update the primary key.

## 2.5 Assumptions and Dependencies

- The administrative section of the hospital will provide the list of available doctors, nurse, wards/beds.
- The login ID and password must be created by the system administrator and communicated to the concerned user confidentially to avoid unauthorized access to the system.

## 2.6 Apportioning of Requirements

- Digital payment mode.
- Lab details.

### 3. Specific Requirements

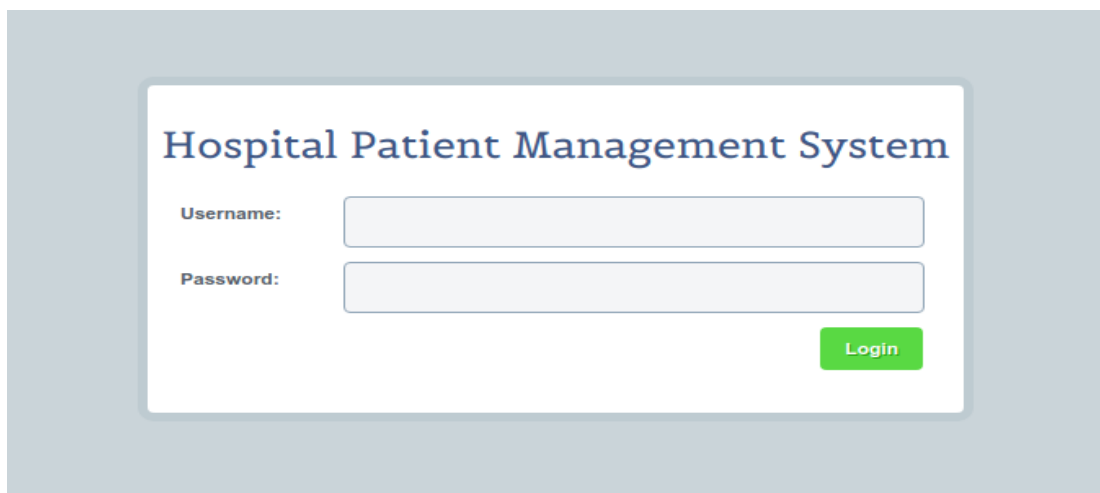
This section contains the software requirements in details along with the various forms to be developed.

#### 3.1 External Interface Requirements

##### 3.1.1 User Interfaces

###### Login

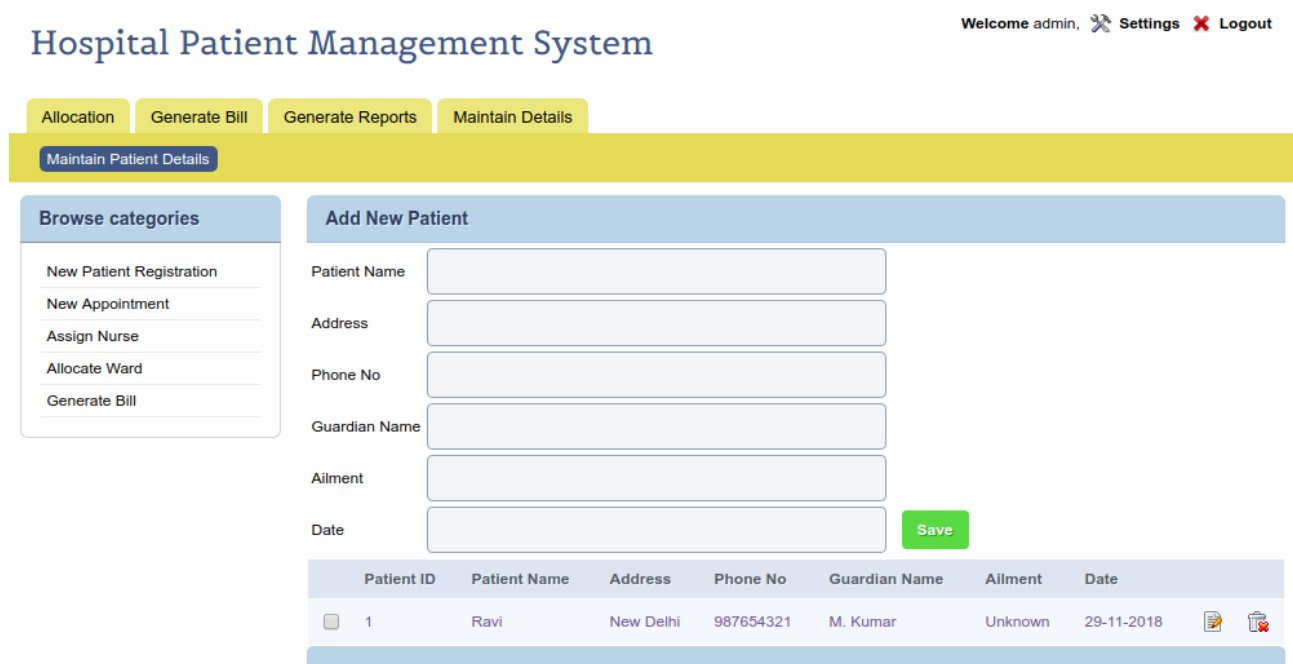
This form will be accessible to system administrator, doctor, nurse, receptionist. It will allow the entry of only authorized users through valid login ID and password.



The login form is titled "Hospital Patient Management System". It features two input fields: "Username:" and "Password:". Below the password field is a green "Login" button. The form is set against a light blue background.

###### Patient registration

This form will be accessible to system administrator, receptionist to register patients to the hospital.





The patient registration interface includes a header with the title "Hospital Patient Management System" and user information "Welcome admin, ⚙ Settings ✖ Logout". Below the header are four tabs: "Allocation", "Generate Bill", "Generate Reports", and "Maintain Details". The "Maintain Patient Details" tab is active.

On the left is a "Browse categories" sidebar with the following options: "New Patient Registration", "New Appointment", "Assign Nurse", "Allocate Ward", and "Generate Bill".

The main area is titled "Add New Patient" and contains the following form fields: "Patient Name", "Address", "Phone No", "Guardian Name", "Ailment", and "Date". A green "Save" button is located at the bottom right of the form.

Below the form is a table displaying patient records:

	Patient ID	Patient Name	Address	Phone No	Guardian Name	Ailment	Date	
<input type="checkbox"/>	1	Ravi	New Delhi	987654321	M. Kumar	Unknown	29-11-2018	 

### Allocate doctor

This form will be accessible to system administrator, receptionist to allocate doctor to the registered patient.

Hospital Patient Management System
Welcome admin, Settings Logout

Allocation
Generate Bill
Generate Reports
Maintain Details

Appointment

Browse categories

Appointment
Assign Nurse
Allocate Ward

Info

Allocate Resource

Create New Appointment

Patient ID
Patient Name
Doctor Name
Select Doctor
Date
Start Time
End Time
Booking Emp ID
Status
Create Appointment

Patient ID	Patient Name	Doctor Name	Date	Start Time	End Time	Booking EmpID	Status
1	S. Kumar	V Kumar	29-11-2018	10.00 AM	10.30 AM	1	Closed
2	Vinay Prakash	R S Kumar	04-12-2018	11.00 AM	11.30 AM	1	Open

### Allocate nurse

This form will be accessible to system administrator, receptionist to allocate nurse to the required patient.

Hospital Patient Management System
Welcome admin, Settings Logout

Allocation
Generate Bill
Generate Reports
Maintain Details

Assign Nurse

Browse categories

Appointment
Assign Nurse
Allocate Ward

Info

Allocate Resource

Assign Nurse

Patient ID
Patient Name
Ward No
Nurse Name
Select Nurse
Nurse ID
Status
Assign Nurse

ID	Patient ID	Patient Name	Ward No	Nurse ID	Nurse Name	Status
1	1	Ravi	N101	1	Shiny	Open



**Allocate ward**

This form will be accessible to system administrator, receptionist to allocate allocate ward/beds to the required patient.

Hospital Patient Management System

Welcome admin, [Settings](#) [Logout](#)

Allocation
Generate Bill
Generate Reports
Maintain Details

Allocate Ward

**Browse categories**

- Appointment
- Assign Nurse
- Allocate Ward

**Info**

- Allocate Resource

Allocate New Ward

Patient ID

Patient Name

Ward Type

Ward No

Rate

Days

Booking Emp ID

Status

Allocate Ward

Patient ID	Patient Name	Ward Type	Ward No	Rate	Days	Booking EmpID	Status
1	Ravi	Single Seater	N101	2000	2	1	Open

**Maintain patient details**

This form will be accessible to system administrator, receptionist to maintain patient details.

Hospital Patient Management System

Welcome admin, [Settings](#) [Logout](#)

Allocation
Generate Bill
Generate Reports
Maintain Details

Maintain Patient Details

**Browse categories**

- New Patient Registration
- New Appointment
- Assign Nurse
- Allocate Ward
- Generate Bill

Add New Patient

Patient Name

Address

Phone No

Guardian Name

Ailment

Date

Save

Patient ID	Patient Name	Address	Phone No	Guardian Name	Ailment	Date	
<input type="checkbox"/> 1	Ravi	New Delhi	987654321	M. Kumar	Unknown	29-11-2018	

## Maintain doctor details

This form will be accessible to system administrator to maintain doctor details.

### Hospital Patient Management System

Welcome admin, [Settings](#) [Logout](#)

[Allocation](#) [Generate Bill](#) [Generate Reports](#) [Maintain Details](#)

[Maintain Doctor Details](#)

#### Browse categories

- Maintain Doctor
- Maintain Nurse
- Maintain Ward
- Maintain Patient
- Maintain Treatment Type
- Maintain Ward Type

#### Info

Maintain Details.

#### Add New Doctor

Doctor Name

Speciality

Location

Days

Time

Save

	Doctor ID	Doctor Name	Speciality	Location	Days	Time		
<input type="checkbox"/>	2	V Kumar	Neurosurgeon	OPD	MoWeFr	9 AM - 1 PM		
<input type="checkbox"/>	3	R S Kumar	Cardio	OPD2	MoTuWe	10 AM - 2 PM		
<input type="checkbox"/>	4	A N Kapoor	General Physician	OPD1	MoTuWeThFrSaSu	10 AM - 4 PM		

## Maintain nurse details

This form will be accessible to system administrator to maintain nurse details.

### Hospital Patient Management System

Welcome admin, [Settings](#) [Logout](#)

[Allocation](#) [Generate Bill](#) [Generate Reports](#) [Maintain Details](#)

[Maintain Nurse Details](#)

#### Browse categories

- Maintain Doctor
- Maintain Nurse
- Maintain Ward
- Maintain Patient
- Maintain Treatment Type
- Maintain Ward Type

#### Info

Maintain Details.

#### Add New Nurse

Nurse Name

Department

Days

Visiting Time

Save

	Nurse ID	Nurse Name	Department	Days	Visiting Time		
<input type="checkbox"/>	1	Shiny	Neuro	MoTuWeTh	10 AM - 5 PM		

## Maintain ward details

This form will be accessible to system administrator and nurse to maintain ward details.

Hospital Patient Management System
Welcome admin, Settings Logout

Allocation
Generate Bill
Generate Reports
Maintain Details

Maintain Ward Details

Browse categories

- Maintain Doctor
- Maintain Nurse
- Maintain Ward
- Maintain Patient
- Maintain Treatment Type
- Maintain Ward Type

Add New Ward

Ward No
Ward Name
Department
Location
Incharge
Rate/Day
Status

Save

	Ward No	Ward Name	Ward Type	Location	Incharge	Rate/Day	Status		
<input type="checkbox"/>	N101	Alpha	Neuro	First Floor	Vijay	5000	Open		
<input type="checkbox"/>	N201	Beta	Cardio	Second Floor	Ramesh	5000	Open		

## Generate bill

This form will be accessible to system administrator and receptionist to generate bill.

Hospital Patient Management System
Welcome admin, Settings Logout

Allocation
Generate Bill
Generate Reports
Maintain Details

Generate Bills

Info

- Generate Bills

Create New Bill

Patient ID
Patient Name
Bill Type
Treatment Type
TAmount
Ward Type
Ward Rate/Day
Days

Generate Bill

Bill ID	Patient Name	Treatment	TAmount	Ward Type	Ward Charges	Days
1	S. Kumar	General Consultation	500	Single Seater	2000	2

## Generate reports

This form will be accessible to system administrator and receptionist to generate reports.

The screenshot displays the HPMS web interface. At the top, the title 'Hospital Patient Management System' is on the left, and 'Welcome admin, Settings Logout' is on the right. Below the title is a navigation bar with tabs: 'Allocation', 'Generate Bill', 'Generate Reports' (selected), and 'Maintain Details'. Under the 'Generate Reports' tab, there is a 'Generate Reports' button. The main content area is divided into two columns. The left column has a 'Browse categories' section with a 'Manage Reports' input field, and a 'Text Section' with a 'Manages Reports' input field. The right column is titled 'Reports' and contains a table with two columns: 'Doctor' and 'Current Appointment'. The table lists three doctors: A N Kapoor (appointment 1), R S Kumar (appointment 2), and V Kumar (appointment 2).

Reports	
Doctor	Current Appointment
A N Kapoor	1
R S Kumar	2
V Kumar	2

### 3.1.2 Hardware Interfaces

As stated in section 2.1.3

### 3.1.3 Software Interfaces

As stated in section 2.1.4

### 3.1.4 Communication Interfaces

As stated in section 2.1.5

## 3.2 Functional Requirements

LOGIN	
A. Use CaseDescription	
1.	<b>Introduction:</b> This usecase documents the steps that must follow in order to log into the HMPS.
2.	<b>Actors:</b> Receptionist, Administrator, Doctor, Nurse
3.	<b>Precondition:</b> The user must have a valid login ID and password.
4.	<b>Postcondition:</b> If this use case is successful, the actor is logged into the system. If not, the system remains unchanged.
5.	<p><b>Flow of events:</b></p> <p><b>5.1. Basic Flow:</b></p> <p>Starts when the actor wishes into login into the HPMS.</p> <ol style="list-style-type: none"> <li>1. The system requests that the actor specify the function he/she would like to perform (either Login, Change Password).</li> <li>2. Once the actor provides the requested information, one of the following flows is executed: <ul style="list-style-type: none"> <li>• If the actor selects “Login”, the <b>Login</b> flow is executed.</li> <li>• If the actor selects “Change Password”, the <b>Change Password</b> flow is executed.</li> </ul> </li> </ol> <p><b>Basic Flow 1: Login</b></p>

	<ol style="list-style-type: none"> <li>1. The system requests that the actor enter his/her login ID and password information.</li> <li>2. The actor enters his/her login ID and password.</li> <li>3. The actor enters into the system.</li> </ol> <p><b>Basic Flow 2: Change Password</b></p> <ol style="list-style-type: none"> <li>1. The system requests that the actor enter login ID, old password, new password and confirm the new password information.</li> <li>2. The actor enters login ID, old password, new password and confirms the new password information.</li> <li>3. The system validates the entered new password and password is confirmed.</li> </ol> <p><b>5.2. Alternative Flow:</b></p> <p><b>Alternative Flow 1: Invalid Login ID/Password</b> If in the Login flow, the actor enters an invalid login ID and/or password empty, the system displays an error message. The actor returns to the beginning of the basic flow.</p> <p><b>Alternative Flow 2: Invalid Entry</b> If in the Change Password flow, the actor enters an invalid login ID, old password, new password or the new password does not match with the confirm password, the system displays an error message. The actor returns to the beginning of the basic flow.</p> <p><b>Alternative Flow 3: User Exits</b> This allows the user to exit during the use case. The use case ends.</p>
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> None
<b>B. Validity Checks</b>	
	<ol style="list-style-type: none"> <li>i. Every user will have a unique login ID.</li> <li>ii. Login ID cannot be blank.</li> <li>iii. Login ID can only have 4 to 15 characters.</li> <li>iv. Login ID will not accept special characters and blank spaces.</li> <li>v. Password cannot be blank.</li> <li>vi. Length of password can only be 4 to 15 digits.</li> <li>vii. Alphabets, digits, hyphen and underscore characters are allowed in the password field.</li> <li>viii. Password will not accept blank spaces.</li> </ol>
<b>C. Sequencing Information</b>	
None	
<b>D. Error Handling/Response to Abnormal Situations</b>	
If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.	

<b>PATIENT REGISTRATION</b>	
<b>A. Use Case Description</b>	
1.	<b>Introduction:</b> This usecase documents the steps that the Receptionist must follow in order to register patient.
2.	<b>Actors:</b> Receptionist
3.	<b>Precondition:</b> The Receptionist must be logged onto the system before the use case begins.

4.	<b>Postcondition:</b> If the registration is successful, a patient ID must be allocated and the database is updated, else the system remains unchanged.
5.	<b>Flow of events:</b> <b>5.1. Basic Flow: New Patient Registration</b> <ol style="list-style-type: none"> <li>1. The patient details is entered manually.</li> <li>2. Patient is registered by allocating a patient ID to new patient.</li> <li>3. Billing details is generated and amount paid in cash.</li> <li>4. The new patient details is saved into the database.</li> </ol> <b>5.2. Alternative Flow:</b> <b>Alternative Flow 1: User Exits</b> This allow the user to exit at any time during the use case. The use case ends.
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login, Generate Bills
<b>B. Validity Checks</b>	
<ol style="list-style-type: none"> <li>i. The receptionist will be authorized to access the Patient Registration module.</li> <li>ii. Every patient will have unique patient ID.</li> <li>iii. Patient ID cannot be blank.</li> <li>iv. Patient ID can only have value from 10 to 9999 digits.</li> <li>v. Patient name cannot be blank.</li> <li>vi. Length of the name can be of 3 to 300 characters.</li> <li>vii. Relative name cannot be blank.</li> <li>viii. Phone number cannot be blank.</li> <li>ix. Phone can be upto 13 digits.</li> <li>x. Address cannot be blank.</li> <li>xi. Address can have length up to 10 to 200 characters.</li> <li>xii. Ailments cannot be blank.</li> <li>xiii. Doctor assigned cannot be blank.</li> <li>xiv. Visiting date cannot be blank.</li> </ol>	
<b>C. Sequencing Information</b>	
Doctor details should be available in the system.	
<b>D. Error Handling/Response to Abnormal Situations</b>	
If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.	

<b>ALLOCATE DOCTOR</b>	
<b>A. Use CaseDescription</b>	
1.	<b>Introduction:</b> This usecase documents the steps that the Receptionist must follow in order to allocate doctor to new or old patients.
2.	<b>Actors:</b> Receptionist
3.	<b>Precondition:</b> The Receptionist must be logged onto the system before the use case begins. Patient must be registered with hospital.

4.	<b>Postcondition:</b> If the use case is successful, doctor visiting hour is allocated to patient and the database is updated, else the system remains unchanged.
5.	<p><b>Flow of events:</b></p> <p><b>5.1. Basic Flow: Allocate Doctor</b></p> <ol style="list-style-type: none"> <li>1. Select the specified doctor.</li> <li>2. Empty slot with date for doctor is obtained.</li> <li>3. Doctor is allocated to the visiting patient and information is saved into the database.</li> </ol> <p><b>5.2. Alternative Flow:</b></p> <p><b>Alternative Flow 1: Doctor Not Available</b> If the required doctor is not found on particular date inform patient, "doctor not available" on that particular date and exit.</p> <p><b>Alternative Flow 2: Slot Not Available</b> If the required doctor is found then check for empty slot on particular date. If the empty slot is not found on particular date, inform patient, "slot not available" on that particular date and exit.</p> <p><b>Alternative Flow 3: User Exits</b> This allow the user to exit at any time during the use case. The use case ends.</p>
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login, Patient Registration
<b>B. Validity Checks</b>	
<ol style="list-style-type: none"> <li>i. Only the administrator/receptionist will be authorized to access the Allocate Doctor module.</li> <li>ii. Every patient will have a unique patient ID.</li> <li>iii. Patient ID cannot be blank.</li> <li>iv. Doctor ID cannot be blank.</li> <li>v. Ailment list cannot be blank.</li> <li>vi. Date cannot be blank.</li> </ol>	
<b>C. Sequencing Information</b>	
Patient and Doctor details should be available in the system.	
<b>D. Error Handling/Response to Abnormal Situations</b>	
If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.	

<b>ALLOCATE NURSE</b>	
<b>A. Use CaseDescription</b>	
1.	<b>Introduction:</b> This usecase module describe the flow of events for allocating nurses to the required patients.
2.	<b>Actors:</b> Receptionist
3.	<b>Precondition:</b> The Receptionist must be logged onto the system before the use case begins. Patient must be pre-allocated doctor.

4.	<b>Postcondition:</b> If the use case is successful, a nurse is assigned to the patient and the database is updated, else the system remains unchanged.
5.	<b>Flow of events:</b> <b>5.1 Basic Flow: Allocate Nurse</b> <ol style="list-style-type: none"> <li>1. Select from the available nurses in particular department.</li> <li>2. A nurse is allocated to the admitted patient and the information is saved.</li> </ol> <b>5.2. Alternative Flow: User Exits</b> This allow the user to exit at any time during the use case. The use case ends.
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login, Allocate Doctor
<b>B. Validity Checks</b>	
<ol style="list-style-type: none"> <li>i. Only the administrator/receptionist will be authorized to access the Allocate Nurse module.</li> <li>ii. Every patient will have a unique patient ID.</li> <li>iii. Patient ID cannot be blank.</li> <li>iv. Nurse ID cannot be blank.</li> <li>v. Date cannot be blank.</li> <li>vi. Ward list cannot be blank.</li> </ol>	
<b>C. Sequencing Information</b>	
Patient and Nurse details should be available in the system.	
<b>D. Error Handling/Response to Abnormal Situations</b>	
If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.	

<b>ALLOCATE WARDS</b>	
<b>A. Use CaseDescription</b>	
1.	<b>Introduction:</b> This use case module describes the flow of events for allocating beds or wards to the required patients.
2.	<b>Actors:</b> Receptionist
3.	<b>Precondition:</b> The Receptionist must be logged onto the system before the use case begins. Patient must be pre-allocated doctor and nurse. Bed or ward type that the patient need must be known prior to allocation.
4.	<b>Postcondition:</b> If the use case is successful, a ward is allocated to the patient and the database is updated, else the system remains unchanged.
5.	<b>Flow of events:</b> <b>5.1. Basic Flow: Allocate Ward</b> <ol style="list-style-type: none"> <li>1. Specified ward type is selected.</li> <li>2. The system display the list of available beds or wards.</li> <li>3. Ward is allocated as mentioned by patient and information is saved into the database.</li> </ol>



	<b>5.2. Alternative Flow:</b> <b>Alternative Flow 1: Ward Not Available</b> If the required bed or ward is not available, then the request for allocation is denied and use case ends. <b>Alternative Flow 2: User Exits</b> This allow the user to exit at any time during the use case. The use case ends.
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login
<b>B. Validity Checks</b>	
i. Only the administrator/receptionist will be authorized to access the Allocate Ward module. ii. Every patient will have a unique patient ID. iii. Patient ID cannot be blank. iv. Ward ID cannot be blank. v. Ward location cannot be blank. vi. Date of allocation cannot be blank. vii. Amount cannot be blank. viii. Days for allocation cannot be blank.	
<b>C. Sequencing Information</b>	
Patient and Ward details should be available in the system.	
<b>D. Error Handling/Response to Abnormal Situations</b>	
If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.	

MAINTAIN DOCTOR DETAILS	
A. Use CaseDescription	
1.	<b>Introduction:</b> This usecase documents the steps that the administrator must follow in order to maintain doctor details and add, update, delete and view doctor information.
2.	<b>Actors:</b> Administrator
3.	<b>Precondition:</b> The administrator must be logged onto the system before this use case begins.
4.	<b>Postcondition:</b> If the use case is successful, then the doctor information is added, updated, deleted or viewed. Otherwise, the system state is unchanged.
5.	<b>Flow of events:</b> <b>5.1. Basic Flow:</b> This use case starts when the administrator wishes to add/update/delete/view doctor information. <ol style="list-style-type: none"> <li>1. The system requests that the administrator specify the function he/she would like to perform (either add, update, delete or view a doctor).</li> <li>2. Once the administrator provides the requested information, one of the subflows is executed.</li> </ol>

- If the administrator selects “Add a Doctor”, the **Add a Doctor** subflow is executed.
- If the administrator selects “Update a Doctor”, the **Update a Doctor** subflow is executed.
- If the administrator selects “Delete a Doctor”, the **Delete a Doctor** subflow is executed.
- If the administrator selects “View a Doctor”, the **View a Doctor** subflow is executed.

#### **Basic Flow 1: Add a Doctor**

The system requests that the administrator enter the doctor information. This includes:

- Doctor ID
- Doctor Name
- Specialization
- Sitting Location
- Visiting Days
- Visiting Time

Once the administrator provides the requested information, the doctor is added to the system.

#### **Basic Flow 2: Update a Doctor**

1. The system requests that the administrator enter the Doctor ID.
2. The administrator enters the Doctor ID.
3. The system retrieves and display the doctor information.
4. The administrator makes the desired changes to the doctor information. This includes any of the information specified in the **Add a Doctor** subflow.
5. Once the administrator updates the necessary information, the system updates the doctor information with the updated information.

#### **Basic Flow 3: Delete a Doctor**

1. The system requests that the administrator specify the Doctor ID.
2. The administrator enters the Doctor ID. The system retrieves and display the doctor information.
3. The system prompts the administrator to confirm the deletion of the doctor record.
4. The administrator verifies the deletion.
5. The system deletes the record.

#### **Basic Flow 4: View a Doctor**

1. The system requests that the administrator specify the Doctor ID.
2. The system retrieves and displays the doctor information.

### **5.2. Alternative Flow:**

#### **Alternative Flow 1: Invalid Entry**

If in the **Add a Doctor** or **Update a Doctor** flow, the actor enters invalid Doctor ID/Doctor Name/Specialization/Sitting Location/Visiting Days/Visiting Time or leaves the Doctor ID/Doctor Name/Specialization/Sitting Location/Visiting Days/Visiting Time empty, the system displays an appropriate error message. The actor returns to the basic flow and may reenter the invalid entry.

#### **Alternative Flow 2: Doctor Already Exists**

If in the **Add a Doctor** flow, a doctor with a specified Doctor ID already exists, the system displays an error message. The administrator returns to the basic flow and may reenter the doctor.

#### **Alternative Flow 3: Doctor Not Found**

If in the **Update a Doctor** or **Delete a Doctor** or **View a Doctor** flow, the doctor

	<p>information with the specified Doctor ID does not exist, the system displays an error message. The administrator returns to the basic flow and may reenter the Doctor ID.</p> <p><b>Alternative Flow 4: Update Cancelled</b> If in the <b>Update a Doctor</b> flow, the administrator decides not to update the doctor, the update is cancelled and the <b>Basic Flow</b> is restarted at the beginning.</p> <p><b>Alternative Flow 5: Delete Cancelled</b> If in the <b>Delete a Doctor</b> flow, the administrator decides not to delete the doctor, the delete is cancelled and the <b>Basic Flow</b> is restarted at the beginning.</p> <p><b>Alternative Flow 6: User Exits</b> This allows the user to exit at any time during the use case. The use case ends.</p>
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login
<b>B. Validity Checks</b>	
	<ol style="list-style-type: none"> <li>Only the administrator will be authorized to access the Maintain Doctor Details module.</li> <li>Every doctor will have unique doctor ID.</li> <li>Doctor ID cannot be blank.</li> <li>Doctor ID can only have value from 10 to 9999 digits.</li> <li>Doctor name cannot be blank.</li> <li>Doctor specialization cannot be blank.</li> <li>Length of the name can be of 3 to 300 characters.</li> </ol>
<b>C. Sequencing Information</b>	
	None
<b>D. Error Handling/Response to Abnormal Situations</b>	
	If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.

MAINTAIN NURSE DETAILS	
A. Use Case Description	
1.	<b>Introduction:</b> This usecase documents the steps that the administrator must follow in order to maintain nurse details and add, update, delete and view nurse information.
2.	<b>Actors:</b> Administrator
3.	<b>Precondition:</b> The administrator must be logged onto the system before this use case begins.
4.	<b>Postcondition:</b> If the use case is successful, then the nurse information is added, updated, deleted or viewed. Otherwise, the system state is unchanged.
5.	<p><b>Flow of events:</b></p> <p><b>5.1. Basic Flow:</b></p> <ol style="list-style-type: none"> <li>This use case starts when the administrator wishes to add/update/delete/view nurse information.</li> <li>The system requests that the administrator specify the function he/she would like to perform (either add, update, delete or view a nurse).</li> <li>Once the administrator provides the requested information, one of the subflows is</li> </ol>

executed.

- If the administrator selects “Add a Nurse”, the **Add a Nurse** subflow is executed.
- If the administrator selects “Update a Nurse”, the **Update a Nurse** subflow is executed.
- If the administrator selects “Delete a Nurse”, the **Delete a Nurse** subflow is executed.
- If the administrator selects “View a Nurse”, the **View a Nurse** subflow is executed.

#### **Basic Flow 1: Add a Nurse**

The system requests that the administrator enter the nurse information. This includes:

- Nurse ID
- Nurse Name
- Department
- Visiting Days
- Visiting Time

Once the administrator provides the requested information, the nurse is added to the system.

#### **Basic Flow 2: Update a Nurse**

1. The system requests that the administrator enter the nurse ID.
2. The administrator enters the nurse ID.
3. The system retrieves and display the nurse information.
4. The administrator makes the desired changes to the nurse information. This includes any of the information specified in the **Add a Nurse** subflow.
5. Once the administrator updates the necessary information, the system updates the nurse information with the updated information.

#### **Basic Flow 3: Delete a Nurse**

1. The system requests that the administrator specify the nurse ID.
2. The administrator enters the nurse ID. The system retrieves and display the nurse information.
3. The system prompts the administrator to confirm the deletion of the nurse record.
4. The administrator verifies the deletion.
5. The system deletes the record.

#### **Basic Flow 4: View a Nurse**

1. The system requests that the administrator specify the nurse ID.
2. The system retrieves and displays the nurse information.

### **5.2. Alternative Flow:**

#### **Alternative Flow 1: Invalid Entry**

If in the **Add a Nurse** or **Update a Nurse** flow, the actor enters invalid Nurse ID/ Nurse Name/Specialization/Department/Visiting Days/Visiting Time or leaves the Nurse ID/ Nurse Name/Specialization/Department/Visiting Days/Visiting Time empty, the system displays an appropriate error message. The actor returns to the basic flow and may reenter the invalid entry.

#### **Alternative Flow 2: Nurse Already Exists**

If in the **Add a Nurse** flow, a nurse with a specified nurse ID already exists, the system displays an error message. The administrator returns to the basic flow and may reenter the nurse.

#### **Alternative Flow 3: Nurse Not Found**

If in the **Update a Nurse** or **Delete a Nurse** or **View a Nurse** flow, the nurse information with the specified nurse ID does not exists, the system displays an error message. The

	<p>administrator returns to the basic flow and may reenter the nurse ID.</p> <p><b>Alternative Flow 4: Update Cancelled</b> If in the <b>Update a Nurse</b> flow, the administrator decides not to update the nurse, the update is cancelled and the <b>Basic Flow</b> is restarted at the beginning.</p> <p><b>Alternative Flow 5: Delete Cancelled</b> If in the <b>Delete a Nurse</b> flow, the administrator decides not to delete the nurse, the delete is cancelled and the <b>Basic Flow</b> is restarted at the beginning.</p> <p><b>Alternative Flow 6: User Exits</b> This allows the user to exit at any time during the use case. The use case ends.</p>
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login
<b>B. Validity Checks</b>	
	<ol style="list-style-type: none"> <li>i. Only the administrator will be authorized to access the Maintain Nurse Details module.</li> <li>ii. Every nurse will have unique nurse ID.</li> <li>iii. Nurse ID cannot be blank.</li> <li>iv. Nurse ID can only have value from 10 to 9999 digits.</li> <li>v. Nurse name cannot be blank.</li> <li>vi. Visiting days cannot be blank.</li> <li>vii. Visiting time cannot be blank.</li> <li>viii. Length of the name can be of 3 to 300 characters.</li> </ol>
<b>C. Sequencing Information</b>	
	None
<b>D. Error Handling/Response to Abnormal Situations</b>	
	If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.

<b>MAINTAIN PATIENT DETAILS</b>	
<b>A. Use CaseDescription</b>	
1.	<b>Introduction:</b> This usecase documents the steps that the Administrator/Receptionist/Doctor/Nurse must follow in order to maintain patient details and add, update, delete and view patient information.
2.	<b>Actors:</b> Receptionist, Administrator, Doctor, Nurses
3.	<b>Precondition:</b> The user must be logged onto the system before this use case begins.
4.	<b>Postcondition:</b> If the use case is successful, then the patient information is added, updated, deleted or viewed. Otherwise, the system state is unchanged.
5.	<p><b>Flow of events:</b></p> <p><b>5.1. Basic Flow:</b> This use case starts when the administrator wishes to add/update/delete/view patient information.</p> <ol style="list-style-type: none"> <li>1. The system requests that the administrator specify the function he/she would like to perform (either add, update, delete or view a patient).</li> <li>2. Once the administrator provides the requested information, one of the subflows is</li> </ol>

executed.

- If the administrator selects “Add a Patient”, the **Add a Patient** subflow is executed.
- If the administrator selects “Update a Patient”, the **Update a Patient** subflow is executed.
- If the administrator selects “Delete a Patient”, the **Delete a Patient** subflow is executed.
- If the administrator/Receptionist/Doctor/Nurses selects “View a Patient”, the **View a Patient** subflow is executed.

#### **Basic Flow 1: Add a Patient**

The system requests that the administrator enter the patient information. This includes:

- Patient ID
- Patient Name
- Relative Name
- Phone No
- Address
- Ailments
- Doctor Assigned
- Visiting Date

Once the administrator provides the requested information, the patient is added to the system.

#### **Basic Flow 2: Update a Patient**

1. The system requests that the administrator enter the patient ID.
2. The administrator enters the patient ID.
3. The system retrieves and display the patient information.
4. The administrator makes the desired changes to the patient information. This includes any of the information specified in the **Add a Patient** subflow.
5. Once the administrator updates the necessary information, the system updates the patient information with the updated information.

#### **Basic Flow 3: Delete a Patient**

1. The system requests that the administrator specify the patient ID.
2. The administrator enters the patient ID. The system retrieves and display the patient information.
3. The system prompts the administrator to confirm the deletion of the patient record.
4. The administrator verifies the deletion.
5. The system deletes the record.

#### **Basic Flow 4: View a Patient**

1. The system requests that the Administrator/Receptionist/Doctor/Nurses specify the patient ID.
2. The system retrieves and displays the patient information.

### **5.2. Alternative Flow:**

#### **Alternative Flow 1: Invalid Entry**

If in the **Add a Patient** or **Update a Patient** flow, the actor enters invalid Patient ID/Patient Name/Relative Name/Phone No/Address/Ailments/Doctor Assigned/Visiting Date or leaves the Patient ID/Patient Name/Relative Name/Phone No/Address/Ailments/Doctor Assigned/Visiting Date empty, the system displays an appropriate error message. The actor returns to the basic flow and may reenter the invalid entry.

#### **Alternative Flow 2: Patient Already Exists**

If in the **Add a Patient** flow, a patient with a specified patient ID already exists, the system

	<p>displays an error message. The administrator returns to the basic flow and may reenter the patient.</p> <p><b>Alternative Flow 3: Patient Not Found</b>          If in the <b>Update a Patient</b> or <b>Delete a Patient</b> or <b>View a Patient</b> flow, the patient information with the specified patient ID does not exist, the system displays an error message. The administrator returns to the basic flow and may reenter the patient ID.</p> <p><b>Alternative Flow 4: Update Cancelled</b>          If in the <b>Update a Patient</b> flow, the administrator decides not to update the patient, the update is cancelled and the <b>Basic Flow</b> is restarted at the beginning.</p> <p><b>Alternative Flow 5: Delete Cancelled</b>          If in the <b>Delete a Patient</b> flow, the administrator decides not to delete the patient, the delete is cancelled and the <b>Basic Flow</b> is restarted at the beginning.</p> <p><b>Alternative Flow 6: User Exits</b>          This allows the user to exit at any time during the use case. The use case ends.</p>
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login
<b>B. Validity Checks</b>	
	<ul style="list-style-type: none"> <li>i. The administrator will be fully authorized to access the Maintain Patient Details module.</li> <li>ii. Receptionist will be authorized to access to the Maintain Patient Details module with add, update and view operations.</li> <li>iii. Doctor will be authorized to access the Maintain Patient Details module with view operations.</li> <li>iv. Nurse will be authorized to access the Maintain Patient Details module with view operations.</li> <li>v. Every patient will have unique patient ID.</li> <li>vi. Patient ID cannot be blank.</li> <li>vii. Patient ID can only have value from 10 to 9999 digits.</li> <li>viii. Patient name cannot be blank.</li> <li>ix. Length of the name can be of 3 to 300 characters.</li> <li>x. Relative name cannot be blank.</li> <li>xi. Phone number cannot be blank.</li> <li>xii. Phone can be upto 13 digits.</li> <li>xiii. Address cannot be blank.</li> <li>xiv. Address can have length up to 10 to 200 characters.</li> <li>xv. Ailments cannot be blank.</li> <li>xvi. Doctor assigned cannot be blank.</li> <li>xvii. Visiting date cannot be blank.</li> </ul>
<b>C. Sequencing Information</b>	
None	
<b>D. Error Handling/Response to Abnormal Situations</b>	
If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.	

MAINTAIN WARD DETAILS	
A. Use CaseDescription	
1.	<b>Introduction:</b> This usecase documents the steps that the administrator must follow in order to maintain ward details and add, update, delete and view ward information.
2.	<b>Actors:</b> Administrator, Nurse
3.	<b>Precondition:</b> The Administrator/Nurses must be logged onto the system before this use case begins.
4.	<b>Postcondition:</b> If the use case is successful, then the ward information is added, updated, deleted or viewed. Otherwise, the system state is unchanged.
5.	<p><b>Flow of Events:</b></p> <p><b>5.1. Basic Flow:</b> This use case starts when the administrator wishes to add/update/delete/view ward information.</p> <ol style="list-style-type: none"> <li>1. The system requests that the administrator specify the function he/she would like to perform (either add, update, delete or view a ward).</li> <li>2. Once the administrator provides the requested information, one of the subflows is executed. <ul style="list-style-type: none"> <li>• If the administrator selects “Add a Ward”, the <b>Add a Ward</b> subflow is executed.</li> <li>• If the administrator selects “Update a Ward”, the <b>Update a Ward</b> subflow is executed.</li> <li>• If the administrator selects “Delete a Ward”, the <b>Delete a Ward</b> subflow is executed.</li> <li>• If the administrator/Nurse selects “View a Ward”, the <b>View a Ward</b> subflow is executed.</li> </ul> </li> </ol> <p><b>Basic Flow 1: Add a Ward</b> The system requests that the administrator enter the ward information. This includes:</p> <ul style="list-style-type: none"> <li>• Ward No.</li> <li>• Ward Name</li> <li>• Depatment</li> <li>• Location</li> <li>• Incharge</li> <li>• Status</li> <li>• Rate</li> </ul> <p>Once the administrator provides the requested information, the ward is added to the system.</p> <p><b>Basic Flow 2: Update a Ward</b></p> <ol style="list-style-type: none"> <li>1. The system requests that the administrator enter the ward No.</li> <li>2. The administrator enters the ward No.</li> <li>3. The system retrieves and display the ward information.</li> <li>4. The administrator makes the desired changes to the ward information. This includes any of the information specified in the <b>Add a Ward</b> subflow.</li> <li>5. Once the administrator updates the necessary information, the system updates the ward information with the updated information.</li> </ol> <p><b>Basic Flow 3: Delete a Ward</b></p> <ol style="list-style-type: none"> <li>1. The system requests that the administrator specify the ward No.</li> <li>2. The administrator enters the ward No. The system retrieves and display the ward</li> </ol>



	<p>information.</p> <ol style="list-style-type: none"> <li>The system prompts the administrator to confirm the deletion of the ward record.</li> <li>The administrator verifies the deletion.</li> <li>The system deletes the record.</li> </ol> <p><b>Basic Flow 4: View a Ward</b></p> <ol style="list-style-type: none"> <li>The system requests that the administrator/Nurse specify the ward No.</li> <li>The system retrieves and displays the ward information.</li> </ol> <p><b>5.2. Alternative Flow:</b></p> <p><b>Alternative Flow 1: Invalid Entry</b>          If in the <b>Add a Ward</b> or <b>Update a Ward</b> flow, the actor enters invalid ward No/ ward Name/Department/Location/Incharge/Status or leaves the ward No/ ward Name/Department/Location/Incharge/Status empty, the system displays an appropriate error message. The actor returns to the basic flow and may reenter the invalid entry.</p> <p><b>Alternative Flow 2: Ward Already Exists</b>          If in the <b>Add a Ward</b> flow, a ward with a specified ward No already exists, the system displays an error message. The DOE returns to the basic flow and may reenter the ward details.</p> <p><b>Alternative Flow 3: Ward Not Found</b>          If in the <b>Update a Ward</b> or <b>Delete a Ward</b> or <b>View a Ward</b> flow, the ward information with the specified ward No does not exist, the system displays an error message. The administrator returns to the basic flow and may reenter the ward No.</p> <p><b>Alternative Flow 4: Update Cancelled</b>          If in the <b>Update a Ward</b> flow, the administrator decides not to update the ward, the update is cancelled and the <b>Basic Flow</b> is restarted at the beginning.</p> <p><b>Alternative Flow 5: Delete Cancelled</b>          If in the <b>Delete a Ward</b> flow, the administrator decides not to delete the ward, the delete is cancelled and the <b>Basic Flow</b> is restarted at the beginning.</p> <p><b>Alternative Flow 6: User Exits</b>          This allows the user to exit at any time during the use case. The use case ends.</p>
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login
<b>B. Validity Checks</b>	
	<ol style="list-style-type: none"> <li>The administrator will be fully authorized to access the Maintain Ward Details module.</li> <li>Nurse will be authorized to access the Maintain Ward Details module with only view operation.</li> <li>Every ward will have a unique ward ID.</li> <li>Ward ID cannot be blank.</li> <li>Ward ID can only have value from 10 to 9999 digits.</li> <li>Ward name cannot be blank.</li> <li>Ward location cannot be blank.</li> <li>Department cannot be blank.</li> <li>Ward Incharge cannot be blank.</li> <li>Status will be boolean value – booked/empty.</li> </ol>
<b>C. Sequencing Information</b>	
	None
<b>D. Error Handling/Response to Abnormal Situations</b>	

If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.

GENERATE BILLS	
A. Use CaseDescription	
1.	<b>Introduction:</b> This usecase module describe the flow of events for generating bills.
2.	<b>Actors:</b> Receptionist, Administrator
3.	<b>Precondition:</b> The user must be logged onto the system before the use case begins. Patient must be registered.
4.	<b>Postcondition:</b> If the use case is successful, a bill is generated and the database is updated, else the system remains unchanged.
5.	<b>Flow of events:</b> <b>5.1 Basic Flow: Generate Bill</b> <ol style="list-style-type: none"> <li>1. A bill amount is calculated based on the kind of treatment.</li> <li>2. A bill is generated with a unique bill ID.</li> <li>3. Bill details along with patient ID is saved in the database.</li> </ol> <b>5.2. Alternative Flow: User Exits</b> This allow the user to exit at any time during the use case. The use case ends.
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login
B. Validity Checks	
i.	The administrator/receptionist will be authorized to access the Generate Bill module.
ii.	Every bill will have a unique bill ID.
C. Sequencing Information	
None	
D. Error Handling/Response to Abnormal Situations	
If any of the validation flows does not hold true, appropriate error message will be prompted to the user for doing the needful.	

GENERATE REPORTS	
A. Use CaseDescription	
1.	<b>Introduction:</b> This usecase module describe the flow of events for generating reports.
2.	<b>Actors:</b> Receptionist
3.	<b>Precondition:</b> The user must be logged onto the system before the use case begins.
4.	<b>Postcondition:</b> If the use case is successful, the system displays the details based on selected operations. No changes are made to the database. Else, no report is displayed.

5.	<b>Flow of events:</b> <b>5.1 Basic Flow: Generate Reports</b> <ol style="list-style-type: none"> <li>The operator issues the command to generate the below reports. <ul style="list-style-type: none"> <li>Details of all registered patients.</li> <li>Details of appointments.</li> <li>Details of allocated beds/wards.</li> </ul> </li> <li>The system displays a report including details of all the operations.</li> </ol> <b>5.2. Alternative Flow</b> None
6.	<b>Special requirements:</b> None
7.	<b>Associated use cases:</b> Login

### 3.3 Performance Requirements

- Should support at least 10 terminals.
- Should support at least 10 user simultaneously.
- Should run on 500 Mhz, 512 MB RAM machine
- Responses should be within 2 seconds.

### 3.4 Logical Database Requirements

The following information will be placed in a database.

Table Name	Description
MaintainPatient	Records the details of the patients in the hospital.
MaintainDoctor	Records the details of the doctors in the hospital.
MaintainNurse	Records the details of the nurses in the hospital.
MaintainWard	Records the details of the wards in the hospital.
Login	Records the login details of the user.
Bill	Records the bills generated in the hospital.
Users	Records employee details.
Department	Records the details of the different departments.
Appointment	Records of the appointment.
AssignNurse	Records of the nurse scheduled to patient.
AllocateWard	Records of the allocated wards.

### 3.5 Design Constraints

None

### **3.6 Software System Attributes**

#### **Usability**

The application will be user-friendly and easy to operate and the functions will be easily understandable.

#### **Reliability**

The application have a high degree of fault tolerance.

#### **Security**

The application will be password protected. User will have to enter correct login ID and password to access the application.

#### **Maintainability**

The application will be designed in a maintainable manner. It will be easy to incorporate new requirements in the individual modules.

#### **Portability**

The application will be easily portable on any windows-based system that has SQL Server installed.

### **3.7 Other Requirements**

None