# ***Ahsanullah University of Science & Technology***

Department of Computer Science & Engineering



**Hospital Management System**

CSE 3224

Information System Design & Software Engineering Lab

Submitted By:

Shadman Sakib 16.01.04.092

A.K.M. Sadman Ali 16.01.04.093

Tasmia Tuj Juha 16.01.04.095

Mehedi Hasan Abir 16.01.04.102

Date of Submission: **05 March, 2019**

**Contents:**

1) Project Motivation ……………………………………………………………………………3

2) Actors……………………………………………………………………………………………….4

3) Use Case According to Primary Actors………………………………….…………….5

5) Class Diagram……………………………………………………………………………………9

6) Classes Explanation…………………………………………………………………………

**Project Motivation:**

The management of most hospitals of our countries are slow, inefficient and too reliant on paper-based information. Employees struggle to cope with their daily tasks due to the workloads. Crowdy atmosphere prevails due to haphazard appointment scheduling. A lot of hand-written work have to be done, which consumes a lot of time.

Having done a survey we have realized that most people are dissatisfied with the current system. It seemed apparent that people would hugely appreciate a system which allowed features like online appointment-making, digital prescriptions and digital reports.

The unfavorable condition of existing systems along with the huge dissatisfaction from people acted as our main motivation for this project. Since the existing systems are clearly unfavorable for people, we provide a new online system which targets to give a solution to the existing management problems by providing fast and efficient management online. It aims to minimize hand-written works and to take care of the essential functionalities of the hospital. It will be a huge time saver and will facilitate proper communication amongst management, staff and patient parties. By providing a user-friendly website, our project will help the employees and patients by saving their valuable time, and will also provide them with more functionalities and comfort.

**Actors:**

Every system has some actors which can be a person, organization, or a subsystem. These actors play vital role in the system. There are two kinds of actors for every system.

These two types of actors in our system are described below:

**1) Primary Actors:**

They are the actors who stimulate the system and initiate various events. They use the system’s main functions to achieve a goal. The primary actors of our project are:

* **Visitors:** They are the unregistered users of the system. They can view certain sections of the site but they have to register to use the system.
* **Admins:** They will be responsible for the management of the website.
* **Patients:** They are the registered users of the website. They can perform many activities to initiate the system.
* **Doctors:** They are the registered users of the website. They can perform many activities to initiate the system.

**2) Secondary Actors:**

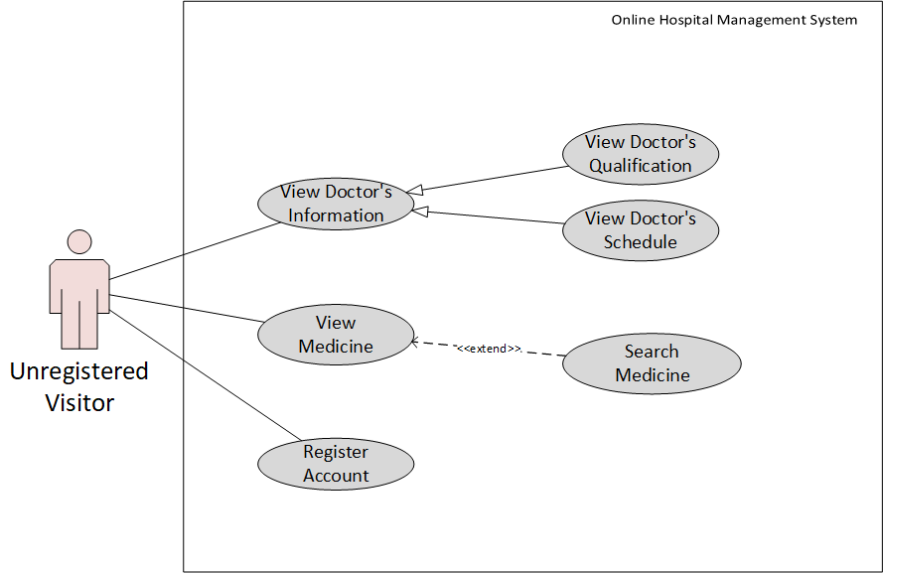
They are the actors that only receive stimuli from the system. They mainly support the system to achieve primary actor’s goals. The secondary actors of our project are:

* **Payment Service:** The service will receive stimuli from the system whenever a customer pays through online credit card system.
* **Cashier:** This actor will be active whenever a customer pays physically at the hospital counter.
* **Lab Employee:** This actor will be active whenever a patient takes tests in the hospital.

**Use cases according to the primary actors:**

Our system has four primary actors. These three actors have different use cases according to the goals they want to achieve. Their use case diagrams are described in the further sections.

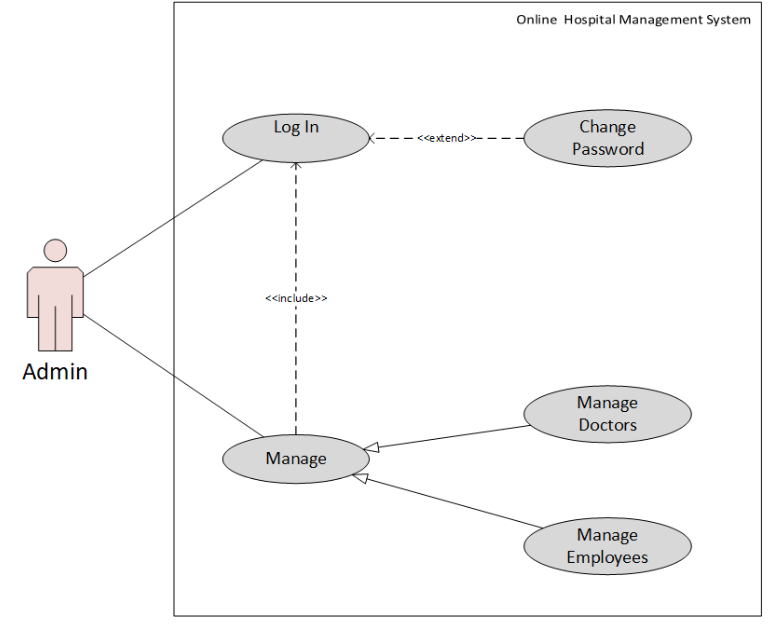
**1)Use Case Diagram for Visitors:**



From the above diagram we can see that a visitor can-

* View doctor related information such as doctor’s qualifications and schedule.
* View medicine related information.
* Register to the website by creating an account.

**2)Use Case Diagram for Admins:**

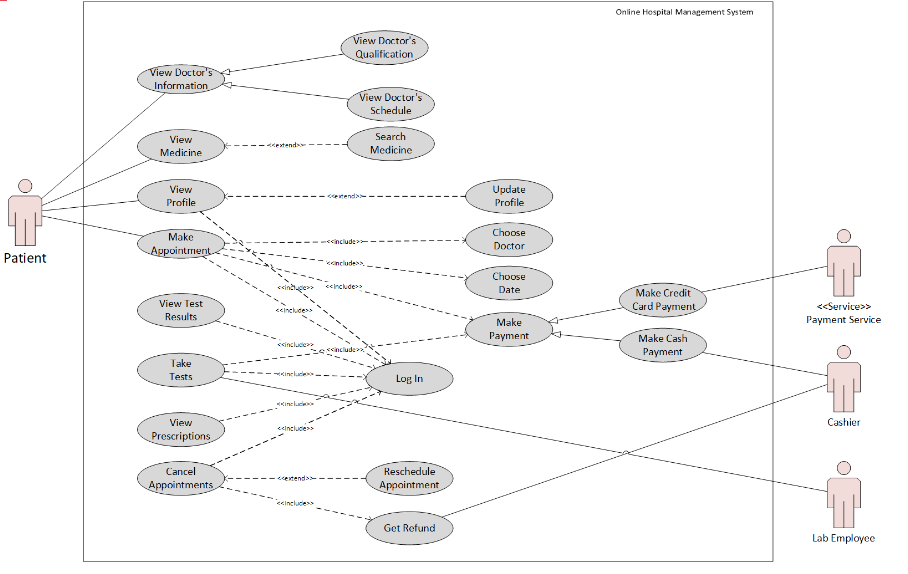


From the above diagram we can see that an admin can-

* Log into his/her account for the maintenance of the website
* Manage the website after logging into account
* Managing will include:

1. Managing Doctors
2. Managing Employees

**3)Use Case Diagram for Patients:**



From the above diagram we can see that-

* Patients can view doctor and medicine related information just like a visitor.
* They can view their own profile after log in. They can also update it if they want to.
* Patients can make appointments via the following procedure:

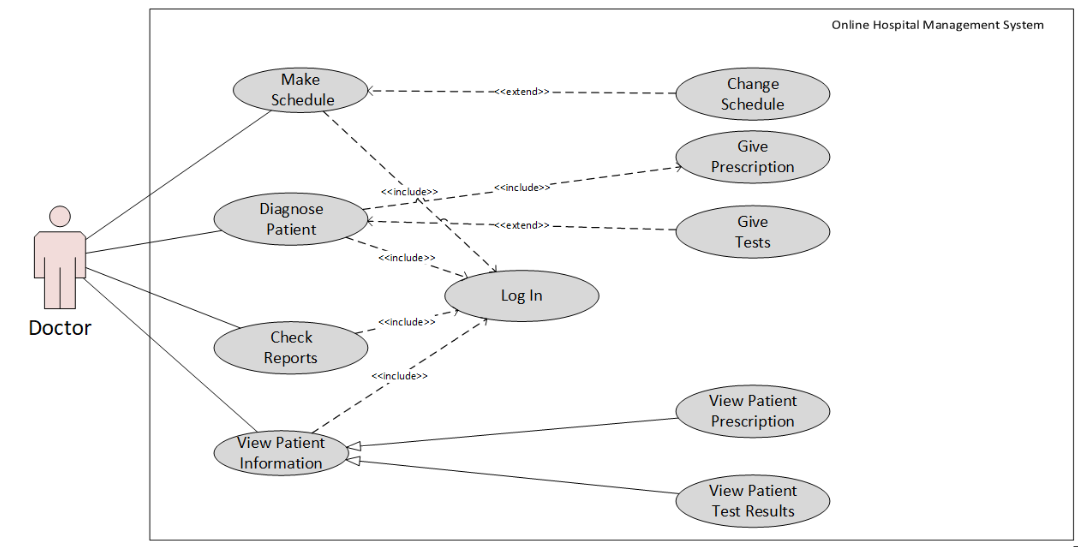
1. Choose doctor
2. Choose date
3. Make Payment

* Patients can make payments in the following ways:

1. Via Credit Card
2. Via Cash

* Payment Via Credit Card is associated with the secondary actor: Payment System. And Payment Via Cash is associated with the secondary actor: Cashier.
* Patients can take tests, which also requires payment. The use case Take Test is also associated with the secondary actor: Lab Employee.
* They can cancel their appointment if it’s necessary. In that case, they will get refund. They can also reschedule the appointment if they want to.
* They can view all their prescriptions and test results.
* Patients must log in first in order to do the above-mentioned activities

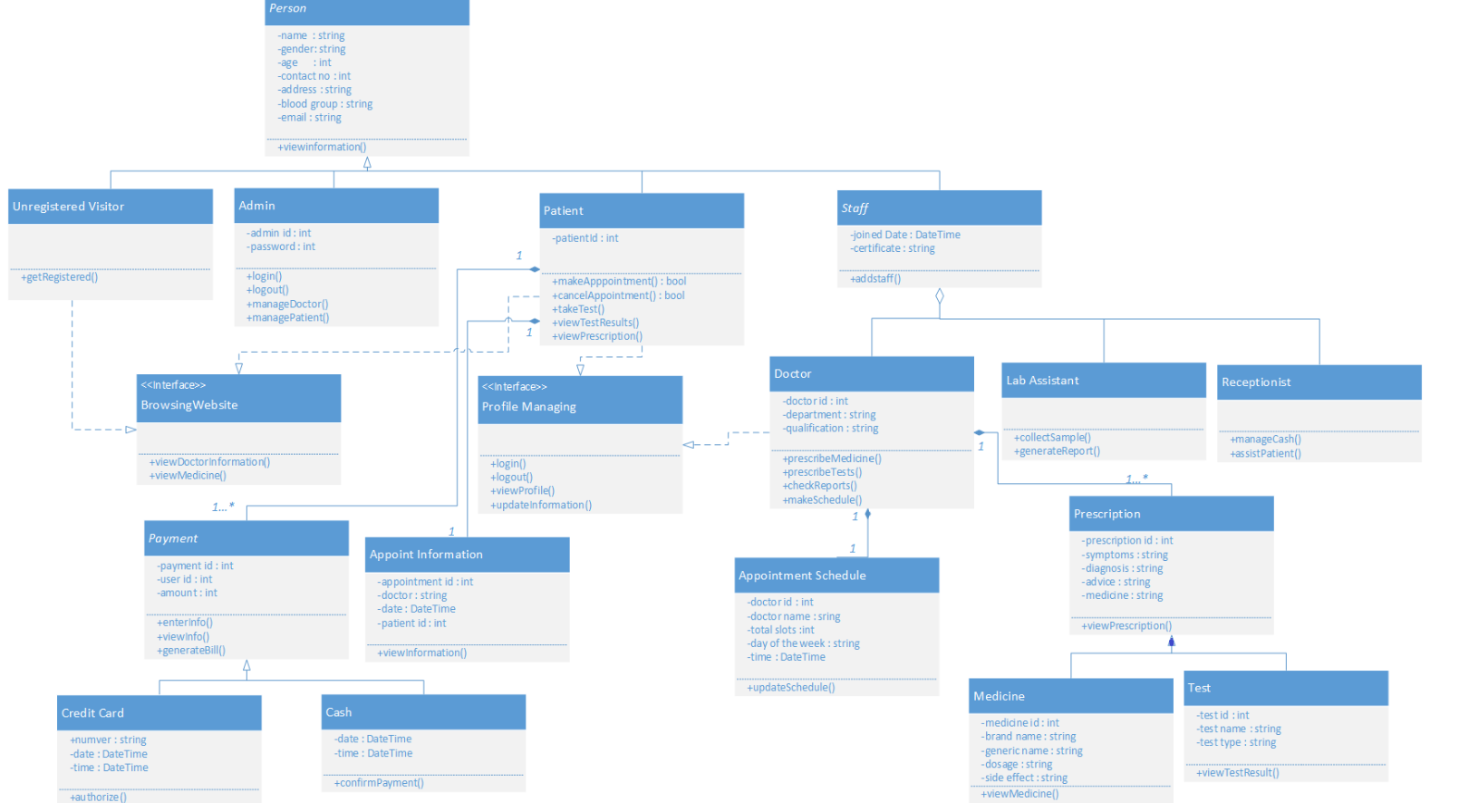
**4)Use Case Diagram for Doctors:**

****

From the above diagram we can see that-

* Doctors can make their own schedule and update it if necessary.
* Doctors can give prescription to patients. And if necessary, they can give tests.
* They can view patient information such as patient prescriptions and test results.
* They can check reports.
* Doctors must log into their account first in order to do the above-mentioned activities.

**Class Diagram:**

****

**List of attributes and methods with visibility:**

abstract Class *Person*:

Attributes:

* -name: string
* -gender: string
* -age: int
* -contact no: int
* -address: string
* -blood group: string
* -email: string

Methods:

* +viewinformation()

Class Unregistered Visitor:

Methods:

* +getRegistered()

Class Admin:

Attributes:

* -admin id: int
* -password: int

Methods:

* +login()
* +logout()
* +manageDoctor()
* +managePatient()

Class Patient:

Attributes:

* -patient id: int

Methods:

* +makeAppointment()
* +cancelAppointment()
* +takeTest()
* +managePatient()
* +viewTestResult()
* +viewPrescription()

Abstract class *Staff:*

Attributes:

* -joined Date: DateTime
* -certificate: String

Methods:

* +addStaff()

<<interface>> Browsing Website:

Methods:

* +viewDoctorInformation()
* +viewMedicine()

<<interface>> Profile Managing:

Methods:

* +login()
* +logout()
* +viewProfile()
* +updateInformation()

Class Doctor:

Attributes:

* -doctor id: int
* -department: String
* -qualification: String

Methods:

* +prescribedMedicines()
* +prescribedTests()
* +checkReport()
* +makeSchedule()

Class Lab Assistant:

Methods:

* +collectSample()
* +generateReport()

Class Receptionist:

Methods:

* +manageCash()
* +assistPatient()

Abstract class *Payment:*

Attributes:

* -payment id: int
* -user id: int
* -amount: int

Methods:

* +enterInfo()
* +viewInfo()
* generateBill()

Class Credit Card*:*

Attributes:

* -number: String
* -date: DateTime
* -time: DateTime

Methods:

* +authorize()

Class Cash*:*

Attributes:

* -date: DateTime
* -time: DateTime

Methods:

* +confirmPayment()

Class Appoint Information*:*

Attributes:

* -appointment id: int
* -doctor: String
* -date: DateTime
* -patient id: int

Methods:

* +viewInformation()

Class Appointment Schedule*:*

Attributes:

* -doctor id: int
* -doctor name: String
* -total slots: int
* -day of the week: String
* -time: DateTime

Methods:

* +updateSchedule()

Class Prescription*:*

Attributes:

* -prescription id: int
* -symptoms: String
* -diagnosis: String
* -advice: String
* -medicine: String

Methods:

* +viewPrescription()

Class Medicine*:*

Attributes:

* -medicine id: int
* -brand name: String
* -generic name: String
* -dosage: String
* -side effect: String

Methods:

* +viewMedicine()

Class Test*:*

Attributes:

* -test id: int
* -test name: String
* -test type: String

Methods:

* +viewTestResult()

**Conclusion:**

The use case and class diagrams we created helped us to provide a clear view of our project. Specifically, the class diagram with classes and relations among them helps to get a clear object-oriented view of the project. Hence, creating these diagrams certainly boosted our progress.