 **SOFTWARE DESIGN & ANALYSIS**

**PROJECT REPORT**

*Submitted By*

Tahawar ihsan, M. Mohsin Raza, Matti Ur Rehman

(19P-0097) (19p-0072) (19p-0048)

*Submitted to*

USAMA MUSHARAF

(INSTRUCTOR CS)

**DEPARTMENT OF COMPUTER SCIENCE**

**FAST NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCINCES, PESHAWAR**

Session 2019-2023

The Hospital Management System (HMS) is a web application, which used for the control of hospital services. The HMS web application can be accessed by either mobile or computer browser. The HMS application combines all details regarding doctors, patients, nurses, hospital

administrative, etc. into one software. HMS System allows the patients to register via a registration module (form), which gathers and stores all required patient’s data such as name, e-mail, gender, etc. Registered patients can skip this step and login directly using their username and password through the login module. Nevertheless, unregistered users can only take advantage of major system features such as viewing the hospital timings. After the patient creates an account and register, he can access the allowed system features/functionalities for patients. Patients can view available appointments, book an appointment and manage his/her own profile. After the patient book an appointment, he can visit the hospital according to his appointment. Once the patient reaches the hospital, the receptionist will issue a clinic number for him since the receptionist has access to the system to view the appointments list and status with nurses and doctors. The HMS system also allows the receptionist to create patient accounts and book an appointment, referring to the doctors’ schedule, for unregistered patients. Once patient’s turn came, the patient can explain his condition to the consulting nurse, so that the nurse performs the pre-assessment examinations to diagnose the problem and then redirect him to the concerned doctor/clinic. The HMS system enables the nurse to allot patients for the concerned doctors, to view doctors’ status and to update patients’ account. Then, the concerned doctor will diagnose the patient, and then enter the prescription needed for the patient. If the doctor sees that the patient needs any further examinations like collecting and processing specimens, the system allows the doctor to redirect the patient to the Nurse again. After the nurse collects the specimens, the specimens will be sent to the laboratory so that the lab assistant can process, analyze the specimens, and then generate and enter the test results into the system. Furthermore, the doctor can redirect the patient to the lab assistant if there is a need to perform examinations such as X-Ray images, CT scan, MRI. The lab assistant can access the system and generate test reports regarding the examinations or test performed.On the other hand, the doctor keeps track of the examination results entered by the lab assistant and then recommend further actions to be taken if required, as well as enters a new prescription for the patient. The system also allows the patient to access his account to see prescription details and view his reports along with doctor advice. This feature is very useful since test reports usually take a long time to be generated, so that the patient may leave the hospital and view the results along with doctor’s advice through his account without the need of going to the hospital again. Once the prescription is ready, the pharmacist will prepare the medicines for the patient and enters the dose and guidelines of each medicine into the system. When the patient goes to the pharmacy of the hospital, he/she will find the medicines ready so

that he/she can pick and go easily. The patient has two options to know the dose and guidelines of each medicine, either by asking the pharmacist directly or by accessing his/her account to see it. This will help the patient be aware of the medicines’ dose if he/she forgets it. Finally, the patient will need to go to the cashier to pay for his/her visit. The system allows the cashier to create and order invoice for payment through the billing module. In addition, the cashier can watch the payment history of the patients.

Requirement Analysis

USE CASE DIAGRAM

Diagram

Description automatically generated

**Activity Diagram of Doctor**

**Diagram

Description automatically generated**

Activity Diagram of Patient:

**Diagram

Description automatically generated**

Activity Diagram of Nurse

**Diagram

Description automatically generated**

Activity Diagram of Reception

Diagram

Description automatically generated

Activity Diagram of Lab Assistant

**Diagram

Description automatically generated**

Activity Diagram of Pharmacist

Diagram

Description automatically generated

Activity Diagram of Cashier

**Diagram

Description automatically generated**

Activity Diagram of Adminstrator:

**Diagram

Description automatically generated**

Swim lane diagram

**Diagram

Description automatically generated**

**Design Model**

**Identify a possible set of classes/objects in the HMS case study.**

**Name of classes and attribute are:**

\* Doctor, patient

\* Nurse, pharmacists,

\* lab\_assitant,

\* Cashier, reception,

\* administrator, person

\* pharmacy, lab,

\* department,

**ASSOCIATION**

**Diagram

Description automatically generated**

**Class diagram:**

Diagram

Description automatically generated

**AFTER ANALYSIS AND USING SOLID principles and design patterns.**

1. **SOLID Principles**
2. Open Close Principle:

In our class diagram we apply Open Close Principle in our class diagram we can add new classes with out changing data in other class. You can simply see this Principle in our diagram. For example you add new class inherit with person class without changing into any other class.

1. LISKOV SUBSTITUTION PRINCIPLE:

In our class diagram we apply LISKOV SUBSTITUTION PRINCIPLE by adding this we inherit that class together which have same function under same class and other in different class and at the end we combine these classes with super class. In our case we have function getattribute() and setattribute() which are use with their base class

1. Design pattern
2. Builder Design Pattern

In our class diagram we use Builder Design Pattern. Like for HMS we have different classes and combine them in that way at the end it make for us complete HMS.

2)Bridge Design Pattern

In our class diagram we use Bridge Design Pattern for everything we use sperate class like lab class and pharmacy class we divide every class in that way and connect it with super class that at the time we call and do things.

1. Decorator Design Pattern

We also use Decorator Design Pattern in our class

1. Chain of Responsibility

In our class diagram we also have Chain of Responsibility in that after main class we have different classes after some choice we move to next class and after using different class we move to next class .

**Diagram

Description automatically generated**

**Screenshot of working project**

Graphical user interface

Description automatically generated

Graphical user interface, text

Description automatically generated

A screenshot of a computer

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, application, PowerPoint

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface

Description automatically generated

Graphical user interface, application, PowerPoint

Description automatically generated

**![Graphical user interface

Description automatically generated]()![Graphical user interface

Description automatically generated with low confidence]()![Graphical user interface

Description automatically generated]()![Graphical user interface

Description automatically generated]()![Graphical user interface

Description automatically generated]()![Graphical user interface, text, application

Description automatically generated]()![Table

Description automatically generated]()![Text

Description automatically generated with medium confidence]()![Graphical user interface

Description automatically generated]()**