**PAPERLESS HOSPITAL SERVICE**

(12bce0007, 12bce0002, 12bce0027)

SOFTWARE DESIGN DOCUMENT

TABLE OF CONTENTS

# 1) INTRODUCTION

## 1.1- PURPOSE

## 1.2-SCOPE

## 1.3-OVERVIEW

# 2) DESIGN OUTLINE

## 2.1- DESIGN DECISION

## 2.2-SYSTEM ARCHITECTURE OVERVIEW

2.2.1-CONTEXT MODEL

2.2.2-ER DIAGRAM

## 2.3-DESIGN DETAILS

2.3.1-COMPONENTS/MODULES

2.3.2-PURPOSE OF MODULES/ COMPONENTS

2.3.3-INTERACTION BETWEEN COMPONENTS

3) DESIGN PROCESS

4) DESIGN ISSUES

5) DESIGN-DETAILS

5.1) CLASS DIAGRAM

5.2) SEQUENCE-DIAGRAM

5.3) ACTIVITY-DIAGRAM

5.4) UI MOCKUPS

**1) INTRODUCTION**

**1.1-PURPOSE**

This software design document aims to explain the scope, design architecture, design components, interactions and design issues (functional and non functional) of the web application Paperless Hospital Service. This document also includes the diagrams showing the high level architecture of the system and diagrams to explain the interaction between various components. The alternate design details will also be discussed.

**1.2-SCOPE**

Objective of this project is to leverage paperless hospital service where patient need not do any paper-work while getting admitted to the hospital by providing seamless application that will handle the thousands of patient information and provide efficient healthcare service. Through this application, the main aim is to digitize every operation of hospital and reducing the unnecessary burden of maintain physical records on paper which are not only cumbersome to manage but also do not provide a secure means to safe confidential data.

**1.3-OVERVIEW**

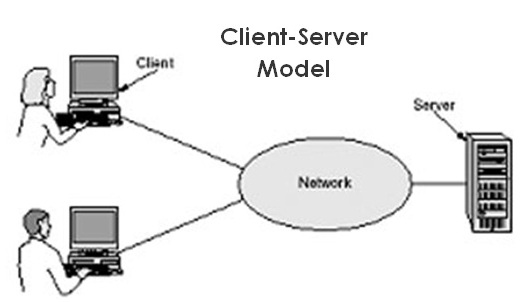
This application is being developed for the Healthcare/Medicine domain. The effectiveness of this application will be decided by its ability to provide secure access to all the users and also limiting the access to the confidential data. Depending on the role a particular user plays in the hospital, access will be granted. The main tasks involved are storing and updating patient’s information, calculating the expense, retrieving doctors information, doctors updating patient’s tests results and reports and checking whether patient has insurance or not.

**2) DESIGN OUTLINE**

**2.1- DESIGN DECISION**

Since the project aims to develop a web application, Modern Web based Client-Server Model is the most preferable and convenient option to handle this application. The Web-based system is composed of individual nodes, each performing different functions and often their roles can be dynamic. In such systems, the client component handles the user interface and the server provides back-end processing, such as database access, printing, and so on.

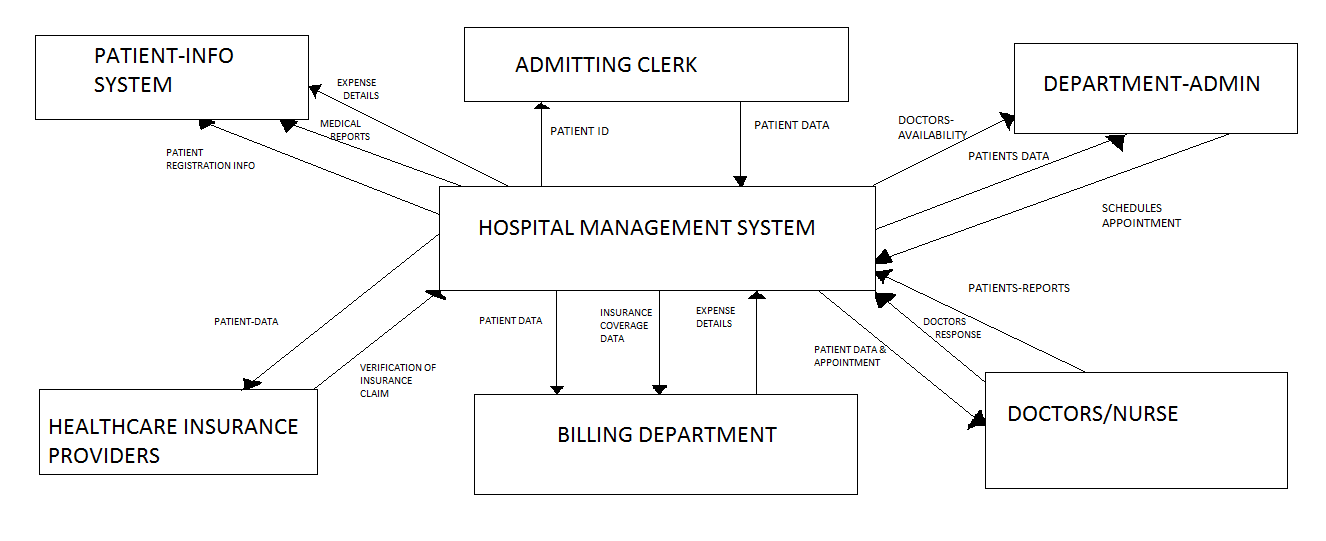
Having individual nodes allows splitting software systems into multiple components with each component running on a different computer and performing a specialized function. This approach simplifies development, management, administration, and often improved performance and robustness, since failure in one computer did not necessarily disable the entire system.

****

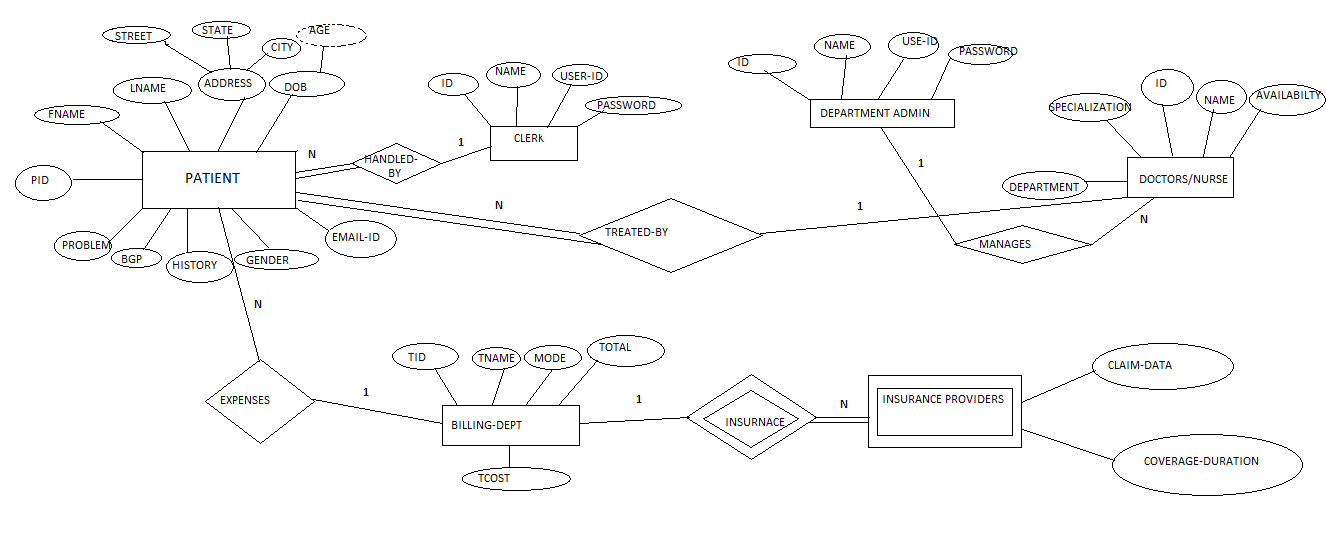
**Client –Server Model**

**2.2-SYSTEM ARCHITECTURE OVERVIEW**

**1- CONTEXT-MODEL**

****

**2- ER-DIAGRAM**

****

**2.3-DESIGN DETAILS**

*2.3.1-COMPONENTS/MODULES*

* + - PATIENT/RELATIVE
    - ADMITTING CLERK
    - DEPARTMENT ADMINISTRATOR
    - DOCTOR/NURSE
    - BILLING DEPARTMENT
    - INSURANCE PROVIDER

*2.3.2-PURPOSE OF EACH COMPONENT*

*PATIENT/RELATIVE*

Patient gets admitted by giving their information to the Admitting Clerk. If Patient is in serious state, relative will help in giving information about the patient to the Admitting Clerk. Also in case of emergency only essential details will be recorded. Patient will be requested to give the following details.

*ADMITTING CLERK*

The Admitting Clerk logs in the Web Portal by giving his unique username and password. Once he is logged in, he checks whether the patient is new or not. If the incoming patient is New, then Admitting Clerk will create a new record in Web Portal and store the patient information by collecting all necessary details and also about health history. Else, he just collects information about the type of his emergency. Also he will provide Patient ID which will be sent via sms to patient’s mobile number.

*DEPARTMENT ADMINISTRATOR*

Department Administrator checks and keeps track of all the doctors in various departments like ENT, Cardio, and General -Surgery etc. The department admin logs in the system by giving his unique username and password. The department admin after login gets to know the patient details. He can also see how many doctors are currently available or not. Accordingly, he checks their timings and the emergency of the patient and schedules their appointment.

*DOCTOR/NURSE*

Doctors will get patient information by querying on Patient ID and will conduct series of tests and will update their test report along with comments in the Application. The data about medical treatments and other diagnoses are a matter of privacy. Hence only patient and doctor can view this data and nobody else will have access to this data. All these details will be sent by mail to the patient to his email-id.

*BILLING DEPARTMENT*

Billing Department will calculate the expense and will be responsible to verify if Patient has Insurance Policy, if so then they will open a secure session to charge the cost to Insurance providers. If Patient doesn’t have Insurance Policy, then Billing Department Admin will send a message via sms to pay either in cash or credit/debit cards .The Billing Department will only get the names of the tests performed and essential patient-information and not any further details.

*INSURANCE PROVIDER*

Insurance Providers are external actors. They will get necessary patient-information for verifying the claim and will confirm whether the Insurance ID and policy coverage is valid or not.

2.3.3-INTERACTION BETWEEN COMPONENTS

(THROUGH SEQUENCE DIAGRAMS)



**3) DESIGN PROCESS:-**

The design process is the task of creating or generating design artifacts and subsequently evaluating, refining, integrating, and modifying these artifacts until the result satisfies the requirements of the problem definition. In essence, the design process is the task of mapping problem requirements into design solutions. The design process should be guided by a productive, economic, and controllable methodology that will ensure a high quality product.

**DIFFERENT PROBLEMS:-**

**The Paradigm Problem**: The paradigm problem refers to the failure to recognize and develop a manageable, productive, economically feasible process model for SE. Much attention has been focused on the development of new software engineering paradigms but no results have proven completely satisfactory for SE development. Any successful model must deal with the interdependent facets of making design decisions while recognizing the need for adequate leverage and project management.

**Design Evaluation**: In order to make good design decisions, one must have the ability to assess the validity of a particular design decision or weigh the relative merits of competing design alternatives. The lack of evaluation ability leads to inadequacies in assessing the impacts of a design decision on all levels of the design process. Under the umbrella title of evaluation we include testing, validation, verification and, as a specific instance, prototyping.

**The Representation Problem**: The representation problem is a fundamental requirement for advancement in each of the areas mentioned above. The issue is the ability to express, manipulate and make inferences about design objects and processes. In current practice, major development takes place at very low levels of design for at least two reasons. First, current methods of software engineering encourage designers to think in terms of low level issues such as databases, data structures, performance measures, screens, and interfaces because low level representations are the only mechanisms that provide feasibility measures.

**4) DESIGN ISSUES:-**

Some of the major issues associated with the automation of software development occur with respect to requirements specification, design, maintenance, reusability, validation, verification, and testing.

**FUNCTIONAL ISSUES:-**

* Define clear & complete test requirements.
* Manage changes to requirements.
* Functional gaps in test plans.
* Difficulty in review by the development team for large test plans
* To monitor the exact hardware-software configurations for the test environment.
* Blocking issues found in the product functional areas lead to extensive re-planning putting the initial overall plan at a stake.
* Inconsistency in the structure.

**SOLUTIONS:-**

* Arrange for product feature presentations from development team.
* In case of any requirement changes, make corresponding modifications to program.
* Institutionalize usage of re-usable test environments through various tools.
* Establish a knowledge repository of various problems encountered during test environment setup.
* Provide a better method to reduce inconsistency in the structure.

**NON-FUNCTIONAL ISSUES:-**

* Availability of system.
* Security issues.
* Backup issues.
* Maintainability issues.
* Stability issues.
* connection issues.

**SOLUTIONS:-**

* System should be free from any vulnerable attacks.
* Authentication measure should be included in the program.
* Backup of all the data/information should be taken in regular interval of time.
* Less maintenance should be there.
* Program should be able to run on any system/version.
* Connection between UI & database has to be made in separate file so it can be used at anytime.

**UI ISSUES:-**

* Not readable.
* Login interface error.
* color mix up.

**SOLUTIONS:-**

* Design should be done carefully & preciously.
* Divide the interfaces into layers.
* Colors should be selected carefully so it can attract the user.