**DIAGNOSTIC CENTER CLIENT MANAGEMENT SYSTEM**

**SOFTWARE DESIGN SPECIFICATIONS**

**Document Owner – Rhiya Ramesh**

**Table of Contents**

[1. INTRODUCTION 2](#_Toc22683615)

[1.1. DOCUMENT OUTLINE 2](#_Toc22683616)

[1.2. DOCUMENT DESCRIPTION 2](#_Toc22683617)

[1.2.1. Introduction 2](#_Toc22683618)

[1.2.2. System Overview 2](#_Toc22683619)

[2. DESIGN CONSIDERATIONS 3](#_Toc22683620)

[2.1. ASSUMPTIONS AND DEPENDENCIES 3](#_Toc22683621)

[2.2. GENERAL CONSTRAINTS 4](#_Toc22683622)

[2.3. GOALS AND GUIDELINES 4](#_Toc22683623)

[2.4. DEVELOPMENT METHODS 5](#_Toc22683624)

[3. ARCHITECTURAL STRATEGIES 5](#_Toc22683625)

[3.1. OVERVIEW 5](#_Toc22683626)

[3.2. RATIONALE 5](#_Toc22683627)

[4. SYSTEM ARCHITECTURE 6](#_Toc22683628)

[4.1. SUBSYSTEM ARCHITECTURE 6](#_Toc22683629)

[5. DETAILED SYSTEM DESIGN 6](#_Toc22683630)

[5.1. CLASSIFICATION 6](#_Toc22683631)

[5.2. RESPONSIBILITIES 7](#_Toc22683632)

[5.3. CONSTRAINTS 8](#_Toc22683633)

[5.4. COMPOSITION 8](#_Toc22683634)

[5.5. USES/INTERACTIONS 9](#_Toc22683635)

[5.6. RESOURCES 10](#_Toc22683636)

[5.7. INTERFACE 10](#_Toc22683637)

[6. BIBLIOGRAPHY 10](#_Toc22683638)

# INTRODUCTION

## DOCUMENT OUTLINE

This software design documentation provides a description of a software created to facilitate analysis, planning, implementation. This document can be considered as a blueprint or model of the system. The software design document describes the type of the software being developed and approach used in the development, the design considerations, detailed system architecture.

## DOCUMENT DESCRIPTION

## Introduction

The Diagnostic Center Client Management Website aims at bridging the gap between end-users and clients. The website intends to be a one-stop solution and a hassle-free experience when it comes to booking appointments for lab tests.

This is a digitized diagnostic service application that provides various healthcare solutions.

This software/application will automate the medical diagnostic services and the diagnostic system.

## System Overview

This SDS will cover the software for Diagnostic Centre Client Management.

The tool will help end users(patients) book appointments for conducting medical tests with our clients (diagnostic centers). These tests can be scheduled to be conducted at the center or at home.

End User Mode:

The end-user will be able to login and key in the location manually or allow the application to automatically detect the user’s location.

Once logged in the user will be able to view all the diagnostic center in and around the area and the tests available in each of them.

On choosing the diagnostics center and the test, depending on test the user will either be allowed to take it up at home or schedule an appointment.

The end-user will be given the option of either paying by cash or card.

If he chooses the latter, he will be guided to a payment gateway and a bill will be generated.

The user will be allowed to cancel or reschedule the appointment, given he does so 12 hours in advance.

Client Mode:

The client is provided a template. He creates the schedule based on this template and is allowed to upload it using the application.

The schedule for the current day, the next day and the day after is made visible to the client and the user.

The client will be able to upload the report onto the application.

# DESIGN CONSIDERATIONS

## ASSUMPTIONS AND DEPENDENCIES

* Resources:
  + The client will have a set of employees who have been trained to use our system
* [Budget](https://projectbliss.net/project-budget-template/):
  + Project costs will stay the same as initially [budgeted costs](https://projectbliss.net/project-budget-template/)
  + Training will be conducted internally with no additional training costs incurred
* Finances:
  + Funding for licenses will be provided by various departments as needed
* Scope:
  + [Project scope](https://projectbliss.net/project-scope-creep/) will not change once the stakeholders sign off on the scope statement
* Operational
  + The webpage must be allowed to access the location in order to accurately search for the diagnostic centers nearby.

## GENERAL CONSTRAINTS

* Platform:
  + must work on Macintosh and PC
* Operating system:
  + for Macintosh: must have OS X.4
  + for PC: must have at least Windows 8 or above
* Hardware:
  + at least 128 MB of RAM
  + minimum 800x600 screen resolution with 256 colors

## GOALS AND GUIDELINES

* The Diagnostic Center for Client Management focuses on developing and supporting healthcare software solutions.
* It aims to be the first choice of doctors, hospitals and the patients looking forward to a one-point solution for healthcare software services which are delivered through efficient use of knowledge and technology.
* Our goal is to digitize the healthcare industry which will help the doctors, hospitals and the patients, by maintaining all activities that an onsite diagnostic center performs like collecting samples, performing tests, preparing reports, collection of amounts etc.
* This Diagnostic center for client management software can be used for the automation of any medical diagnostic center.
* It also aims to provide an easy and user-friendly GUI to help its end users use the services effortlessly and quickly.

## DEVELOPMENT METHODS

* The software development method used for this project is the V model which stands for validation and verification model. In this method each phase of software development life cycle starts after the completion of earlier phase.
* In this method both the test and development of the software starts in parallel so as to avoid the downward flow of defects.
* Product – the end-to-end flow of the application is smooth leading to a high degree of certainty.
* Process – the development and testing can be done simultaneously because of the number of independent modules. Therefore, the degree of certainty is quite high.
* Resource – since the resources required, vary from client to client, the degree of certainty is relatively low.
* From the above points, it can be concluded that the **V model** will be apt for the project.

# ARCHITECTURAL STRATEGIES

# OVERVIEW

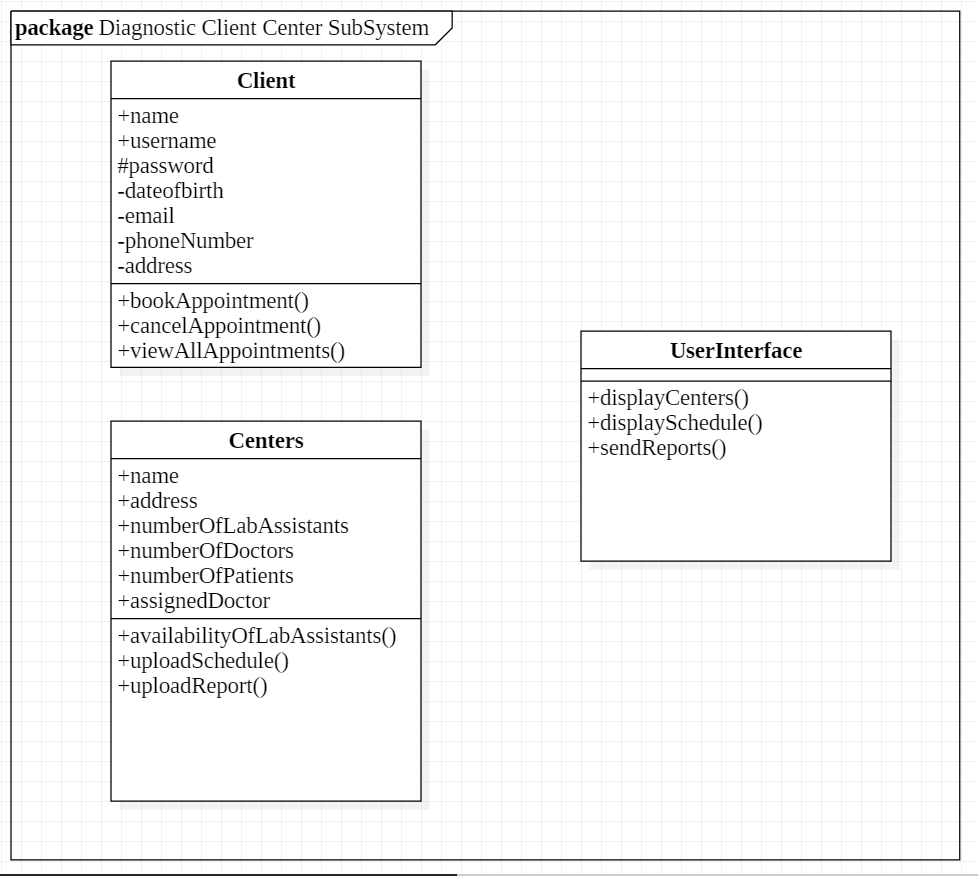
The system will follow the three-tier architectural style and be organized into three layers: the interface layer, application layer and the storage layer. The interface layer will be a graphical user interface that allows users to interact with the system. It will be implemented using the Python and Flask framework, and will allow the users to add tests, book slots, show tests, display report and download reports. The application layer will contain the logic and rules for storing data in the database layer and also retrieving it in accordance with the user’s needs. This is the layer that will contain the data file parsers and will allow controlled access to the data files. Finally, the storage layer will store the metadata required for the system.

# RATIONALE

The three-tier architecture style shall be used because it not only separates the user interface and the metadata, but also provides an application logic layer. The application layer provides a middle layer that allows the data files and the GUI components to be loosely coupled. The application layer has to be modified if there are any changes to the format of the data files and the interface layer will need little or no modification. This will make it easy for clients of this software to modify the data file format and attributes for further research purposes if they wish to do so. This layer makes the system more maintainable and reusable and also hides the complexity of processing data from the users.

# SYSTEM ARCHITECTURE

# SUBSYSTEM ARCHITECTURE



# DETAILED SYSTEM DESIGN

## CLASSIFICATION

The whole application is divided into different modules:

* Login/Register Module.
* Deletion Module.
* Report Module
* Booking Module
* Payment Module
* Scheduling Module

## RESPONSIBILITIES

1. Login/Register Module
   1. This module enables user to create an account and login by entering the right credentials.
2. Deletion Module
   1. This module enables user to delete their account from the system database.
3. Report Module
   1. Uploading of the report is done by the client.
   2. Report will be sent to end-user via a third-party application.
4. Booking Module
   1. It provides flexibility in choosing location, tests, diagnostic center, date, time, and whether it can be done in the comfort of the end user's home or at the diagnostic center.
   2. It also provides flexibility in canceling an appointment.
   3. Confirmation message to be sent – via email and/or notification.
5. Payment Module
   1. Mode of payment can be chosen by the user which is then redirected to payment gateway.
   2. Payment can be done via cash/card.
6. Scheduling Module
   1. Provides interface to upload schedule to the client.
   2. Also provides flexibility to change the schedule every 72 hours and
   3. Flexibility to filter the schedule according to patients and/or lab tests.
   4. CONSTRAINTS
7. Login Module
   1. A single person can have a single account for given mail address.
   2. The username given can consist of 25 characters maximum.
   3. Passwords must consist of five characters minimum and must contain at least one special character.
8. Report Module
   1. The reports uploaded by the client are to be confidential and each client is assigned a passkey.
   2. Only through the passkey, the report uploading component is accessed by the client. This passkey is generally assigned by the client to single personnel to maintain confidentiality.
9. Booking Module
   1. A user can cancel the bookings up to 18 hours prior to the appointment.
10. Scheduling Module
    1. The diagnostic center must upload the schedule every 72 hours.

## CONSTRAINTS

1. Login Module
   1. A single person can have a single account for given mail address.
   2. The username given can consist of 25 characters maximum.
   3. Passwords must consist of five characters minimum and must contain at least one special character.
2. Report Module
   1. The reports uploaded by the client are to be confidential and each client is assigned a passkey.
   2. Only through the passkey, the report uploading component is accessed by the client. This passkey is generally assigned by the client to single personnel to maintain confidentiality.
3. Booking Module
   1. A user can cancel the bookings up to 18 hours prior to the appointment.
4. Scheduling Module
   1. The diagnostic center must upload the schedule every 72 hours.

## COMPOSITION

1. Login/Register Module:
   1. The register module consists of a register component that enables first-time visitors to create an account by filling the necessary details. The login module enables registered users to log in and use the web application.
2. Report Module:
   * 1. The report module consists of mainly two components:
   1. Uploading component: This provides an interface for the client to upload the reports using a passkey.
   2. Notification component: This sends the report to the destined user(patient) via a third-party application.
3. The booking module consists of the following components:
   1. Booking component: This component displays the available centers in the entered location and displays other details as time slots for the particular test in that center. The user can confirm a booking.
   2. Backend component: On confirmation, it updates the available slots in the center and updates the relevant databases.
4. Payment module:
   1. Selection component: This asks for the user to select among the various payment options. On selection, this redirects to the respective payment gateway.
5. Scheduling module:
   1. Uploading component: This component provides an interface to the clients to upload their schedule for the next 72 hours or even update them.
   2. Viewing component: The clients can view the upcoming schedule and also filter based on time slots or kinds of tests using this component.

## USES/INTERACTIONS

The viewing component of the scheduling module refers to the booking database to update itself. The booking database is actually updated by the booking module on confirmations from users.

The payment module interacts with the corresponding bank server through the payment gateway for authentication and authorization to make payment.

The rest of the modules are grouped in a way that the dependencies or direct interactions with the other modules are very less.

## RESOURCES

Resources required to run the application are:

* Python IDE to run the code
* Flask server
* Web service
* Users should have a system with an internet connection to use our application

## INTERFACE

We’ll need user interfaces for:

Login: using a form we can get the details and validate the user

Register: using a form we can get the details and update in the database

Report: pdf or an image to view their report

Booking: we can use a form to confirm the appointment by the user who wants to book a test from any diagnostic center.

Payment gateway: we can ask the user to select the mode of payment by using a list and then move ahead with the mode he/she has chosen.

Scheduling tasks: diagnostic center admin can upload a csv file which gives the details about the person who is available.

Hardware interface:

The system must run over the internet. Can be connected via Wi-Fi, WLAN etc.

Software interface:

Any window-based operating system which can be used to perform basic tasks.

## BIBLIOGRAPHY

## [SDS](https://lostclouds.com/2Communicate/project/SDS.pdf)