DAMS
0.1v

Doctor Appointment Management Software

The Purpose of this project is to build doctor appointment management software for the patient.

Document Control:

Project Revision History

Date	Version	Author	Brief Description of Changes	Approver Signature
13/11/2022	0.1v	Group 6	Initial draft	
14/11/2022	0.2v	Group 6	Added flowchart	

1. INTRODUCTION 5

1.1. Intended Audience 5
1.2. Acronyms/Abbreviations 5
1.3. Project Purpose 5
1.4. Key Project Objectives 5
1.5. Project Scope and Limitation 5
1.5.1. In Scope 5
1.5.2. Out of scope 5
1.6. Functional Overview 6
1.7. Assumptions, Dependencies & Constraints 6
1.8. Risks 6

2. DESIGN OVERVIEW 6

2.1. Design Objectives 6
2.1.1. Recommended Architecture 6
2.2. Architectural Strategies 7
2.2.1. Design Alternative 9
2.2.2. Reuse of Existing Common Services/Utilities 9
2.2.3. Creation of New Common Services/Utilities 9
2.2.4. User Interface Paradigms 9
2.2.5. System Interface Paradigms 10
2.2.6. Error Detection / Exceptional Handling 11
2.2.7. Memory Management 11
2.2.8. Performance 11
2.2.9. Security 11
2.2.10. Concurrency and Synchronization 11
2.2.11. Housekeeping and Maintenance 11

3. SYSTEM ARCHITECTURE 12

3.1. System Architecture Diagram. (Not Necessary) 12
3.2. System Use-Cases 13-15
3.3. Subsystem Architecture 15
3.4. System Interfaces 15
3.4.1. Internal Interfaces 15
3.4.2. External Interfaces 15

4. DETAILED SYSTEM DESIGN 15

4.1. Key Entities 16
4.2. Detailed-Level Database Design 16-18
4.2.1. Data Mapping Information 19
4.2.2. Data Conversion 19
4.3. Archival and retention requirements 19
4.4. Disaster and Failure Recovery 19
4.5. Business Process workflow 19
4.6. Business Process Modeling and Management (as applicable) 19
4.7. Business Logic 19
4.8. Variables 19
4.9. Activity / Class Diagrams (as applicable) 20
4.10. Data Migration 21
4.10.1. Architectural Representation 21

4.10.2. Architectural Goals and Constraints 21
4.10.3. Logical View 21
4.10.4. Architecturally Significant Design Packages 21
4.10.5. Data model 21
4.10.6. Deployment View 21
5. ENVIRONMENT DESCRIPTION 22
5.1. TIME ZONE SUPPORT 22
5.2. LANGUAGE SUPPORT 22
5.3. USER DESKTOP REQUIREMENTS 22
5.4. SERVER-SIDE REQUIREMENTS 22
5.4.1. Deployment Considerations 22
5.4.2. Application Server Disk Space 22
5.4.3. Database Server Disk Space 22
5.4.4. Integration Requirements 22
5.4.5. Jobs 22
5.4.6. Network 22
5.4.7. Others 22
5.5. CONFIGURATION 23
5.5.1. Operating System 23
5.5.2. Database 23
5.5.3. Network 23
5.5.4. Desktop 23
6. REFERENCES 23
7. APPENDIX 23

1. Introduction

This project is intended to implement the doctor appointment management software. This DAMS can be easily used by all the users irrespective of their age. A Doctor Appointment system is a self-service tool for booking appointments.

1.1. Intended Audience

This document can be shared or viewed across the following members: CG employees, BU SME's, internal SME's.

CG employee	
BU SME's	
Internal SME's	

1.2. Acronyms/Abbreviations

DAMS	Doctor appointment management software
DFD	Data flow diagram

1.3. Project Purpose

The Purpose of this project is to build an appointment management software for the doctor.

1.4. Key Project Objectives

- Patients can book their appointment with the specialized doctor through admin.
- Doctors can check patient's medical history.
- Doctors can check their appointment slot.
- Admin can manage both doctor and patient database.

1.5. Project Scope and Limitation

This project is used in creating appointments to the patients and they no longer have to be worry about their slot times. In this project Doctor and patient can plan their schedule in accordance with their availability, and patients do not need to wait more time to meet their respective doctor.

1.5.1. In Scope

This project aims to create the Doctor Appointment Management Software application in which patients can enter their details and appointment can be scheduled as per availability and need of the patient. Doctor can see the appointment list and plan his day accordingly. Admin can save the patient's profile with date and time for further communication.

1.5.2. Out of scope

NA

1.6. Functional Overview

DAMS is an organized well-structured representation of managing appointments to the patients. This project includes appointment for patients, and it deals with doctor, patient appointment database. This project is flexible, easy to use, and it is designed for doctors and patients.

1.7. Assumptions, Dependencies& Constraints

- The System should have Linux installed.
- This project works on laptop or desktop.
- Constraint is such that this software is only created for the people of Capgemini.

1.8. Risks

- Lack of accessibility to the resources.
- System bugs.

2. Design Overview

The doctor appointment management software is made in such a way that it will have a login page from where doctors or patients can login into the database. There will be a main menu which will have sub menus like -creation for doctor, creation for patient, create an appointment, display, search, and exit. So, in the patient part we will receive a unique id for a particular patient and based on the illness the doctor will be appointed as per the availability. Then there will be the following panels: -

- Doctor Panel-Doctor can search for patient, check if his booking is made for the entire day or not and check if the appointment for the patient is completed or not.
- Patient Panel-Here the patient can get appointed to the doctor for a particular slot based on their illness and based on the availability of the doctor.
- Admin Panel-Admin panel has the access to both the doctor and the patient panel. It can check for the doctor's schedule for that particular day. It can search for the patients slot, make a new booking and also check if the appointment for the patient is completed or not.

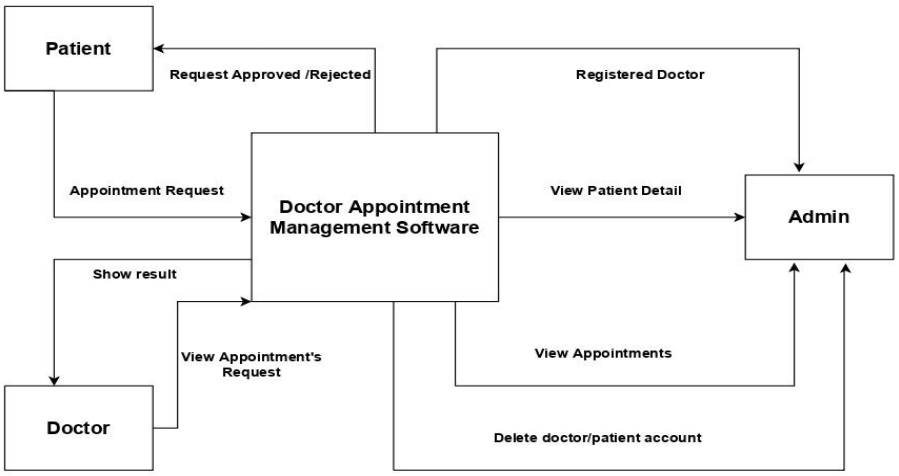
2.1. Design Objectives

- The objective of this design is to use time for both the patient and the doctor wisely.
- Doctors and patients have a unique id to make the search or new entry easier.

2.1.1. Recommended Architecture

The recommended architecture will be DFD (Data Flow Diagram). It is because it helps us to understand the overview of the system without going into much detail. It is a graphical representation which is very easy to understand as it helps visualize contents. Data Flow Diagram represent detailed and well explained diagram of system components.

2.2. Architectural Strategies



Admin:

- Create Doctor profile
- Create patient profile
- Update doctor profile
- Update patient profile
- Delete doctor profile
- Delete patient profile
- Schedule appointment
- Cancel appointment

Doctor:

- Doctor Registration
- Doctor Login
- View profile
- Update information
- View appointment
- Search patient

Patient:

- Patient Registration
- Patient Login
- View appointment
- View profile

2.2.1 Design Alternative

NA

2.2.2 Reuse of Existing Common Services/Utilities

The Already existing doctors and patients can directly login into the main menu with their Unique Id.

2.2.3 Creation of New Common Services/Utilities

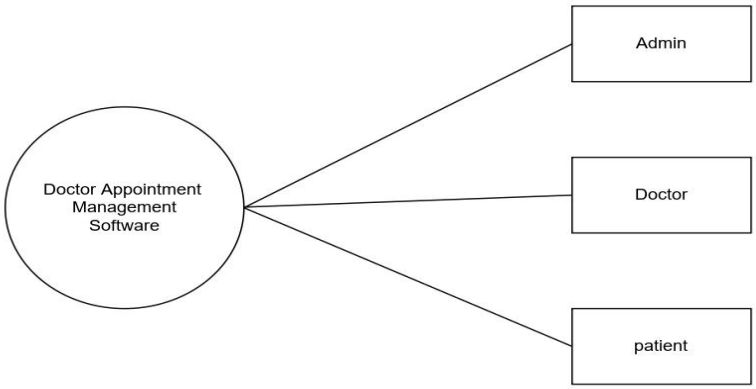
All the new doctors and patients need to make a data entry for the first time to get their personal unique id such that it can act as a reference for them to login to the system the next time.

2.2.4 User Interface Paradigms

A User interface is how a program (including the operating system) communicates with a person. The two common interfaces are the Console and the Graphical User Interface GUI. Each of these has a different approach that needs to be considered.

2.2.5 System Interface Paradigms

The Architecture of System interface Paradigm:



2.2.6. Error Handling / Exceptional Handling

- All the errors detection can be managed using Valgrind.
- Unit testing is also a method used for error detection.

2.2.7. Memory Management

In this Doctor Appointment Management Software, data are stored in System safe and in a secure manner. Memory is always used to store the data collected from the doctor and patient. Once the data is deleted by the admin, it will be deleted from memory also. We will use file management for storing the data of doctors and patients.

To achieve a degree of multiprogramming and proper utilization of memory, memory management is important. Many memory management methods exist, reflecting various approaches, and effectiveness of each algorithm depends on the situation.

2.2.8 Performance

- Quick and easier method rather than visiting the hospital and taking an appointment and waiting in the line for long hours.
- User friendly.
- This project is optimized in every aspect thus works perfectly.

2.2.9 Security

- Only Admin Panel can access the records and data of both the doctor and the patient.
- Patient can only check for his / her details and previous appointments by using their unique id.
- Doctor can check their appointments and schedule for the day by using his / her unique id.

2.3 Concurrency and Synchronization

NA

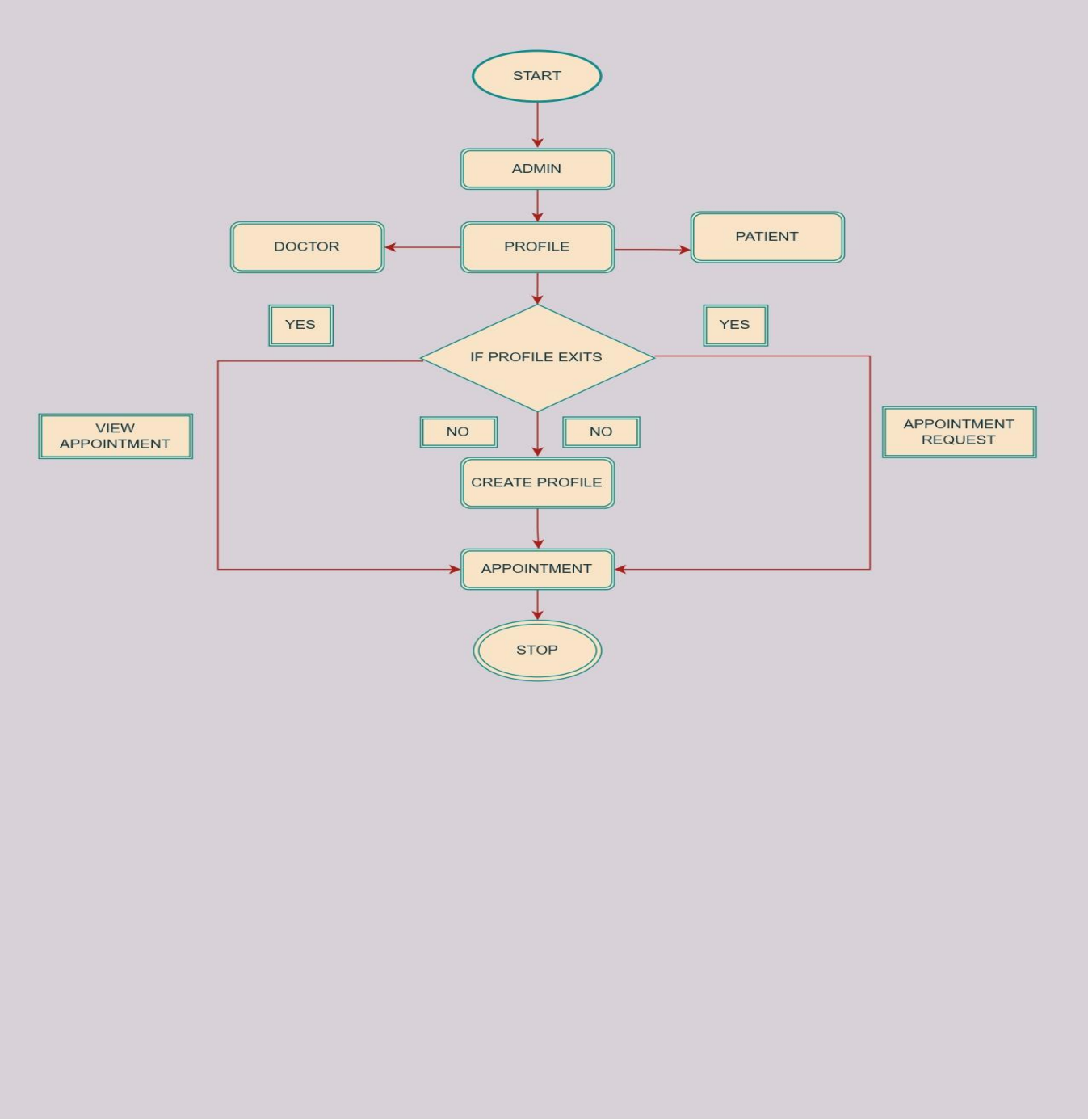
2.4 Housekeeping and Maintenance

All the past records of the doctors and patients are maintained here for further Reference

3. System Architecture

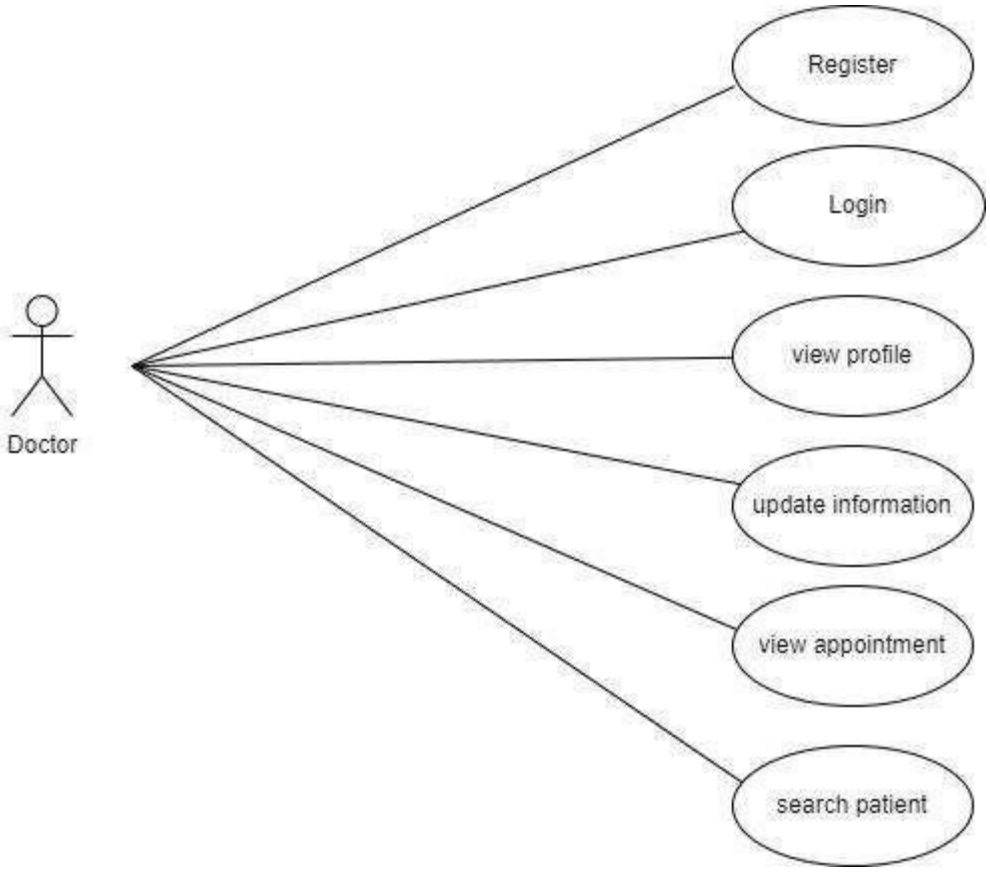
System architecture is the conceptual model that defines the Structure, behavior, and more views of a system.

3.1. System Architecture Diagram. (Not Necessary)

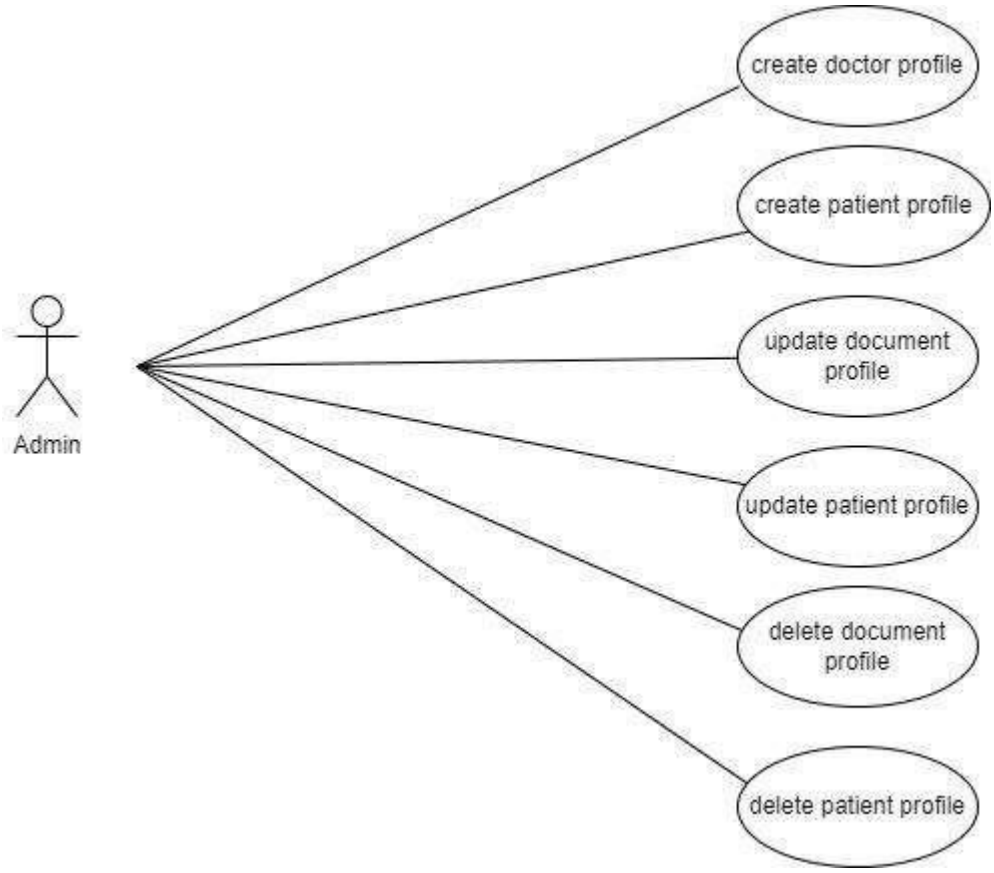


3.2. System Use-Cases

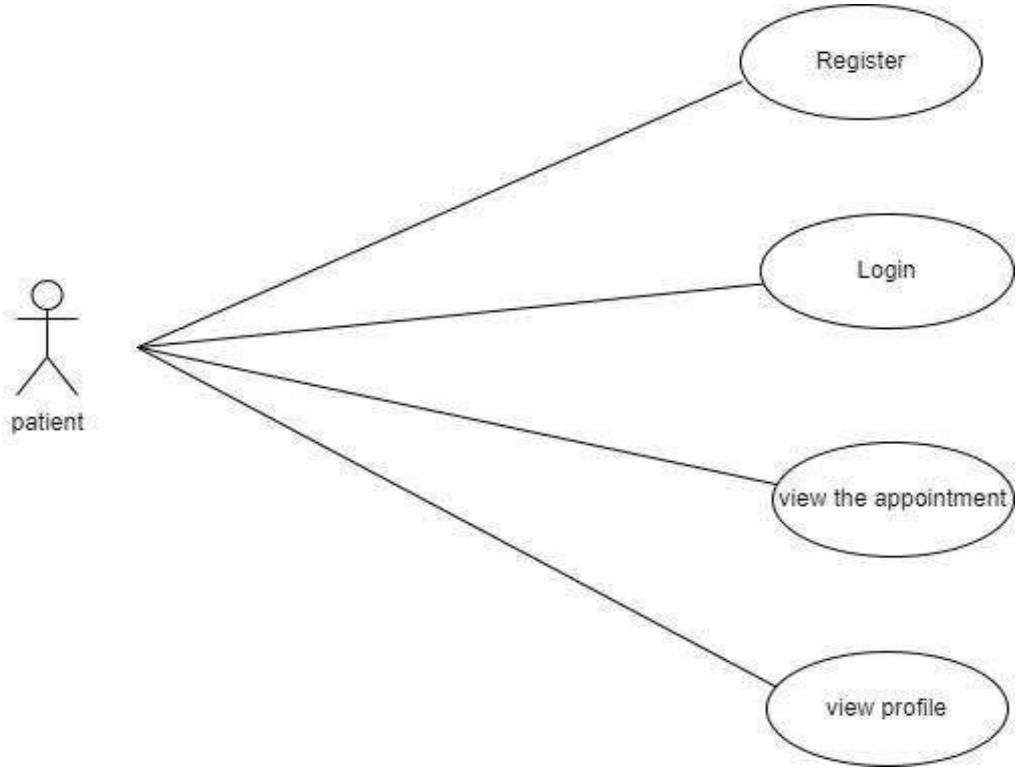
Doctor Use -Case:



Admin Use-case:



Patient Use-Case:



3.3 Subsystem Architecture

NA

3.4 System Interfaces

NA

3.4.1 Internal Interfaces

NA

3.4.2 External Interfaces

NA

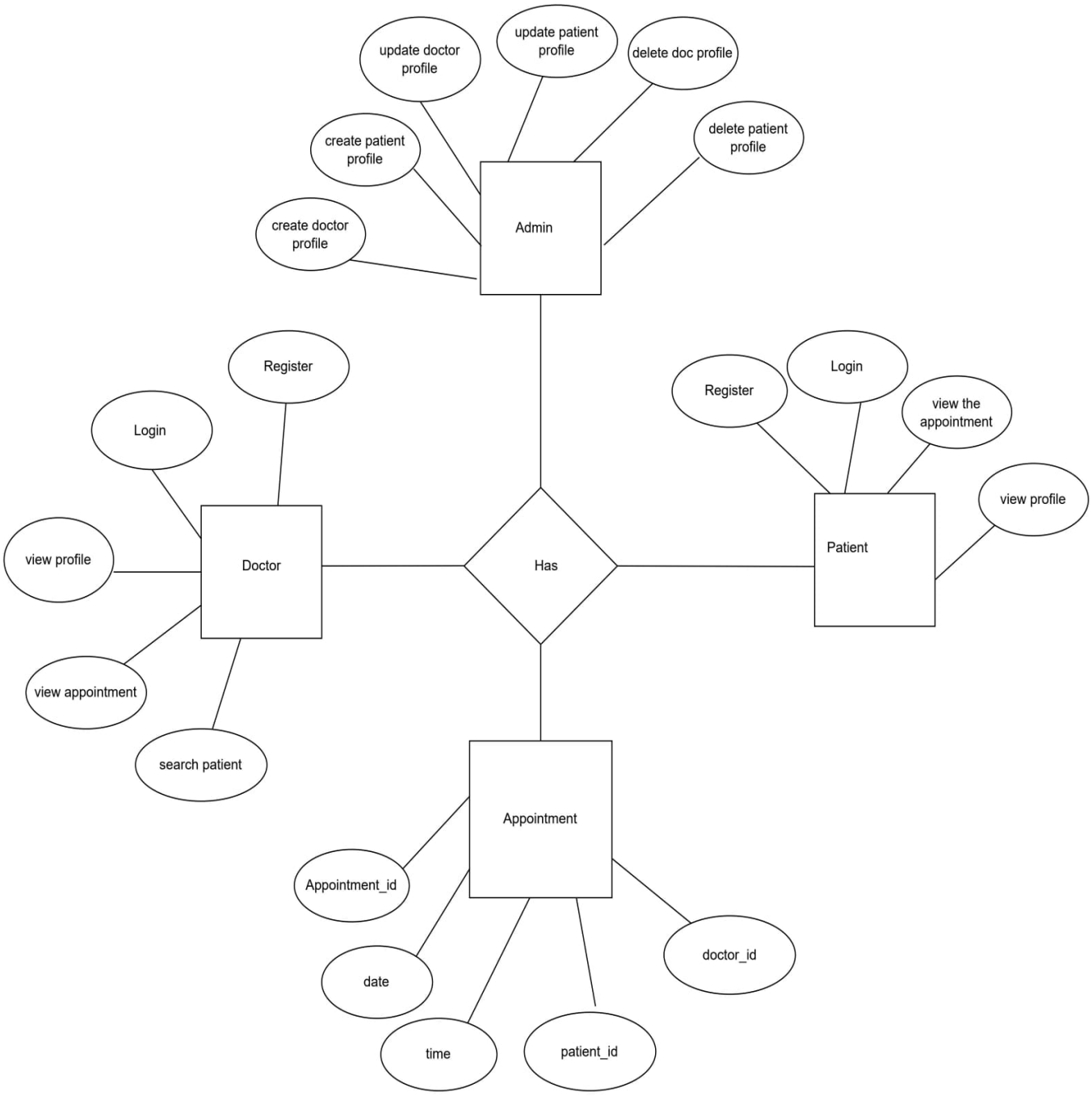
4 Detailed System Design

NA

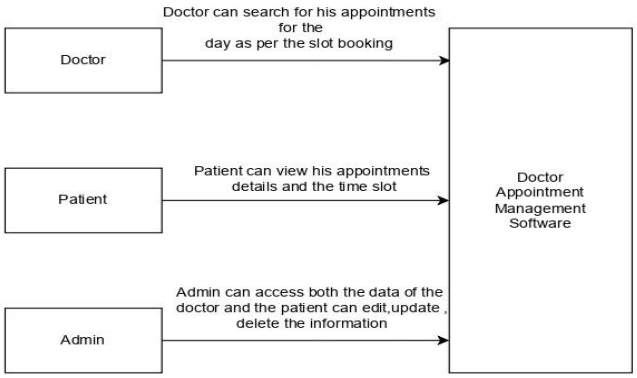
4.1 Key Entities

NA

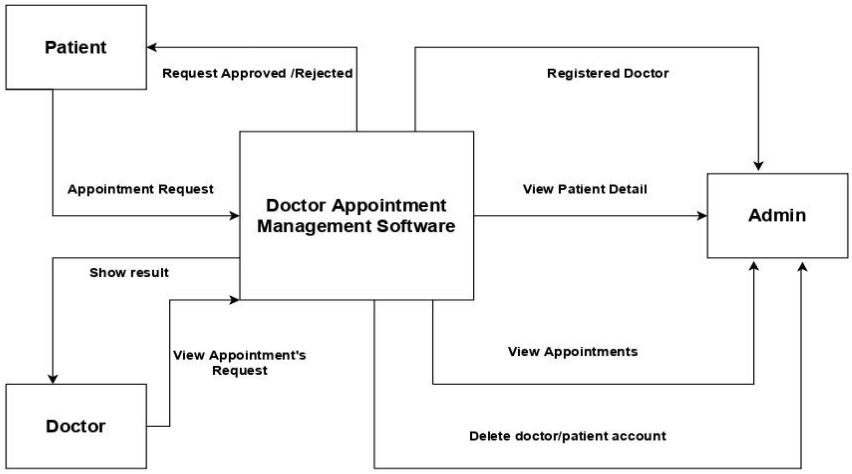
4.2 Detailed-Level Database Design



Level-0 DFD:



Level-1 DFD:



4.2.1 Data Mapping Information

NA

4.2.2 Data Conversion

NA

4.3. Archival and retention requirements

NA

4.4. Disaster and Failure Recovery

NA

4.5. Business Process workflow

NA

4.6. Business Process Modeling and Management (as applicable)

NA

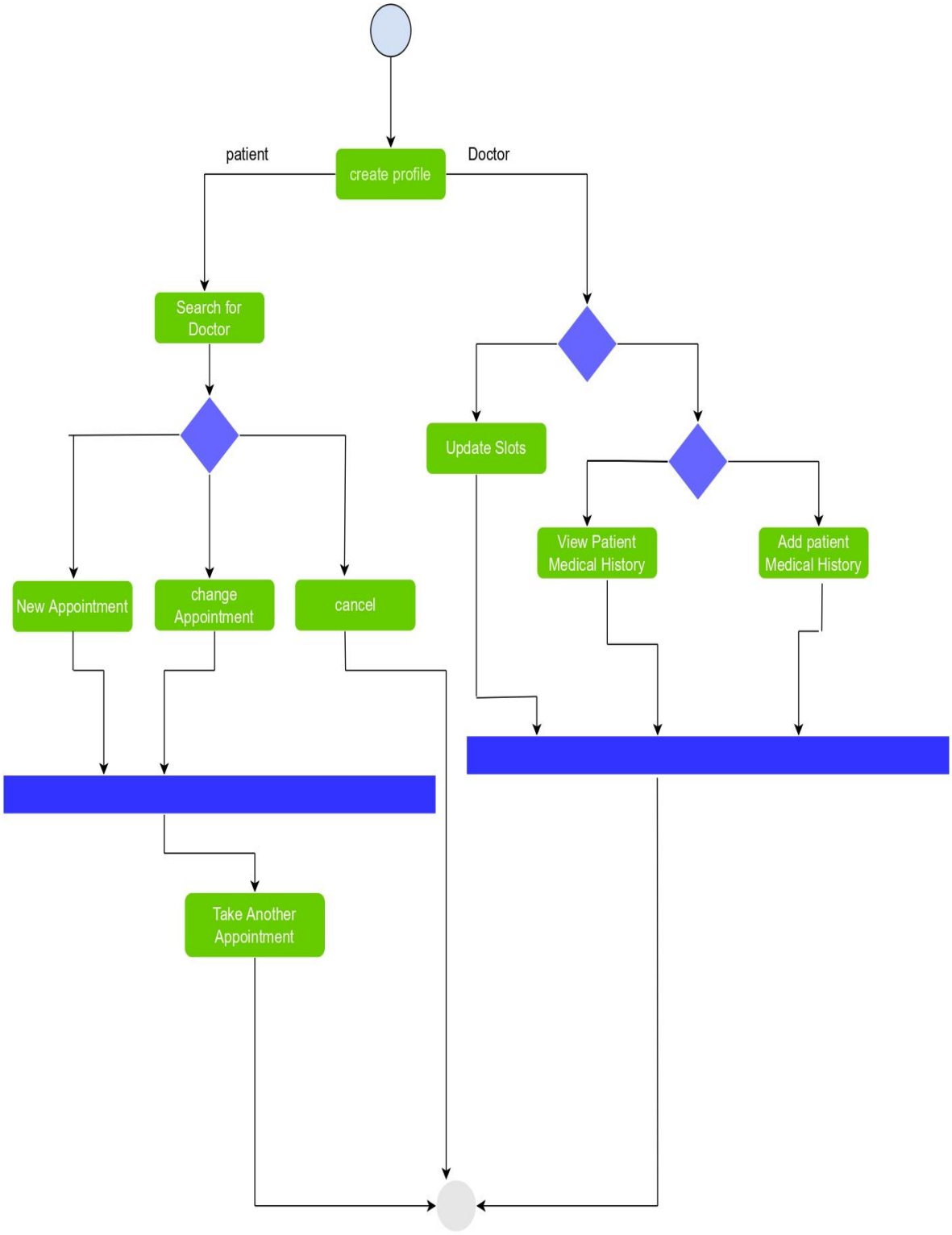
4.7. Business Logic

NA

4.8.Variables

NA

4.9 Activity /Class Diagrams (as applicable)



4.10. Data Migration

NA

4.10.1 Architectural Representation

NA

4.10.2 Architectural Goals and Constraints

NA

4.10.3 Logical View

NA

4.10.4 Architecturally Significant Design Packages

NA

4.10.5 Data model

NA

- **Legacy system data model**
- **Proposed system data model**
- **Interface data model**

4.10.6 Deployment View

NA

5. Environment Description

5.1 Time Zone Support

It supports the time zone as per Indian Standard Time (IST).

5.2 Language Support

This project only supports the English Language.

5.3 User Desktop Requirements

The requirement for user's desktop - Linux should be installed

5.4 Server-Side Requirements

NA

5.4.1 Deployment Considerations

NA

5.4.2 Application Server Disk Space

NA

5.4.3 Database Server Disk Space

NA

5.4.4 Integration Requirements

NA

5.4.5 Jobs

NA

5.4.6 Network

NA

5.4.7 Others

NA

5.5 Configuration

5.5.1 Operating System

- OS-Linux (Ubuntu)-64 bit
- RAM- 8GB
- i3/i5 processor

5.5.2 Database

NA

5.5.3 Network

NA

5.5.4 Desktop

Windows 10

6. Reference

- https://in.coursera.org/specializations/c-programming-linux
- https://www.javatpoint.com/linux-tutorial
- https://www.tutorialspoint.com/cprogramming/index.htm
- https://www.javatpoint.com/c-programming-language-tutorial

7. Appendix

NA

Change Log

DAMS Template Version Control (Maintained by QA)			
Date	**Version**	**Author**	**Description**
13-11-2022	0.1	Group06	Initial Version