Patient’s Appointment Scheduling System Architecture

- Deliverable III

Term Project Group 2

Team Members,:
Nabin Niroula
Vinesh Reddy Kankanalapally
Satya Teja Nimmakayala
Shree Chandan Sahu

Executive Summary

Description:

The purpose of this project is to create a robust, easily accessible and user-friendly patient appointment scheduling system that will mitigate the challenges that patients face while scheduling appointments to visit their healthcare providers.

The main objectives of this project are ➖

1. Designing a patient appointment scheduling system so that hospitals using this system would be benefited from having options to schedule appointments for their patients at other participating hospitals if they do not have facilities for the needed treatments.
2. Helping patients save their time that they otherwise would have to waste trying to get connected to medical representatives either in-person or over the phone and get scheduled for their medical needs.
3. Providing a record management system which can keep medical records of patients for better medical services and could be referenced for future if necessary.
4. Providing a centralized appointment scheduling system so that inter-hospital patient’s appointment scheduling becomes as easy as possible for hospitals.

Stakeholders:

1. Patients consulting with their doctors or other medical care providers for their treatments.
2. Software architectures(ourselves) designing the overall architecture of the system.
3. Developers developing the system.
4. Hospital administrators in search of a centralized patient appointment scheduling system.
5. Medical Science students(e.g. residential doctors/nurses), working under doctor’s supervision, who have access to a patient's medical record.
6. Future software architects who could be a part of this system maintenance team.
7. The users of the system, including healthcare providers, appointment handling staff, and hospital management.

Business Goals:

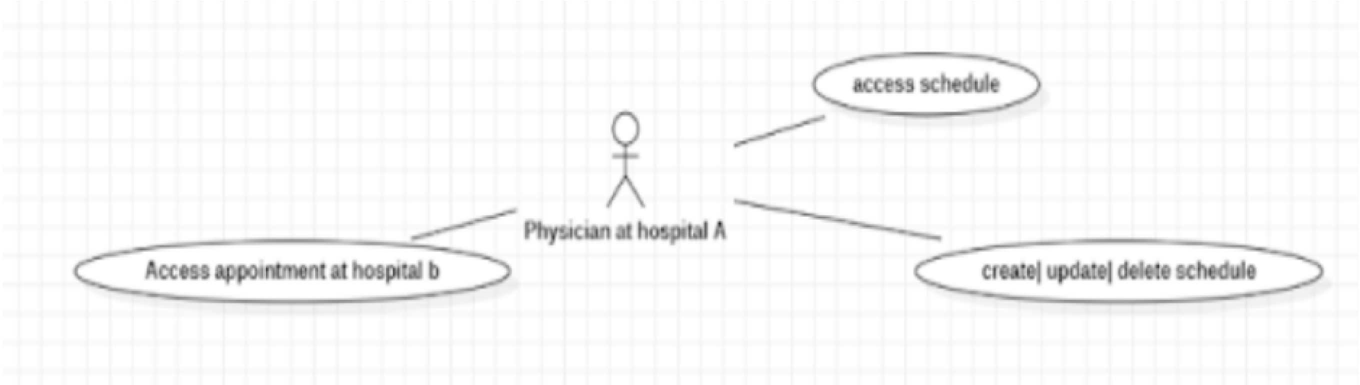
1. To create a robust, easily accessible and user-friendly patient appointment scheduling system that minimizes the challenges patients face while scheduling medical appointments.
2. To design and construct a centralized web-based system that allows users(healthcare providers and other hospital employed staff incharge of handling patients’ appointments) to help create, update, access, or cancel medical appointments of their patients at different hospitals.
3. To construct a cost-effective and distributed system that is readily available online that reduces the technical cost of installing different devices.

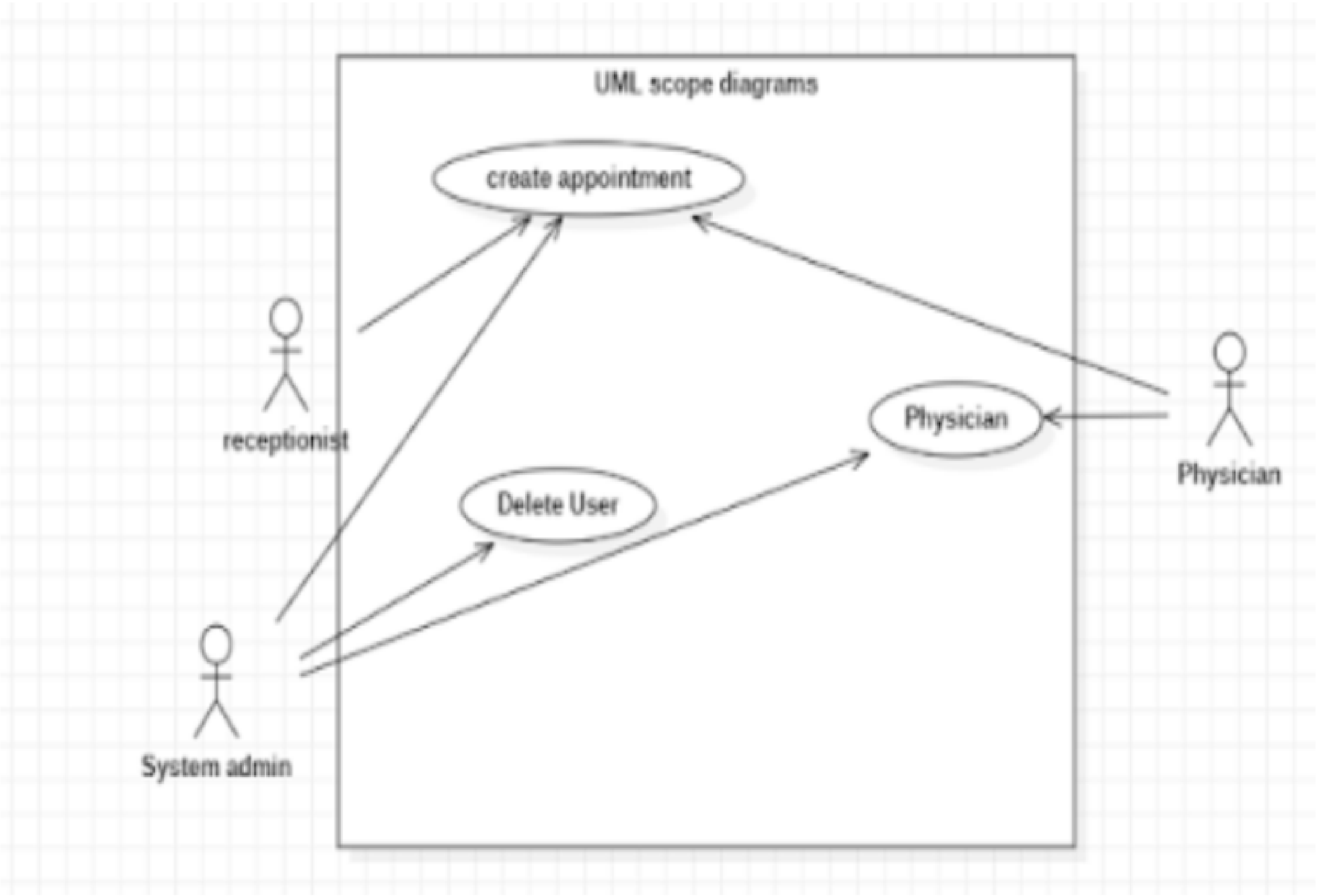
3) Introduction

i) Purpose:

The main purpose of the project is to create a robust, easily accessible, and user-friendly patient appointment scheduling system that will mitigate the challenges that patients face while scheduling appointments to visit their healthcare providers.The project extends to implement a concept of database mirroring to provide temporary duplication of the main database so that in case of database failure, the system provides continuous service such that users are unaware of database providing its functionalities based on temporary back up of it. In addition to aforementioned purposes, the project is targeted towards making the system a web-based structure.

ii) Scope: The scope of the project means what is included in each deliverable incrementally, what design decisions were made, who will use the system, how it is deployed, and many more. It is secure, reliable as well as easy to use. So, any hospital within a given region can participate to use this benefit of a centralized system to handle hospital patient appointment management. It is not available online for anyone to use. Hospitals are the targeted main users of this system.


access schedule
Physician at hospital A
Access appointment at hospital b
create| update| delete schedule



iii) Stakeholders: The following are the main stakeholders for the project

1. Patients consulting with their doctors or other medical care providers for their treatments.
2. Software architectures(ourselves) designing the overall architecture of the system.
3. Developers developing the system.
4. Hospital administrators in search of a centralized patient appointment scheduling system.
5. Medical Science students(e.g. residential doctors/nurses), working under doctor's supervision, who have access to a patient's medical record.
6. Future software architects who could be a part of this system maintenance team.
7. The users of the system, including healthcare providers, appointment handling staff, and hospital management.

iv) Definitions:

1. **Users:** The targeted users of the system are healthcare providers, hospital administrators, and hospital staff like front-desk employees/receptionists.
2. **PASS:** Patient Appointment Scheduling System. This is the system under design.
3. **FR:** Functional Requirement
4. **ASR:** Architecturally Significant Requirement

v) References:

“Software Architecture in Practice,” Fourth Edition by Len Bass, Paul Clements, and Rick Kazman.

4) Project Overview, Goals, Constraints, and Functional Requirements

i) Project Goals: The majority of healthcare service providers operate during the day, making it difficult for people with demanding schedules like working professionals, parents, and students to schedule appointments. It's because many healthcare organizations have office hours that are inconvenient for those with busy schedules—8:00 am to 5:00 pm local time. One of those busy people suffers from both skin-related and brain-related issues. Except in cases where the hospital is sufficiently large to offer both services in one location, the patient will unquestionably need to call at least two different hospitals or clinics to schedule separate appointments to visit two providers separately. Not every hospital is prepared and equipped to offer every service in a single location.

However, our goal is to make this system fully operational, after which the aforementioned issue will be reduced, and patients won't have to call various hospitals to make different appointment requests. They only need to make one phone call to request multiple appointments at various participating hospitals saving their lot of time

ii) Project Constraints: The main constraints in our system is database failure, if one hospital’s database fails than they should be having the backup standalone which would be able to store data on a temporal basis and whenever the database is recovered all data should be retrieved back to the database within a short period of time. Also one other constraint we can see in our system is the uneven cancellation of timing of doctor or the patient scheduled within the system. In that case, the system should book the patient who is in the queue waiting to be scheduled for his appointment.

iii) Functional requirements

The following are some important functional requirements of the system.

FR 1. Sign Up a user

- The system allows hospital management to **create** a user.

FR 2. Log in a user

- Authenticated users are allowed to log in to the system using username and password.

FR 3. Log out a user

- Logged out user’s all access rights will be deactivated until she logs in back again.

FR 4. Retrieve a user

- The system allows hospital administrators to retrieve a user’s information. But the Non - administrative users are not allowed to access other users' info.

FR 5. Update a user

- Users can change her password, and other information. A user is not allowed to update other user’s information unless she is authorized to do so.

FR 6. Delete a user

- System administrators/hospital management can delete a user from the system. A user cannot delete herself.

FR 7. Create an appointment

- It enables a user at one hospital to access and **create** an appointment at another hospital on behalf of the patient. It enables a user at one hospital to access and **create an appointment at another hospital** on behalf of the patient.

FR 8. Update an appointment

- It allows the user at one hospital to **update** the patient's appointment either in his hospital or at another.

FR 9. Delete an appointment

- This allows a user to **delete** an appointment on behalf of the patient.

FR 10. Search an appointment

- The system allows a user to **retrieve** a patient's schedule.

FR 11. Make slot available for new Scheduling

- If an appointment gets **canceled** , this makes that space available for new scheduling.

FR 12. Access patient's data

- It enables the system administrator or the authorized user to view the data of the enrolled patients

FR 13. Maintain patient's account

- This allows an authorized user to create, update, delete, and retrieve a patient's general and medical information.

FR 14. Faults and logs management

- Log the faults and whenever the system's database fails, this immediately creates the temporary backup of that database and enhances the system's functionality. Once the database is repaired, it transfers the data to the main database and erases the temporary database.

FR 15. System Maintenance

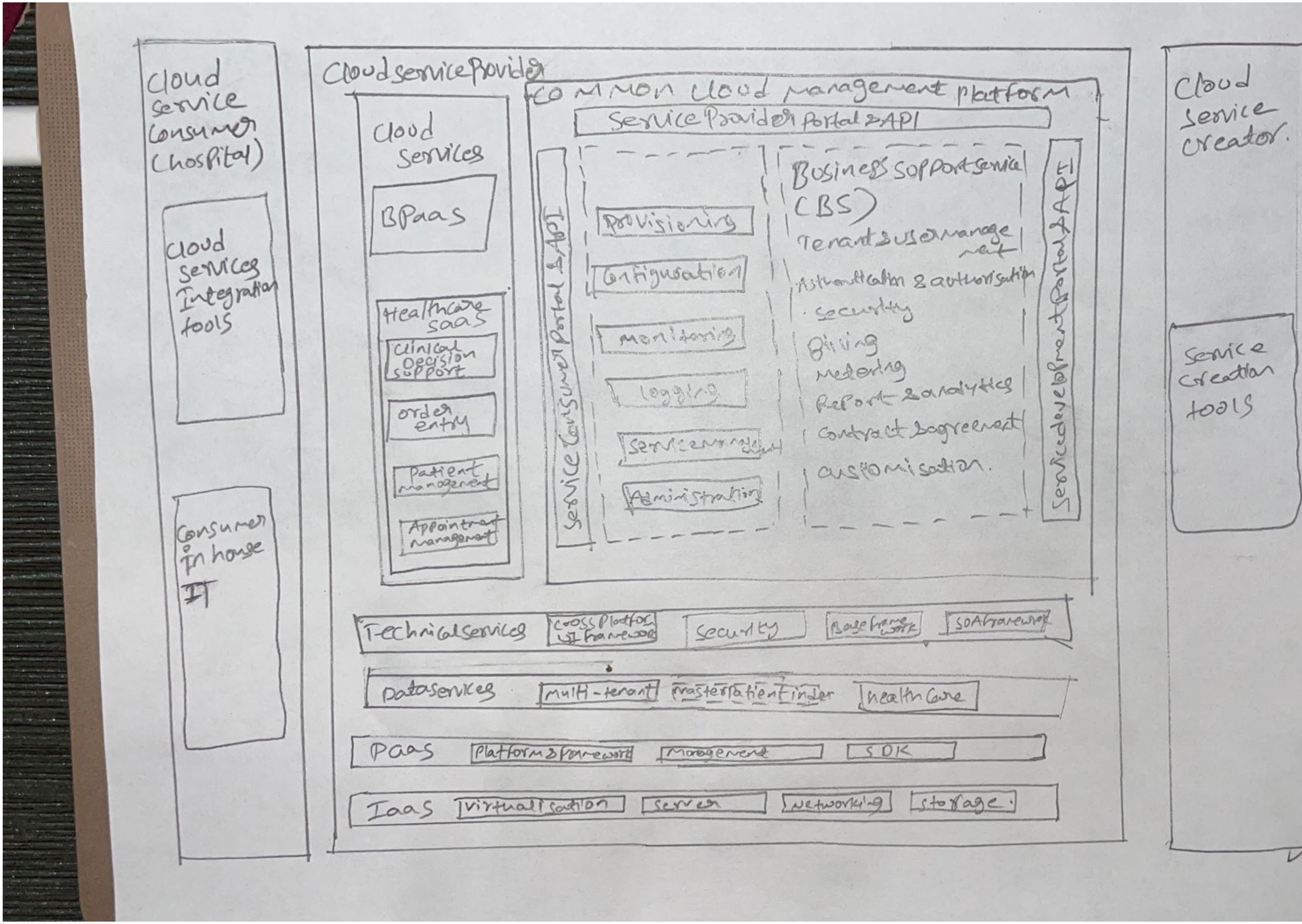
- It switches back to the previous working version if the system is inoperational for 30 minutes while it is being maintained.

iv) Architecture Overview

The project intends to create a system that provides users with the benefit of being able to schedule a medical appointment from one participating hospital at another participating hospital. The participating hospital means the hospital that uses this system to fulfill their needs of having a centralized system such as PASS. The core intention of the PASS is to bring all the hospital systems within a region into one centralized system in terms of managing their patients' appointments across different hospitals. The benefit of this system is that it mitigates the challenges that patients are currently facing when it comes to scheduling their medical appointments. One example of those challenges includes having a patient call different hospitals to find if treatment that patient is looking for is available. With the PASS, the patient would save her time simply by visiting one participating hospital and having a medical staff help her schedule an appointment at another hospital in the system, one that provides the specialty treatment she is looking for.

Architecturally, the system brings all appointment scheduling systems of different hospitals into a common one. This means the hospital staff can access the system online as it is a cloud based system. The temporary database duplication is the key asset to enhancing the availability of the system in case the database failure happens. What happens with this approach is that a backup of the main database is created by means of mirroring it, and in case it fails, the backup comes into effect thereby minimizing the database unavailability, the consequence of which the performance of the system improves.

So, we implemented a 3 tier architectural system, which includes User interfacing tier, Business logic tier, and Persistence tier.


Cloud service Consumer (hospital)
Cloud services Integration tools
Consumer in house IT
Cloudservice Provider
Cloud services
BPaas
Healthcare saas
Clinical Decision support
order entry
Patient management
Appointment management
Common cloud management platform
Service Provider portal & API
Service Consumer Portal & API
Provisioning
Configuration
monitoring
logging
Service management
Administration
Business support service (CBS)
Tenants & user management
Authentication & authorisation
security
Billing
metering
Report & analytics
contract & agreement
customisation.
Service development Portal & API
Technical services
cross Platform UI framework
security
Base framework
SOA framework
Dataservices
multi-tenant
master Patient index
health care
Paas
Platform & framework
Management
SDK
Iaas
virtualisation
server
networking
storage.
Cloud service creator.
Service creation tools

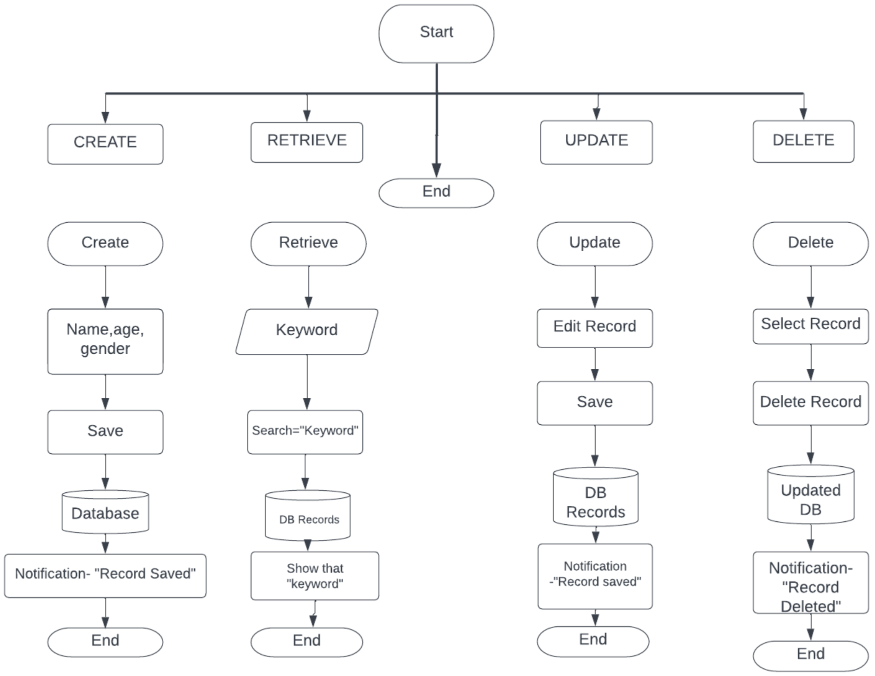
5) Use-Case Views

a) List of the system's use cases.

1) Add a hospital into the system: Given a hospital, this system should add that hospital's patient appointment scheduling system to the database of this system. Under normal operation, the system should accomplish this task within 45 seconds.

2) Update hospital in the system: Given a hospital that was previously added to this system integrates some features to its system. Now this system should adapt that update as well. So, the updating of those kinds of features should be done within 15 seconds when the system is running in its normal condition.

3) Delete a hospital from the system: Given a hospital was added to the database and now needs to be removed from the system for a reason such as a hospital not under operation anymore, then the system should remove this hospital from the database within 45 seconds of its normal operation.



4) Reopen a canceled slot for rescheduling a new appointment: A patient has an appointment on a given day, but a doctor cancels it on the same day and vice versa.

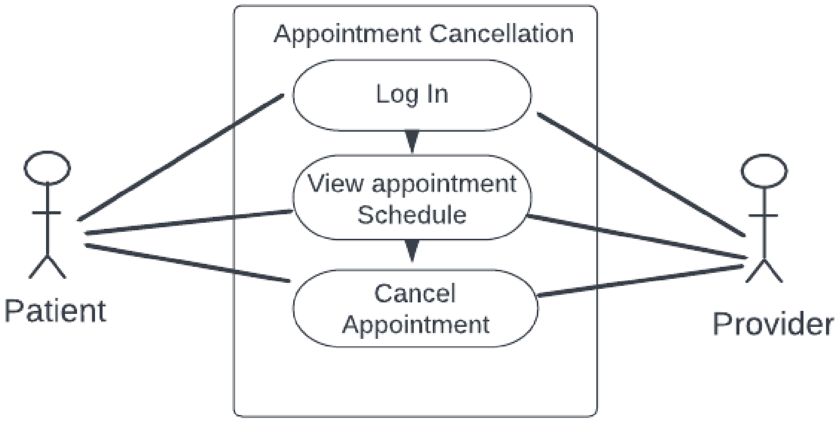
Given a patient has an appointment with a provider at a given hospital on a given day at 11:00am, and the provider cancels the appointment on the same day at 10:00am. Under its normal operation, the system should update it in the database within 5 seconds, and should notify the patient, the doctor, and the hospital within a minute. Similarly, when a front desk associate or a receptionist cancels the appointment on behalf of the patient in the same fashion, then the aforementioned functionality applies exactly.

5) Search slot availability at another hospital: If a hospital does not have a service for some type of treatment then the hospital can schedule an appointment of a patient at another hospital with another provider.

Consider that a patient under his/her treatment in a given hospital needs to go for a further diagnosis with a radiologist, but the radiology service is not available in that hospital. Then this system should be able to search for a nearby hospital that has a radiology service within 30 seconds under its normal operation. The result should be limited to 20 hospitals at maximum at a time. Then it should transfer all the referrals and schedule an appointment within two minutes.

6) Schedule appointment at another hospital: If a hospital has a specialty doctor, but does not have equipment for diagnosis, it should schedule a patient's appointment at a different hospital.

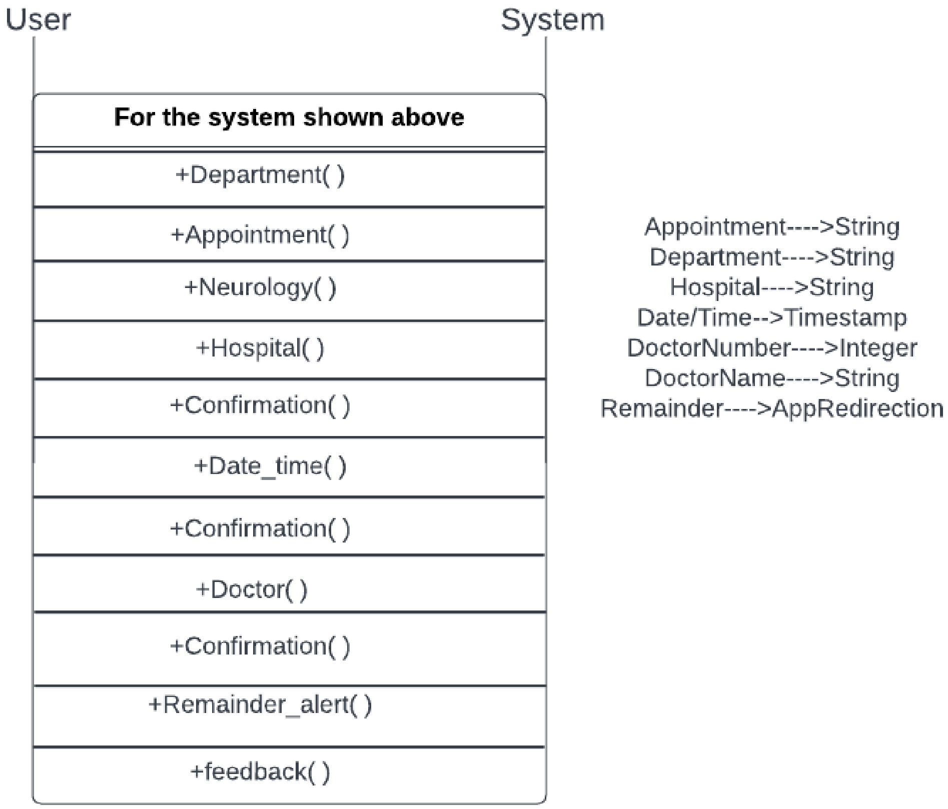
For instance, a hospital such as a primary care provider's clinic may have a doctor to read or interpret the result of an x-ray, but does not have an x-ray machine to take an x-ray. In a normal operation, the system should search for a nearby hospital that has x-ray service within 15 seconds. The result should be limited to 20 hospitals or clinics at maximum at a time.



7) Handle Overlapping appointments: When two appointments overlap, then the service should be provided on the basis of first come first serve.

Given two patients A and B have appointments at a given hospital on a given day at 9:00am to visit the same provider. If patient B checks in at 8:30am and A checks in at 8:45am, then the system should put them in a queue in the order they checked in within 15 seconds after the second patient checks in provided that the first one has already checked in. The system should then notify the provider within that 15 seconds as well.

The following diagram depicts the bare minimum skeleton of functional implementation of the proposed system.



8) Consider a health provider's lunch breaks: A receptionist receives a call to schedule an appointment for a patient on a given day at a given hospital at a given time. Given that she puts the day, date, time, and hospital into the system, then the system should notify her if that particular provider will be on a lunch break at a given time. The system should accomplish this task in less than 15 seconds.

9) **Save patient's time:** This system should save a patient's time that she otherwise would have to spend calling different hospitals at different times to set up an appointment to visit a provider.
Given a patient needs a CT Scans appointment. She does not have to call different hospitals asking for the earliest possible availability. When CT Scans is searched, this system should provide a list of different hospitals available for the earliest possible CT Scans schedule.
This should be done in less than 15 seconds and should limit the search result to 20 at maximum a time.

10) Retrieve the schedule

Given that a patient wants to schedule an appointment to visit a particular provider in a given hospital on a given day, however, there are no slots available anymore for that day. Then the system should provide a list of the schedule of that particular provider in that particular hospital for the next 30 days so that a patient can choose one that works for her the best. The system should pull up the schedule list in less than 15 seconds in its normal operation.

b) Architecturally Significant Use Cases

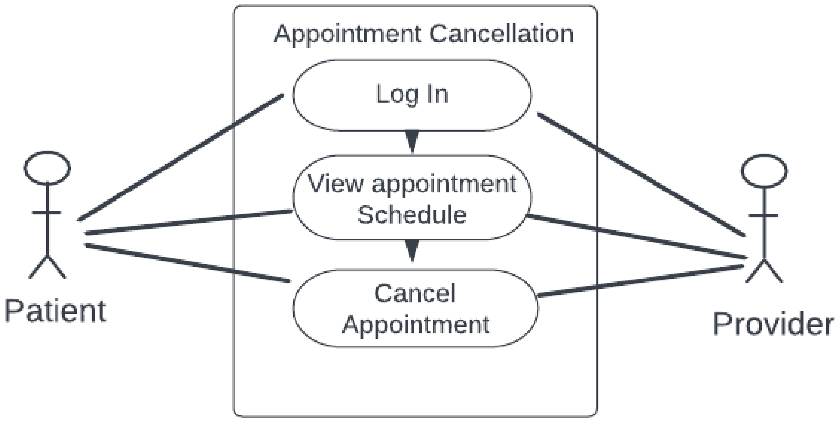
The architecturally significant use cases are as follows:

1) Add a hospital into the system:

This system should add the patient appointment scheduling software used by a given hospital to the system's database. The system should complete this task in 45 seconds when operating normally.

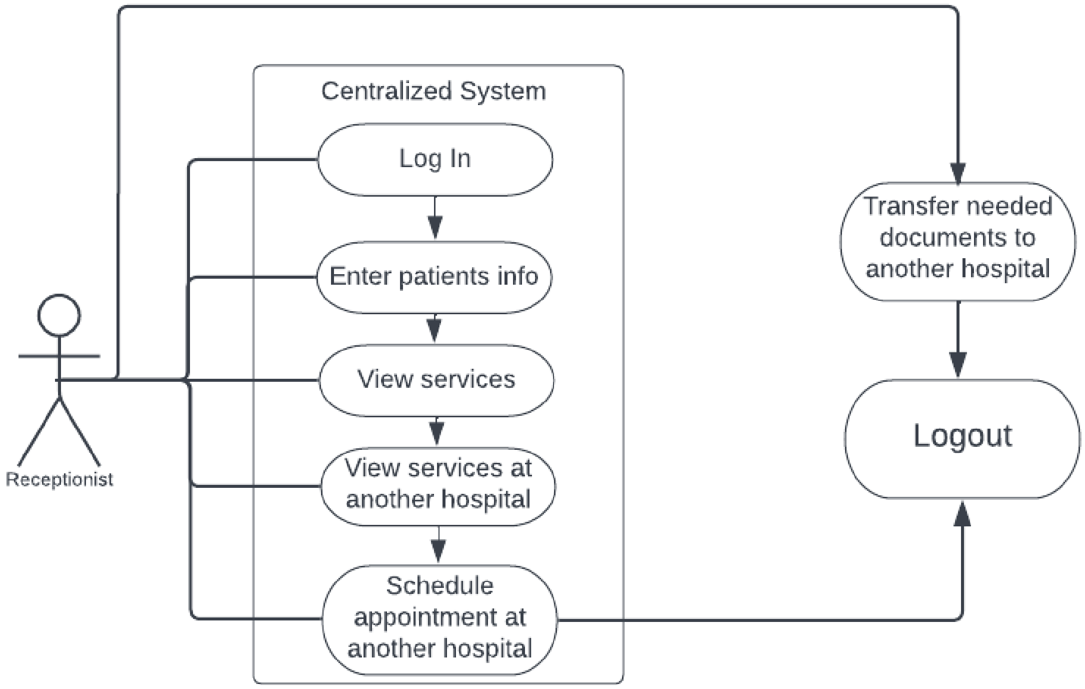
2) Update hospital in the system:

A hospital that was previously added to this system has integrated some features into its system. Now, this system ought to incorporate that update.

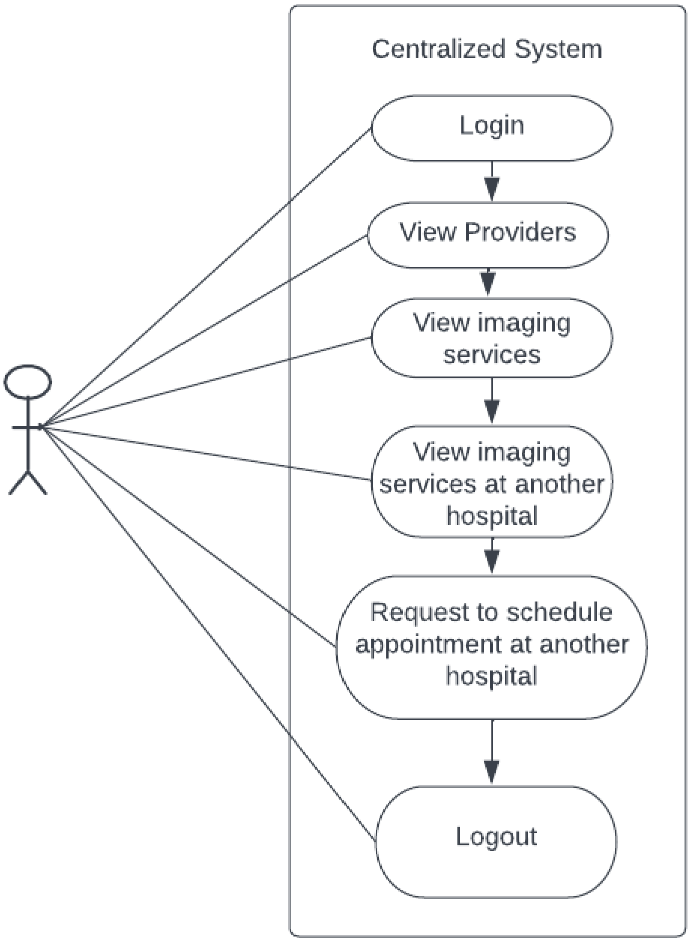


3) Delete a hospital from the system:

If a hospital was added to the database and must now be removed from the system for a reason like a hospital that is no longer in operation, the system should do so within 45 seconds of the hospital's normal operation.

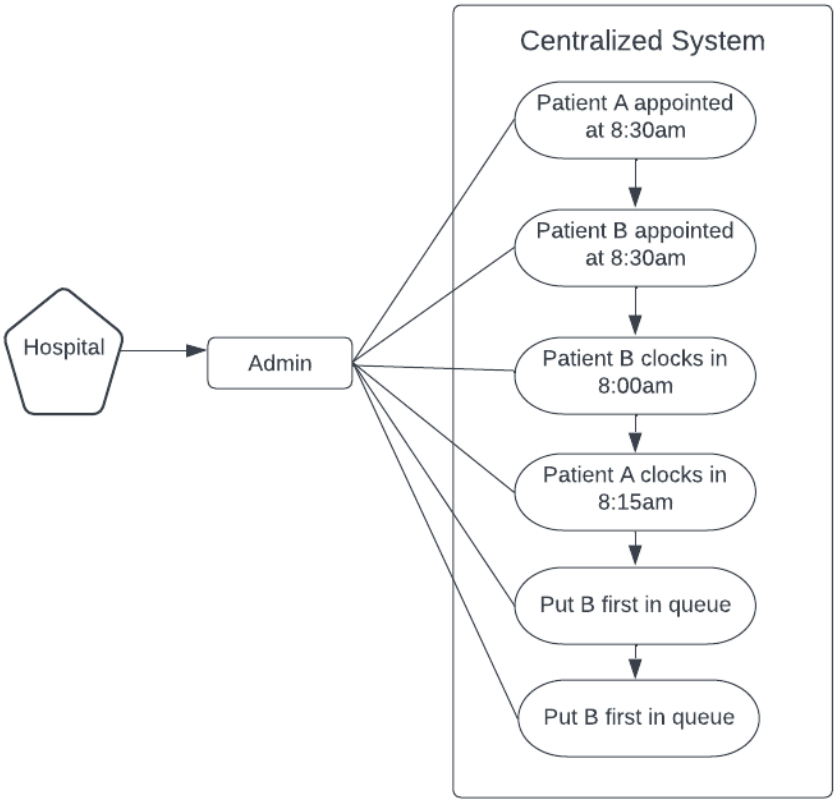

Centralized System
Log In
Enter patients info
View services
View services at another hospital
Schedule appointment at another hospital
Transfer needed documents to another hospital
Logout
Receptionist

4) Reopen a canceled slot for rescheduling a new appointment: A patient has an appointment on a given day, but a doctor cancels it on the same day and vice versa.


Centralized System
Login
View Providers
View imaging services
View imaging services at another hospital
Request to schedule appointment at another hospital
Logout

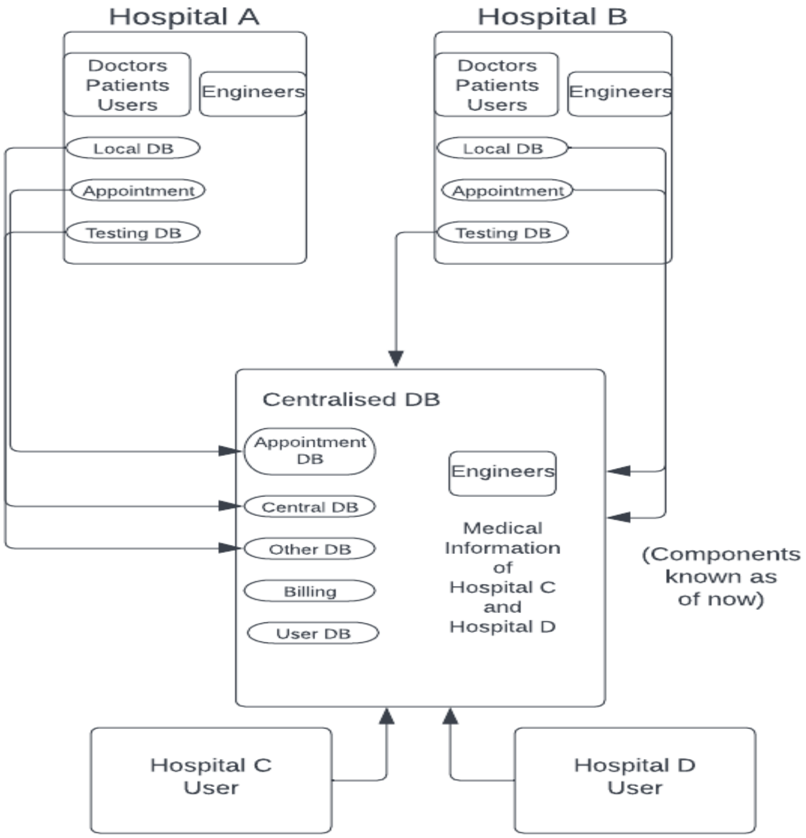
5) Search slot availability at another hospital: If a hospital does not have a service for some type of treatment then the hospital can schedule an appointment of a patient at another hospital with another provider.

6) Schedule appointment at another hospital: If a hospital has a specialty doctor, but does not have equipment for diagnosis, it should schedule a patient's appointment at a different hospital.
For instance, a hospital such as a primary care provider's clinic may have a doctor to read or interpret the result of an x-ray, but does not have an x-ray machine to take an x-ray. In a normal operation, the system should search for a nearby hospital that has x-ray service within 15 seconds. The result should be limited to 20 hospitals or clinics at maximum at a time.


Centralized System
Hospital
Admin
Patient A appointed at 8:30am
Patient B appointed at 8:30am
Patient B clocks in 8:00am
Patient A clocks in 8:15am
Put B first in queue
Put B first in queue

7) Retrieve the schedule

Given that a patient wants to schedule an appointment to visit a particular provider in a given hospital on a given day, however, there are no slots available anymore for that day. Then the system should provide a list of the schedule of that particular provider in that particular hospital for the next 30 days so that a patient can choose one that works for her the best. The system should pull up the schedule list in less than 15 seconds in its normal operation.



c) Architecturally Significant Requirements (ASRs)
i) QA scenarios for those ASRs.

Data Privacy general scenario-

SSO encryption provides a user with secured access privileges

Scenarios portion	**Description**
Scenario	1.Patients data is to be protected and only authorized users, by means of SSO Encryption whenever he logins into our system.
Source	Hackers or unknown users

Stimulus	Alert users of unauthorized attempt
Artifact	System information (Database)
Environment	System
Response	1. Data cannot be altered or manipulated without authorization 2. Date is protected from unauthorized access 3. Databases can be accessed only by authenticated users.
Response measure	1. How much data is lost or compromised is to be ensured? 2. How much data is accessed or vulnerable to the attack?

Accessed privilege general scenario-

Users who are authorized and have permission get access to limited resources needed to get her job done and nothing more.

Scenarios portion	**Possible Values**
Scenario	The patient information, medical history, payment invoice are to be securely stored in the database
Source	Receptionist or people from organization
Stimulus	Unauthorized attempt to system resources
Artifact	Database
Environment	System
Response	1.Data cannot be altered or manipulated without authorization 2.Date is protected from unauthorized access. 3.Databases can be accessed only by authenticated users.
Response measure	1.How much data is lost or compromised is to be ensured?

	2.How much data is accessed or vulnerable to the attack?

User authentication and authorization general scenario-

A user must wait five minutes before making another attempt if their credentials are not authenticated in three consecutive attempts

Scenarios portion	**Possible Values**
Scenario	The patient data is to be protected from any unauthorized user trying to login into the system.
Source	End user or receptionist
Stimulus	Unauthorized attempt into the system
Artifact	System services
Environment	System
Response	1.Data and services are protected from unauthorized access. 2.Data cannot be altered or manipulated without authorization.
Response measure	1.User will not be able to login after 3 unsuccessful attempts.

User appointment management general scenario-

A user at one hospital can retrieve, create, delete, and update appointments either at his hospital or any other hospital..

Scenarios portion	**Possible Values**
Scenario	Should be able to create, read, update and delete data of patient into the database
Source	Receptionist or organization’s employee

Artifact	System
Environment	System running in normal mode
Response	1.System returns a response when a query is updated. 2.System returns an error or generates no response if there is an overload in the system, increasing the wait time.
Response measure	1. Time the response takes 2. The number of requests that get updated and the requests that go unsatisfied.

Throughput general scenario-

Defines the request-response cycle per second and how fast the system responds back

Scenarios portion	**Possible Values**
Scenario	System should not lag and operate during its peak time with an expectancy of < 4000 user request per day
Source	External part of system
Stimulus	System receives more request than it can handle
Artifact	System database
Environment	System in normal condition
Response	System blocks additional concurrent requests from the user and displays a prompt message telling the user to try accessing the system after some time.
Response measure	1. Should not take more than 2 sec for the user to reattempt. 2.System should not crash for maximum request-response.

Steady state availability general scenario-

System should be available for most of the time and if it gets corrupt the downtime should be minimal.

Scenarios portion	**Possible Values**
Scenario	System should be available 99% for the users to be accessible
Source	Server of system or the software
Stimulus	Server crash or incorrect response
Artifact	Server, database storage, communication channels
Environment	Normal operations or shutdown system
Response	Record the primary cause of failure, address any flaws, cover up the flaw, and repair any harm the flaw causes.
Response measure	1. Time to detect the failure, if it's in the server or the system 2. Time to recover from the failure. 3. Mean-time the system prevents itself from failing.

Fault and log management general scenario-

Scenarios portion	**Possible Values**
Scenario	If there is a fault that has occurred the system must work in degraded mode until the fault is fixed
Source	Server of system or the software
Stimulus	Server crash or incorrect response
Artifact	Server, database storage, communication channels
Environment	Degraded mode or under normal operation
Response	Record the primary cause of failure, address any flaws, cover up the flaw, and repair any harm the flaw causes.

Response measure	1. Time to detect the failure, if it's in the server or the system 2. Time to recover from the failure. 3. Mean-time the system prevents itself from failing.

Continuous Integration general scenario-

Scenarios portion	**Possible Values**
Scenario	System should be able to accept new changes and new updated components and features within a month and if any defect arises it should be resolved within 4 person days.
Source	Stakeholders, developers,testers and architects.
Stimulus	When a new functionality needs to be implemented, changing business logic, change in UI of system or platform
Artifact	Data, code and components
Environment	Running time, deprecated mode, build time or during testing phase
Response	Modified changes in the code should be integrated and available to the end user and any change in database should be available to the receptionist or organization.
Response measure	1. Budget allocated for the change cycle 2. Time spent during designing, developing, testing and integrating the new changes. 3. Defects introduced during modifying or changing the code.

Upgrade version general scenario-

Scenarios portion	**Possible Values**
Scenario	Within every four months, the system must be updated to the most recent stable version of the

	dependencies. Ex. Database version upgrade, UI upgrade or business logic backend update.
Source	Architects and developers
Stimulus	The main goal of version control is to keep upto date with the latest dependency and latest code stack used in the market.
Artifact	Data, code and components
Environment	Build time, compile time
Response	The upgraded version should be available to the end user and any change in database should be available to the receptionist or organization in the deployed version.
Response measure	1. Budget allocated for the change cycle 2. Time spent during designing, developing, testing and integrating the new changes. 3. Defects introduced during modifying or changing the code.

Continuous delivery general scenario-

Scenarios portion	**Possible Values**
Scenario	The system should be available for more than 98% of the time to the users.
Source	Complete system
Stimulus	Any type of fault- system crash, server, database crash.
Artifact	Server, database, system, communication routes
Environment	Normal operation, shutdown or overloaded request-response.
Response	Record the error or the cause of the problem, then take corrective action. Reverse the damages that were done.

Response measure	1. Time it takes to detect the fault and failures affecting the system. 2. Time to repair the failure. 3. Mean-time the system prevents itself from failing.

ii) Utility Tree and the ASR prioritization.

Utility Tree

Quality Attribute	Attribute Refinement	ASR Scenario
Security	Access Privilege	Principle of least privilege is applied. For instance, a receptionist can update a patient's appointment, but not the medical records (H, H). → **ASR4**
	Data Privacy	The legitimacy of the source of incoming query requesting patient's information is strictly verified by means of SSO(Single Sign On) encryption(H, H). → **ASR1**
	User authentication and authorization	If a user's credentials are not authenticated in 3 consecutive attempts to login, a user needs to wait 5 minutes to make next attempt(H, M)→ **ASR5**

Performance	Appointment Management	A user at one hospital can retrieve, create, delete, and update appointments at another hospital. (H, H). → **ASR2**
	Throughput	When the system is busy, it should handle 100 request-response cycles per second. (M, M).→ **ASR9**
Availability	System’s Steady State Availability (SSA)	System recovery time from faults is very small. Availability percentage must be 99%, so that the downtime must be 5 mins and 15 secs which is the most desired one for our system. (H, H). → **ASR3**
	Fault detection and log management.	System logs the faults, if detected any. System works in degraded mode until the fault is fixed. For detection of fault Monitor shall be used. During the fault SSA will not be affected. (M, H) → **ASR6**
Modifiability	Continuous Integration	The system should be able to accept changes in its components as and when new features are integrated into the system. If any defects arise, it should be fixed within 4 person days. (H, M)→ **ASR7**
	Upgraded version	In every 4 months a new upgraded version of the system is to be installed with the stable

		version of new dependencies.Ex. UI upgrade, backend with added hospitals into the database. (M, M)→ **ASR10**
Deployability	Continuous delivery	Every five weeks, production must be updated with code updates, new functionality, or bug/defect fixes if any. (H, M)→ **ASR8**

ASR Prioritization

ASR1: The legitimacy of the source of incoming query requesting patient’s information is strictly verified by means of SSO(Single Sign On) encryption(H, H).

ASR2: A user at one hospital can retrieve, create, delete, and update appointments at another hospital. (H, H).

ASR3: System recovery time from faults is very small. Availability percentage must be 99%, so that the downtime must be 5 mins and 15 secs which is the most desired one for our system. (H, H).

ASR4: Principle of least privilege is applied. For instance, a receptionist can update a patient’s appointment, but not the medical records (H, H).

ASR5: If a user’s credentials are not authenticated in 3 consecutive attempts to login, a user needs to wait 5 minutes to make the next attempt(H, M).

ASR6: System logs the faults, if detected any. System works in degraded mode until the fault is fixed. For detection of fault Monitor shall be used. During the fault SSA will not be affected. (M, H).

ASR7: The system should be able to accept changes in its components as and when new features are integrated into the system. If any defects arise, it should be fixed within 4 person days. (H, M).

ASR8: Every five weeks, production must be updated with code updates, new functionality, or bug/defect fixes if any. (H, M)

ASR9: When the system is busy, it should handle 100 request-response cycles per second. (M, M).

ASR10: In every 4 months a new upgraded version of the system is to be installed with the stable version of new dependencies.Ex. UI upgrade, backend with added hospitals into the database. (M, M).

6) Architectural Decisions

a) Justification for architectural decisions

The justifications presented here, all reference the deliverable 2.

Design Decision for implementing functional requirement of Managing Patient's Account and Data

In this iteration in our deliverable II, we talked about creating, updating, deleting, and searching for a patient in the proposed system. We needed to maintain a high level data confidentiality of the patient's personal, medical, and financial record. We chose Data Privacy as one of the main qualities of the proposed system to be addressed, by seriously considering business goals as well as technical risks. We do not want, for example, a receptionist to access a client's medical record unless it is approved by a healthcare provider or other management personnel. There are many advanced tech stacks implemented by the developer team to avoid unauthorized access to confidential documents.

Design Decision for implementing functional requirement of Managing Users

There is no doubt that user authentication and authorization is the most sought after requirement for any application based system designs. This deals not only with user's security at individual level, but also at system level. So, the emphasis was put on the security quality attribute while making decisions here.

Design Decision for implementing functional requirement of Managing Appointments

For any system, the request response cycle time is one of the important aspects because every transaction depends upon the time. Because it is an important task to do, i.e. if the front desk officials have to book an appointment at another hospital, then they have to send a query and they have to get the reply within a specific time, if not they will resend it or the failure occurs. This involves database CRUD functionalities. So, the request response cycle time was given utmost priority, and the design decision was drawn that appointment management holds the highest priority both by means of technical risk assessment as well as the project goal. We put emphasis on performance because latency issues are faced by the system if there is high request response traffic to and from the system and also that resources are shared between the different elements of the system and the database, which is connected to the cloud because ours is the web based system.

Design Decision for implementing functional requirement of Managing Faults and Logs

In this phase we talk about performance and how it becomes a highly sought after attribute for our system. Performance is the main quality attribute for the system. Latency issues are faced by the system if there is high request response traffic to and from the system. Resources are shared between the different elements of the system and the database, which is connected to the cloud because ours is the web based system.

Design Decision for implementing functional requirement of System Maintenance

For the system updation and deletion is the common thing for the admins and other people to do. So, there are some situations in which the database does not behave in a desired way. For that, we have an alternate database in which it can be used as the primary one and the system can work in the degraded mode while the system works on the repair.

b) ASR prioritization justifications

The justification for ASR prioritization is based on the utility tree provided in the section 5 above.

The **ASR1** prioritization deals with user authentication and authorization in addition to other important security risk factors as mentioned in the scenarios in the utility tree. Since this deals with the data confidentiality or the privacy of the data, there is an associated risk in implementing it as well as meeting the business goal without any adversaries as we decided that it is (H, H).

ASR2 also deals with data privacy as it has to do with accessing schedules of one participating hospital from another participating hospital. The technical risk analysis and the business goal assessment indicated that this holds the second priority in the list of architecturally significant requirements because we categorized it as (H, H).

ASR3 deals with the System recovery time from faults. Availability percentage must be 99%, so that the downtime must be 5 mins and 15 secs which is the most desired one for our system. We prioritized it as (H, H), and thus considered that it has to do with the performance of the proposed system.

ASR4 has to do with access privileges. This then references security quality attributes. We decided that the principle of least privilege must be applied and thus assessed it to be (H, H).

ASR5 talks about user authentication and authorization. It is the security feature and is thus assessed to be (H, M). Since it is technically moderate to implement whereas from a business perspective most needed, we prioritized it to be in the top 5 ASRs.

ASR6 encompasses the idea of system maintenance where it deals with fault and log management. Getting correct faults logged to appropriate files is very important for future system maintenance, the decision was made that it holds the risk analysis of (M, H).

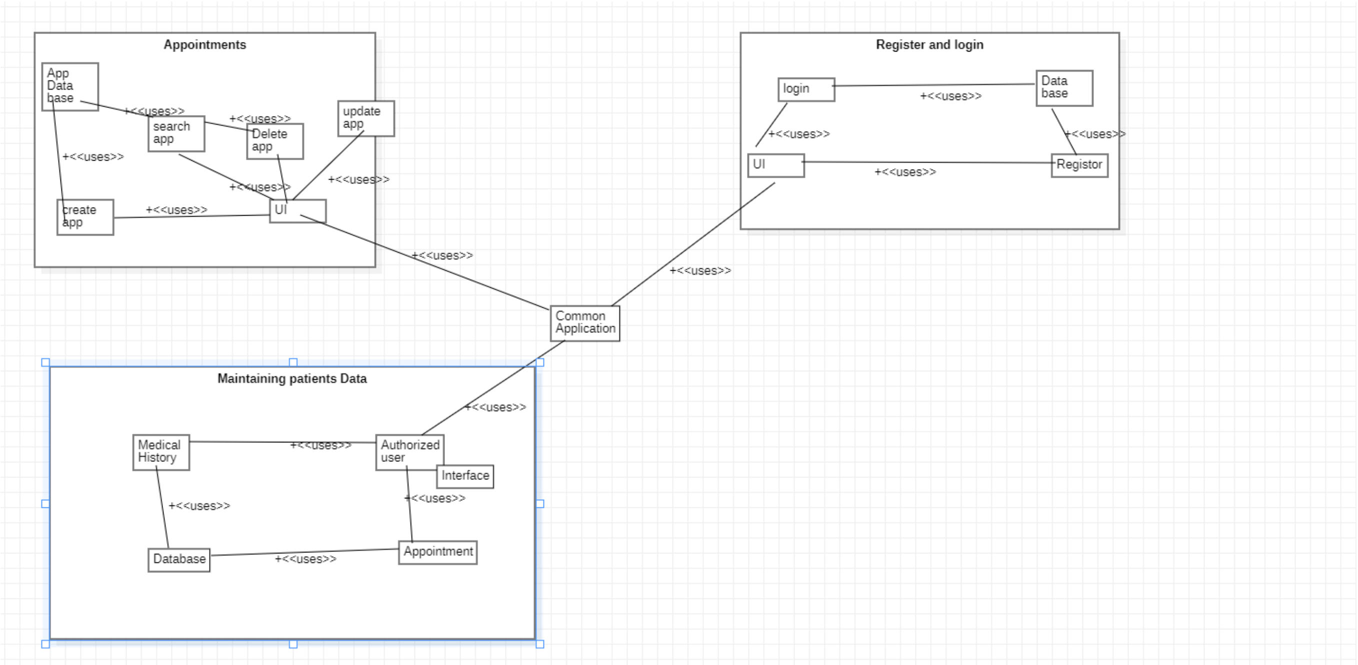
ASR7 works with the quality attribute of integration. It specifically deals with continuous integration. So, a decision was made to consider it should be analyzed from the view of deployability. Since deployment is an essential attribute for any system to get its life on the internet, it is risk analyzed to be (H, M).

ASR8 considers the deployability quality attribute and deals with continuous delivery. Since deployment is very important for any interactive systems, the risk analysis helped us make the decision to be (H, M).

ASR9 has to do with performance. It specifically deals with throughput of the system. Since performance is equally important for the system to render user friendly service in the application based system, we risk analyzed and concluded that it should be (M, M). We concluded that without this the system won't fail, however, will have some consequences.

ASR10 emphasizes the quality attribute of modifiability. The approach of modifying the system is crucial for the web-based system as it causes severe system downtime if the upgradation of the system fails due to unforeseen reasons. However, our risk analysis concluded that it should be considered (M, M).

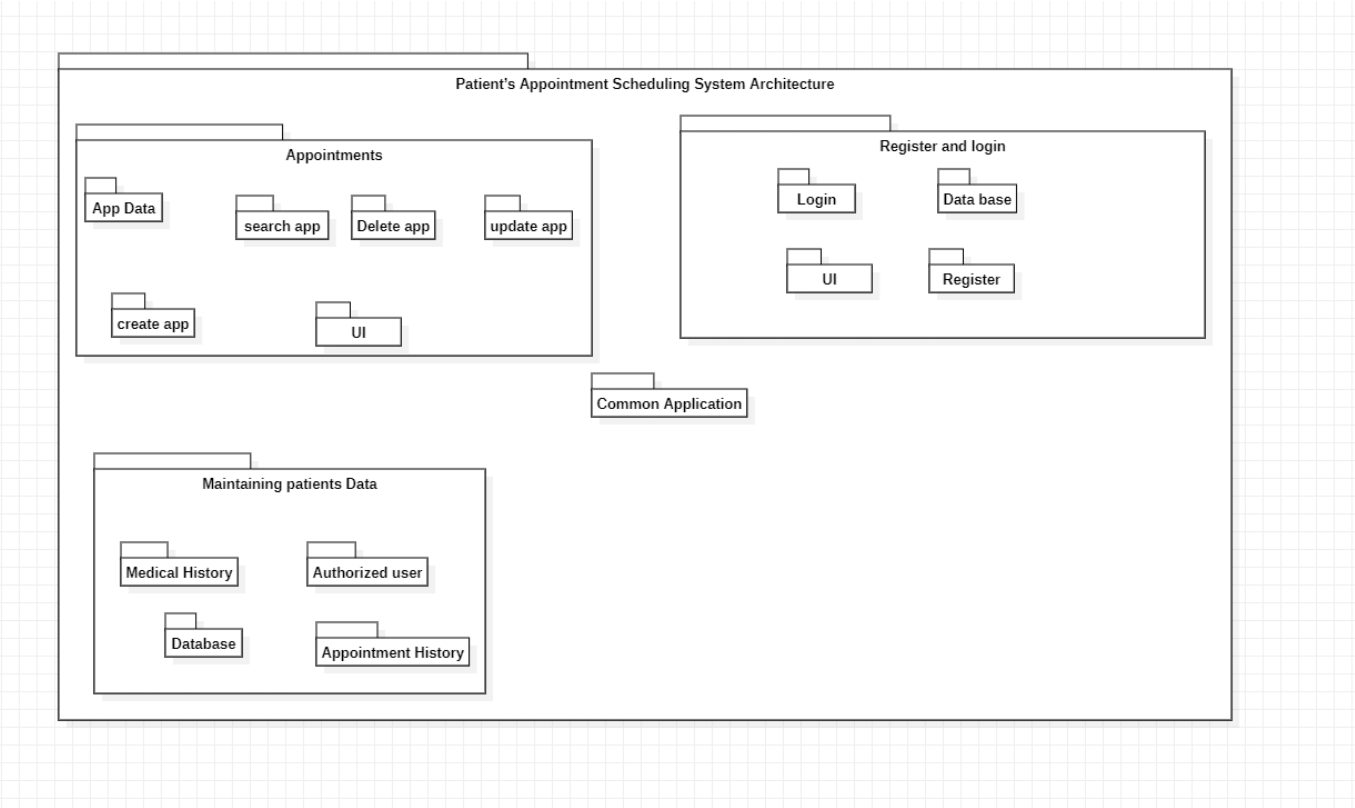
7) Module Views



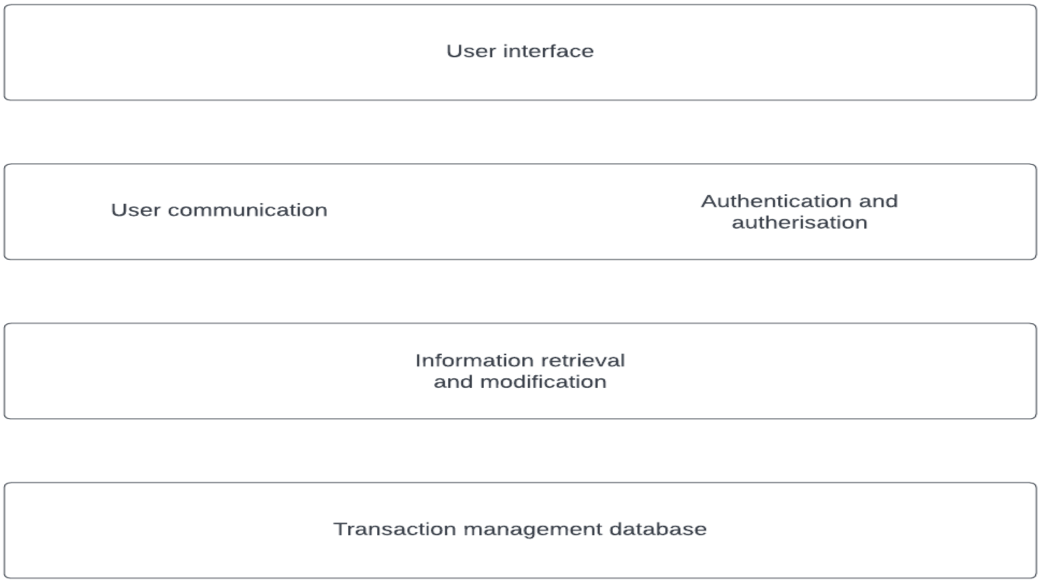
This is the module view for our architecture and they can be divided into various segments like the register and login, Appointments, Maintaining patients data like these are the most critical modules and their organization of that.

Modules and the Layers responsibilities are as follows:

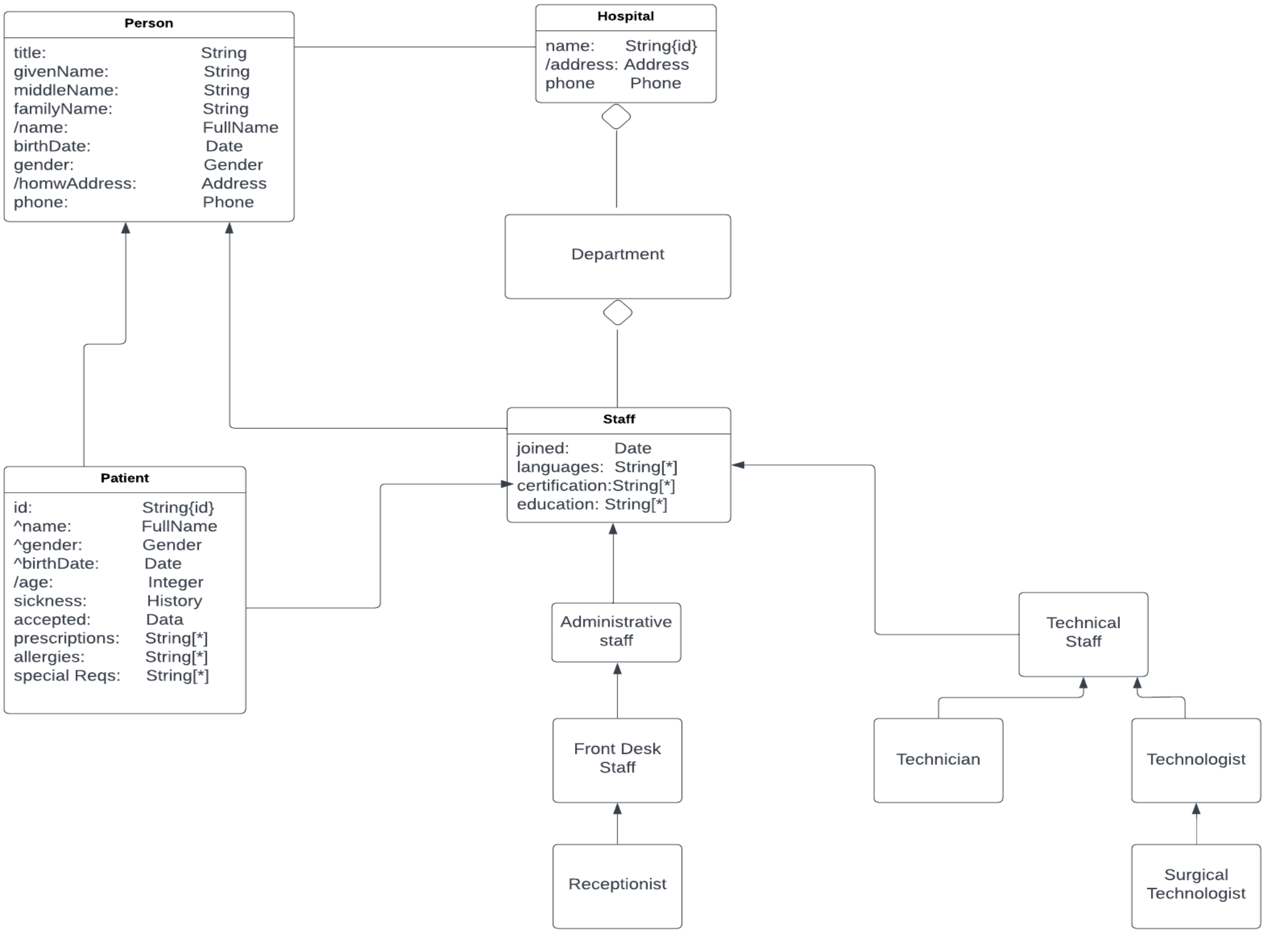
Modules are as follows like the Appointment management, Register and login and maintaining patients data. The appointments must be scheduled properly without any disturbances and the clashes in the schedule and the responsibilities are allocated based on that. For Register and Login it is as follows as the User Interface must be friendly and easy to understand. Maintaining patients' data is like keeping all the data in the particular fashion, So that they can be handled easily and they can be taken from the database whenever there is the need to use it.



This is the entire system module view and the above one tells about which one uses which one but this one tells about the entire representation of the system in the module view. The common app mainly tells about the Application which is nothing but the software which we developed in this process and what it does in the entire process like how it coordinates with the next ones. Like the appointments and the login and register these both must be worked on the singled hand like there must be the to and fro connection between them so that they can give the maximum efficiency from that.


User interface
User communication
Authentication and autherisation
Information retrieval and modification
Transaction management database

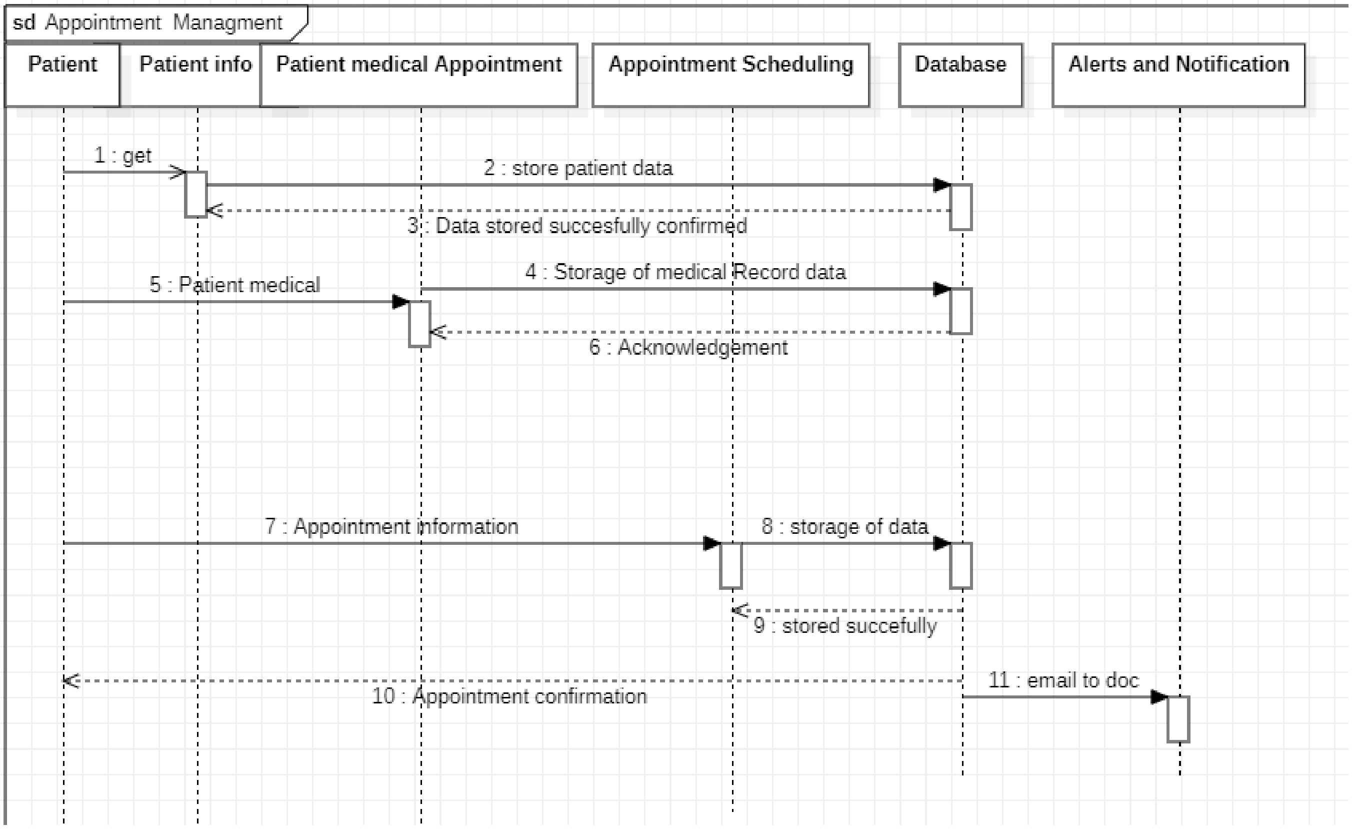
This is the Layered view for our architecture. Which has layers as per the diagram in which we have the User Interface and the levels are as follows which is like the user communication and the information retrieval and modification. Which last has the Transaction Management Database which has the following which is the top down approach from the top to down.



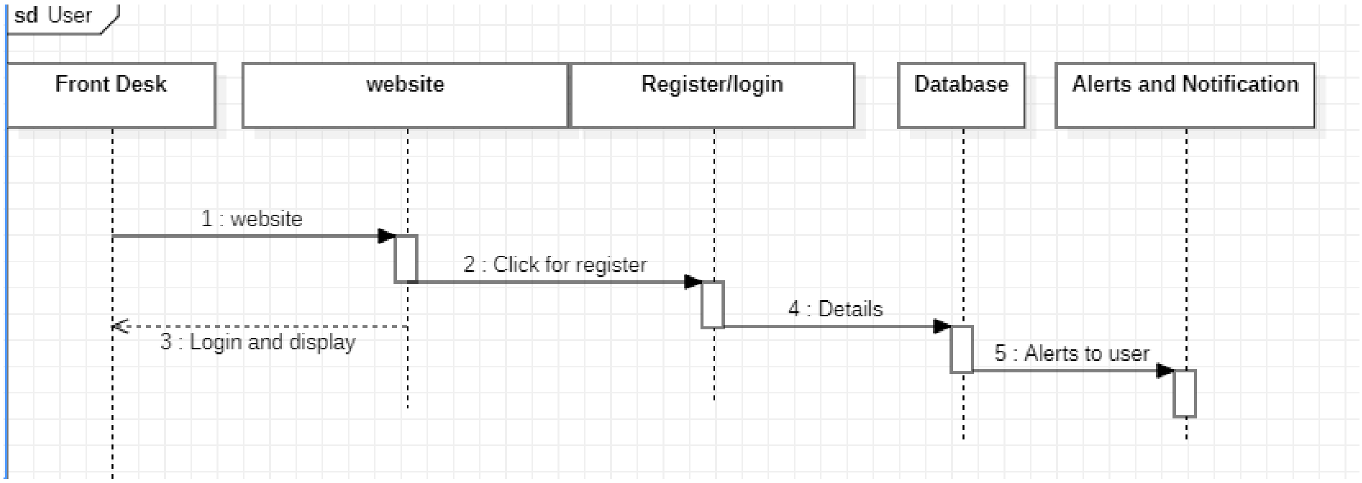
This is the class diagram for our project. It states all classes and the attributes for the classes of the hospital, staff, patient and many more. Like the administrative staff , these are things which we need to concentrate on like the Id, sickness etc . This helps the classification of the users very easily and the properties also can be extracted from this very easily so that they can be done.

8) Behavioral Views

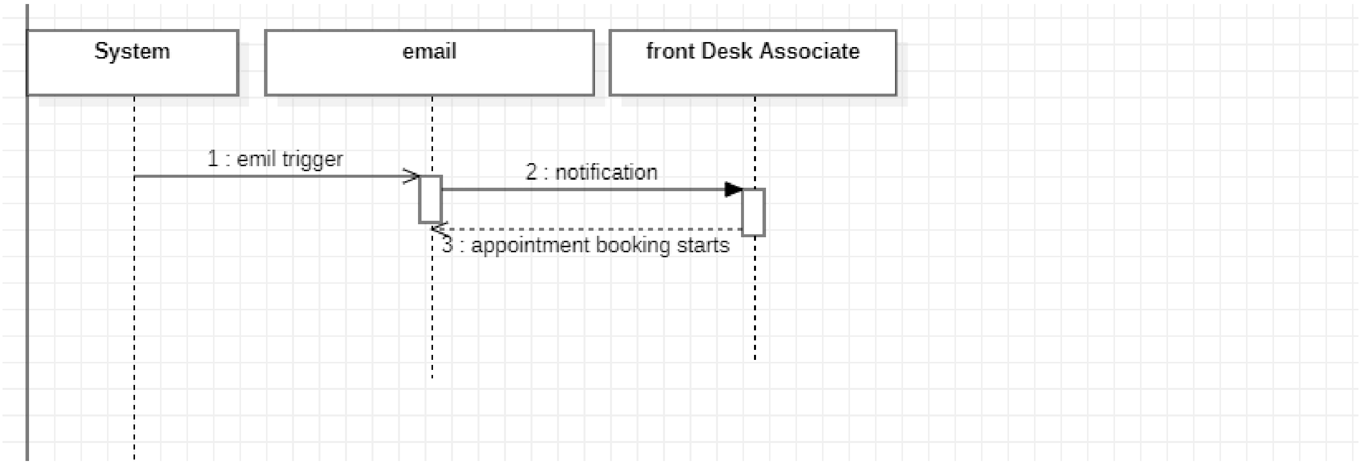
a)



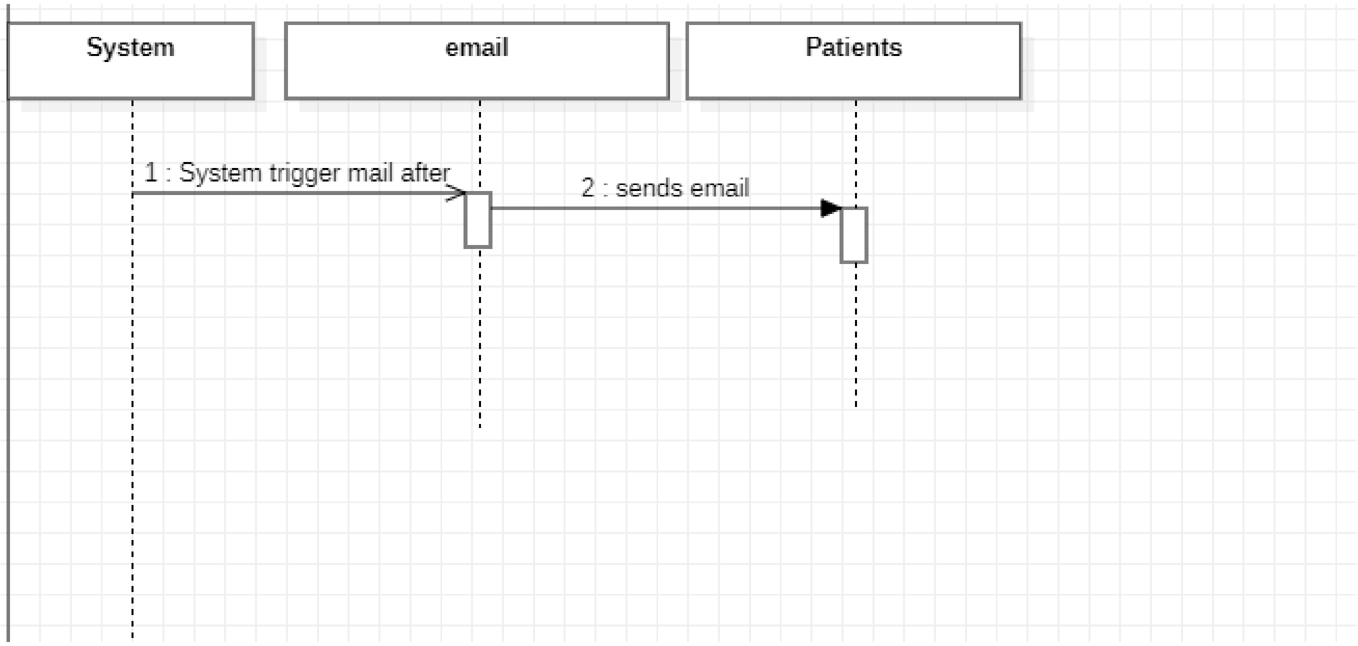
This above image shows the behavioral view of the Appointment management of our software. As we can see that the steps are indicated very clearly and the basic general process is done by that. The First step is to get the patient's data and then we have to store it in the Database for our future reference. These are the steps which are followed by the system in the sequential manner. For each step there will be acknowledgement from the components. Here in this case if the new patient data is created then we have to take the personal details and also the medical history of the patient stating that this is the patient history and she has the particular disease which we have to cure. So in that case we have to modify the data. Means if there is a request by the patient stating that they have to reschedule the appointment so we have to modify the existing data so the tactics for this we used here is to reduce coupling like that. Reduce coupling is nothing but Encapsulating the direct access to some part of the components and restricting dependencies among services. For instance, a receptionist can only view the appointments of the patient's, she does not have access to the medical records and the financial invoice of any patient. So we encountered the situation of the appointment management.



This is the sequence diagram for the users. But in our software the users are mainly the front desk receptionists. SO for them to register and login they have to go to the particular process a=and we can call it as the sequence and it must be followed based on these they can be accessing the records. As here we can see that if the user is new then they will go to the website and then they will login and they will register for that.

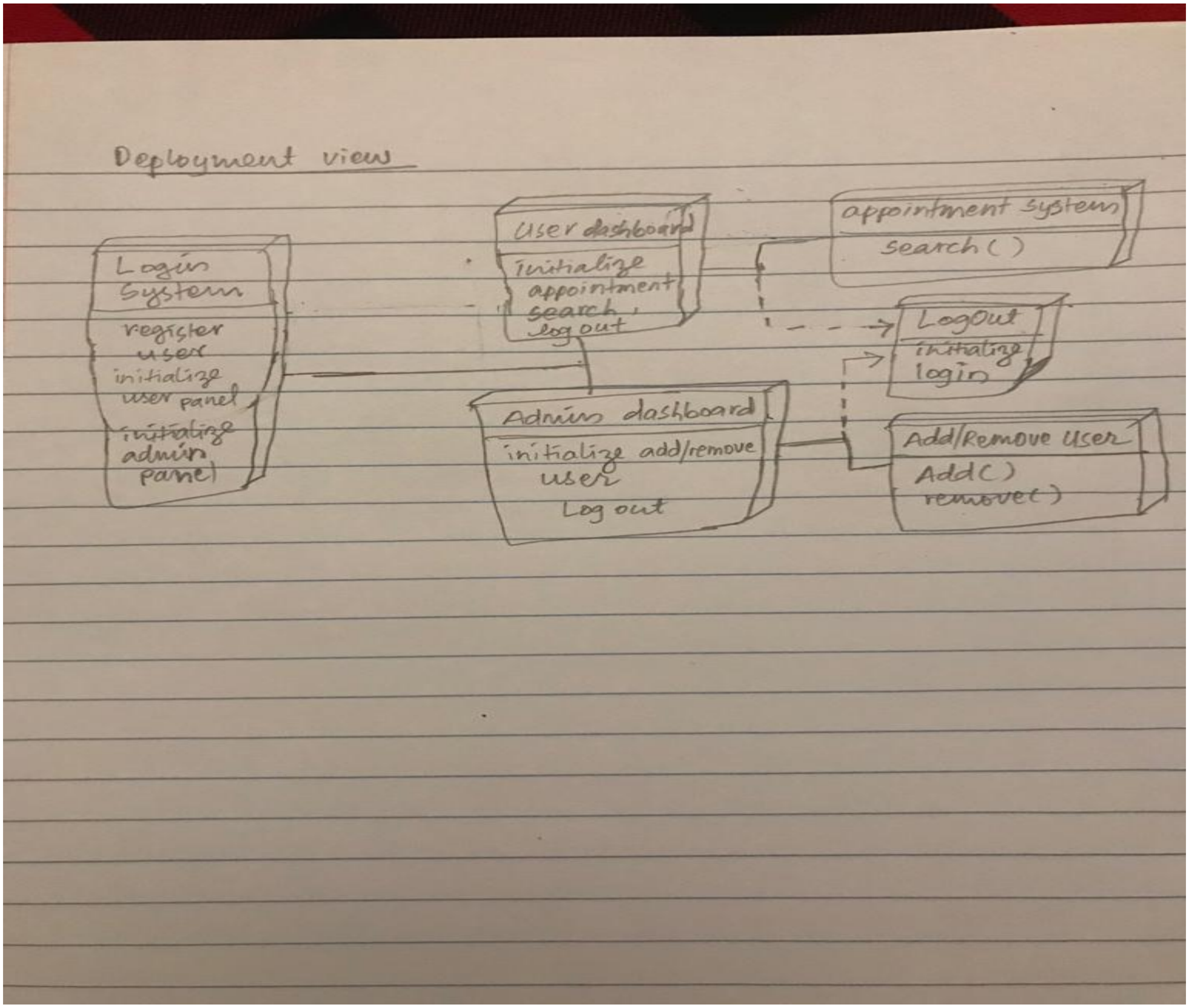


This is about the sequence diagram between the system and the front desk associate so that the communication between them can be smooth and they can be done. In this case it is about the email trigger to the user in our case it is the front desk associate. It is because in this case the notification is sent to the front desk when they try to login stating that the login is successful and then after that the entire process is done. Like the updation of the records and the appointments bookings like that will be started from that point.



This is about the sequence diagram between the system and email with the patients which states that the Whenever the appointment is booked with doctors on the particular date then on that day prior to the appointment both the doctor and the patient will get the notification. We can say that this is the reminder for both doctor and patient.

9) Deployment View



Physical nodes are the physical computers that users use to interact with the proposed system. So, the physical nodes include devices such as receptionist's desktop computer, healthcare provider's computer, and hospital administrator's devices. There are two types of tactics to be considered here.

Managing Deployment Pipeline-

1-Scale Rollouts: Release the new version of the service in stages, under close supervision, and without notifying users one by one. This method mitigates some of the fallout from launching a subpar service.

2-Rollback: When a deployment is flawed and fails to meet users' expectations, the change is rolled back.

3-Commands for Script Deployment: Long and complicated deployments usually rely on a series of commands that have been meticulously planned and rehearsed.

Managing Deployment System-
1-Manage Service Interactions: It is possible to deploy and simultaneously run multiple versions of a system service. In order to independently release both the old and new versions of the resources, it is recommended to make a copy of them.

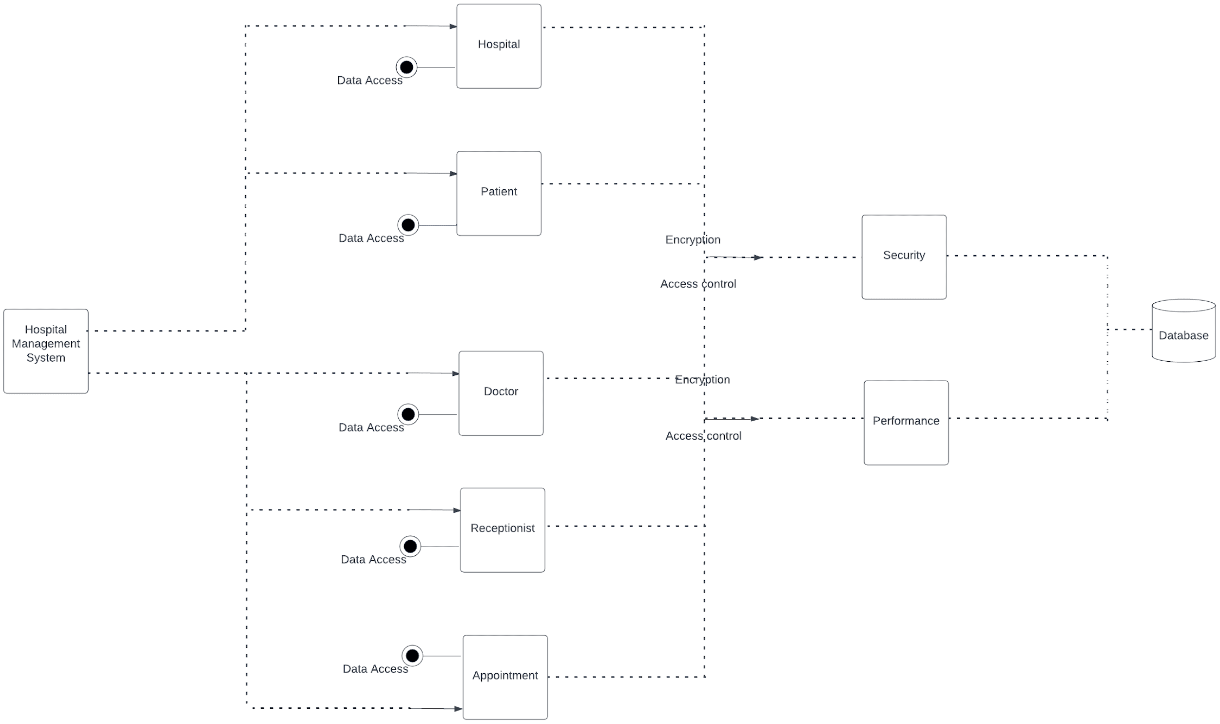
2-Package Dependencies: contains everything needed for installation in a single file. Dependency packaging can be done in a number of ways, including containers, pods, and virtual machines.

3- Toggle Features-In the event of an issue during deployment, this process will be executed to deactivate the updated function.

Patterns-
Based on the trends we've seen, changes should be seamlessly integrated with any subsequent changes made by other developers and pushed to the server instantly. Due consideration must be given to the structure in determining which services must be deployed and how.

Microservice Architecture: The microservices architecture allows for the creation, deployment, and upkeep of microservices architecture diagrams and services independently from the rest of the application.

Complete replacement of services: It is feasible to implement changes all at once or in phases, and to do so in a way that reduces environmental impact. to the extent that the most recent updates supersede all prior versions. Services are only "replaced" when both old and new functionality are given access to the user.



This is the Component view of our architecture and they are shown as above which has the Hospital management system and which deals with the doctor, Patients, Receptionists, Appointments these are the things which are needed to take care. Data access is the access point which is needed to take care. Then the data is encrypted for access.