Hospital Management System

System Design

Version 1.0

27.12.2020

Group 5

Prepared for

SE301 Software Engineering



**Table of Contents**

[1.](#_heading=h.gjdgxs) Introduction 1

[1.1.](#_heading=h.30j0zll) Purpose of the System 1

[1.2.](#_heading=h.1fob9te) Design Goals 1

[1.3.](#_heading=h.3znysh7) Definitions, Acronyms, and Abbreviations 1

[1.4.](#_heading=h.2et92p0) References 1

[2.](#_heading=h.tyjcwt) Current Software Architecture 1

[3.](#_heading=h.3dy6vkm) Proposed Software Architecture 1

[3.1.](#_heading=h.1t3h5sf) Overview 1

[3.2.](#_heading=h.4d34og8) System Decomposition 1

[3.3.](#_heading=h.17dp8vu) Hardware Software Mapping 2

[3.4.](#_heading=h.3rdcrjn) Persistent Data Management 2

[3.5.](#_heading=h.26in1rg) Access Control and Security 2

[3.6.](#_heading=h.lnxbz9) Global Software Control 2

[3.7.](#_heading=h.35nkun2) Boundary Conditions 2

[4.](#_heading=h.1ksv4uv) Subsystem Services 2

[5.](#_heading=h.44sinio) References 2

SYSTEM DESIGN DOCUMENT

# Introduction

Hospital Management System is an online hospital system that handles different directions of clinic workflows. It manages the smooth healthcare performance along with administrative, medical, legal and financial control.

## Purpose of the System

The system is designed for making the connection between users and hospitals better and easier. The hospital management system’s main users are admin, doctors and patients. The users need to create an account and login to the system to be able to use the system functions.   
The doctors that work in a hospital which uses the hospital management system can easily view their calendar and appointments, can view their patient’s information and can inform the patient about their treatment online.  
The patients can book, view, change and cancel appointments, view the doctor that will examine them, also can view their Patient Page which shows their test results, diagnosis and all information about their health.  
The Admin will use the system to control the system functions and will control the Doctor and Patient’s movements.

## Design Goals Usability: The system should be easy to use for every type of user. In first login, users

## should easily understand the functions of the system and should be able to use every function of the system without being lost in the system. It should be accessible, easily understandable and should contain a description or a mini tutorial about the system. The system’s interface should contain all of the main functions on the main page. When the user login to the system, they should see Appointment, Doctor List, Patient Page, Profile Page navigations and should be able to find the answer to their needs.

## 

## Reliability: We need that the system should be reliable. It means that the system needs to have data loss tolerance, it should save the backup data daily and this backup data should be accessible immediately if needed but the system should not allow anyone to abuse the backup data or the actual data. The system should easily handle the exceptions. No one needs a system that crashes easily.

## Security: The system's safety and security is so important because the system will save the data such as personal information like addresses, phone numbers, identity numbers. For example, any patient should not be able reach to the other patient’s or the doctor’s personal information.

## Speed: The response time of the system is so important. When a user's request an operation and response time goes like 1 minute or even more, the user is not going to like the bad performance and is going to look for a better system. So it is so important to answer the users' needs.

## Performance: There will be multiple users that use the system at the same time and we expect from the system to respond to every user quickly and effectively.

## 

## 

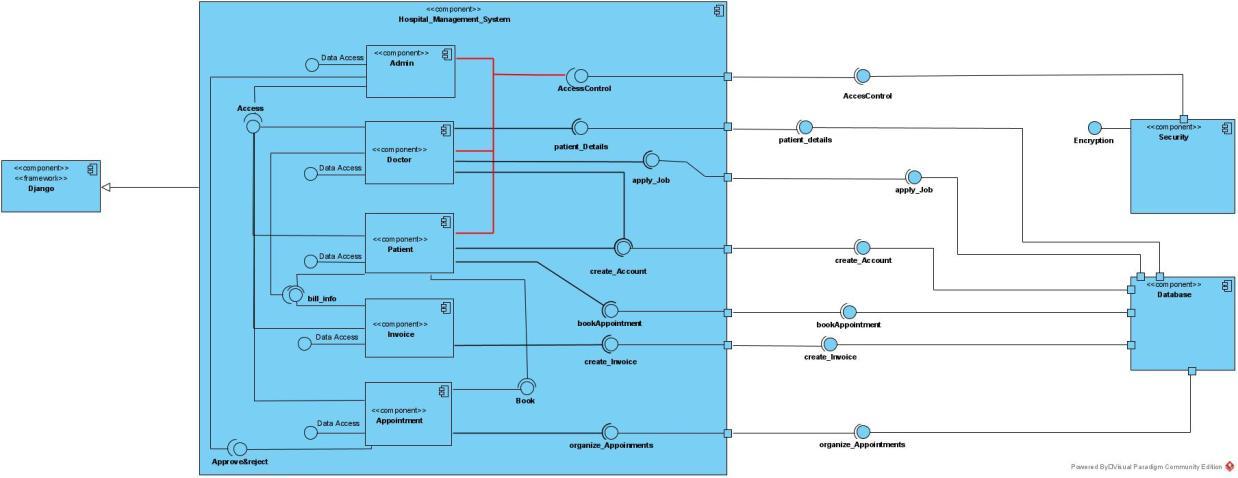
## References

The web pages we researched for our SDD:

1. <https://www.scientificamerican.com/article/fact-fiction-encryption-prevents-digital-eavesdropping/#:~:text=SSL%20uses%20a%20cryptographic%20system,browser%20to%20the%20Web%20server.&text=(Note%20that%20URLs%20requiring%20an,instead%20of%20%E2%80%9Chttp%E2%80%9D.)>
2. <https://www.wi-fi.org/discover-wi-fi/security>
3. <https://tr.wikipedia.org/wiki/Eavesdropping>
4. <https://www.techopedia.com/definition/13612/eavesdropping>
5. <https://www.w3schools.com/sql/sql_injection.asp>
6. <https://realpython.com/prevent-python-sql-injection/>
7. <https://portswigger.net/web-security/sql-injection>
8. <https://www.theguardian.com/technology/2016/dec/15/passwords-hacking-hashing-salting-sha-2#:~:text=Salting%20is%20simply%20the%20addition,same%20salt%20for%20every%20password>.
9. <https://auth0.com/blog/adding-salt-to-hashing-a-better-way-to-store-passwords/>
10. <https://www.thesslstore.com/blog/difference-encryption-hashing-salting/>
11. <https://medium.com/@ahmetseyhan/sha-256-kriptografik-hash-algoritmas%C4%B1-294d13f0ca5d>
12. <https://wpamelia.com/login-page-design/>

# Proposed Software Architecture

## Overview



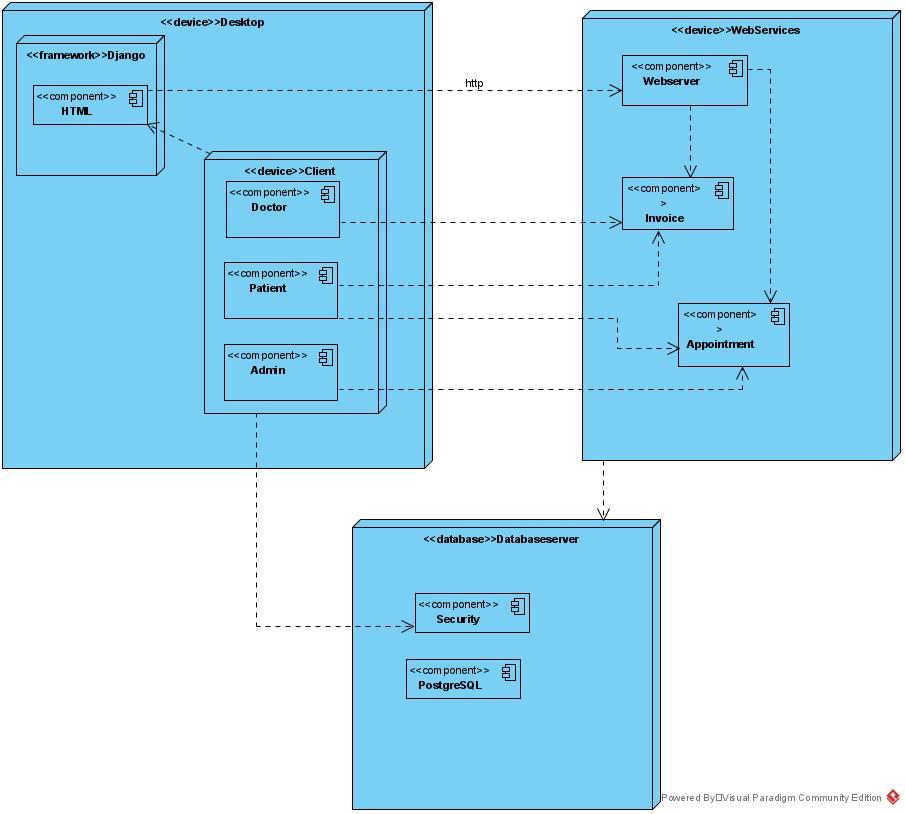
## System Decomposition

Our system for this project is Hospital Management System. In our system, we have 3 different user types; Admin, Patient and Doctor.   
First of all, we designed a system where patients can create their account and login to the system to use the system functionally. They can book appointments for their medical treatment by viewing and choosing the available dates on the appointment page, they can view the Doctor List and can choose the doctor they want for their medical treatment and can view and choose the date for that doctor. They can view and cancel their awaiting appointments easily. Patients have their personal page where they can enter their necessary personal information.   
Patients will have a Patient Page separate from their personal page which they can view and download their treatment information, test results, diagnosis, prescription information and invoice pdf.  
Our main goals for the second user, Doctor, are to make it easy to view their daily appointments, and keeping the information about the patients in one place like their medical history, their test information and results etc. They can use the system to save their patients’ diagnosis information and view them later on or next appointments easily.  
Doctors will have an account which is created by the system Admin and they can login to the system by using that account. They will have a personal page where they can enter their personal information, their education and job history etc.

They can view their appointments on the Appointment Page which are booked by the patients and their patients’ information like their medical history, previous treatments and diagnosis, the medicines they use and other information about the patients. They can change the Patient Page and enter the diagnosis and prescription information for the patient on the Patient Page. They can view the test results on the Patient Page.  
For the last one, Admin is capable of controlling the system, can organize things about the system, can use the Admin Page. They can create accounts for Doctors, and they can approve the patient’s accounts which are created by patients. They will also have an account that is specialized for admins where they will can view the patient’s and doctor’s personal information and can change the information if necessary.   
Admin can book appointments for patients, or they can approve or reject the appointment requests. They can change and cancel the appointment which is mostly based on patients’ requests. They can edit and view the Patient Page same as the Doctor. They can also edit the Doctor List. Admin can create and edit Invoice Pdf which can be viewed and downloaded by patients.

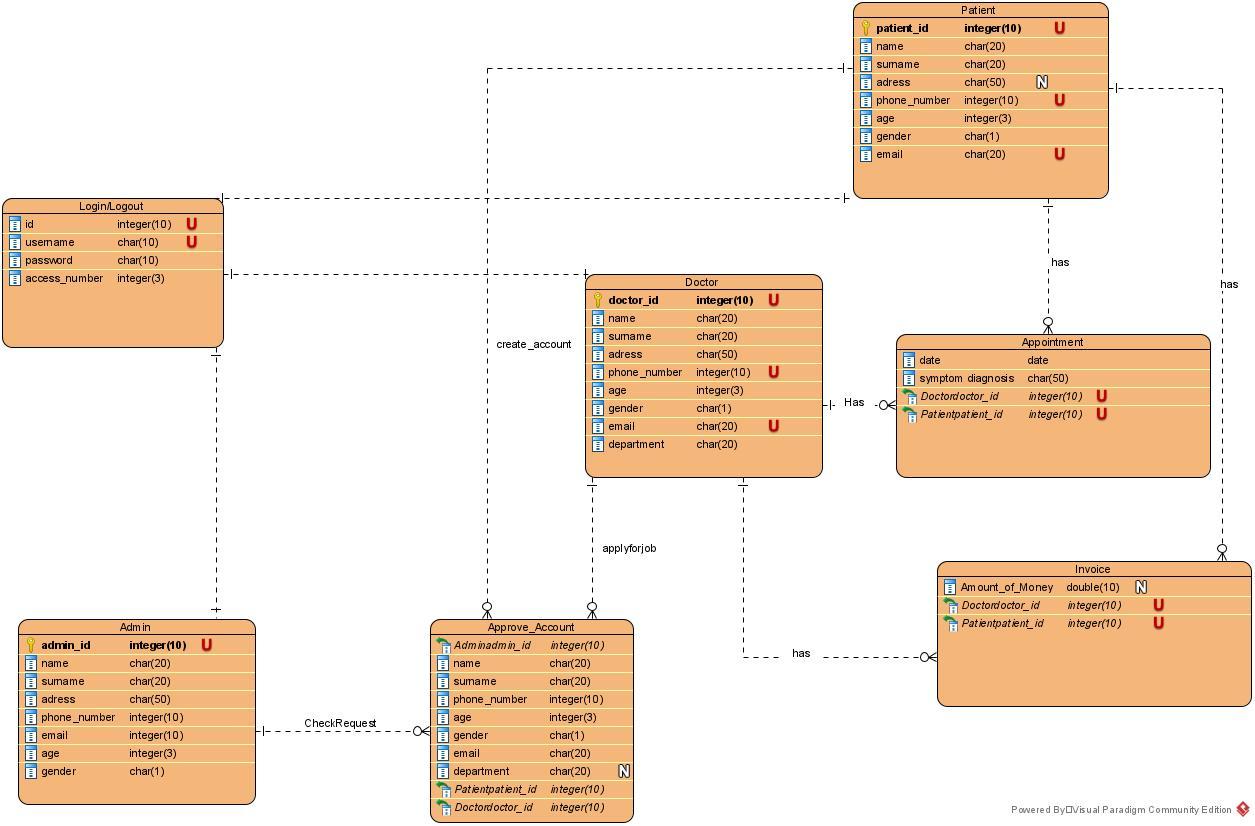
## Hardware Software Mapping

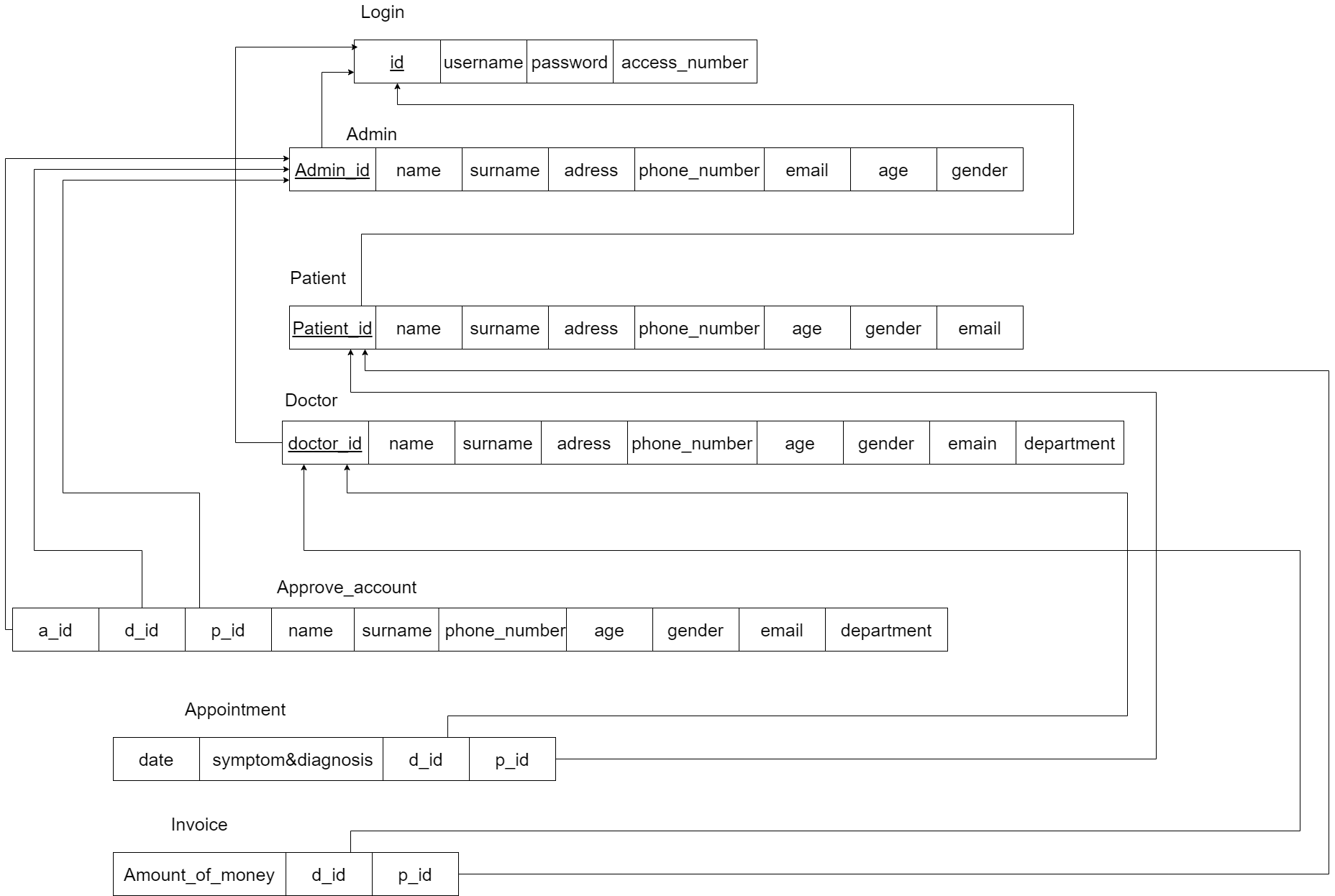
## The photo below shows the hardware-software mapping. The relations between subsystems and components used in our system are shown.



## Persistent Data Management

## In the photo associated between the database tables below, the tables required for our system and the relationships between them are shown. The login\_logout table is the table set up to log into the system. Admin patient doctor tables are for recording data according to user types. The Appointment table is for making the operations related to the appointment. The Invoice table is for the transactions to be made about the invoiced information. The Approve\_Account table is for the patient and doctor's applications under the control of admin.

****

****

## Access Control and Security

**Access Control:**

We created an Access Control Matrix which describes access control relation between functions and the users. It shows the capability of each user that uses the hospital management system.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 🡪 | **Appointment** | **Invoice** | **Create\_Account** | **Patient\_List** | **Doctor\_List** | **Enter&Exit\_The\_System** | **Edit\_profile** | **Get\_payment** | **Sypmtom and Diagnosis** |
| Admin | Approve()  Reject()  Book()  Cancel()  View() | Create()  View() | SignUp()  ApproveTheAccount() | View()  Edit() | View()  Edit() | Login/logout() | X | Control() | View() |
| Doctor | View() | View() | Applyforjob() | View()  Edit() | View()  Edit() | Login/logout() | Edit() | Take() | Create()  View() |
| Patient | Book()  Cancel() | View() | SignUp() | View() | View() | Login/logout() | Edit() | Pay() | View() |

**Security:**

Login process is for registered users. When the user sends a request, the system checks the relevant data in the database. The system allows or denies the user to enter the system according to the information stored in the database. If a user with a username and password is not found in the system, the system is not allowed to log in, if the user exists in the database and the username and password match, the system is allowed to log in. If these checks are not made. Hackers can break into our system. For example, they can overload the system by logging in with a billion different IPs at the same time.

Salting process: It is very important to use a strong encryption system. The important point here is the security of a user's information. It is necessary to protect users' information against all attacks from outside. Basically, hackers try to hack users' passwords with Dictionary attacks or brute force attacks. To prevent this situation, we will salt users' passwords with SHA256, so it would be very difficult to hack users' passwords. On the other hand, even people who have access to the database in this way will not be able to see users' passwords.

Another problem is SQL injection .SQL injection is a code injection technique that can destroy your database. SQL injection is one of the most common web hacking techniques. SQL injection is the injection of malicious code into SQL statements through web page input. SQL injection usually occurs when you ask a user for login such as username / user id, and instead of username / id it gives an SQL statement that you will unknowingly run in your database.For example:

txtUserId = getRequestString ("UserId");

txtSQL = "SELECT \* FROM Users WHERE UserId =" + txtUserId;

Here if the hacker enters the statement “105 or 1 = 1”,

Then, the SQL statement will look like this

SELECT \* FROM Users WHERE UserId = 105 OR 1 = 1;

It will understand that this can authorize access to the system or even delete all the data in the system. We will use SQL parameters to protect a website from SQL injection. SQL parameters are values ​​that are added in a controlled manner to a SQL query at execution time.For example:

txtUserId = getRequestString ("UserId");

txtSQL = "SELECT \* FROM Users WHERE UserId = @ ";

db.Execute (txtSQL, txtUserId);

Finally , the term 'eavesdropping' is used to refer to the interception of communication between two parties by a malicious third party. To prevent this 'eavesdropping' event, we will encrypt the transmitted data before sending it to one user to another user. So that any third party trying to read the data will only see a string of meaningless words. We are going to use WPA3 wireless encryption.

## Global Software Control

Our global software control service for our Hospital Management System is a website which uses web browser applications such as Google Chrome, Mozilla Firefox, Internet Explorer etc. If users want to use our system, they need to have internet connection and a web browser on their computer or smartphones.   
Admin can use the system to control the system backend.   
Users send requests to the system database and system replies related questions such as appointments, profile pages, doctor lists which are saved on our database. As replies to the requests from users, the server replies it with corresponding services or continues listening to the next requests.

## Boundary Conditions

**Initialization:**

**Start-Up:** When the system starts working, the servers and the database will start working and ready to accept the requests and reply to them, record the data for every change that is made on the system about patients, doctors, appointments.

**Login:** Hospital Management System has three types of users; Patient, Doctor and Admin. Users can login to the system by using a web browser on their PC. Users will have three different login pages and they will need to choose the user type to login the system. First, users need to choose the user type to login to the system. After that, users need to enter their ID and password in order to login to the system. If ID and password match an entry, they will successfully login to the system. If the ID or password is not correct or there is not a match, there will be an error message and the user will not be logged in.

**Termination:**

**System Shutdown:** If there is maintenance for the system, the system needs to be shutted down and once it shutted down it will not be available for users except for the admins of the system and while shutted down, any changes on the data will not be made in the system. That should be done to handle the updates and not to lose any data during these updates and maintenance. System can unexpectedly shutdown itself because of overload or an unexpected system failure. If the system has shutted itself down, we should see the reason for that shutdown and should fix it as quickly as possible.

**Logout:** There will be a logout button on the web page. If a user wants to log out, they will need to use the logout button. They can choose “kept logged in” while logging in the system, they will be kept logged in every time they enter the web page until they hit the logout button.

Admin will be automatically logged out once they close the system to prevent lack of security since they can reach every information of critical functions in the system.

**Failure:**

If there is a connection error, it will end up with a loss of data that needs to be saved on the database. Patients or doctors need to save their profile page after editing the page, or other pages they can reach and edit. If the users crashed because of an internet connection error, the system should keep the previous data and not make any change since the changes are not saved and the data should be missing or wrong. After the connection is successfully established, users should be notified about the unsaved changes and the system should ask users to make the changes and save them again with a proper connection.

If there is an error while showing the info about appointments, doctor list, profile page etc. the system should handle an exception should notify the users and admins about the failure.

System should be updated regularly to avoid major failures. If there is a major failure on the system, the system should be shutted down, and started again after the failure is removed successfully.

# Subsystem Services

The Hospital Management System uses the Django framework.

The sub components of the HMS; Admin, Doctor, Patient, Invoice and Appointment.

Among these components,

* Admin : Can access Doctor, Patient, Invoice and Appointment components with access services and control appointments such as approve or reject with approve reject function. Admin has access number to, control Login with AccessControl function. Access control function works in conjunction with the security component and the security component is encrypted.
* Doctor : Can view Payment information with bill\_info service from Invoice subcomponent. Doctor subcomponents can apply for a job to work in Hospital using ApplyJob function and If Doctor accepted Hospital to work, can create a new account with CreateAccount function. Doctors can also view and edit Patient Diagnosis, Symptom and Informations using Patient\_Details function.
* Patient : Can view own Payment information with bill\_info service from Invoice subcomponent. Patient subcomponent must register first, If the patient is going to book an appointment at this hospital for the first time using CreateAccount function. Then patients can book appointments with bookAppointment functions and view Appointment Details using Book service.
* **Invoice:** Can generate Invoices which include Payment Details using CreateInvoice function. Also Invoice subcomponent is the provider of Payment Details to Doctors and Patients using bill\_info function.
* **Appointment:** Can cancel and view appointments using OrganizeAppointment function.

Subcomponents work in conjunction with each other (data flows between each other). Data required for the operation of these functions,provided from the database component.

# 