Online Application Title: Hello Doctor

Team: 1) **ANURAG NANDKISHOR KANOJE (PRN: 210541281081)**
2) J K BIRAJDAR (PRN: 210541281029)

SRS Document:

Introduction

Hello Doctor

Purpose:

The advancement in Information Technology and internet penetration has greatly enhanced various business processes and communication between patients and doctors. This Hello Doctor is developed to provide the following services:

To be able to use internet technology to project to the global world instead of limiting their services to their local domain alone, many hospitals and doctors can use this so as to make the processes easy simple and convenient. People can take online appointment from any location by searching and by checking the doctors available time slots. Blood donors registered by admin in this project so as to avoid any false information and facility to search area and city wise available blood donors of registered blood donors to help the needy during the emergency.

Scope:

This project traverses a lot of areas ranging from city, state and country wide. One single place where patients can see multiple doctors with their specialization and can contact or can directly take an appointment with that doctor. This project will be useful to any small clinics, medium and big hospitals which gives convenience to the people. People can search blood donors from their area or from their city by choosing the blood group.

Definitions:

AES --> Administration of E-Placement System

TPO --> Training Placement Officer

SRS --> Software Requirement Specification

GUI --> Graphical User Interface

Portal --> Personalized Website

Stack holder --> The person who will participate in the System. And Owner of system

Ex. Admin, Doctor, Patient

UML---> Software Engineering Notation for visualising System in the form diagrams

SSL--->Secure Socket Layer used for providing restricted access to application.

BOD---> Board of Directors (Management).

RDBMS --> Relational Database Management System.

CLUSTERS---> Group of independent servers.

Overview:

This system is aim at developing a web application for the Hospitals and doctors. The system is a web application that can be accessed throughout the nation with proper login provided. This system can be used as a web application for the Doctor with any specialization to serve appointment service for patient with convenience. Patient logging should be able to take the appointment with doctor and also check the doctor details such as fees and can also see and search for blood donor. The key feature of this project is that it is a onetime registration. Our project provides the facility of maintaining the details of the Patient and Doctors. Search for blood donor by entering city facility is available. Administrator logging in can add, view or delete the doctor, patients and blood donors. This project is helpful to any doctor and small clinic and hospitals also can use.

General Description:

The aim of the proposed system is to develop a system with improved facilities. This system can overcome all the limitation of the existing system, such as staying in queue and wastage of time, it gives more security to data, ensures data accuracy, reduces paper work and save time, only verified doctors and blood donors added, it makes information flow efficient and paves way for easy report generation, reduce the space. proposed system is cost effective.

Functional Requirement:

This section provides requirement overview of the system. Various functional modules that can be implemented by the system will be-

Description:

Registration if user wants to login then he/she must be registered, Unregistered user cannot get login. Login user logins to the system by entering valid user id and password for going to welcome page.

After login, user will be identified if user is Administrator, then he/she can see the admin dashboard and Administrator logging in can add, view or delete the doctor, patients and blood donors. If login user is doctor then he/she can see the doctor dashboard where doctor can see the current active appointment along with patient details, appointment history, can generate time slots for appointments and can also update profile and check todays time slots.

And if login user is patient then he/she redirected to patient dashboard where patient can take the appointment with the doctor by selecting the doctor along with available time slot, also can see the active appointment and appointment history and patient can also see the

blood donor can also search city wise and blood group wise and finally patient can update his/her profile.

Technical Issues:

This system will work on client-Server architecture. It will require an internet server.

The system should support some commonly used browser such as Chrome etc.

Interface Requirement Various interfaces for the product could be

1.Login Page,

2.Registration form

There will be a screen displaying information about Patient that the eligible.

The user may select the different options which will be open in another screen as,

i. Login Page
ii. Registration Form
iii. About us
iv. Contact us

Hardware Interface:

The System must run over the internet and All the hardware shall require to connect to internet will be hardware interface for the system.

e.g. modem, WAN, LAN

Specialized Server Infrastructure Hardware

The system should use distributed servers i.e. cloud for managing large amount of data so as to make it appear as single unit for end-user.

The system should have proper clusters for backup.

Software Interface:

The system is on server so it requires the any scripting language like JSP or PHP or ASP, ETC.

The system should be able to exchange data using XML, JASON or any advance technology.

The system requires Database also for the store the any transaction of the system like MySQL or oracle, or SQL server etc.

System also require DNS (Domain Name space) for the naming on the internet.

At the end-user need web browser for interact with the system.

Performance Requirement:

There is no performance requirement in this system, because the server request and response to client is totally based on internet connection of end-user.

Design Constrains:

This system should be developed using Standard Web Page Development Tool, which conforms GUI standards such like HTML, XML, JSON, etc.

The system should support various RDMS and Cloud Technologies.

Non-Functional Requirements

1.Security:

SSL

The System use SSL (Secure Socket Layer) in all transaction that include any confidential customer information.

The system must automatically log out all customers after a period of inactivity.

The system should not leave any cookies on the customer's computer containing user's password.

The system's back-end servers shall only be accessible to authenticated administrators.

Sensitive data will be encrypted before being sent over insecure connections like internet.

The proper firewalls should be developed to avoid intrusions from the internal or external sources.

2.Reliability:

The system provides storage of all databases on redundant computers with automatic switchover.

The main pillar of reliability of the system is the backup of the database

which is continuously maintained and update to reflect the most recent changes.

3: Availability:

The system should be available at all times. Meaning the user can access it using web browser,

only restricted by the down time of the server on which the system runs.

In case of a of a hardware failure or database corruption, a replacement page will be shown.

uptime: It mean 24 * 7 availability

100%----------

99.9%

99.999%

99.9999%

4: Maintainability:

A commercial database is used for maintaining the database and application server takes care of the site.

The maintainability can be done efficiently.

5.Portability:

The application is HTML and scripting language based (JavaScript). So the end user part is fully portable and any system using

any web browser should be able to use the features of the system, including any hardware platform that is available

or will be available in the future.

An end-user is used this system on an OS; either it is Windows or Linux.

The System shall run on PC, Laptops and PDA, etc.

The technology should be transferable to different environments easily.

6.Accessibility:

Only registered users should be allowed to see their profile after authentications.

Only GUI access of the system should be permitted to end users.

7.Policies:

The system should adhere to all the legal formalities of the particular countries.

The system should maintain security related to sensitive data.

8.Efficiency:

The system should provide good throughput and response to multiple users without burdening the system by using appropriate number of servers.

9.Safety:

Software should not harm ethical and environmental conditions of the end users machine.

10.Modulariy:

The system should have user friendly interface.

It should be easily updated, modified and reused.

Operational Scenario:

Admin:

- Admin can login to the system.
- View the list of all doctors.
- Add new doctor.
- Delete Flat doctor.
- Add new blood donor.
- View list of patients.
- View list of donors.
- Delete patients.
- Logout the admin.

Doctor:

- Doctor can login to the system.
- Can view and update his/her details.
- View active appointments.
- View appointment history.
- Create time slots for appointments.
- View todays time slots.
- Logout from the system.

Patient:

- Patient can login to system.
- Patient can update his/her profile.
- Can book appointment with doctor.
- Show current appointment.
- View appointment history.
- Can search and get info blood donor.
- Logout from the system.

Preliminary Schedule:

1. Login
2. Manage dashboard
3. Manage doctor database
4. Manage patient database
5. Manage blood donor databse
6. add or remove doctors
7. update information
8. Take an appointment
9. Logout
10. See blood donor list
11. Create new account
12. View account details
13. Registration
14. Check doctor available time slots
15. Appointment history and current active appointment
16. Manage appointment database along with time slots

ER listing:

Application Architecture:

Application = Logic + data

Logic = (UI Logic + Business Logic + Data Access Logic)

Data = (structured data, Non Structured data)

Online Application:

Web based Application deployed on web and accessed by users from anywhere Administration of E-Placement System (OAPMS)-----Web portal-- used remotely by HOD, TPO, Students, Company.

Logic:

UI Logic:

Web Pages + HTML controls + Web Components (Angular)

Navigation: (UI Routing) HTML Links, Routing mechanism

Data Binding: DOM + JSP tags (JSTL) + {{}} model,

Event Binding: action handlers

HTTP Request: GET: ----------Do get

POST: ----------Do post

PUT:

DELETE: Client Side UI----HTML, CSS, JavaScript, bootstrap

UI (Client Side UI Framework)

Angular, React, Vue

Web Logic: (Server Side processing)

Server UI---------- JSP, servlet, (classical java web technology)

spring MVC (to take advantage of MVC design Pattern using readymade formwork)

Model, View, Controller

Router

(SOA layer)

Spring Boot API

CRUD REST API

ORM Technique: Hibernate (ORM)

JPA

JDBC (database Connectivity)

State management

Client Side state management

cookies, query string, form collection, hidden variables

local storage, session storage, Web sql,

Server Side state management

session, Cache,

database

Business Logic:

Java console application will be used to test your business Logic

Core Java:will contain

1.business query processing

2.business operation management

3.Business data manipulation

from Administration of E-Placement System (OAPMS) system point of view Modules

1.Registration:

: Patient Registration

create, insert, delete, update

2.Skill Mapping

: Doctor and Patients

get patient details

see active and history of appointments

get doctor details and blood donor details

3.Scheduling

: Doctor

: get time slots

: send email notification to the patient of appointment

: Patient

: see the available time slots for doctor

4.Authentication

: validate user

: identify user

Data:

Structured Data

RDBMS

fields

tables

constraints

Add some dummy records in your newly created database

write reusable SQL queries against those database tables to check business Queries

Test those SQL queries on existing dummy database you built

List of Tables

admin_tbl

Fields:

Id, name, email and password.

Primary Key: Id

appointment_tbl

Fields:id, appoint_time, appointment_type, doctor_id, patient_id.

Primary Key: Id

blood_donor_tbl

Fields:

Id,blood_group,city,contact_number,email,name,state.

Primary Key: Id

doctor_tbl

Fields:

Id,area,city,dob,email,first_name,gender,last_name,mobile_number,password,state,username,fees,language,qualification,specialization,time_slot_id

Primary Key: Id

doctor_time_table_available_slots:

Fields:

doctor_time_table_id,available_slots,available_slots_key.

Primary Key: doctor_time_table_id, available_slots_key.

doctor_time_table_holidays

Fields:

doctor_time_table_id,holidays

doctor_timetable_tbl

Fields:

id,break_time,end_time,end_date,slot_duration,start_date,start_time

Primary Key: id

Patient_tbl

Fields:

id,area,city,dob,email,first_name,gender,last_name,mobile_number,password,state,username,blood_group.

Primary Key: id

Unstructured:

NoSQL

Social Media

MongoDB