Project - 01

Project Title: Hospital Management System

JS, HTML, CSS

Module Code	DSMP-101				
Module					
Year	2023	Stage	03	Date	14.09.2023

Nature of the project	Individual Assignment
Project Submission Date	21.09.2023
Evaluation Date	-

Objectives

1. Improve frontend development skills in HTML, CSS, and JavaScript.
2. Learn backend development by connecting the frontend to Firebase Firestore and implementing CRUD operations.
3. Understand database design principles, including structuring data in collections and documents.
4. Enhance problem-solving abilities in real-world hospital management scenarios.
5. Gain experience in project management, including planning, organization, and testing.

Assignment Requirements

Detail the specific requirements and features of the Hospital Management System, such as:

1. User authentication and authorization.
2. Patient registration and management.
3. Doctor registration and management.
4. Appointment scheduling.
5. Billing and invoicing.
6. Medicine inventory management.
7. Report generation (e.g., patient records, billing reports).

Introduction

Explain the purpose of the project, which is to build a simplified Hospital Management System, and clarify that user authentication won't be included in this version.

Project Overview

Provide a high-level overview of the Hospital Management System, including its main features and functionalities, without mentioning user authentication.

Tools and Technologies

List the technologies and tools the students will need to complete the project, including JavaScript, HTML, CSS, and Firebase for data storage

Firebase Setup:
Guide students through setting up a Firebase project and configuring Firestore for data storage, but skip the Firebase Authentication setup.

Database Design:
Discuss the structure of the Firestore database, including collections and documents for storing patient, doctor, appointment, and other relevant data.

Frontend Development:
Provide guidance on creating the user interface using HTML, CSS, and JavaScript.
Encourage students to use a responsive design for better user experience.

Backend Development:
Explain how to connect the frontend to Firebase and how to perform CRUD (Create, Read, Update, Delete) operations on the Firestore database.
Patient Management:

Instructions

Patient Management:

Create a "Patients" section in the interface to display a list of patients.
Add a form for registering new patients, including fields for name, age, gender, contact information, and medical history.
Implement the ability to view, edit, and delete patient records.
Ensure that patient records are stored in the Firestore database under a "patients" collection.

Doctor Management:

Create a "Doctors" section in the interface to list registered doctors.
Design a form for registering new doctors, including fields for name, specialization, contact details, and availability.
Implement functionality to view, edit, and delete doctor profiles.
Store doctor information in the Firestore database under a "doctors" collection.

Appointment Scheduling:

Develop an "Appointments" section for scheduling appointments between patients and doctors.
Include date and time picker elements for selecting appointment details.
Implement logic to prevent double-booking and conflicts.
Store appointment data in Firestore, associating it with both the patient and doctor.

Billing and Invoicing:

Create a "Billing" module to calculate charges for medical services.
Design an invoice generation system that includes patient details, services provided, costs, and payment status.
Implement tracking of outstanding balances for patients.
Store billing and invoice data in Firestore under an appropriate collection.

Medicine Inventory Management:

Develop a "Medicine Inventory" section to manage the hospital's medicine stock.
Create forms for adding, updating, and removing medicines with fields like name, quantity, price, and expiration date.
Implement validation to prevent negative quantities or invalid inputs.
Store medicine data in Firestore under a "medicine" collection.

Report Generation:

Design a reporting module to generate various types of reports, such as patient records, billing reports, and appointment schedules.
Provide options for users to select report criteria, such as date ranges.
Generate reports dynamically based on user selections.
Allow users to download or print reports as PDFs or other formats.

User-Friendly Interface:

Ensure that the user interface is intuitive and responsive, with a consistent layout and design.
Implement navigation menus, buttons, and forms for easy interaction.
Use CSS for styling to create an appealing and professional appearance.
Optimize the interface for both desktop and mobile devices.

Database Integration:

Connect the frontend to Firebase Firestore using Firebase SDK.
Implement Create, Read, Update, and Delete (CRUD) operations to interact with Firestore collections.
Use Firebase Firestore queries to retrieve and display data efficiently.

Testing:

Conduct thorough testing of all system functionalities.
Test different scenarios, including edge cases and potential errors.
Debug and resolve any issues or bugs encountered during testing.
Ensure that data integrity and security are maintained.

Deployment:

Choose a web hosting platform or Firebase Hosting for deployment.
Follow deployment guides for the selected platform.
Configure domain settings if applicable.
Monitor the deployed system to ensure it functions correctly in the production environment.