Hospital Management System – Salesforce CRM Project Documentation

## Project Overview

The Hospital Management System (HMS) is a Salesforce-based CRM solution designed to streamline operations across all hospital departments. It integrates patient registration, doctor scheduling, appointment booking, billing, and reporting within a single secure cloud platform. The HMS replaces manual and paper-based processes with automated workflows and real-time dashboards, ensuring improved patient care, better staff coordination, and adherence to healthcare data privacy regulations. By leveraging Salesforce's flexibility and scalability, the system supports future enhancements such as integration with insurance providers and telemedicine services.

## Objectives

The objective of this project is to create a unified and secure environment for managing hospital operations. Key goals include: • Centralizing patient records for easy access and updates. • Providing automated appointment scheduling and doctor availability tracking. • Ensuring strict role-based access and compliance with health data privacy laws. • Enabling management to make data-driven decisions through advanced reporting and dashboards. • Building a scalable platform capable of integrating future services such as pharmacy inventory or insurance APIs.

## Phase 1: Problem Understanding & Industry Analysis

Extensive stakeholder meetings were conducted with hospital administrators, doctors, and reception staff to capture key requirements. We identified issues such as inconsistent patient records, overlapping appointments, and time-consuming billing processes. Industry analysis focused on compliance with HIPAA-like data privacy standards and the need for 24/7 access to patient data. Business process mapping detailed each step: patient registration, appointment scheduling, doctor consultation, treatment, and billing. This phase concluded with a detailed requirement document that guided the subsequent configuration and development.

## Phase 2: Org Setup & Configuration

A Salesforce Developer Org was provisioned to serve as the project environment. The company profile was set with appropriate time zones and currencies. Business hours were configured to include emergency services running 24/7 and custom holiday settings for staff scheduling. Roles were designed to establish a clear hierarchy—Admin at the top, followed by Doctors, Receptionists, and Billing Staff. Profiles and permission sets were created to ensure each user type had access only to the data relevant to their role. Org-wide defaults and sharing rules were implemented to secure sensitive patient information while enabling smooth collaboration among staff.

## Phase 3: Data Modeling & Relationships

Data modeling involved creating custom objects: Doctor, Patient, Appointment, and an optional Billing object for financial tracking. Fields included demographic details, medical history for patients, specialties for doctors, and status/fee details for appointments. Lookup relationships were established between Appointment and both Doctor and Patient to ensure accurate scheduling. Schema Builder was utilized to visually verify relationships, while record types were introduced to differentiate appointment types such as routine check-ups and emergency visits. Compact layouts were designed for mobile users to quickly view key information like appointment time, doctor, and patient name.

## Phase 4: Process Automation

Automation was key to reducing manual workload. Validation rules prevented invalid data entry, such as past appointment dates or negative fees. Flows handled appointment confirmations, sending email and SMS notifications to patients. Scheduled flows generated daily reminders for doctors about their upcoming appointments. Approval processes were introduced for high-cost treatments, ensuring that managerial oversight was obtained before expensive procedures. These automations improved accuracy, reduced missed appointments, and enhanced patient communication.

## Phase 5: Apex Programming

Advanced business logic was implemented using Apex. Triggers ensured that no overlapping appointments could be scheduled for the same doctor. Apex classes managed complex calculations such as monthly revenue summaries. Batch Apex jobs were scheduled to perform nightly cleanups of old or cancelled appointments, improving system performance. Queueable and Scheduled Apex handled asynchronous tasks like sending periodic reports to hospital management. Robust test classes were developed to maintain high code coverage and ensure reliable deployments.

## Phase 6: User Interface Development

A user-friendly Lightning App was created to bring all custom objects under a single navigation. Lightning App Builder allowed the creation of customized record pages tailored to doctors and receptionists. Lightning Web Components (LWC) were built to enable quick patient searches and appointment bookings, enhancing usability. Home page dashboards provided at-a-glance metrics such as upcoming appointments and daily patient counts, improving staff productivity.

## Phase 7: Integration & External Access

While the initial phase did not require external integrations, the system was prepared for future connectivity. Named Credentials and Remote Site Settings were configured as placeholders for potential integration with insurance verification services or external billing providers. API limits were reviewed and monitored through the Salesforce System Overview to ensure the platform can scale as integrations are added.

## Phase 8: Data Management & Deployment

Data import strategies were planned to safely migrate existing patient and doctor records into Salesforce. The Data Import Wizard was used for smaller datasets, while Data Loader handled bulk uploads. Duplicate rules were enforced to prevent redundant patient records. Change Sets and SFDX were employed for moving configurations from sandbox to production, ensuring a smooth deployment pipeline.

## Phase 9: Reporting, Dashboards & Security Review

Powerful reports were developed to track KPIs like daily appointment counts, revenue by department, and doctor utilization rates. Dashboards combined these metrics into a single visual interface for hospital administrators. Dynamic dashboards allowed each doctor to view only their own schedule and performance. A comprehensive security review confirmed that patient data was protected using field-level security, strict sharing settings, session controls, and audit trails.

## Phase 10: Quality Assurance Testing

A complete test plan was executed covering all functionalities—from record creation to approval processes and automated flows. Each test case documented the input data, expected output, and actual system behavior. Screenshots of both input and output validated successful test completion. This rigorous testing phase ensured that the system was stable, secure, and ready for hospital-wide deployment.

## Conclusion

The Salesforce-based Hospital Management System provides a unified platform for managing doctors, patients, appointments, and billing with maximum security and efficiency. Automation and custom Apex code reduce manual tasks, while advanced reports and dashboards deliver real-time operational insights. The detailed phase-wise approach—from requirements gathering to final testing—ensures that the system is scalable, compliant, and adaptable to future needs such as insurance integration or telemedicine services. This documentation serves as a comprehensive guide for administrators, developers, and future maintenance teams to understand the system's architecture and functionality.