WeCare Hospital Management System

SAN JOSÉ STATE
UNIVERSITY

Hospital Management System Proposal

Team #19: Nicholas Hsiao, Huynh Phan, Diana Sok

September 5, 2019

Instructor: Dr. Mike-Wu

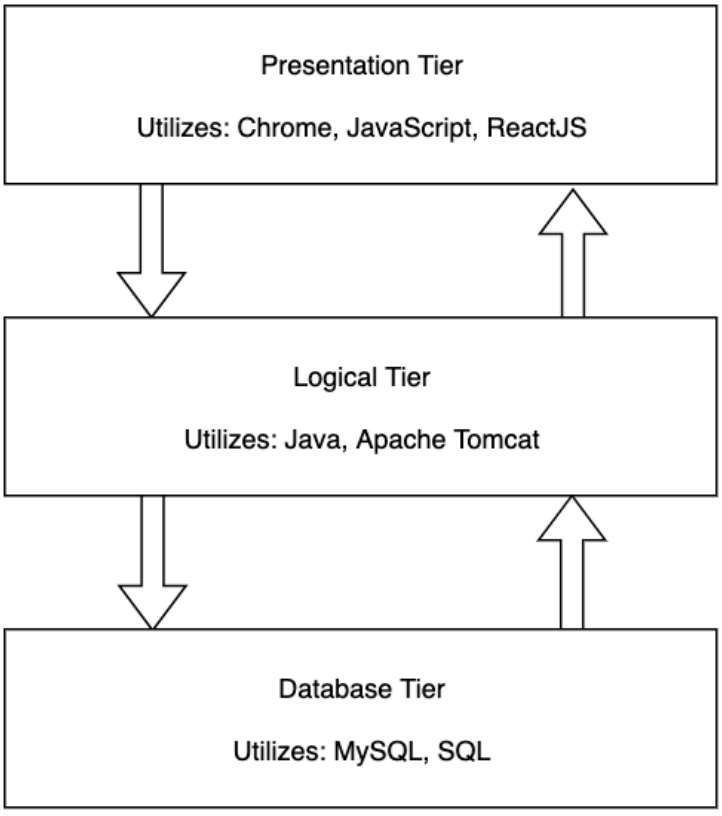
Project Proposal

1. Project Overview

For this database application project, we will be developing a hospital management system with both user and administrative functions. Users (patients) will be able to register as a new user and fill in a medical profile or continue as a registered user and have access to their medical profile and appointments. Users will also be able to schedule and update appointments. Administrators (doctors and nurses) will have read and write access patient medical records and be able to confirm, view and update all upcoming appointments. This project will be a great opportunity to build a real-world full stack application.

2. System Environment

Our application will follow the three-tier architecture structure presented below. For client browser we will be using Google Chrome. The front-end and user interface will be built using JavaScript and ReactJS as the framework. Our project will use Apache Tomcat and Java for hosting the web server. Finally, we will be using SQL and the MySQL RDBMS for our database tier.



As we are only mimicking an actual three-tier architecture and using LocalHost, the entire three-tier architecture set-up must be replicated to run this application. Listed below are the hardware specifications of the laptops each member is using to run an Apache TomCat Server on.

Nick:

Component	Specification
Central Processing Unit	2 GHz Intel Core i5
Number of Processors	1
Memory	8 GB
Operating System	macOS High Sierra (10.13.6)

Huynh:

Component	Specification
Central Processing Unit	Intel Core i5-8250U CPU @ 1.60 GHz
Number of Processors	4
Memory	8 GB 1600MHz DDR3
Operating System	Windows 10 Home 64-bit

Diana (MacBook):

Component	Specification
Central Processing Unit	2 GHz Intel Core i5
Number of Processors	1
Memory	8 GB 1867MHz LPDDR3
Operating System	macOS Mojave (10.14.5)

3. Functional Requirement

Patient Functional Requirements

Requirement	ID	Detail Leveled Requirements
Create Account	1P	Allow user to register and create an account
	2P	Prevent re-registering with an existing account
	3P	Prompt user in case of invalid registration details
	4P	Allow user to enter in personal information and fill out medical history
	5P	Assign new users a unique patient id
Login/ Logout	6P	Allow users to log in after providing valid credentials
	7P	Prompt user of error in case of invalid credentials
	8P	Allow users to log out of their account
	9P	Allow users to recover or reset their password via email
	10P	A Recover or reset password attempt on an account not in the system fails
Personal Info and Medical History/Profile Access	11P	Allow users to edit or update their personal information and medical history
	12P	Allow users to review their personal information and medical history
	13P	Allow users to have read access to their own medical profiles written by doctors
Schedule appointments	14P	Allow user to schedule an appointment based on doctor.
	15p	Allow user to schedule appointment based on time.
	16P	Prevent scheduling an appointment conflicting with patient's existing appointments.
	17P	Prevent scheduling an appointment conflicting with the doctor's schedule.

	18P	Allow patient to schedule multiple appointments.
View/Update appointments	19P	Allow user to view upcoming appointments.
	20P	Allow user to view past appointments.
	21P	Allow user to cancel an appointment.
	22P	Allow user to change appointment date.
	23P	Allow user to change appointment time.
	24P	Updating appointment date cannot conflict with scheduled doctor's existing schedule.
	25P	Updating appointment date cannot conflict with patient's existing schedule.
	26P	Updating appointment time cannot conflict with scheduled doctor's existing schedule.
	27P	Updating appointment time cannot conflict with patient's existing schedule.

Administrative (Doctors/Nurses) Functional Requirements

Requirement	ID	Detail Leveled Requirements
Create Account	1A	Allow administrator to register and create an account upon providing their Employee ID
	2A	The Employee ID must be 9 digits long
	3A	Allow administrators to provide their first and last name, pick a department, and create a password
	4A	Upon creation, a generated work email will be provided from the information and added to the profile
Login/Logout	5A	Enable administrators to log in upon providing their Employee ID and password

	6A	Allow administrators to log out of their account
	7A	Enable administrators to submit a support ticket to change their password
Access medical profiles	8A	Administrators will be allowed to view all patient profiles
	9A	Administrators will be allowed to add to the medical history of a patient profile
	10A	Administrators will be allowed to view updates to a patient profile given by the patient
	11A	Administrators will be allowed to view all employee profiles
	12A	Administrators will be allowed to provide their own hours of restricted availability
	13A	Administrators will be allowed to add a prescription to a patient profile
	14A	Administrators will be allowed to view the appointments of other employee profiles
View/Update appointments	15A	Allow user to view upcoming appointments.
	16A	Allow user to view past appointments.
	17A	Allow user to cancel an appointment.
	18A	Allow user to specify chunks of time that cannot be scheduled.
	19A	Update affected patient of appointment cancellation.

4. Non-Functional Requirements

1. Execution qualities (Qualities which are observable during operation)

   a. Security
      i. There will be no broken authentication or broken access control points through which admin privileges are given to non-admin users
      ii. Patients will not be able to access restricted data
   b. Privacy
      i. Patients cannot view other patients' data
      ii. Patients cannot view any of the doctor's private data, such as their patient list
   c. Performance
      i. The database program shall execute in timely fashion, returning queries in a reasonable amount of time
   d. Constancy
      i. The program will not need to be executed separately or restarted in a single user's session as well as a series of concurrent users of variable permission levels. The program will run in a perpetual state throughout its use.
2. Evolution qualities
   a. Documentation
      i. The system will feature an organization of code with descriptions such that each component can be easily understood as a constituent of the system
   b. Testability
      i. The system source code will follow the 3-tier architecture. Using that to the advantage of self-testing, the divided infrastructure will enable the project team to easily identify the point of vulnerability or error as one of the three classifications once error/exception handling is implemented