Hospital Management System
Design Document

Name : Aathira Thankamoney dev
Student ID : 21161041
Email : aathira.thankamoneydev@mail.bcu.ac.uk

Table of Contents

Overview

The Hospital Management System is a software application that is designed to manage various aspects of a hospital, including patient records, doctor appointments, and administrative tasks. The system has three main user roles: Admin, Doctor, and Patient.

Key Features

Admin

- Admin login
- Register/view/update/delete doctors
- View patients
- Assign doctors to patients
- Discharge patients from the system when their treatment is complete
- View a list of discharged patients
- Update their own personal information (name and address), username and password
- Patients are grouped into families by the admin
- Patients can be relocated from one doctor to another
- Admins can request a management report, which includes information such as:
  - Total number of doctors in the system
  - Total number of patients per doctor
  - Total number of appointments per month per doctor
  - Total number of patients based on illness type

Doctor

- View patient details assigned to them by the admin
- View their appointments

Patient

- Apply to be admitted to the hospital
- Book an appointment
- View their assigned doctor
- Check the status of their appointment (approved, pending)

Data Model

The Hospital Management System stores data about patients, doctors, and appointments in .txt files. The following is a summary of the main data entities and their attributes:

Classes

Person

- First Name
- Last Name
- Email
- Age
- Contact Number
- Address Line 1

- Address Line 2

Patient – Inherits from Person

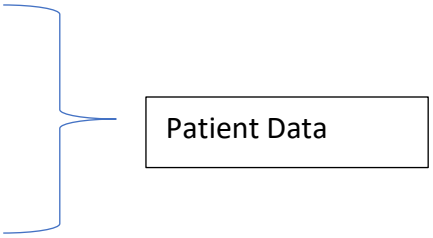
- Patient ID (unique identifier)
- Doctor ID
- Appointment Date
- Symptoms

Doctor - Inherits from Person

- Doctor ID (unique identifier)
- Specialty

Appointment

- Appointment ID (unique identifier)
- Patient ID
- Doctor ID
- First Name
- Last Name
- Email
- Age
- Contact Number
- Address Line 1
- Address Line 2



- Appointment Date
- Status - Possible values are Pending, Confirmed and Completed.
- Admit – Possible values ‘Requested’, ‘Rejected’ , ‘Discharge’ and ‘’ if there is no admit request

Modules

File_operaions

Generic methods to perform file read, write operations. Name: file_operations.py

Validator_functions

Generic functions to validate use input fields. Name: validator_functions.py

User Interface

Main Screens

The Hospital Management System will have a graphical user interface (GUI) that is implemented using the Tkinter library. The interface will be divided into different frames for the different user roles, with each screen providing the appropriate functionality for that role. You can use Tkinter to design the GUI for the application. The GUI of the application should have the following pages/screens:

- Login screen: This screen should have fields for the user to enter their login id and password, and a login button.
- Admin Dashboard: This screen should display options for the admin to perform various actions such as registering a doctor, viewing/updating/deleting a doctor, viewing patients, assigning doctors to patients, discharging patients, viewing the discharged patients list, and updating their own information.

- Doctor home screen: This screen should display options for the doctor to view their assigned patients and appointments. (not functional, part of future scope)
- Patient home screen: This screen should display options for the patient to apply for admission, book an appointment, view their assigned doctor, and check the status of their appointment.

Detailed Functionalities

Functionality	Technical Implementation
The GUI has separate screens for admin login and doctor. Login is not supported for patient. The screens for each role have the relevant features and information for that role.	Admin login is implemented via login method in Admin. The method verifies login credentials, which takes a username and password as input and returns a Boolean indicating whether the login is successful or not.
Patient can book an appointment by filling in necessary details. Patient can also check the appointment status	Implemented using book_appointment and search_appointment method in Patient class
Admin can confirm appointment request. Admin can also process admit request	Implemented via methods in Admin, Patient , Doctor and Appointment classes
Admin can add, view , edit and delete doctors	
Admin can generate reports	
Admin can update own information	
Grouping patients of same family	Default system groups patients from same Post Code. Admin can delete those default groups or can create new groups

Backend

The Hospital Management System will store its data in text files. Each file will contain the data for a specific entity (e.g., patients, doctors, appointments). The system will use file I/O functions to read and write data to these files.

Files

1. appointment.txt
2. doctor.txt
3. patient.txt
4. admin.txt
5. patient_group.txt

Sample Data Storage in file

Appointment

First Name=Claire || Last Name=Thomas || Contact Number=7123321111 || Age=49 || Email=cthomas@gmail.com || Symptoms=Cardiology || Address Line 1=Cheshire House || Address Line 2=Cheshire || Status=Completed || Post Code=SE352 || Appointment ID=d13cc || Patient ID=d13cc || Admit= || Doctor ID=1aab3 || Appointment Date=2023-02-02

First Name=Zani || Last Name=Aman || Contact Number=7123131313 || Age=34 || Email=zaman@gmail.com || Symptoms=Cardiology || Address Line 1=Sheffield Wednesday || Address Line 2=Sheffield || Status=Completed || Post Code=SH345 || Appointment ID=b7fe4 || Patient ID=b7fe4 || Admit= || Doctor ID=1aab3 || Appointment Date=2023-02-04

Dependencies

The Hospital Management System will depend on the following libraries. All are default included in Spyder 3.1.5 and Python 3.9

GUI

- Tkinter: for creating the GUI
- import tkinter as tk
- from tkinter import ttk
- from tkinter import *
- from tkinter.messagebox import showinfo - *messagebox for error display*

Reports - Charts

- import matplotlib.pyplot as plt – *For charts*
- from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg – *For Charts*
- from collections import defaultdict – *For patients per doctor per month chart*
- from datetime import datetime - *For patients per doctor per month chart*
- import calendar - *For patients per doctor per month chart*

For generating unique ID

- import uuid
- import hashlib

File Handlin

- import os

Testing

The Hospital Management System was tested manually.

Limitation of the Design

1. The system backend is implemented using .txt files. Using files reduces the scalability of the system. Changing the backend to relational databases like MySQL or MongoDB enhances the system usage, security and performance
2. Loading all data elements as part of the system load. This is followed from the partial design. After changes to the data elements, it needs to be refreshed. At present it is handled by refreshing data in show_frame method. This may have performance considerations. A centralised class or MySQl can optimise this.

3. System compromises on security as the possibility of masking sensitive data in backend is limited in txt files
4. Tkinter GUI comes with it's limitation. E.g – Without installing additional packages like tkcalendar it is not possible to have date picker
5. Due to the timeline constraints, haven't fully utilised the opportunity to leverage inheritance for tkinter frames
6. Look and feel of the GUI needs to be improvised and would be considered as part of future developments.

Conclusion

The Hospital Management System is a powerful tool that will help hospitals to manage their operations more efficiently and effectively. By providing features such as patient records, doctor appointments, and administrative tasks, the system will help hospitals to streamline their workflows and improve patient care.