RESEARCH ARTICLE OPEN ACCESS

HOSPITAL MANAGEMENT SYSTEM USING PYTHON

Ms. K. Kowsika[1], Mr. C. Balaji [2]

[1] Department of Commerce (Business Analytics), Dr. N.G.P Arts and Science college (Autonoumous) Affiliated to Bharathiar University, Coimbatore, Tamilnadu
(Mail Id-kowsikakalimuthu1829@gmail.com)

[2] Assistant Professor, Department of Commerce (Business Analytics), Dr. N.G.P Arts and Science college (Autonoumous) Affiliated to Bharathiar University, Coimbatore, Tamilnadu
(Mail Id: bala3mca@gmail.com)

Abstract:

The Hospital Management System (HMS) is a Python-based application designed to enhance hospital administration by automating patient records, appointment scheduling, and doctor assignments. The system leverages Python, Tkinter (GUI), and SQLite to provide an efficient and secure data management solution. By integrating a random allotment number generator, the system ensures error-free appointment scheduling. This project aims to reduce administrative workload, minimize errors, and improve hospital efficiency through a user-friendly digital interface. Future advancements, including billing systems, pharmacy management, and AI-powered analytics, will further optimize healthcare operations, making the HMS a comprehensive solution for modern hospitals.

Keywords: Hospital Management System, Patient Data Automation, Appointment Scheduling, Python, Tkinter, SQLite, Healthcare Technology.

1. **Introduction**:

Efficient hospital management is crucial for delivering high-quality healthcare services. Traditional manual systems for patient record-keeping, appointment scheduling, and resource management often lead to inefficiencies, errors, and security concerns. To address these challenges, the Hospital Management System (HMS) using Python has been developed. This system aims to enhance hospital workflow by automating patient data handling, streamlining appointment scheduling, and ensuring real-time data access. The system is designed to be secure, scalable, and easy to use, making it a valuable tool for healthcare institutions.

1.1 **Motivation:**

Innovative methods like hospital automation and digital record-keeping are gaining popularity for improving efficiency and accuracy in healthcare administration. Relying solely on manual record systems may result in errors, inefficiencies, and delays in patient care. The objective here is to replace traditional methods with a digitalized approach, ensuring real-time access to patient records, secure data management, and efficient hospital operations

1.2 **Objectives:**

This initiative aims to predict Alzheimer's disease and obtain more precise and reliable outcomes. It will utilize Convolutional Neural Networks

(CNN) and Support Vector Machine (SVM) algorithms. The Python programming language will be used for implementing machine learning techniques to carry out this task.

1.3 **Problem Statement:**

There is a lack of sufficient awareness about Alzheimer's Disease. As individuals age, they may experience gradual changes in their physical abilities, affecting activities like walking, sitting, and eventually swallowing. As memory and cognitive functions decline, they may require significant help with everyday tasks. At this point, individuals may need round-the-clock support for personal care and daily routines. As dementia progresses, it impairs the ability to communicate, adjust to surroundings, and move, making it increasingly difficult for them to express pain through words or gestures.

2. **Literature Review**

The role of digital systems in healthcare has been widely explored, with numerous studies highlighting the benefits of automation in medical record management, appointment scheduling, and hospital resource allocation. Existing research suggests that electronic health record (EHR) systems significantly enhance data accuracy and retrieval speed, reducing human errors. However, many proprietary HMS solutions suffer from high costs and complexity. This study focuses on an open-source, lightweight alternative using Python.

3. System Design and Architecture

The HMS is built on a three-tier architecture comprising:

- **Presentation Layer:** The user interface is developed using Tkinter, featuring an interactive dashboard, patient registration forms, and an appointment scheduling system.
- **Business Logic Layer:** Python handles the core functionalities, including CRUD (Create, Read, Update, Delete) operations on patient records.
- **Data Layer:** SQLite, a lightweight and embedded database, ensures secure and efficient data management.

The system is modular, allowing future enhancements such as integration with billing systems, pharmacy management, and real-time notifications.

4. Methodology

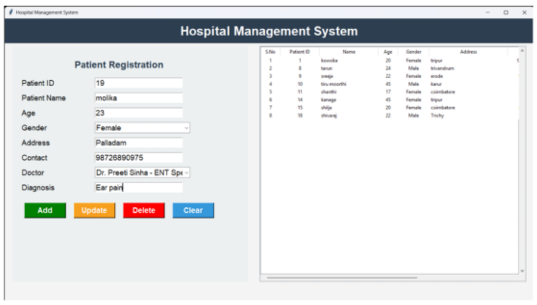
The system was developed using an agile methodology, with iterative design and testing to ensure functionality and usability. Key modules implemented include:

4.1 Database Management: CRUD Operations

The Hospital Management System implements full CRUD (Create, Read, Update, Delete) functionality for managing patient records.

4.1.1 Adding a New Patient Record

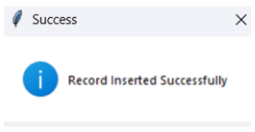
```
import sqlite3

def add_patient(name, age, gender, address, contact, doctor, diagnosis):

    conn = sqlite3.connect("hospital.db")

    cursor = conn.cursor()

    cursor.execute("""

        INSERT INTO patients (name, age, gender, address, contact, doctor, diagnosis)

        VALUES (?, ?, ?, ?, ?, ?, ?)

    """, (name, age, gender, address, contact, doctor, diagnosis))

    conn.commit()

    conn.close()
```


Hospital Management System

Patient Registration

Patient ID	19
Patient Name	molika
Age	23
Gender	Female
Address	Palladam
Contact	98726890975
Doctor	Dr. Preeti Sinha - ENT Spe
Diagnosis	Ear pain

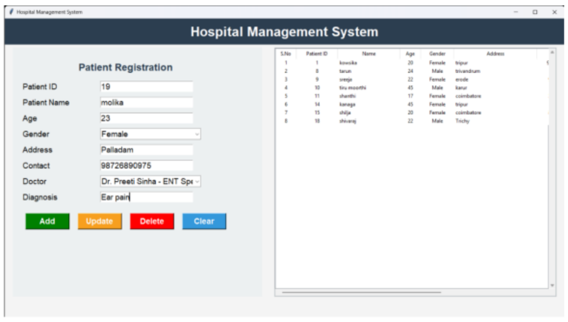
S.No	Patient ID	Name	Age	Gender	Address
1	1	kowsika	20	Female	tripur
2	8	tarun	24	Male	trivandrum
3	9	sreeja	22	Female	erode
4	10	tiru moorthi	45	Male	karur
5	11	shanthi	17	Female	coimbatore
6	14	kanaga	45	Female	tripur
7	15	shilja	20	Female	coimbatore
8	18	shivaraj	22	Male	Trichy


Success

Record Inserted Successfully

4.1.2 Fetching Patient Records

```
def fetch_patients():
    conn = sqlite3.connect("hospital.db")
    cursor = conn.cursor()
    cursor.execute("SELECT * FROM patients")
    records = cursor.fetchall()
    conn.close()
    return records
```


Hospital Management System

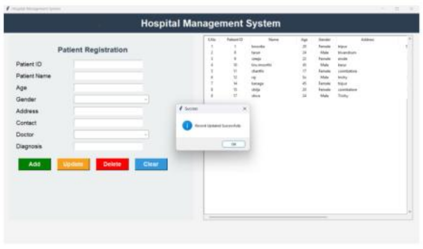
Patient Registration

Patient ID	19
Patient Name	molika
Age	23
Gender	Female
Address	Palladam
Contact	98726890975
Doctor	Dr. Preeti Sinha - ENT Spe
Diagnosis	Ear pain

S.No	Patient ID	Name	Age	Gender	Address
1	1	kowsika	20	Female	tripur
2	8	tarun	24	Male	trivandrum
3	9	sreeja	22	Female	erode
4	10	tiru moorthi	45	Male	karur
5	11	shanthi	17	Female	coimbatore
6	14	kanaga	45	Female	tripur
7	15	shilja	20	Female	coimbatore
8	18	shivaraj	22	Male	Trichy

4.1.3 Updating a Patient Record

```
def update_patient(patient_id, name, age, gender, address, contact, doctor, diagnosis):
    conn = sqlite3.connect("hospital.db")
    cursor = conn.cursor()
    cursor.execute("""
        UPDATE patients SET name=?, age=?, gender=?, address=?, contact=?, doctor=?, diagnosis=?
        WHERE id=?
    """, (name, age, gender, address, contact, doctor, diagnosis, patient_id))
    conn.commit()
    conn.close()
```


Hospital Management System

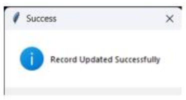
Patient Registration

Patient ID	
Patient Name	
Age	
Gender	
Address	
Contact	
Doctor	
Diagnosis	

S.No	Patient ID	Name	Age	Gender	Address
1	1	kowsika	20	Female	tripur
2	8	tarun	24	Male	trivandrum
3	9	sreeja	22	Female	erode
4	10	tiru moorthi	45	Male	karur
5	11	shanthi	17	Female	coimbatore
6	12	[illegible]	[illegible]	Male	trichy
7	14	kanaga	45	Female	tripur
8	15	shilja	20	Female	coimbatore
[illegible]	17	[illegible]	[illegible]	Male	Trichy

Success

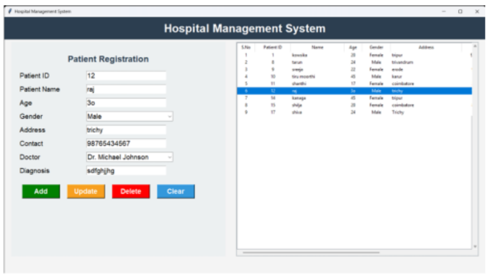
Record Updated Successfully


Success

Record Updated Successfully

4.1.4 Deleting a Patient Record

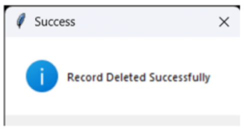
```
def delete_patient(patient_id):
    conn = sqlite3.connect("hospital.db")
    cursor = conn.cursor()
    cursor.execute("DELETE FROM patients WHERE id=?", (patient_id,))
    conn.commit()
    conn.close()
```



Patient Registration

Patient ID	12
Patient Name	raj
Age	3o
Gender	Male
Address	trichy
Contact	98765434567
Doctor	Dr. Michael Johnson
Diagnosis	sdfghjjhg

S.No	Patient ID	Name	Age	Gender	Address
1	1	kowsika	20	Female	tirpur
2	8	tarun	24	Male	trivandrum
3	9	sreeja	22	Female	erode
4	10	tiru moorthi	45	Male	karur
5	11	shanthi	17	Female	coimbatore
6	12	raj	3o	Male	trichy
7	14	kanaga	45	Female	tirpur
8	15	shilja	20	Female	coimbatore
9	17	shiva	24	Male	Trichy



Record Deleted Successfully

5. Results and Discussion

The HMS was tested for performance, usability, and data integrity. Results indicate a significant reduction in administrative workload and improved accessibility to patient data. The automated patient allotment system minimizes errors, while the SQLite database ensures reliable data storage. The system's real-time update feature enhances operational efficiency by synchronizing patient records instantly.

[Insert Screenshot: Database Management Module]

6. Conclusion and Future Enhancements

The HMS successfully addresses inefficiencies in traditional hospital administration by providing a cost-effective, automated, and user-friendly solution. Future work includes integrating AI-driven analytics for predictive healthcare insights, enhancing security measures, and expanding functionalities to include billing and pharmacy management.

References:

This paper presents an efficient, lightweight, and scalable approach to hospital management, offering a practical solution for healthcare institutions seeking digital transformation.

[1]. Lutz, M. (2011). Programming Python (4th ed.). O'Reilly Media.

[2]. Zelle, J. (2016). Python programming: An introduction to computer science (3rd ed.). Franklin, Beedle & Associates.

[3]. Driscoll, M. (2020). Python GUI programming with Tkinter. Packt Publishing.

[4]. Python, M. (2021). SQLite for beginners: A comprehensive guide to SQLite database programming. Independently Published.

[5]. Luba Novic, B. (2019). Introducing Python: Modern computing in simple packages. O'Reilly Media.

[6]. Sussman, G. J., & Wisdom, J. (2020). Functional programming in Python. The MIT Press.

[7]. Silberschatz, A., Korth, H. F., & Sudarshan, S. (2010). Database system concepts (6th ed.). McGraw-Hill.

[8]. Pressman, R. (2014). Software engineering: A practitioner's approach (8th ed.). McGraw-Hill.

[9]. Sommerville, I. (2015). Software engineering (10th ed.). Pearson.

[10]. Python Software Foundation. (n.d.). Python official documentation. Retrieved from https://docs.python.org/3/

[11]. TkDocs. (n.d.). Tkinter GUI programming. Retrieved from https://tkdocs.com/

[12]. Software Testing Help. (n.d.). Software testing techniques. Retrieved from https://www.softwaretestinghelp.com/

[13]. ResearchGate. (n.d.). Hospital management system overview. Retrieved from https://www.researchgate.net/

[14]. IEEE Xplore. (n.d.). Hospital management system research papers. Retrieved from https://ieeexplore.ieee.org/

[15]. Haux, R. (2018). Health information systems – Past, present, future. International Journal of Medical Informatics, 79(9), 599-610. https://doi.org/10.1016/j.ijmedinf.2010.06.003

[16]. Kuo, A. M. (2011). Opportunities and challenges of cloud computing to improve health care services. Journal of Medical Internet Research, 13(3), e67. https://doi.org/10.2196/jmir.1867

[17]. Mehta, N., & Pandit, A. (2018). Convergence of big data analytics and artificial intelligence in healthcare. International Journal of Medical Informatics, 114, 57-59. https://doi.org/10.1016/j.ijmedinf.2018.03.006

[18]. Pinho, J. S., Santos, M. F., & Costa, C. (2020). Healthcare information systems: A comprehensive review of integration strategies. Health Informatics Journal, 26(4), 2745-2760. https://doi.org/10.1177/1460458219899471

[19]. Raghupathi, W., & Raghupathi, V. (2014). Big data analytics in healthcare: Promise and potential. Health Information Science and Systems, 2(1), 3. https://doi.org/10.1186/2047-2501-2-3