KGISL MICROCOLLEGE

PY MED BILLING

R.HIRAN - 60724CBR 006

DATA SCIENCE-JUL-RE

2024

ABSTRACT

Hospital billing involves various complex processes, including patient registration, service billing, and bill generation. Manual systems are prone to errors, time-consuming, and often result in inaccurate billing. With the increasing demand for automation in healthcare, this project seeks to provide a basic yet effective solution.

This beginner-friendly project aims to design a simple hospital billing system using Python and Tkinter. The system will enable hospital staff to efficiently manage patient billing information, calculate total costs, and generate bills.

CONTENTS

S.NO	TOPICS	PAGE NO.
1	INTRODUCTION	4
2	SOFTWARE REQUIREMENT AND SPECIFICATION	5
3	PROJECT PLAN	5
4	FUTURE WORK	11
5	CONCLUSION	11

INTRODUCTION:

The Hospital Billing System is a simple and user-friendly application designed to manage patient billing information efficiently. This project aims to create a computerized system that replaces traditional manual billing methods, reducing errors and increasing productivity.

Objectives:

1. Design a user-friendly graphical user interface (GUI) using Tkinter.
2. Implement patient registration and billing information management.
3. Develop a system to calculate total costs and generate bills.

Scope:

This project focuses on the core functionality of hospital billing, including:

1. Patient registration
2. Service billing (consultation, medication, lab tests, etc.)
3. Bill generation and display

SOFTWARE REQUIREMENT AND SPECIFICATION:

Python can be used in healthcare for a variety of purposes, including data analysis, management, and predictive modeling .Hospitals have a variety of expenses.Python and Tkinter offer a powerful combination for developing hospital billing systems. Python's readability, versatility, and extensive libraries make it an ideal choice for handling complex data processing and calculations. Tkinter, a GUI toolkit, provides a simple and intuitive interface for interacting with the system.

PROJECT PLAN:

- Improves patient satisfaction
- Helps facilities run more efficiently
- Improves cash flow
- Ensures accurate payments
- Reduces debts

Objectives for the Users:

1. Hospital administrators: Efficiently manage patient billing information.
2. Hospital staff: Easily generate accurate bills and reduce manual errors.
3. Patients: Receive clear and transparent billing information.

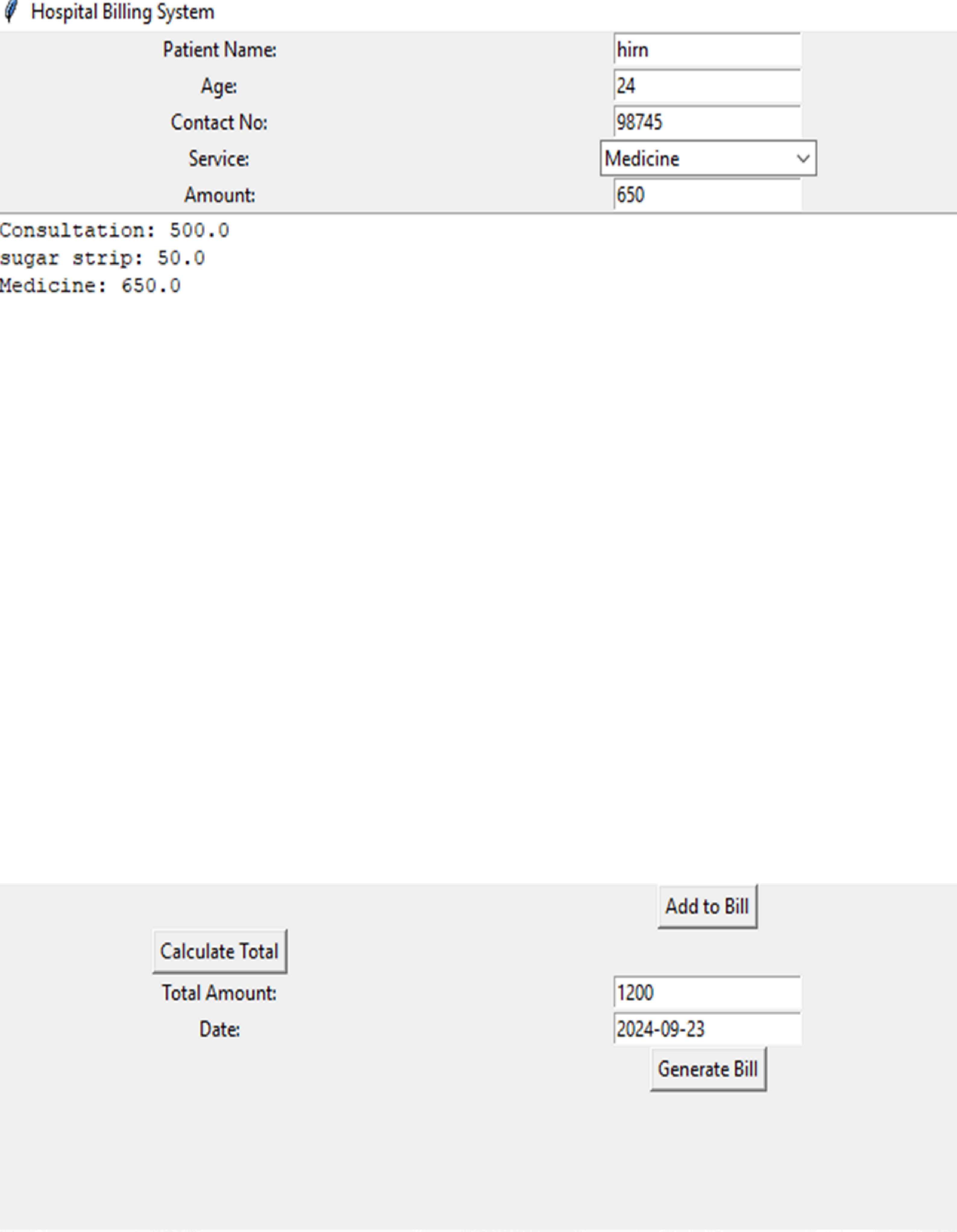
Technical Specifications:

1. Programming Language: Python 3.x
2. GUI Library: Tkinter

CODE:

```
import tkinter as tk
from tkinter import ttk, messagebox
from datetime import datetime

class HospitalBillingSystem:
    def __init__(self, root):
        self.root = root
        self.root.title("Hospital Billing System")

        self.patient_name = tk.StringVar()
        self.age = tk.StringVar()
        self.contact_no = tk.StringVar()
        self.service = tk.StringVar()
        self.amount = tk.StringVar()
        self.total_amount = tk.StringVar()
        self.date = tk.StringVar()
        self.date.set(datetime.now().strftime("%Y-%m-%d"))

        self.create_widgets()

    def create_widgets(self):
        tk.Label(self.root, text="Patient Name:").grid(column=0, row=0)
        tk.Entry(self.root, textvariable=self.patient_name).grid(column=1, row=0)

        tk.Label(self.root, text="Age:").grid(column=0, row=1)
        tk.Entry(self.root, textvariable=self.age).grid(column=1, row=1)

        tk.Label(self.root, text="Contact No:").grid(column=0, row=2)
        tk.Entry(self.root, textvariable=self.contact_no).grid(column=1, row=2)
tk.Label(self.root, text="Service:").grid(column=0, row=3)
            ttk.Combobox(self.root, textvariable=self.service, values=["Consultation", "Surgery",
"Medicine"]).grid(column=1, row=3)

        tk.Label(self.root, text="Amount:").grid(column=0, row=4)
        tk.Entry(self.root, textvariable=self.amount).grid(column=1, row=4)

        self.bill_text = tk.Text(self.root)
        self.bill_text.grid(column=0, row=5, columnspan=2)

        tk.Button(self.root, text="Add to Bill", command=self.add_to_bill).grid(column=1, row=6)
        tk.Button(self.root, text="Calculate Total", command=self.calculate_total).grid(column=0,
row=7)

        tk.Label(self.root, text="Total Amount:").grid(column=0, row=8)
        tk.Entry(self.root, textvariable=self.total_amount).grid(column=1, row=8)

        tk.Label(self.root, text="Date:").grid(column=0, row=9)
        tk.Entry(self.root, textvariable=self.date).grid(column=1, row=9)

            tk.Button(self.root, text="Generate Bill", command=self.generate_bill).grid(column=1,
row=10)
    def add_to_bill(self):
        try:
            service = self.service.get()
            amount = float(self.amount.get())
            self.bill_text.insert(tk.END, f"{service}: {amount}\n")
        except ValueError:
            messagebox.showerror("Error", "Amount must be a number")
```

```
 def calculate_total(self):
 try:
 total = 0
 for line in self.bill_text.get("1.0", tk.END).splitlines():
 total += float(line.split(":")[1].strip())
 self.total_amount.set(str(total))
 except ValueError:
 messagebox.showerror("Error", "Invalid bill items")

 def generate_bill(self):
 bill = f"Patient Name: {self.patient_name.get()}\nAge: {self.age.get()}\nContact No:
{self.contact_no.get()}\nDate: {self.date.get()}\n\n{self.bill_text.get('1.0',
tk.END)}\nTotal Amount: {self.total_amount.get()}"
 print(bill)
 messagebox.showinfo("Success", "Bill generated successfully")

root = tk.Tk()
hospital_billing_system = HospitalBillingSystem(root)
root.mainloop()
```



Patient Name: hirn

Age: 24

Contact No: 98745

Service: Medicine

Amount: 650

```
Consultation: 500.0
sugar strip: 50.0
Medicine: 650.0
```

Total Amount: 1200

Date: 2024-09-23

Expected Outcomes:

1. A functional hospital billing system with GUI.
2. Improved efficiency and accuracy in patient billing.
3. Enhanced user experience for hospital staff and patients.

Limitations:

1. Basic security measures (no encryption).
2. Limited scalability.
3. No integration with existing hospital management systems.

Project Timeline:

- Planning and design: 2 days
- Implementation: 7 days
- Testing and debugging: 3 days
- Documentation and reporting: 2 days

Project Objectives and Success Criteria:

- Successful implementation of patient registration and billing functionality.
- Accurate calculation and generation of bills.
- User-friendly GUI with clear navigation.
- Timely completion within the specified timeline.

Conclusion and Future Enhancements

User-Friendly Interface
The Tkinter-based GUI provided an intuitive and easy-to-navigate interface for both administrators and patients.

Patient Management:
The system effectively managed patient records, including personal information, medical history, and insurance details.

Billing Calculations:
Accurate calculations were implemented for various billing components such as consultations, procedures, medications, and hospital stays

Reports and Analytics:
Detailed reports were generated to provide insights into billing trends, identify areas for improvement, and support financial decision-making.