

“GST BILLING APPLICATION”

A

Project Report

*submitted in partial fulfillment of the
requirements for the award of the degree of*

BACHELOR OF COMPUTER APPLICATION

in

COMPUTER SCIENCE & ENGINEERING

by

Name
Arpit Gupta

Roll No.
500122077



School of Computer Science

University of Petroleum & Energy Studies

Bidholi, Via Prem Nagar, Dehradun, Uttarakhand

April– 2024

ABSTRACT

A “**GST billing application**” automates the process of generating invoices and managing transactions in compliance with Goods and Services Tax regulations. It enables businesses to accurately calculate and apply GST rates to their products and services, ensuring legal compliance and reducing manual errors. The application also maintains detailed records of transactions, facilitating seamless tax filing and auditing.

- **Automation of Invoicing:** The application automates the process of generating invoices, saving time and reducing errors.
- **GST Compliance:** Ensures compliance with Goods and Services Tax regulations by accurately calculating and applying GST rates to products and services.
- **Transaction Management:** Maintains detailed records of transactions, simplifying tax filing and auditing processes.
- **User-Friendly Interface:** Provides a user-friendly interface for generating and printing bills, improving operational efficiency.
- **Financial Management:** Enhances overall financial management by streamlining invoicing processes and improving tax compliance.

USE CASES:

- **Invoicing:** Generate invoices quickly and accurately, including GST details, for products and services sold.
- **Tax Compliance:** Automatically calculate and apply the correct GST rates, ensuring compliance with tax regulations.
- **Record Keeping:** Maintain detailed records of transactions, including invoices, payments, and taxes, for easy access and reference.
- **Financial Reporting:** Generate reports on sales, taxes collected, and other financial data to aid in decision-making and tax filing.
- **Inventory Management:** Integrate with inventory systems to track stock levels and reconcile sales with available inventory.
- **Customer Management:** Maintain customer profiles and purchase history for personalized service and targeted marketing.
- **Efficiency:** Streamline billing processes, reduce manual errors, and improve overall efficiency in managing sales transactions.

TABLE OF CONTENTS

S.NO.	Heading	Page No.
1.	Introduction	1
2.	Motivation	2
3.	Working diagram	3
4.	Methodology	6
5.	Source code	9
6.	Output	15
7.	Conclusion	17
8.	References	18

1.INTRODUCTION

A GST billing application is a crucial tool for businesses to manage their invoicing and taxation processes efficiently. With the implementation of the Goods and Services Tax (GST) in many countries, including India, businesses are required to adhere to specific invoicing guidelines to comply with the law. A GST billing application simplifies this process by automating the calculation of GST, generating invoices, and maintaining accurate records.

It ensures compliance with GST regulations and provides businesses with the necessary tools to track their sales, manage inventory, and generate financial reports.

Furthermore, a GST billing application improves accuracy and reduces the risk of errors in billing and taxation. It streamlines the entire invoicing process, from creating invoices to recording payments, making it easier for businesses to maintain their financial records and stay compliant with GST laws.

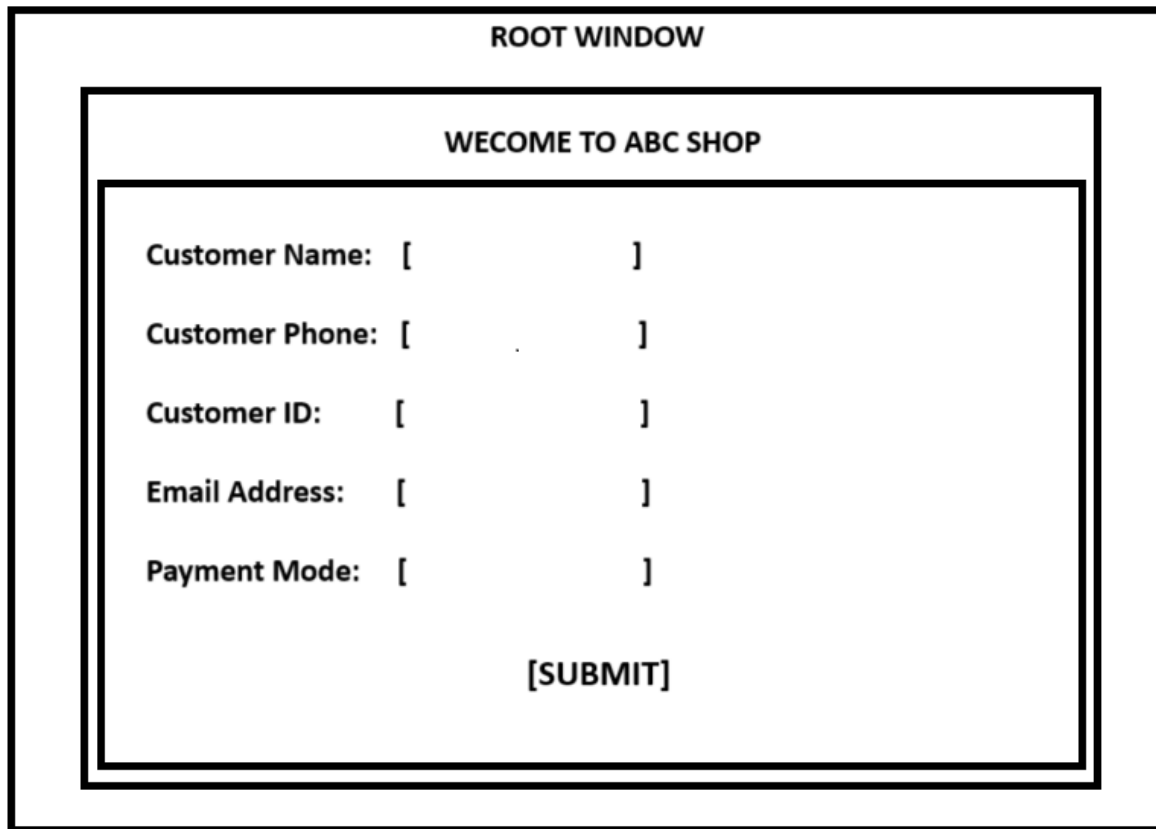
2.MOTIVATION

The motivation behind developing a GST billing application from the need to simplify and streamline the invoicing and taxation processes for businesses. Here are some key motivations:

1. **Compliance:** GST laws mandate businesses to maintain accurate records and issue invoices in a prescribed format. A GST billing application helps businesses comply with these regulations by automating the invoicing process and ensuring that invoices are GST-compliant.
2. **Efficiency:** Manual invoicing processes are time-consuming and prone to errors. By automating invoicing and tax calculations, a GST billing application saves time and reduces the risk of errors, allowing businesses to focus on their core activities.
3. **Accuracy:** GST involves complex tax calculations, including different tax rates for different products and services. A GST billing application ensures accurate tax calculations, reducing the risk of under or overpayment of taxes.
4. **Record-keeping:** GST laws require businesses to maintain detailed records of their transactions. A GST billing application helps businesses keep track of their sales, purchases, and taxes, making it easier to generate reports and comply with audit requirements.
5. **User Satisfaction:** A well-designed GST billing application can enhance the user experience by providing clear and transparent invoices. This can improve user satisfaction .

Overall, the motivation behind a GST billing application is to simplify invoicing ensure compliance with GST laws, and improve the efficiency and accuracy of business operations.

3.Working Diagram



The diagram illustrates the layout of the 'ROOT WINDOW'. It consists of three nested rectangular frames. The outermost frame is labeled 'ROOT WINDOW' at the top center. Inside it is a frame labeled 'WECOME TO ABC SHOP' at the top center. The innermost frame contains five input fields, each with a label followed by a bracketed space: 'Customer Name: []', 'Customer Phone: []', 'Customer ID: []', 'Email Address: []', and 'Payment Mode: []'. Below these fields, centered, is a button labeled '[SUBMIT]'.

Root Window:

- 1.This is the main window of the application where the user enters customer details such as name, phone number, ID (optional), email address, and payment mode.
- 2.The user clicks the "Submit" button to save these details and proceed to enter item details.

DETAILS WINDOW

ENTER ITEM DETAILS

Item Name : []

Quantity: []

Rate (per pc's): []

GST (In %): []

[Add item]

[Next]

Details Window:

- 1.After submitting customer details, this window allows the user to enter details for each item purchased.
- 2.The user enters the item name, quantity, rate (per piece), and GST rate (in percentage).
- 3.Clicking "Add Item" adds the item to the invoice and allows the user to enter details for another item. Clicking "Next" moves to the final receipt window.

4.Methodology

The methodology for developing the GST billing application involves several steps, including designing the user interface, implementing functionality for entering customer and item details, calculating taxes, and generating the final receipt. Here's a general methodology for developing the application:

1.Requirement Analysis:

- Identify the requirements of the application, including the ability to enter customer details, add items to the invoice, calculate taxes (SGST, CGST), and generate a final receipt.

2.Design User Interface:

- Design the user interface using Tkinter, including windows for entering customer details, item details, and displaying the final receipt.

3.Implement User Interface:

- Use Tkinter to implement the designed user interface, including labels, entry fields, buttons, and frames for organizing the layout.

4.Implement Functionality:

- Implement functionality for entering customer details and saving them to a file (records.txt).
- Implement functionality for entering item details and saving them to a file (details.txt).
- Calculate taxes (SGST, CGST) based on the entered item details and display the total amount including taxes.
- Generate the final receipt with customer details, itemized list of items purchased, and total amount including taxes.

5.Testing:

- Test the application to ensure that it works as expected, including entering customer details, adding items, calculating taxes, and generating the final receipt.

6.Deployment:

- Once the application is tested and working correctly, deploy it for use by businesses for their GST billing needs.

7.Maintenance:

- Regularly update the application to comply with any changes in GST regulations and to fix any bugs or issues that may arise.

8.User Training:

- Provide training to users on how to use the application effectively for their GST billing requirements.

5.SOURCE CODE

```
from tkinter import *
from tkinter import ttk
import datetime
import os
```

```
date = datetime.date.today()
root = Tk()
root.title('RECIPT APP')
root.configure(bg='#070F2B')
root.geometry("644x344")
```

```
def getvals():
    print("Submitting form")
    customer_data = {
        "Name": namevalue.get(),
        "Phone": phonevalue.get(),
        "ID": idvalue.get(),
        "Email": emailvalue.get(),
        "Payment Mode": paymentmodevalue.get()
    }
    print(customer_data)
    with open("records.txt", "a") as f:
        f.write(str(customer_data) + "\n")
```

```
def submit_close():
    getvals()
    root.destroy()
    on_closing()
```

```
Label(root, text="Welcome to abc shop", font="comicsansms 13 bold", pady=15,
fg='white', bg='#070F2B').grid(row=0, column=3)
```

```
name = Label(root, text="Customer Name", bg='#070F2B', fg='white')
```

```
phone = Label(root, text="Customer Phone Number", bg='#070F2B', fg='white')
id_label = Label(root, text="Customer ID (optional)", bg='#070F2B', fg='white')
email = Label(root, text="Email Address", bg='#070F2B', fg='white')
paymentmode = Label(root, text="Payment Mode", bg='#070F2B', fg='white')
```

```
name.grid(row=1, column=2)
phone.grid(row=2, column=2)
id_label.grid(row=3, column=2)
email.grid(row=4, column=2)
paymentmode.grid(row=5, column=2)
```

```
namevalue = StringVar()
phonevalue = StringVar()
idvalue = StringVar()
emailvalue = StringVar()
paymentmodevalue = StringVar()
```

```
nameentry = Entry(root, textvariable=namevalue, bg='#070F2B', fg='white')
phoneentry = Entry(root, textvariable=phonevalue, bg='#070F2B', fg='white')
identry = Entry(root, textvariable=idvalue, bg='#070F2B', fg='white')
emailentry = Entry(root, textvariable=emailvalue, bg='#070F2B', fg='white')
paymentmodeentry = Entry(root, textvariable=paymentmodevalue, bg='#070F2B',
fg='white')
```

```
nameentry.grid(row=1, column=3)
phoneentry.grid(row=2, column=3)
identry.grid(row=3, column=3)
emailentry.grid(row=4, column=3)
paymentmodeentry.grid(row=5, column=3)
```

```
Button(root, text="Submit", bg='white', fg='#070F2B',
command=submit_close).grid(row=7, column=3)
```

```
def on_closing():
    win = Tk()
    win.title('DETAIL')
    win.configure(bg= '#070F2B')
    win.geometry("644x344")
```

```

Label(win,text="Enter item details ", font="arial 13 bold",
pady=15,fg='white',bg= '#070F2B').grid(row=0, column=3)
def getitem():
    print("Submitting detail of item ")
    item_data = {
        "Item Name": itemvalue.get(),
        "Quantity": qtyvalue.get(),
        "Rate": ratevalue.get(),
        "GST": gstvalue.get()
    }
    print(item_data)

    with open("details.txt", "a") as g:
        g.write(str(item_data) + "\n")

item = Label(win, text="Item Name",bg= '#070F2B',fg='white')
qty = Label(win, text="Quantity",bg= '#070F2B',fg='white')
rate= Label(win,text= "Rate(per pc's)",bg= '#070F2B',fg='white')
gst = Label(win, text="GST (in %)",bg= '#070F2B',fg='white')

item.grid(row=1, column=2)
qty.grid(row=2, column=2)
rate.grid(row=3, column=2)
gst.grid(row=4, column=2)

itemvalue = StringVar()
qtyvalue = IntVar()
ratevalue = IntVar()
gstvalue = IntVar()

itementry = Entry(win, textvariable=itemvalue,bg= '#070F2B',fg='white')
qtyentry = Entry(win, textvariable=qtyvalue,bg= '#070F2B',fg='white')
rateentry = Entry(win, textvariable=ratevalue,bg= '#070F2B',fg='white')
gstentry = Entry(win, textvariable=gstvalue,bg= '#070F2B',fg='white')

itementry.grid(row=1, column=3)
qtyentry.grid(row=2, column=3)
rateentry.grid(row=3, column=3)
gstentry.grid(row=4, column=3)

```

```

    Button(win,text="Add item",bg= 'white',fg='#070F2B',
command=getitem).grid(row=7, column=3)
    def next():
        win.destroy()
        final()

    Button(win,text="Next",command=next).grid(row=8,column=3)
    win.mainloop()

```

```

def final():
    import tkinter as tk
    from tkinter import ttk
    final=tk.Tk()

    final.attributes("-fullscreen",True)
    final.title("Final receipt")
    lbl = Label(final, bg='black')
    customer_frame = ttk.Frame(final, padding="20")
    customer_frame.pack(pady=20)

```

```

    tk.Label(customer_frame, text="Customer Details", font=("Arial", 20),
fg='black').grid(row=0, column=0, columnspan=2, pady=10)

```

```

    tk.Label(customer_frame, text="Customer Name:", font=("Arial", 14),
fg='black').grid(row=1, column=0, sticky="e")
    tk.Label(customer_frame, text=namevalue.get(), font=("Arial", 14),
fg='black').grid(row=1, column=1, sticky="w")

```

```

    tk.Label(customer_frame, text="Customer Phone:", font=("Arial", 14),
fg='black').grid(row=2, column=0, sticky="e")
    tk.Label(customer_frame, text=phonevalue.get(), font=("Arial", 14),
fg='black').grid(row=2, column=1, sticky="w")

```

```

    tk.Label(customer_frame, text="Customer ID:", font=("Arial", 14),
fg='black').grid(row=3, column=0, sticky="e")
    tk.Label(customer_frame, text=idvalue.get(), font=("Arial", 14),
fg='black').grid(row=3, column=1, sticky="w")

```

```

tk.Label(customer_frame, text="Email:", font=("Arial", 14),
fg='black').grid(row=4, column=0, sticky="e")
tk.Label(customer_frame, text=emailvalue.get(), font=("Arial", 14),
fg='black').grid(row=4, column=1, sticky="w")

tk.Label(customer_frame, text="Date:", font=("Arial", 14),
fg='black').grid(row=5, column=0, sticky="e")
tk.Label(customer_frame, text=datetime.date.today(), font=("Arial", 14),
fg='black').grid(row=5, column=1, sticky="w")

tree_frame = ttk.Frame(final)
tree_frame.pack(pady=10)
columns = ("item_name", "qty", "rate", "amount", "sgst", "cgst",
"amount_with_gst")
tree = ttk.Treeview(tree_frame, columns=columns, show="headings")
tree.heading("item_name", text="Item Name")
tree.heading("qty", text="Quantity")
tree.heading("rate", text="Rate")
tree.heading("amount", text="Amount")
tree.heading("sgst", text="SGST")
tree.heading("cgst", text="CGST")
tree.heading("amount_with_gst", text="Amount (incl. GST)")

total_amount_with_gst = 0
with open("details.txt", "r") as f:
    for line in f:
        item_data = eval(line.strip())
        qty = item_data["Quantity"]
        rate = item_data["Rate"]
        gst = item_data["GST"]
        amount = qty * rate
        sgst = gst / 2
        cgst = gst / 2
        gst_amount = (amount * gst) / 100
        amount_with_gst = amount + gst_amount
        total_amount_with_gst += amount_with_gst
    int(total_amount_with_gst)
    tree.insert("", "end", values=(item_data["Item Name"], qty, rate, amount,
sgst, cgst, amount_with_gst))

```



```
tree.pack()

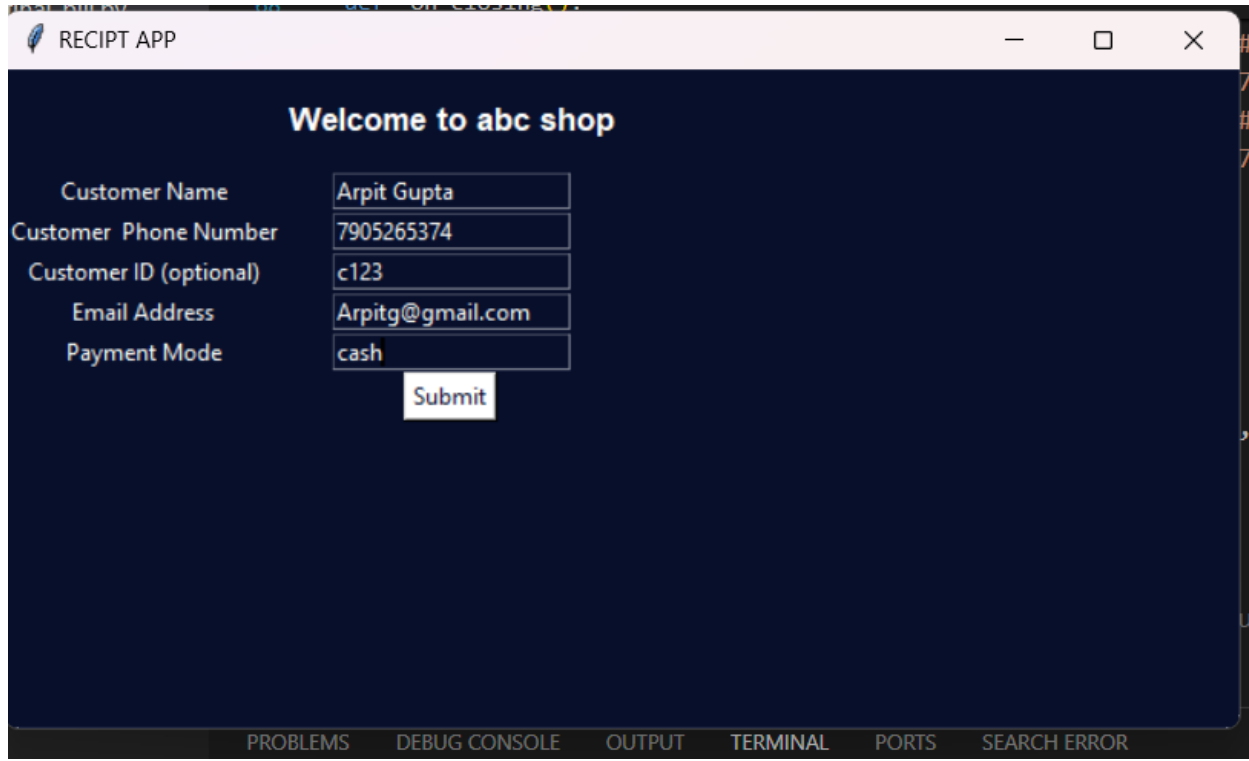
total_frame = ttk.Frame(final, padding="20")
total_frame.pack(pady=20, side="left")

Label(total_frame, text=f"Total Amount (incl. GST):
{'{:.2f}'.format(total_amount_with_gst)}", font=("Arial", 14), fg='black').pack()
Label(total_frame, text=f"Mode of Payment: {paymentmodevalue.get()}",
font=("Arial", 14)).pack()

def delete_file(details):
    os.remove(details)
    delete_file("details.txt")
    delete_file("records.txt")
    final.mainloop()

root.mainloop()
```

6.Output



The screenshot shows a web browser window with the title 'RECIPT APP'. The main content area has a dark blue background with the text 'Welcome to abc shop' in white. Below this, there is a form with five input fields and a submit button. The fields are labeled on the left and contain text on the right. The labels are 'Customer Name', 'Customer Phone Number', 'Customer ID (optional)', 'Email Address', and 'Payment Mode'. The corresponding values are 'Arpit Gupta', '7905265374', 'c123', 'Arpitg@gmail.com', and 'cash'. A 'Submit' button is located below the 'Payment Mode' field. At the bottom of the window, there is a dark grey bar with several tabs: 'PROBLEMS', 'DEBUG CONSOLE', 'OUTPUT', 'TERMINAL', 'PORTS', and 'SEARCH ERROR'. The 'TERMINAL' tab is currently selected.

Customer Name	Arpit Gupta
Customer Phone Number	7905265374
Customer ID (optional)	c123
Email Address	Arpitg@gmail.com
Payment Mode	cash

Submit

Fig:First window for recipt app to take input from user

DETAIL

Enter item details

Item Name:

Quantity:

Rate(per pc's):

GST (in %):

Fig: Detail window to take input from user and add item

Customer Details

Customer Name: Arpit Gupta
 Customer Phone: 7905265374
 Customer ID: c123
 Email: arpitg@gmail.com
 Date: 2024-04-16

Item Name	Quantity	Rate	Amount	SGST	CGST	Amount (incl. GST)
bag	10	500	5000	2.5	2.5	5250.0
cover	5	600	3000	2.5	2.5	3150.0

Total Amount (incl. GST): 8400.00
 Mode of Payment: cash

Fig: Final receipt window proceed by entered output

7.Conclusion

In conclusion, the development of a GST billing application using Tkinter provides businesses with a user-friendly and efficient solution for managing their invoicing and taxation processes. By following the methodology outlined above, the application can be designed and implemented to meet the specific requirements of businesses while ensuring compliance with GST regulations.

The application allows businesses to easily enter customer details, add items to the invoice, calculate taxes (SGST, CGST), and generate a final receipt. The use of Tkinter ensures a visually appealing and intuitive user interface, making it easy for users to navigate and use the application.

Overall, the GST billing application provides businesses with a streamlined and automated solution for managing their billing and taxation needs, improving efficiency, accuracy, and compliance with GST laws.

8.References

(As per their appearance in the chapters)

Title: GST Billing Application

Description: A desktop application for managing invoicing and taxation processes, compliant with Goods and Services Tax (GST) regulations.

Technologies:

1. Python for application logic.
2. Tkinter for the graphical user interface.
3. File handling data storage.

References:

1. Youtube (Logical help)
2. GEEK for GEEKS (Syntax help)
3. Grammerly (for report writing help)
4. Class notes (for other doubt)