

BILLING SYSTEM

PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR
THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

Computer Science and Engineering

SUBMITTED BY

(Vansham Pandey -2100460100115)

SESSION: 2023-24



Dr. APJ Abdul Kalam Technical University, Uttar Pradesh
LUCKNOW, INDIA

TABLE OF CONTENTS

• INTRODUCTION	5
• OBJECTIVE	6
• TECHNOLOGY USED:	7
• SYSTEM REQUIREMENT:	13
• FLOWCHART:	18
• LITERATURE RIVIEW:	19
• OUTPUT SCREENSHOTS:	22
• CONCLUSION:	24
• FUTURE SCOPE:	25
• REFRENCES:	28
• APPENDIX:	29

CERTIFICATE

This is to certify that the Project Report on the topic of "**BILLING SYSTEM**" is submitted by **Vansham Pandey CSE 3rd year (2100460100115)** in fulfillment for the award of degree of **BACHELOR OF TECHNOLOGY** in **Computer Science and Engineering** has been found satisfactory and is approved for submission.

- i. Embodies the work of the candidates themselves,
- ii. Has duly been completed,
- iii. Is up to the desired standard both in respect of contents and language for being referred to the examiners.

Dr. Vijay Singh

Associate Professor

(CSE) Department

MPEC

Dr. Atul Kumar

Head of Department CSE

MPEC

ACKNOWLEDGEMENT

It gives us the great sense of pleasure to present the report of the B.Tech project undertaken during B.Tech 3rd year. I owe special debt of gratitude to **Dr. Atul kumar** (Head of Department in Computer Science and Engineering).

The satisfaction that accompanies the successful completion of any task would be incomplete without mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all effort with success.

We are grateful to our project guide **Mr. Vijay Singh** for the guidance, inspiration and constructive suggestions that help us in the preparation of this project.

I also thank my college and colleagues who helped us in successful completion of the project.

Vansham Pandey

(2100460100115)

INTRODUCTION

Customer Billing System Project is a simple console application designed to demonstrate the practical use of python programming language and its features as wells as to generate an application which can be used in any departmental store, shops, cafes etc. for billing to the customer.

You may use this program to keep track of your frequent customers' information such as name, address, phone number, paid amount, due amount, payment date, and so on. You may also add and update accounts at any moment if you have a new client.

The source code of Customer Billing System Project in C has been written in C programming language without of the use of any graphics. The code is complete, totally error free and not that long – just.

OBJECTIVE

We aims to create or develop a system that is capable and reliable in the whole transaction flow such as tracking, retrieving and storing data in an appropriate way.

In particular it aims to:

- Provide a database that will store information.
- Develop a system that will lessen process delay in terms of releasing receipts and customer bill.
- Provide security of two levels of users.
- Provides a convenient solution of Billing Pattern.
- Make an easy-to-use environment for users and Customers.

Technology Used

PYTHON Programming Language

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.

Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0.^[34] Python 2.0 was released in 2000. Python 3.0, released in 2008, was a major revision not completely backward-compatible with earlier versions. Python 2.7.18, released in 2020, was the last release of Python 2.

Python consistently ranks as one of the most popular programming languages.

History of Python programming language

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wickenden & Informatica (CWI) in the Netherlands as a successor to the ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's "benevolent dictator for life", a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a five-member Steering Council to lead the project.

Python 2.0 was released on 16 October 2000, with many major new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released on 3 December 2008, with many of its major features backported to Python 2.6.x and 2.7.x. Releases of Python 3 include the `2to3` utility, which automates the translation of Python 2 code to Python 3.

Python 2.7's end-of-life was initially set for 2015, then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3. No further security patches or other improvements will be released for it. Currently only 3.8 and later are supported (2023 security issues were fixed in e.g. 3.7.17, the final 3.7.x release.

In 2021 (and again twice in 2022), security updates were expedited, since all Python versions were insecure (including 2.7) because of security issues leading to possible remote code execution and web-cache poisoning. In 2022, Python 3.10.4 and 3.9.12 were expedited and 3.8.13, because of many security issues. When Python 3.9.13 was released in May 2022, it was announced that the 3.9 series (joining the older series 3.8 and 3.7) would only receive security fixes in the future. On 7 September 2022, four new releases were made due to a potential denial-of-service attack: 3.10.7, 3.9.14, 3.8.14, and 3.7.14.

As of October 2023, Python 3.12 is the stable release, and 3.12 and 3.11 are the only versions with active (as opposed to just security) support. Notable changes in 3.11 from 3.10 include increased program execution speed and improved error reporting.

Python 3.12 adds syntax (and in fact every Python since at least 3.5 adds some syntax) to the language, the new (soft) keyword `type` (recent releases have added a lot of typing support e.g. new type union operator in 3.10), and 3.11 for exception handling, and 3.10 the `match` and `case` (soft) keywords, for structural pattern matching statements. Python 3.12 also drops outdated modules and functionality, and future versions will too, see below in Development section.

Python 3.11 claims to be between 10 and 60% faster than Python 3.10, and Python 3.12 adds another 5% on top of that. It also has improved error messages, and many other changes.

Features in Python

The features of python are stated below:

1.Free and Open Source

Python language is freely available at the official website and you can download it from the given download link below click on the **Download Python** keyword. Download Python Since it is open-source, this means that source code is also available to the public. So you can download it, use it as well as share it.

2. Easy to code

Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in the Python language and anybody can learn Python basics in a few hours or days. It is also a developer friendly language.

3. Easy to Read

As you will see, learning Python is quite simple. As was already established, Python's syntax is really straightforward. The code block is defined by the indentations rather than by semicolons or brackets.

4. Object-Oriented Language

One of the key features of Python is Object-Oriented programming. Python supports object-oriented language and concepts of classes, object encapsulation, etc.

5. GUI Programming Support

Graphical User interfaces can be made using a module such as PyQt5, PyQt4, wxPython, or Tk in Python. PyQt5 is the most popular option for creating graphical apps with Python.

6. High-Level Language

Python is a high-level language. When we write programs in Python, we do not need to remember the system architecture, nor do we need to manage the memory.

7. Large Community Support

Python has gained popularity over the years. Our questions are constantly answered by the enormous Stack Overflow community. These websites have already provided answers to many questions about Python, so Python users can consult them as needed.

8. Easy to Debug

Excellent information for mistake tracing. You will be able to quickly identify and correct the majority of your program's issues once you understand how to interpret Python's

error traces. Simply by glancing at the code, you can determine what it is designed to perform.

9. Python is a Portable language

Python language is also a portable language. For example, if we have Python code for Windows and if we want to run this code on other platforms such as Linux, Unix, and Mac then we do not need to change it, we can run this code on any platform.

10. Python is an Integrated language

Python is also an Integrated language because we can easily integrate Python with other languages like C, C++, etc.

11. Interpreted Language:

Python is an Interpreted Language because Python code is executed line by line at a time. like other languages C, C++, Java, etc. there is no need to compile Python code this makes it easier to debug our code. The source code of Python is converted into an immediate form called **bytecode**.

12. Large Standard Library

Python has a large standard library that provides a rich set of modules and functions so you do not have to write your own code for every single thing. There are many libraries present in Python such as regular expressions, unit-testing, web browsers, etc.

13. Dynamically Typed Language

Python is a dynamically-typed language. That means the type (for example- int, double, long, etc.) for a variable is decided at run time not in advance because of this feature we don't need to specify the type of variable.

14. Frontend and backend development

With a new project py script, you can run and write Python codes in HTML with the help of some simple tags <py-script>, <py-env>, etc. This will help you do frontend development work in Python like javascript. Backend is the strong forte of Python it's extensively used for this work cause of its frameworks like Django and Flask.

15. Allocating Memory Dynamically

In Python, the variable data type does not need to be specified. The memory is automatically allocated to a variable at runtime when it is given a value. Developers do not need to write `int y = 18` if the integer value 15 is set to y. You may just type `y=18`.

Advantages and Disadvantages of Python

Every programming language comes with benefits and limitations as well. These benefits and limitations can be treated as advantages and disadvantages. Python also has a few disadvantages over many advantages. Let's discuss each here:

Advantages of Python:

- Easy to learn, read, and understand
- Versatile and open-source
- Improves productivity
- Supports libraries
- Huge library
- Strong community
- Interpreted language

Disadvantages of Python:

- Restrictions in design
- Memory inefficient
- Weak mobile computing
- Runtime errors
- Slow execution speed

Applications of Python

Python is a highly-demanding and popular programming language used in various disciplines. Let's talk about a few of the major areas where Python is in great demand.

- Developers can work on **web development** and create amazing and feature-rich websites using Python and its frameworks like Django. Python comes with inbuilt libraries and tools which makes the process of developing web applications easier.
- Data scientists can **build powerful AI models** using Python snippets. Due to its easily understandable feature, it allows developers to write complex algorithms.
- You can also automate your tasks using Python with libraries like Beautiful Soup, pandas, matplotlib, etc. for **scraping and web automation**.
- You can also use Python to work on **CAD (computer-aided designs) designs**, to create 2D and 3D models digitally.
- Using libraries like Pandas, and TensorFlow, experts can work on **data analysis and machine learning applications** for statistical analysis, data manipulation, etc.
- Python can also be used by developers to **build games** using Pygame to develop 2D and 3D games. Some of the popular games built using Python are Pirates of the Caribbean, Battlefield 2, etc.

Conclusion

Python has a lot of reasons which make it a more popular and highly demanding programming language. We have discussed all the pointers that any beginner who's starting to learn Python should focus on. If you want to become a Python developer, learn Python in-depth and try to gain more advanced concepts. Also, developers can use Python in developing feature-rich applications. Gain a deep knowledge of Python and be a successful Python developer.

Software Requirements

For Python we require an IDE that is efficient and useful code editor:

1. Visual Studio Code
2. Windows 10

IDE

- IDE is the Integrated Development Environment that provides the user interface for code development, testing and debugging features. It helps to organize the project artifacts that are relevant to the source code of the software application.
- It provides several tools and features to make the development easy and standardize based upon the programming language the developer writes the code. The IDEs also have the functionalities to compile and interpret the program.
- It is a software application that defines the visual representation of the location of the files easily and makes it more understandable for the user. It contains development tools **Such** as text editors, compilers and test platforms.

PyCharm:

PyCharm is an integrated development environment (IDE) used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supports web development with Django. PyCharm is developed by the Czech company JetBrains.

It is cross-platform, working on Microsoft Windows, macOS and Linux. PyCharm has a Professional Edition, released under a proprietary license and a Community Edition released under the Apache License.^[5] PyCharm Community Edition is less extensive than the Professional Edition.

Features

- Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python code refactoring: including rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Support for web frameworks: Django, web2py and Flask
- Integrated Python debugger
- Integrated unit testing, with line-by-line coverage
- Google App Engine Python development
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with changelists and merge
- Scientific tools integration: integrates with IPython Notebook, has an interactive Python console, and supports Anaconda as well as multiple scientific packages including Matplotlib and NumPy.^{[6][7]}

Hardware Requirement

Processor:

Intel CORE i5 vPRO

CPU @ 1.70GHz

1.90GHz RAM: 8GB is
baseline.

- **Processor:** The processor also known as CPU is a piece of hardware. It is often referred to as the “Brain of a PC” because all the computations and processing are carried out directly or indirectly by the processor containing millions of transistors. It is a single chip that is capable of processing data. In fact, the performance, capability, and pricing of a computer system are largely determined by the processor to present in it. It controls all the components in a PC.
- **Disk (HDD) :** Hard Disk Drive is a storage device which stores the operation system (OS), installed software, and the other computer files. HDD means the data is retained when our computer system is shut down. HDD is also called a fixed disk, hard disk, or hard drive. The HDD was introduced in the year 1956 by IBM.
- **SSD :** A solid-state drive (SSD) is a new generation of storage device used in computers. SSDs store data using flash-based memory, which is much faster than the traditional hard disks they've come to replace. SSDs also have no moving parts, and upgrading to one is a great way to speed up your computer and make it more resilient.

Objectives:

Let us now see some of the objectives of the operating system, which are:

1. **Convenient to use:** One of the objectives is to make the computer system more convenient to use in an efficient manner.
2. **User Friendly:** To make the computer system more interactive with a more convenient interface for the users.
3. **To provide easy access to users for using resources by acting as an intermediary between the hardware and its users.**
4. **For managing the resources of a computer.**
5. **Controls and Monitoring:** By keeping the track of who is using which resource, granting resource requests, and mediating
6. **conflicting requests from different programs and users.**
7. **Providing efficient and fair sharing of resources between the users and programs.**

Characteristics:

Let us now discuss some of the important characteristic features of operating systems:

- **Device Management:** The operating system keeps track of all the devices. So, it is also called the Input / Output controller that decides which process gets the device, when, and for how much time.
- **File Management:** It allocates and de-allocates the resources and also decides who gets the resource.
- **Job Accounting:** It keeps the track of time and resources used by various jobs or users.
- **Error-detecting Aids:** It contains methods that include the production of dumps, traces, error messages, and other debugging and error-detecting methods.

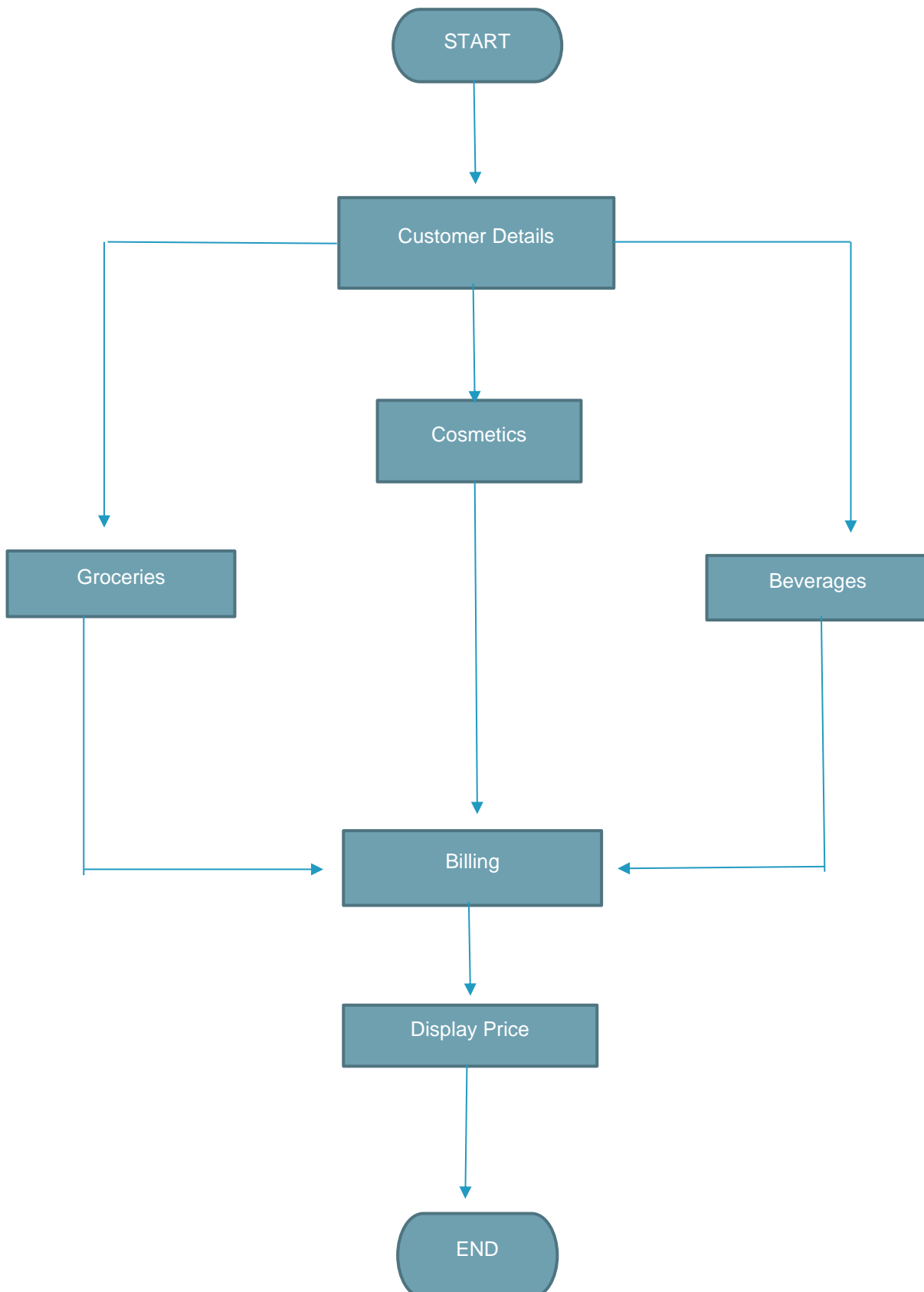
Types of the operating system:

- **Batch Operating System:** This type of operating system does not interact with the computer directly. There is an operator which takes similar jobs having the same requirements and groups them into batches.
- **Time-sharing operating System:** This type of operating system allows many users to share computer resources.

- **Distributed operating System:** This type of operating system manages a group of different computers and makes appear to be a single computer.
- **Network operating system:** This type of operating system running on a server and provides the capability to manage data, users, groups, security, applications, and other networking functions.
- **Real-time operating system:** This type of operating system serves real time system and the time interval required to process and respond to inputs is very small.
- **Processor:**
Intel Pentium and later
- **Disk Space:**
1GB or more

IMPLEMENTATION AND RESULTS

Flowchart:



LITERATURE REVIEW:

A literature review for a billing system project would typically involve an examination of existing research, publications, and relevant sources related to billing systems, particularly in the context of software development and implementation. Below is a generic outline to help you structure your literature review for a billing system project:

1. Introduction to Billing Systems:

- Define what a billing system is and its importance in various industries.
- Discuss the role of billing systems in automating financial transactions and managing revenue streams.
- Highlight the significance of efficient billing systems for businesses.

2. Historical Overview:

- Provide a historical perspective on billing systems, tracing their evolution over time.
- Identify key milestones and technological advancements that have influenced billing systems.

3. Types of Billing Systems:

- Classify billing systems based on various criteria such as industry, scale, and functionality.
- Discuss the differences between manual and automated billing systems.

4. Key Components of Billing Systems:

- Explore the essential features and components that make up a billing system.
- Discuss how these components contribute to the overall efficiency of the system.

5. Challenges and Issues in Billing Systems:

- Examine common challenges faced in the development and implementation of billing systems.
- Discuss issues related to security, accuracy, scalability, and compliance.

6. Technological Trends and Innovations:

- Investigate current trends and innovations in billing system technologies.
- Explore the integration of emerging technologies such as blockchain, AI, and cloud computing in billing systems.

7. Case Studies:

- Present case studies of successful billing system implementations.
- Analyze the strategies adopted, challenges faced, and outcomes achieved in these case studies.

8. Regulatory Compliance:

- Explore the regulatory requirements and standards that billing systems must adhere to.
- Discuss the implications of compliance on system design and implementation.

9. User Experience and Interface Design:

- Examine the importance of user-friendly interfaces in billing systems.
- Discuss best practices in designing interfaces that enhance user experience.

10. Integration with Other Systems:

- Investigate the integration of billing systems with other enterprise systems (e.g., CRM, ERP).
- Discuss the benefits and challenges of seamless integration.

11. Security and Data Privacy:

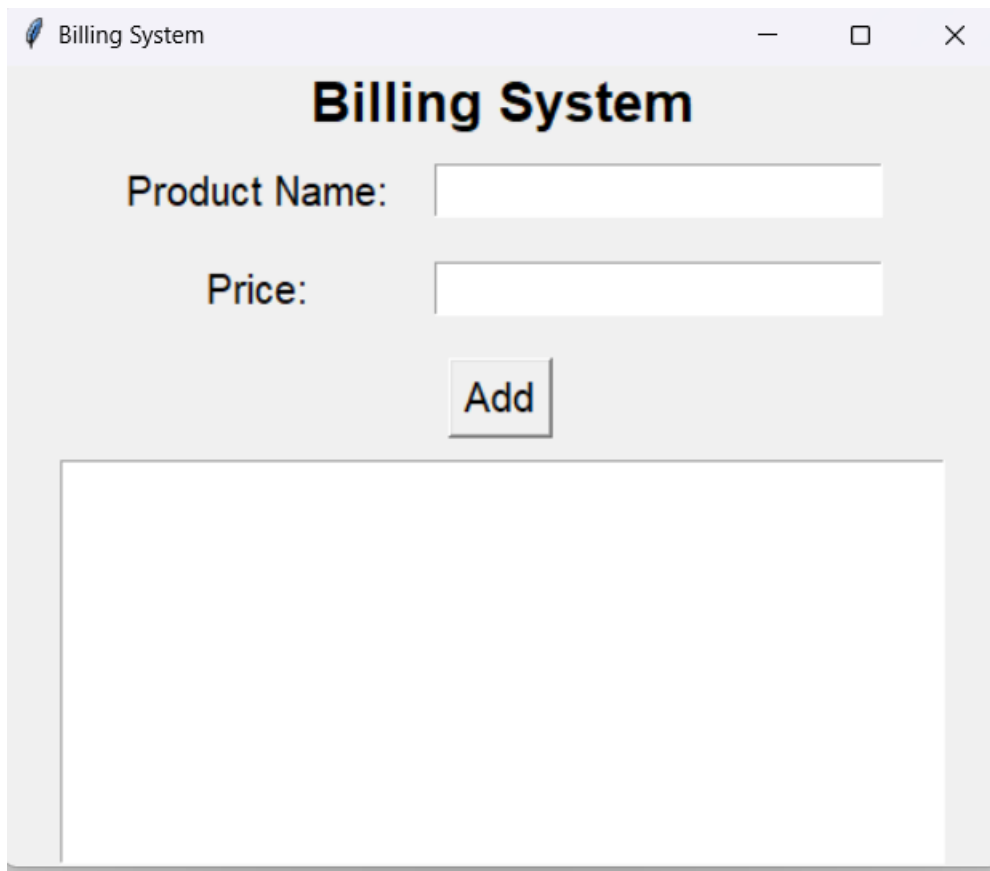
- Explore security measures implemented in billing systems to protect sensitive financial data.
- Discuss the impact of data privacy regulations on billing system design.

12. Future Directions:

- Speculate on future trends and developments in billing systems.
- Consider potential advancements in technology and their implications for billing system architecture.

Remember to critically analyze and synthesize the information gathered from various sources, providing a clear and comprehensive overview of the existing literature related to billing systems.

Output Screenshots:



The screenshot shows a window titled "Billing System" with a standard macOS-style title bar (minimize, maximize, close buttons). The window has a light gray background. At the top, the title "Billing System" is displayed in a large, bold, black font. Below the title, there are two input fields. The first is labeled "Product Name:" and the second is labeled "Price:". Both labels are in a black font. Below the "Price:" field, there is a button labeled "Add" in a black font. At the bottom of the window, there is a large, empty white rectangular area, likely intended for displaying a list of items or a receipt.

```

window.geometry("500x400")

# Create a label for the title
title_label = Label(window, text="Billing System", font="Arial 20 bold")
title_label.pack()

# Create a frame for the input fields
input_frame = Frame(window)
input_frame.pack()

# Create labels and entry boxes for the product name and price
product_label = Label(input_frame, text="Product Name:", font="Arial 15")
product_label.grid(row=0, column=0, padx=10, pady=10)
product_entry = Entry(input_frame, font="Arial 15")
product_entry.grid(row=0, column=1, padx=10, pady=10)

price_label = Label(input_frame, text="Price:", font="Arial 15")
price_label.grid(row=1, column=0, padx=10, pady=10)
price_entry = Entry(input_frame, font="Arial 15")
price_entry.grid(row=1, column=1, padx=10, pady=10)

update_total()

vansh
"C:\Users\VANSHAM PANDEY\PycharmProjects\pythonProject1\venv\Scripts\python.exe" "C:

```

Billing System

Product Name:

Price:

Add

sugar: 45.0
oil: 150.0
vegetable: 230.0

Billing System

Product Name:

Price:

Add

sugar: 45.0
oil: 150.0
vegetable: 230.0

Total: 425.0

CONCLUSION:

This was my project about “Billing System”

Development of this Project takes a lot of efforts. I think this System gave a lot of satisfaction. .Through every task is never said to be perfect in this development field even more improvement may be possible in this System.

I learnt so many things and gained a lot of knowledge about development field.

I hope this will prove fruitful.

FUTURE SCOPE:

The future scope of billing systems is likely to be influenced by ongoing technological advancements, changing business landscapes, and evolving customer expectations. Here are some potential areas of growth and development in the future of billing systems:

Automation and AI Integration:

Increased automation of billing processes using artificial intelligence (AI) and machine learning.

AI-driven predictive analytics for billing cycles, fraud detection, and personalized billing strategies.

Blockchain Technology:

Implementation of blockchain for secure and transparent transactions, reducing fraud and ensuring data integrity.

Smart contracts for automated billing based on predefined conditions.

Cloud-Based Billing Systems:

Growing adoption of cloud-based billing systems for scalability, flexibility, and accessibility.

Integration with other cloud services for seamless data exchange.

Subscription and Usage-Based Billing:

Expansion of subscription-based and usage-based billing models across various industries.

Dynamic pricing strategies based on real-time usage and customer behavior.

Mobile and Contactless Payments:

Integration with mobile payment technologies, including contactless payments and mobile wallets.

Enhanced mobile billing interfaces for a seamless user experience.

Enhanced Security Measures:

Continuous improvement in cybersecurity measures to protect sensitive financial data.

Implementation of biometric authentication and advanced encryption techniques.

Personalization and Customer Experience:

Customized billing experiences tailored to individual customer preferences.

Interactive and user-friendly interfaces to improve overall customer experience.

Regulatory Compliance and Data Privacy:

Continued focus on meeting evolving regulatory requirements and data privacy standards.

Implementation of features to ensure compliance with changing legal frameworks.

Integration with Emerging Technologies:

Integration with emerging technologies such as 5G, Internet of Things (IoT), and edge computing.

Utilization of data from IoT devices for more accurate billing and real-time insights.

Sustainability and Green Billing:

Development of environmentally friendly billing practices to support sustainability goals.

Implementation of eco-friendly billing processes and reduced paper usage.

REFERENCES

Bibliography

- Wikipedia
- Stack Overflow
- Geek for Geeks

Websites

- <https://www.bing.com/ck/a?!&&p=0c22a8d4e0313bd6JmltdHM9MTcwMjI1MjgwMCZpZ3VpZD0yYTMwYTM3Mi00YWFiLTU5ZjQtMDRjYi1iMzg3NGI3NTY4M2ImaW5zaWQ9NTIyNQ&p=3&ver=2&hsh=3&fclid=2a30a372-4aae-69f4-04cbb3874b75683b&psq=mainloop+python&u=a1aHR0cHM6Ly9zdGFja292ZXJmbG93LmNvbS9xdWVzdGlbnMvMjkxNTgyMjAvdGtpbnRlci11bmRlcnN0YW5kaW5nLW1haW5sb29wLw&ntb=1>
- <https://www.bing.com/ck/a?!&&p=d92120426171351aJmltdHM9MTcwMjI1MjgwMCZpZ3VpZD0yYTMwYTM3Mi00YWFiLTU5ZjQtMDRjYi1iMzg3NGI3NTY4M2ImaW5zaWQ9NTI1Mw&p=3&ver=2&hsh=3&fclid=2a30a372-4aae-69f4-04cbb3874b75683b&psq=mainloop+python&u=a1aHR0cHM6Ly93d3cuZ2Vla3Nm3JnZWVrcy5vcmevcHl0aG9uLWd1aS10a2ludGVyLw&ntb=1>
- <https://www.bing.com/ck/a?!&&p=01ec9a1f6cd267f9JmltdHM9MTcwMjI1MjgwMCZpZ3VpZD0yYTMwYTM3Mi00YWFiLTU5ZjQtMDRjYi1iMzg3NGI3NTY4M2ImaW5zaWQ9NTIzMQ&p=3&ver=2&hsh=3&fclid=2a30a372-4aae-69f4-04cbb3874b75683b&psq=wikipedia&u=a1aHR0cHM6Ly93d3cuZ2lraXBIZGlhLm9yZy8&ntb=1>

APPENDIX:

```
# Import tkinter module for GUI
from tkinter import *

# Create a window object

window = Tk()

# Set the window title and size

window.title("Billing System")
window.geometry("500x400")

# Create a label for the title

title_label = Label(window, text="Billing System", font="Arial 20 bold")
title_label.pack()

# Create a frame for the input fields

input_frame = Frame(window)
input_frame.pack()

# Create labels and entry boxes for the product name and price

product_label = Label(input_frame, text="Product Name:", font="Arial 15")
product_label.grid(row=0, column=0, padx=10, pady=10)
product_entry = Entry(input_frame, font="Arial 15")
product_entry.grid(row=0, column=1, padx=10, pady=10)

price_label = Label(input_frame, text="Price:", font="Arial 15")
price_label.grid(row=1, column=0, padx=10, pady=10)
price_entry = Entry(input_frame, font="Arial 15")
price_entry.grid(row=1, column=1, padx=10, pady=10)

# Create a button to add the product to the bill
add_button = Button(input_frame, text="Add", font="Arial 15", command=lambda:
add_product())
add_button.grid(row=2, column=0, columnspan=2, padx=10, pady=10)

# Create a listbox to display the bill items

bill_list = Listbox(window, font="Arial 15", width=40, height=10)
bill_list.pack()
```

```

# Create a label to display the total amount

total_label = Label(window, text="Total: 0", font="Arial 20 bold")
total_label.pack()


# Define a function to add a product to the bill
def add_product():

    # Get the product name and price from the entry boxes
    product = product_entry.get()
    price = price_entry.get()

    # Check if the product name and price are not empty
    if product and price:

        # Try to convert the price to a float value
        try:
            price = float(price)
        except ValueError:
            # If the price is not a valid number, show an error message
            bill_list.insert(END, "Invalid price")
            return

        # Format the product and price as a string
        item = f"{product}: {price}"

        # Add the item to the bill list
        bill_list.insert(END, item)

        # Clear the entry boxes
        product_entry.delete(0, END)
        price_entry.delete(0, END)

        # Update the total amount
        update_total()

    else:
        # If the product name or price is empty, show an error message
        bill_list.insert(END, "Please enter product name and price")

# Define a function to update the total amount
def update_total():

```

```

# Initialize the total amount to zero
total = 0

# Loop through the bill items
for i in range(bill_list.size()):
    # Get the item from the bill list
    item = bill_list.get(i)

    # Split the item by colon
    parts = item.split(":")

    # Check if the item has two parts
    if len(parts) == 2:
        # Try to convert the second part to a float value
        try:

            price = float(parts[1])
            except ValueError:
                # If the price is not a valid number, skip this item
                continue

        # Add the price to the total amount
        total += price

# Format the total amount as a string
total = f"Total: {total}"

# Update the total label
total_label.config(text=total)

# Start the main loop of the window
window.mainloop()

```

