**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| Chapter No. | Description | Page No. |
| 1. | Introduction |  |
| 2. | Objective of the project |  |
| 3. | Concepts Used |  |
| 4. | System Development Life Cycle (SDLC) |  |
| 5. | Phases of SDLC |  |
| 6. | Source Code |  |
| 7. | Output |  |
| 8. | Testing |  |
| 9. | Bibliography |  |

**Chapter 1 – Introduction**

This billing software, developed using **Python** and **MySQL**, provides a streamlined solution for managing billing and inventory operations. The code is divided into two main components: bill.py for generating bills and entry.py for data entry and inventory management. Together, these scripts offer a complete system for handling customer purchases, tracking inventory, and maintaining records efficiently.

* 1. **bill.py: Billing and Invoice Generation**

This script is responsible for generating invoices based on customer purchases. It interacts with the MySQL database to retrieve item details, calculate the total cost, and update stock quantities.

**Key Features:**

* **Database Connection**: Establishes a secure connection to the MySQL database (IPPROJECT) to access stock information.
* **Invoice Creation**: Collects customer details (name, mobile number) and retrieves product information using a barcode.
* **Billing Logic**: Calculates total cost, including an 18% tax on the selling price (SRP), and displays the final invoice.
* **Stock Update**: Updates the stock quantity in the database after a successful transaction.
* **Data Export**: Exports billing details to a CSV file for record-keeping.
* **User Interaction**: Allows users to generate another bill or exit the program seamlessly.
  1. **entry.py: Inventory Data Entry and Management**

This script facilitates adding new items to the inventory or updating existing stock levels.

**Key Features:**

* **Database Connection**: Connects to the MySQL database to fetch existing stock data.
* **Barcode Verification**: Checks if an entered barcode already exists in the stock.
* **Item Addition**: Inserts new items into the STOCK table if the barcode is not found.
* **Stock Update**: Updates the quantity of existing items when the barcode matches an existing entry.
* **Error Handling**: Manages database and input errors gracefully, ensuring data integrity.
* **User Interaction**: Allows users to add/update another entry or exit the program.

**Technology Stack**

* **Python**: Used for the core logic, input handling, and interaction with the MySQL database.
* **MySQL**: Manages inventory data securely and efficiently.

**Benefits of the Codebase**

* **Efficiency**: Automates billing and inventory updates, reducing manual errors.
* **Data Integrity**: Ensures stock quantities are accurate by updating the database in real-time.
* **User-Friendly Interface**: Simple prompts guide users through billing and data entry processes.
* **Scalability**: Easily adaptable for businesses with varying inventory sizes.

**Conclusion**

The combination of bill.py and entry.py creates a powerful and efficient billing system that can be customized and expanded as needed. Whether you’re generating invoices or managing stock, this system ensures accuracy, security, and ease of use, making it an essential tool for any business.

**Chapter 2 – Objectives of the project**

The primary objective of this project is to **develop a robust, efficient, and user-friendly billing and inventory management system** using **Python** and **MySQL** that automates key business processes. This includes generating accurate invoices, managing stock levels, and maintaining secure data records to enhance operational efficiency and minimizing manual errors.

**Specific Objectives:**

1. **Automate Billing Process**:  
   Streamline the generation of invoices by automating tax calculations, total cost computation, and payment summaries.
2. **Inventory Management**:  
   Enable real-time updates to stock levels during sales and new product entries, ensuring accurate inventory tracking.
3. **Data Security and Integrity**:  
   Leverage MySQL for secure, structured storage of customer, product, and transaction data, ensuring data integrity and easy retrieval.
4. **User-Friendly Interface**:  
   Provide a simple, interactive interface that guides users through billing and data entry processes, minimizing the need for technical expertise.
5. **Data Export and Reporting**:  
   Export transaction data to CSV files for record-keeping and generate reports for better business insights and analytics.
6. **Error Handling and Validation**:  
   Implement robust input validation and error handling to prevent data inconsistencies and ensure smooth operation.

**Outcome:**

The software aims to improve the efficiency of billing and inventory processes, reduce manual effort, enhance accuracy, and provide businesses with a reliable tool to manage day-to-day operations seamlessly.

**Chapter 3 – Concepts Used**

1. **Python:**

**Python** is a high-level, interpreted, and general-purpose programming language known for its simplicity, readability, and versatility. Created by **Guido van Rossum** and first released in **1991**, Python has become one of the most popular languages worldwide due to its ease of use and vast range of applications, from web development to data science.

**Key Features of Python**

1. **Simple and Readable Syntax**  
   Python’s syntax is clean and easy to understand, making it an excellent choice for beginners and professionals alike. Its focus on readability allows developers to write clear, concise code.
2. **Interpreted Language**  
   Python code is executed line-by-line by an interpreter, which makes debugging easier and allows rapid development.
3. **Dynamically Typed**  
   Variables in Python are dynamically typed, meaning you don’t need to declare their type explicitly. This flexibility allows faster coding but requires careful error handling.
4. **Extensive Standard Library**  
   Python comes with a vast standard library that includes modules for various tasks such as file I/O, regular expressions, web development, and more.
5. **Cross-Platform Compatibility**  
   Python is platform-independent, meaning code written on one operating system (like Windows) can run on another (like Linux or macOS) without modification.
6. **Open Source and Community Support**  
   Python is open-source and supported by a large, active community. This ensures continuous development and access to countless resources, libraries, and frameworks.

**Applications of Python**

1. **Web Development**  
   Frameworks like **Django** and **Flask** make it easy to build robust web applications quickly.
2. **Data Science and Machine Learning**  
   Python is a favorite in data science, with libraries like **Pandas**, **NumPy**, **Matplotlib**, and **Scikit-learn** providing powerful tools for data analysis and machine learning.
3. **Automation and Scripting**  
   Python’s simplicity makes it ideal for automating repetitive tasks, such as file manipulation, web scraping, and system administration.
4. **Game Development**  
   Libraries like **Pygame** enable developers to create simple games with ease.
5. **Scientific Computing**  
   Python is used in scientific research for simulations and complex computations, with tools like **SciPy** and **SymPy**.
6. **Artificial Intelligence (AI) and Deep Learning**  
   Frameworks like **TensorFlow** and **PyTorch** enable the development of AI and deep learning applications.

**Advantages of Python**

* **Ease of Learning and Use**: Ideal for beginners due to its intuitive syntax.
* **Versatility**: Supports multiple programming paradigms, including procedural, object-oriented, and functional programming.
* **Large Ecosystem**: Extensive libraries and frameworks for nearly every field.

**Conclusion**

Python’s flexibility, simplicity, and powerful ecosystem make it an ideal language for a wide variety of programming tasks. Whether you're a beginner starting your programming journey or a seasoned developer tackling complex projects, Python provides the tools and resources needed to succeed in today’s technology landscape.

1. **MySQL:**

**MySQL** is an open-source relational database management system (RDBMS) widely used for managing and organizing large amounts of data. It is known for its reliability, scalability, and ease of use, making it a popular choice for web applications, data-driven software, and enterprise solutions. Developed by **Oracle Corporation**, MySQL operates on the Structured Query Language (**SQL**) to perform operations such as querying, updating, and managing databases.

**Key Features of MySQL:**

1. **Relational Database**:  
   MySQL organizes data into tables, where relationships between data are maintained using keys (primary and foreign keys).
2. **Scalability**:  
   It supports small applications with minimal data as well as large-scale applications managing terabytes of information.
3. **High Performance**:  
   MySQL is optimized for fast query execution, supporting high-performance read and write operations for demanding applications.
4. **Cross-Platform Compatibility**:  
   MySQL runs on various operating systems, including Windows, Linux, and macOS, making it versatile for different environments.
5. **Data Security**:  
   MySQL ensures data security with robust user authentication, access control, and encryption features.
6. **Replication and Clustering**:  
   Supports replication for creating copies of data across multiple servers and clustering for high availability.
7. **Open Source and Community Support**:  
   As an open-source project, MySQL is free to use, with a large community providing extensive support and continuous improvements.

**Common Uses of MySQL:**

* **Web Applications**: Used by platforms like WordPress, Facebook, and Twitter to store and manage user data, content, and transactions.
* **E-commerce**: Tracks products, customers, orders, and payments.
* **Data Warehousing**: Stores large datasets for analysis and reporting.
* **Content Management Systems (CMS)**: Powers CMS platforms like Joomla and Drupal.

**Basic Components of MySQL:**

1. **Database**: A collection of tables storing related data.
2. **Table**: A structured format that holds data in rows and columns.
3. **Query**: SQL commands used to interact with the database.
4. **Stored Procedures and Triggers**: Scripts that automate database operations and respond to specific events.
5. **Indexes**: Structures that improve the speed of data retrieval.

**Advantages of MySQL:**

* Free and open-source with commercial licensing options.
* Fast and reliable, even under heavy load.
* Comprehensive security features to protect sensitive data.
* Strong integration capabilities with programming languages like Python, PHP, and Java.

**Conclusion:**

MySQL is a powerful and versatile database management system suitable for applications ranging from simple websites to complex enterprise solutions. Its combination of speed, security, and flexibility makes it a go-to choice for developers and businesses seeking efficient data management solutions.

1. **CSV:**

**CSV (Comma-Separated Values)** is a widely-used file format for storing and exchanging tabular data in a simple, text-based structure. Each line in a CSV file represents a row of data, and each data field is separated by a comma (or another delimiter such as a semicolon or tab).

**Key Characteristics of CSV Files:**

1. **Simple Structure**:  
   CSV files store data in plain text format, making them lightweight and easy to read.
2. **Tabular Format**:  
   Data is organized into rows and columns, similar to a spreadsheet, with each line representing a row and commas separating column values.
3. **Platform Independence**:  
   CSV files can be opened and edited in a variety of software, including spreadsheet applications (like Microsoft Excel, Google Sheets) and text editors.
4. **No Complex Formatting**:  
   Unlike formats like Excel or database files, CSV does not support complex features such as formulas, formatting, or embedded media, making it ideal for simple data storage and transfer.

**Benefits of CSV in the Billing Software:**

1. **Data Portability**:  
   CSV files make it easy to transfer billing records to other systems or share them across different platforms.
2. **Compatibility**:  
   They can be opened and processed by numerous applications, ensuring compatibility with various data analysis tools.
3. **Ease of Integration**:  
   CSV files can be easily imported into databases, making them ideal for backup and migration purposes.
4. **Human-Readable**:  
   Since CSV files are plain text, they are easy to understand and modify manually if necessary.

**Use in the Billing Software:**

In the **Billing Software**, CSV files are used to:

1. **Export Billing Data**:  
   Every generated bill is saved in a CSV file with details such as the customer's name, contact information, items purchased, quantity, price, tax, and total amount.
2. **Maintain Records**:  
   CSV files act as a backup for all transactions, providing a simple way to keep historical records of sales.
3. **Facilitate Reporting**:  
   The exported CSV files can be imported into tools like Excel or data analysis platforms for further processing and report generation.

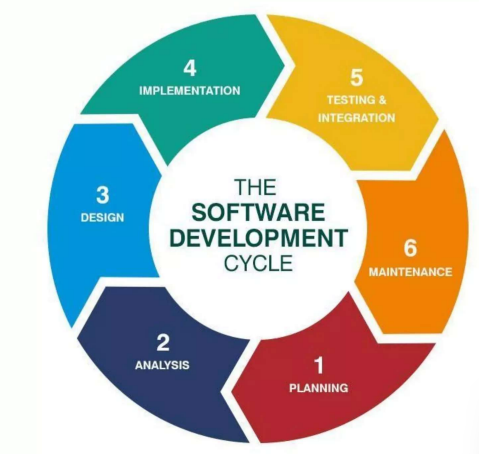
**Conclusion:**

CSV is a powerful and versatile format for data storage and transfer, making it an ideal choice for the **Billing Software** to ensure data portability, simplicity, and broad compatibility.

**Chapter 4 – System Development Life Cycle**

The system development life cycle is a project management technique that divides complex projects into smaller, more easily managed segments or phases. Segmenting projects allow managers to verify the successful completion of project phases before allocating resources to subsequent phases.

Software development projects typically include initiation, planning, design, development, testing, implementation, and maintenance phases. However, the phase may be divided differently depending on the organization involved.



**Chapter 5 – Phases of SDLC**

**5.1 Initiation Phase**

The initiation phase is the beginning of the project. In this phase, the idea for the project is exploredand elaborated. The goal of this phase is to examine the feasibility of the project. In addition, decisions are made concerning who is to carry out the project, which party will be involved and whether the project has an adequate base of support among those who are involved.

In this phase, the current prospective project leader writes a proposal, which contains a description of the above-mentioned matters. Examples of this type of project include business plans and grant applications. The prospective sponsors of the project evaluate the proposal and, upon approval, provide the necessary financing. The project finally begins at the time of approval.

**5.2 System Concept Development Phase**

The Concept Development Phase may begin after the approval of the Concept Proposal and Project Charter, the completion of the Initiation project status review, and the approval to proceed to the Concept Development Phase.

The focus of the phase is two-fold:

* Evaluate the feasibility of alternatives
* Clearly define and approve project scope, including the system, all deliverables, and all required activities.

The Concept Development Phase activities are inputs into the development of ITPR, which is a required output of this phase.

**5.3 Planning Phase**

The planning stage (also called feasibility stage) is the phase in which developers will plan for the upcoming project. It helps to define the problem and scope of any existing systems, as well as determine the objectives for their new systems. By developing an effective outline for the upcoming development cycle, they’ll theoretically catch problems before they affect development. And help to secure the funding and resources they need to make their plan happen.

Perhaps most importantly, the planning stage sets the project schedule, which can be of key importance if development is for a commercial product that must be sent to market at certain times.

**5.4 Analysis Phase**

The analysis stage includes gathering all the specific details required for a new system as well as determining the first ideas for prototypes.

Developers may:

* Define any prototype system requirements
* Evaluating alternatives to existing prototypes
* Perform research and analysis to determine the needs of end-user

Furthermore, developers will often create software requirement specification or SRS documents.

This includes all the specifications for the software, hardware and network requirements for the system they plan to build. This will prevent them from overdrawing funding or resources when working at the same place as other development teams.

**5.5 Design Phase**

The design stage is a necessary precursor to the main developer stage. Developers will first outline the details for the overall application, alongside specific aspects, such as its:

* User interface
* System interface
* Network and network requirements
* Databases

They’ll typically turn the SRS document they created into a more logical structure that can later be implemented in a programing language. Operation, training, and maintenance plans will all be drawn up so that developers know what they need to do throughout every stage of the cycle moving forward.

Once complete, development managers will prepare a design document to be referenced throughout the next phase of the SDLC.

**5.6 Development Phase**

The development stage is the part where developers write code and build the application according to the earlier design documents and outlined specifications.

This is where Static Application Security Testing or SAST tools come into play.

Product program code is built per the design document specifications. In theory, all the prior planning and outlined should make the actual development phase relatively straightforward.

Developers will follow any coding guidelines as defined by the organization and utilize different tools such as compilers, debuggers, and interpreters.

Programing languages can include staples such as Python, C++, PHP, and more. Developers will choose the right programing code to use based on the project specifications and requirements.

**5.7 Testing Phase**

The software must be tested to make sure that there aren’t any bugs and that the end-user experience will not negatively be affected at any point. During the testing stage, developers will go over their software with a fine-tooth comb, noting any bugs or defects that need to be tracked, fixed and later retested.

**5.8 Implementation and Integration Phase**

After testing, the overall design for the software will come together. Different modules or designs will be integrated into the primary source code through developer efforts, usually by leveraging training environments to detect further errors or defects.

The information system will be integrated into its environment and eventually installed. After passing this stage, the software is theoretically ready for market and may be provided to any end-users.

**5.9 Maintenance Phase**

The SDLC doesn’t end when software reaches the market. Developers must now move into a maintenance mode and begin practicing any activities required to handle issues reported by end-users.

Furthermore, developers are responsible for implementing any changes that the software might need after deployment.

This can include handling residual bugs that were not able to be patched before launch or resolving new issues that crop up due to user reports.

A larger system may require longer maintenance stages compared to smaller systems.

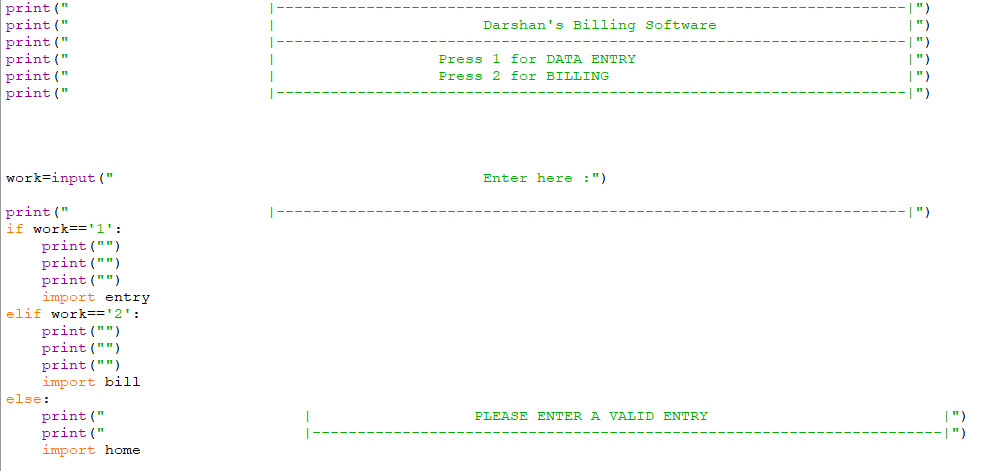
**5.10 Disposition Phase**

The Disposition Phase is the end of an information system’s life cycle. The information system is formally retired according to organizational needs, laws and regulations, and the Disposition Plan. The disposition activities ensure that the information about the system is preserved according to applicable records management regulations and policies for future access.

The decision to proceed with the Disposition Phase is based on recommendations and approvals from an In-process review during the Operations and Maintenance Phase.

**Chapter 6 – Source Code**

**6.1 home.py**

****

**6.2 entry.py (If entered 1)**

**A screen shot of a computer

Description automatically generated**

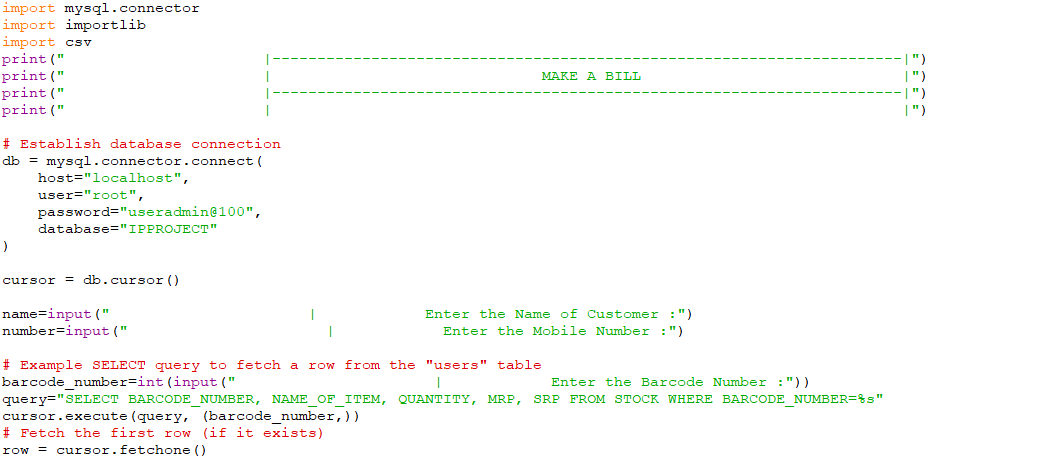
**A screenshot of a computer code

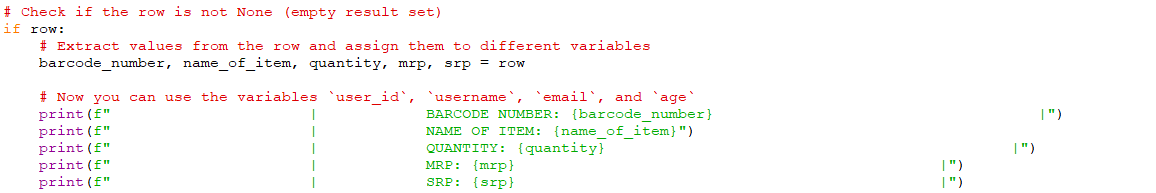
Description automatically generated**

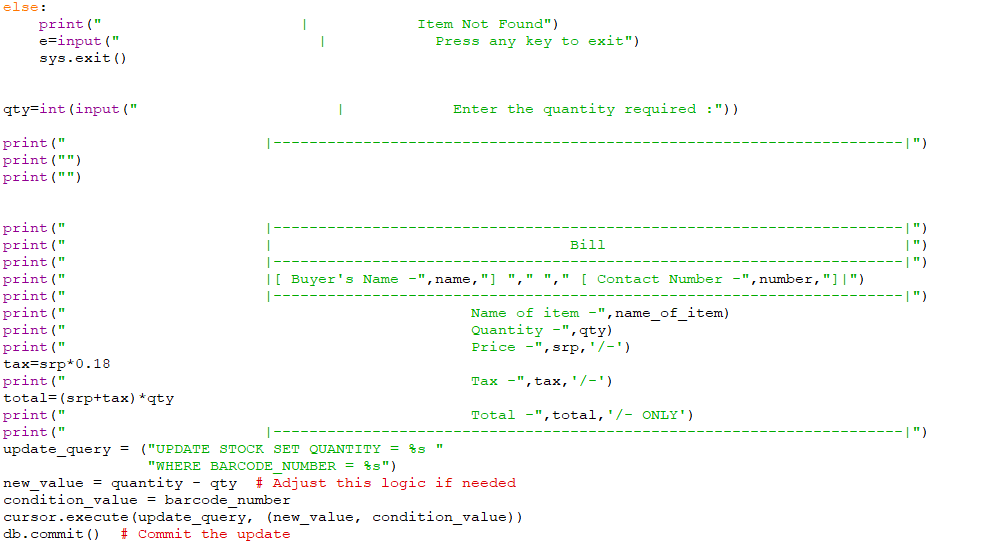
**A close-up of a document

Description automatically generated**

**6.3 bill.py (If entered 2)**

****

****

****

**A screenshot of a computer program

Description automatically generated**

**6.4 MySQL Table**

****

**Chapter 7 – Output**

**7.1 home.py**

**A screen shot of a computer

Description automatically generated**

**7.2 If entered 1 (entry.py)**

**7.2.1 If already existing barcode entered.**

**A screen shot of a computer

Description automatically generated**

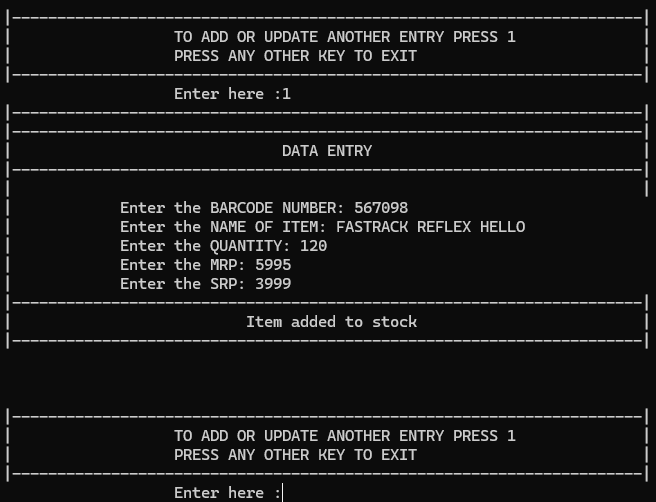
**Entry before:**

****

**Entry after:**

****

**7.2.2 If a new non-existing barcode entered**

****

**Table before:**

**A screen shot of a computer

Description automatically generated**

**Table after:**

**A screen shot of a computer

Description automatically generated**

**7.3 If entered 2 (bill.py)**

**7.3.1 Entries of customer**

**A screenshot of a computer screen

Description automatically generated**

**7.3.2 Generated Bill**

**A screenshot of a computer

Description automatically generated**

**7.3.3 CSV Entry of customer**

**A screenshot of a computer

Description automatically generated**

**7.3.4 If the barcode entered doesn’t exist**

**A screen shot of a black screen

Description automatically generated**