**Project Document**

**Group Name:** SciPy  
**Project Title:** SciPy Bills & Inventory Management

**Team Members:**

1. Atharva Date – Team Lead
2. Chitraksh Chavan – Buddy Developer
3. Jagdish Kachhawah - Buddy Developer

**1. *System Architecture***

The SciPy Billing and Inventory Management system is a desktop application developed using Python's Tkinter library. It utilizes SQLite for efficient database management. The application is structured into two main sections: Product Management (Admin) and Dashboard.

**Features List**

**Login Panel**

* Username and password entry area without logging in user can’t enter product management panel

**Product Management:**

* Add new products with fields for name, ID, price, and quantity.
* Update existing product details.
* Delete products from the database.
* Display a list of all products.

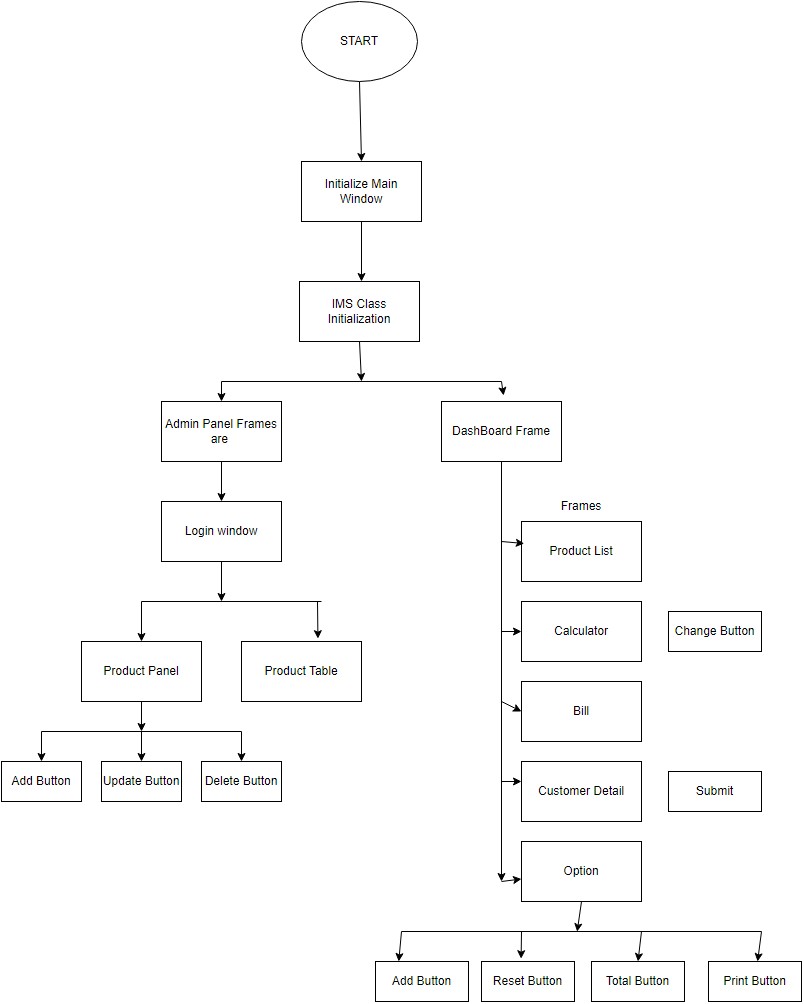
**Dashboard:**

* Enter customer details (name, phone number and address).
* Add products to the billing area using Option panel.
* Calculate the total cost.
* Display the bill with details including product name, quantity, price, and total amount.
* Reset the bill area using reset button.
* View product list to select item and add in bill area.
* Print button save the bill in PC
* Calculator to calculate change

A diagram of a system

Description automatically generated

**2. *Logic Flow Chart***



**3. *Rough Low Fidelity GUI Design***

A blue rectangular object with white text

Description automatically generatedA blue rectangular object with white text

Description automatically generatedA blue rectangular object with white text

Description automatically generated

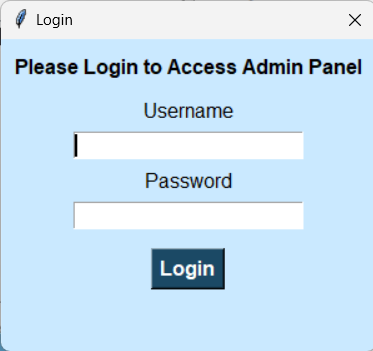
**4. *Technologies Used***

* **Python:** Primary programming language.
* **Tkinter:** GUI package for Python.
* **SQLite:** Lightweight database management system.
* **Pillow:** Python Imaging Library for image processing.

**5. *Requirement List***

* Third-Party Packages Used:
  + Pillow: Install using pip install pillow, used for handling images.

**6.** ***Final UI Screenshots with brief Explanation***

  
-Login screen for logging in admin panel so only authorised personnel can make changes

A screenshot of a computer

Description automatically generated   
The Product Management Panel allows users to manage the inventory of products.

Product Panel: - Name Field: Input for product name.

- ID Field: Input for product ID.

- Price Field: Input for product price.

- Quantity Field: Input for product quantity.

Products Table: Displays products with columns for ID, Name, Price, and Quantity.

Functionalities: - Add Product: Enter details and click "Add".

- Update Product: Select a product, modify details, and click "Update".

- Delete Product: Select a product and click "Delete".

A screenshot of a computer

Description automatically generated

The Dashboard Panel provides an overview of the product list, customer details, billing calculator, and options for managing transactions.

1) Product List: Displays the list of available products with their names and prices.

2) Calculator: - Money Received Field: Input for the amount of money received from the

customer.

- Total Cost Field: Displays the total cost of the selected products.

- Change Button: Calculates and displays the change to be returned to the customer.

3) Bill: Receipt Area Displays the generated receipt for the transaction.

4) Customer Details: - Name Field: Input for the customer's name.

- Phone Number Field: Input for the customer's phone number.

- Address: Input for the customer's address

5) Options: - Add Button: Adds the selected products to the bill.

- Total Cost Field: Displays the total cost of the products.

- Quantity Field: Input for the quantity of the selected product.

- Total Button: Calculates the total cost based on quantity.

- Print Button: Prints the receipt and save it

- Add Button: Add product item in bills

-Reset button: reset the bill

6) Functionalities: - Add Product to Bill: Select a product, enter quantity, and click "Add".

- Generate Bill: Enter customer details, select products, calculate total cost,  
 and click "Bill".

- Calculate Change: Enter money received and click "Change".

- Print Receipt: Click "Print" to print the receipt.

A screenshot of a computer

Description automatically generated Final bill display as a text file with total and balance amount.

**7. *Database Structure with Explanation***

* **Database Name:** ProductList.db
* **Table:** productlist
* **Columns:**
  + pro\_name (TEXT): Name of the product.
  + pro\_id (TEXT): Unique product identifier.
  + price (REAL): Price of the product.
  + quantity (INTEGER): Available quantity of the product.

This table stores essential product details for inventory management.

**8) *List of Functions and Classes***

**Class IMS**

This class manages the main window and handles the GUI for a comprehensive billing and inventory management system.

**\_\_init\_\_ Method**

The initializer sets up the main window, defines its properties, and initializes frames and widgets essential for the application.

* **Parameters:**
  + main\_window: The root window of the Tkinter application.
* **Key Attributes:**
  + self.main\_window: Reference to the main window.
  + self.logged\_in: Tracks user authentication status.

**show\_login\_window Function**

Displays a login window for user authentication.

* **Components:**
  + username\_entry: Entry widget for username input.
  + password\_entry: Entry widget for password input.
  + login: Inner function handling login logic. If credentials match, sets self.logged\_in to True; otherwise, displays an error message.

**show\_frame1 Function**

Displays the admin frame if the user is logged in; otherwise, prompts for login.

**clear\_field Function**

Clears the content of a given entry widget.

* **Parameters:**
  + field: The entry widget to clear.

**addProduct Function**

Adds a new product to the database.

* **Workflow:**
  + Connects to the SQLite database.
  + Checks for existing product ID.
  + Inserts new product if ID is unique.
  + Updates product list display and clears input fields.

**updateProduct Function**

Updates an existing product in the database.

* **Workflow:**
  + Connects to the SQLite database.
  + Verifies product ID existence.
  + Updates product details based on input fields, retaining old values if fields are empty.
  + Updates product list display and clears input fields.

**deleteProduct Function**

Deletes a product from the database.

* **Workflow:**
  + Connects to the SQLite database.
  + Confirms deletion with user if product ID exists.
  + Deletes product and updates product list display.
  + Clears input fields.

**show\_frame2 Function**

Displays the dashboard frame.

**loadProducts Function**

Loads all products from the database into the tree view widget within the admin frame.

**change Function**

Calculates change to be given back to the customer based on total cost and money received.

**populateListBox Function**

Populates the product list in the dashboard frame with product names and prices from the database.

**submit\_customer\_details Function**

Submits customer details and displays them in the text area.

* **Workflow:**
  + Validates all customer detail fields.
  + Displays customer details and placeholder for the invoice in the text area.

**resetDash Function**

Resets the dashboard, clearing all customer and product details.

* **Workflow:**
  + Prompts user confirmation for reset.
  + Clears and resets input fields and text area.

**addItem Function**

Adds a selected product to the invoice.

* **Workflow:**
  + Validates product selection and quantity.
  + Calculates total price for selected quantity.
  + Updates remaining quantity in database.
  + Adds product details to text area and updates total cost.

**finalBill Function**

Finalizes the bill by displaying total cost in the text area.

**GUI Setup in \_\_init\_\_**

* Sets main window title.
* Creates left menu with buttons for Admin Panel and Dashboard.
* Configures Admin Frame with input fields and buttons for product management.
* Initializes Product Table using tree view for product display.
* Sets up Dashboard Frame with product list, calculator, and options for managing invoices.

**Calculator and Options**

* **Calculator:**
  + Fields for entering total cost and received amount.
  + Button to calculate change.
* **Options:**
  + Buttons for adding items to invoice and resetting dashboard.

**9. *File Structure & List of Supporting Files & Dependencies***

* **File Structure:**
  + main.py: Main application code.
  + ProductList.db: SQLite database file.
  + README.md: Project documentation.
* **Dependencies:**
  + Python 3.x
  + Tkinter (included with Python)
  + SQLite (included with Python)
  + Pillow (install using pip install pillow)

**10) *Outcome of the Project & Contribution of Each Team Member***

**Outcome:**

The SciPy Billing and Inventory Management system delivers a robust solution for efficiently managing product inventory and generating accurate customer bills. Key features include seamless functionalities for adding, updating, and deleting products, ensuring streamlined operations and precise billing and change calculations. This project has significantly enhanced our proficiency in Python, specifically in utilizing the Tkinter GUI package and SQLite3 database management. It provided hands-on experience with implementing these technologies in a real-world application.

**Contribution of Each Team Member:**

* **Chitraksh Chavan:**
  + Led the UI design efforts using Canva and contributed extensively to the frontend design with Tkinter. Designed intuitive user interfaces that enhance usability and visual appeal also helped in designing dashboards functions.
* **Atharva Date:**
  + Served as the team lead, providing guidance and support to team members. Played a crucial role in integrating the frontend GUI with the backend database using structured tree data handling. Resolved technical issues and ensured smooth functionality across the application.
* **Jagdish Kachhawah:**
  + Focused on implementing backend functionalities, specifically in the Admin panel. Developed and connected various buttons to database operations, enabling functionalities such as adding, updating, and deleting products. Contributed to the overall functionality and reliability of the system.