

Inventory Management System Documentation

-TEAM HV

Overview

The Inventory Management System is a GUI application designed for managing inventory and generating sales invoices. It includes functionalities for logging in, managing sales, viewing and updating inventory, and additional features like theme switching and password changing.

Table of Contents

- Project Structure
- Installation and Setup
- Running the Application
- Code Walkthrough
- Process Flow
- Screenshots

1. Project Structure

The project consists of the following key components:

- **main.py**: The main script to run the application.
- **lib/**: Directory containing various modules for different functionalities.
 - **login_system.py**: Handles the login system.
 - **main_menu.py**: Contains the main menu of the application.
 - **inventory.py**: Manages inventory functionalities.
 - **sales.py**: Manages sales functionalities.
 - **settings.py**: Contains settings-related functionalities.
 - **theme_engine.py**: Handles theme switching.
- **Screenshots/**: Directory containing screenshots of the application.
- **requirements.txt**: Lists the dependencies required to run the project.

2. Installation and Setup

To set up the project on your local machine, follow these steps:

1. Clone the repository:

```
git clone https://github.com/harshikala/Inventory\_TeamHV.git
```

-
2. (Optional) Create and activate a virtual environment:

```
python -m venv venv
source venv/bin/activate # On Windows, use `venv\Scripts\activate`
```

-
-
3. Install the required dependencies:

```
pip install -r requirements.txt
```

3. Running the Application

To run the application, use the following command:

```
python main.py
```

Login Credentials:

□ Username: **admin**

□ Password: **password**

4. Commands to Run the Code

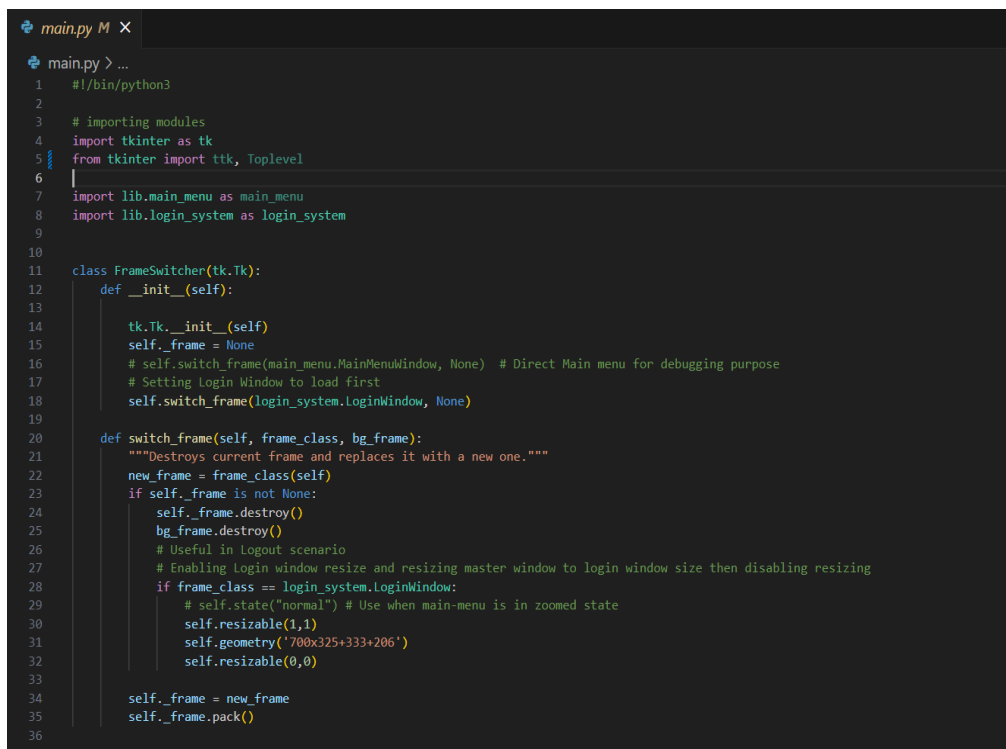
To run the code, ensure you have installed the required dependencies and then use:

```
python main.py
```

5. Code Walkthrough

main.py

The main script initializes the Tkinter application and sets up the initial login window.

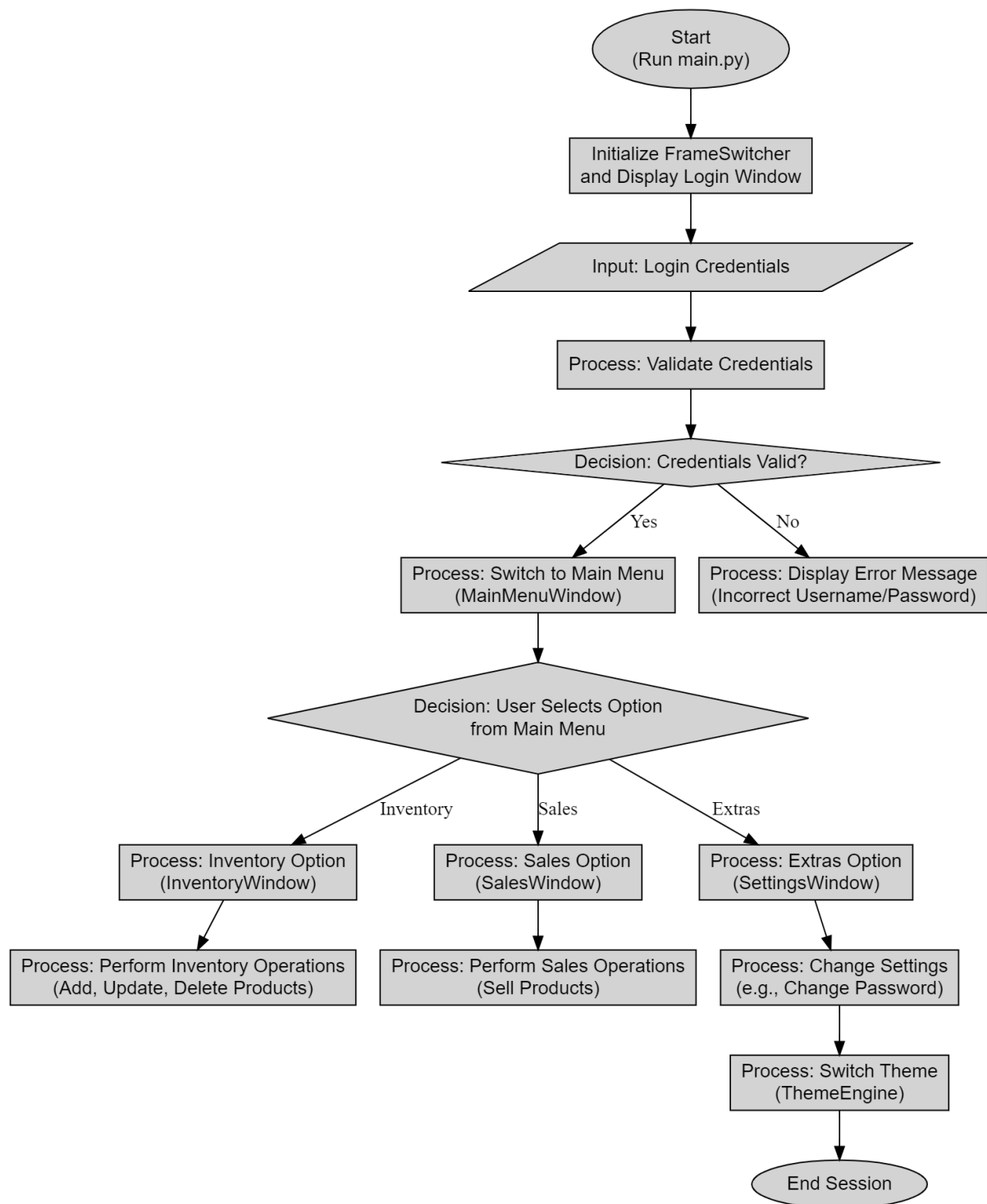


```
main.py M X
main.py > ...
1  #!/bin/python3
2
3  # importing modules
4  import tkinter as tk
5  from tkinter import ttk, Toplevel
6
7  import lib.main_menu as main_menu
8  import lib.login_system as login_system
9
10
11 class FrameSwitcher(tk.Tk):
12     def __init__(self):
13
14         tk.Tk.__init__(self)
15         self._frame = None
16         # self.switch_frame(main_menu.MainMenuWindow, None) # Direct Main menu for debugging purpose
17         # Setting Login Window to load first
18         self.switch_frame(login_system.LoginWindow, None)
19
20     def switch_frame(self, frame_class, bg_frame):
21         """Destroys current frame and replaces it with a new one."""
22         new_frame = frame_class(self)
23         if self._frame is not None:
24             self._frame.destroy()
25             bg_frame.destroy()
26             # Useful in Logout scenario
27             # Enabling Login window resize and resizing master window to login window size then disabling resizing
28             if frame_class == login_system.LoginWindow:
29                 # self.state("normal") # Use when main-menu is in zoomed state
30                 self.resizable(1,1)
31                 self.geometry('700x325+333+206')
32                 self.resizable(0,0)
33
34         self._frame = new_frame
35         self._frame.pack()
36
37
```

□ **FrameSwitcher class:** Manages switching between different frames (login, main menu, etc.).

□ **switch_frame method:** Destroys the current frame and replaces it with a new one.


6. Process Flow




7. Screenshots

Screenshots of different parts of the application can be found in the Screenshots directory.

Login

 Inventory Management System/ Login

USER LOGIN



Username

admin

Password

LOGIN

Main Menu


Inventory Management System

Logout


MAIN MENU

Mon,29/Jul/24
12:38 AM


INVENTORY MANAGEMENT SYSTEM




SALES



INVENTORY



EXTRAS



EXIT

Sales

BACK

SALES

Mon,29/Jul/24
12:38 AM

Customer Details

Customer Name

Contact No.

Options

Clear All

BILL RECORDS

Product Selection

Category

Sub-Category

Product

Quantity

Stock

Unit Price

Total Price

Add to Cart

CART

Product Name	Quantity	Unit Price	Total Price
--------------	----------	------------	-------------

Remove Selected

Generate Bill

Inventory

BACK

INVENTORY

Mon,29/Jul/24
03:18 AM

Product List

No	Product Name	Category	Sub-Category	Quantity	Unit Price	Total Price
1	Fair & Lovely Men, 50g	Cosmetics	Face	993	₹ 129.0	₹ 128097.0
2	Fair & Lovely Women, 50g	Cosmetics	Face	150	₹ 97.0	₹ 14550.0
3	Himalaya Lip Balm, 10gm	Cosmetics	Lips	599	₹ 33.0	₹ 19767.0
4	Nivea Lip Balm, 4.8g	Cosmetics	Lips	127	₹ 105.0	₹ 13335.0
5	Set Wet Hair Gel Cool, 50ml	Cosmetics	Hair	713	₹ 55.5	₹ 39571.5
6	Vaseline, 100ml	Cosmetics	Skin	1485	₹ 200.0	₹ 297000.0
7	Nivea Skin Lotion, 400ml	Cosmetics	Skin	441	₹ 400.0	₹ 176400.0
8	Lakme cream	Cosmetics	Skin	20	₹ 200.0	₹ 4000.0

Product Name

ADD PRODUCT

Menu

Product Name

Quantity

Price(₹)

Select Product

View All Product

Update Product

Delete Product

Extras

BACK

EXTRAS

Mon,29/Jul/24
03:19 AM


CHANGE PASSWORD

CHANGE THEME

Current Password

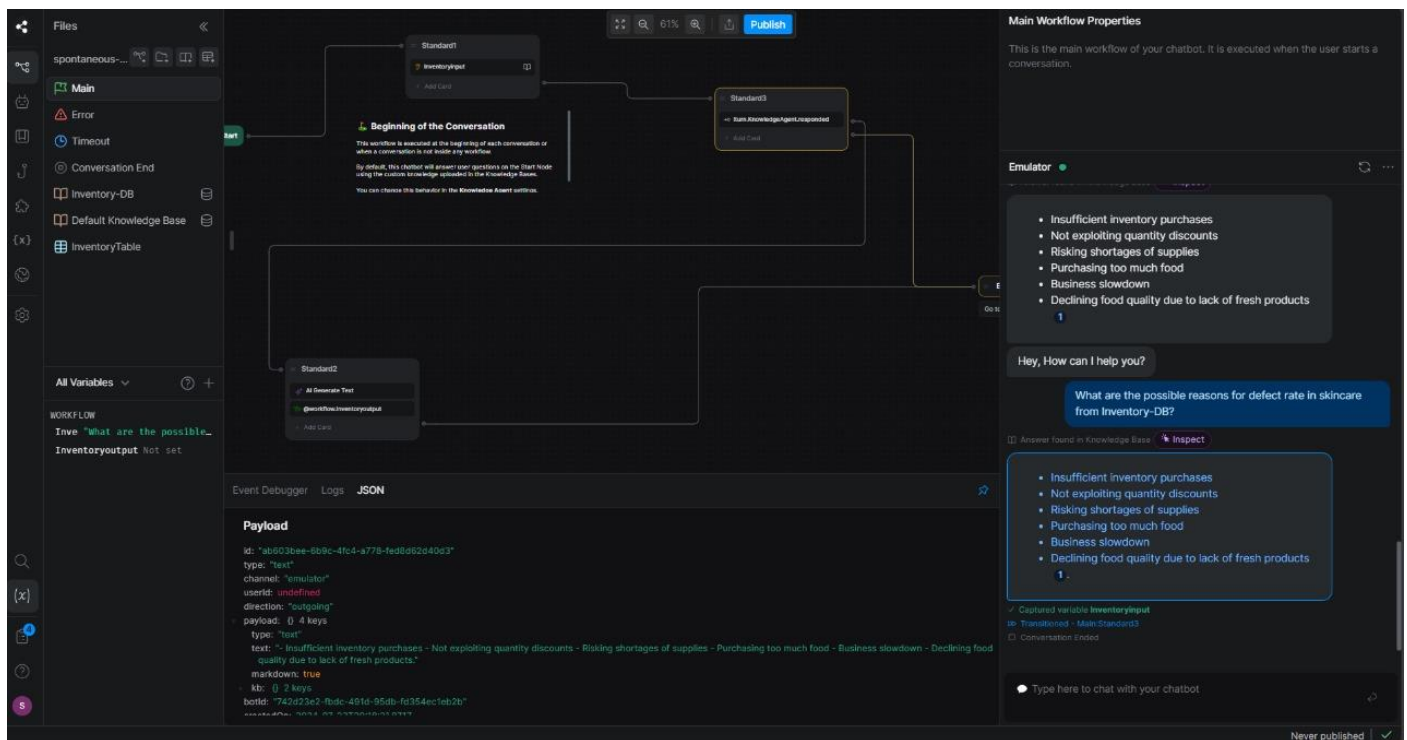
New Password

Change Password



LIGHT MODE

AI ChatBot:



8. Code Overview of lib Folder

login_system.py

This module handles the login functionality of the application. It includes the LoginWindow class, which presents the login interface to the user. The module verifies the entered credentials and grants access to the main menu upon successful login.

main_menu.py

This module contains the main menu interface of the application. The MainMenuWindow class provides options to navigate to different sections such as Sales, Inventory, and Extras. It serves as the central hub from which users can access other functionalities.

inventory.py

The inventory.py module manages all inventory-related operations. It includes functionalities to add new products, update existing product details, and delete products. The module interfaces with the database to ensure that inventory data is accurately maintained.

sales.py

This module handles the sales aspect of the application. The SalesWindow class allows users to manage sales transactions, enter product quantities, and generate sales invoices. It also ensures that sales data is correctly recorded and updated in real-time.

settings.py

The settings.py module provides various settings-related functionalities. This includes options to change the application's password and other configurable settings. It ensures that users can personalize their application experience.

theme_engine.py

The theme_engine.py module manages the theme switching functionality of the application. It allows users to switch between different visual themes (e.g., light mode and dark mode) to enhance the user experience based on their preferences.

9. Summary

This documentation covers the installation, setup, and running of the Inventory Management System application. It includes detailed overviews of the main script and each module in the lib folder, providing a clear understanding of the project's structure and functionality. By following this guide, users can efficiently navigate and utilize the application for inventory and sales management.