CAPSTONE PROJECT REPORT

(Project Term July to december 2023)

Submitted by

Registration Number : 12217293

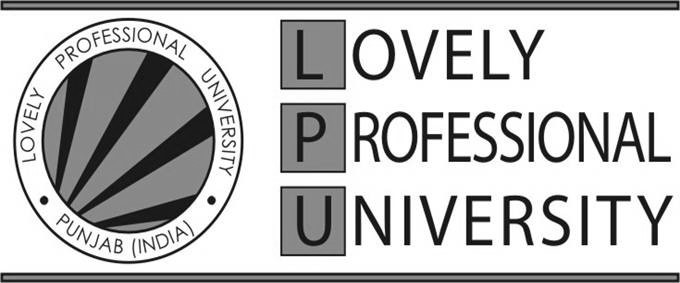
Course Code : INT 216

Under the Guidance of

##### Ved Prakash Chaubey : 63892

(Name of faculty mentor with designation)

School of Computer Science and Engineering



1

DECLARATION

We hereby declare that the project work entitled (GROCERY STORE AND SALES MANAGEMENT SYSTEM”) is an authentic record of our own work carried out as requirements of Capstone Project for the award of B.Tech degree in CSE Data Science(DATASCIENCE and Ml) .(Programme Name) from Lovely Professional University, Phagwara, under the guidance of (Dr.VED PRAKASH), during july to November 2023. All the information furnished in this capstone project report is based on our own intensive work and is genuine.

Name of Student : pallapatipavankumar

Registration Number: 12217293

(Signature of Student 1) Date:

ii

CERTIFICATE

This is to certify that the declaration statement made by this group of students is correct to the best of my knowledge and belief. They have completed this Capstone Project under my guidance and supervision. The present work is the result of their original investigation, effort and study. No part of the work has ever been submitted for any other degree at any University. The Capstone Project is fit for the submission and partial fulfillment of the conditions for the award of B.Tech degree in CSE Data Science(DATASCIENCE And ML) (Programme Name) from Lovely Professional University, Phagwara.

Signature and Name of the Mentor

Designation

School of Computer Science and Engineering, Lovely Professional University, Phagwara, Punjab.

Date :

iii

### TABLE OF CONTENTS

Inner first page……………………………………………………………………….(i)

Declaration…...……………………………………………………………………..(ii)

Certificate……..…….………………………………………………………………(iii)

Acknowledgement…………………...………………………………………….......(v)

Table of Contents……………………...……………………………………………(vi)

[1.ABSTRACT 1](#_Toc10441)

[2. INTRODUCTION 1](#_Toc10442)

[3. OBJECTIVE OF PROJECT 2](#_Toc10443)

[4.DESCRIPTION OF THE PROJECT 3](#_Toc10444)

[5.SOURCE CODE 4](#_Toc10445)

6.INPUT/OUTPUT 18

7.SCOPE OF PROJECT 27

8.FUTURE IMPROVEMENTS 27

9.CONCLUSION 27

iv

# 1.ABSTRACT

The project is a grocery store inventory and sales management system with a graphical user interface. It is implemented using the tkinter and pandas libraries in Python. The system allows users to add, update, and delete products from the inventory, sell products and update sales data, generate reports, and display inventory and sales data. The GUI includes labels and entry fields for entering product details and buttons for performing various tasks. The project aims to provide a functional solution for managing inventory and sales data in a grocery store, although additional features and error handling may be required depending on specific requirements.

# INTRODUCTION

My project is a GUI-based grocery store inventory and sales management system that allows store owners to manage their inventory and sales data easily. The application is built using the Python programming language and the tkinter and pandas libraries. It provides a user-friendly interface with labels and entry fields for adding, updating, and deleting product details such as name, price, and quantity.

The system also allows store owners to sell products and update sales data, as well as generate reports for inventory and sales. The reports can be saved to CSV files for future reference. The application aims to simplify the inventory and sales management process for grocery store owners, providing them with an efficient and effective tool to keep track of their store's operations.

# OBJECTIVE OF PROJECT

Create a user-friendly graphical interface for managing a grocery store inventory and sales data using the tkinter library.

Use the pandas library to handle and manipulate inventory and sales data stored in CSV

files.

Implement functions for adding, updating, and deleting products from the inventory.

Implement a function for selling products and updating sales data.

Develop a function for generating reports on inventory and sales data, which can be displayed to the user.

Provide functionality for saving changes made to the inventory and sales data to CSV

files.

Develop error handling and input validation to ensure the user is entering valid data and prevent crashes.

Test the system to ensure its functionality and identify and fix any bugs or issues. Optimize the system for performance and efficiency, so it can handle large amounts of data and users without slowing down or crashing.

Incorporate any additional features or requirements as requested by the grocery store.

# 

# 4.DESCRIPTION OF THE PROJECT

It is a GUI application designed for managing inventory and sales data in a grocery store. It provides an easy-to-use interface that allows users to add, update, and delete products from the inventory, record sales, and generate reports. The application uses the tkinter library for creating the graphical user interface and the pandas library for handling the data in CSV files.

The GUI includes labels and entry fields for entering product details such as name, price, and quantity. Users can perform various tasks by clicking on the buttons provided, such as adding new products, updating existing ones, deleting products, selling products, displaying the inventory, and generating sales reports.

The application is a functional implementation of a grocery store inventory and sales management system, but it may require additional features and error handling depending on the specific needs of the grocery store. Overall, the project is an excellent example of how technology can be used to streamline operations and improve efficiency in retail businesses.

# 5.SOURCE CODE

import pandas as pd import tkinter as tk from tkinter import messagebox

# load inventory data from CSV file inventory\_file = 'inventory.csv' try:

inventory\_df = pd.read\_csv(inventory\_file, index\_col=0) except FileNotFoundError:

inventory\_df = pd.DataFrame(columns=['Name', 'Price', 'Quantity'])

# load sales data from CSV file sales\_file = 'sales.csv' try:

sales\_df = pd.read\_csv(sales\_file, index\_col=0) except FileNotFoundError:

sales\_df = pd.DataFrame(columns=['Name', 'Price', 'Quantity',

'Total'])

def add\_product():

name = name\_entry.get().strip() price = float(price\_entry.get()) quantity = int(quantity\_entry.get())

# check if product already exists if name in inventory\_df.index:

messagebox.showerror('Error', f"Product '{name}' already exists in inventory.")

return

# add new product to inventory inventory\_df.loc[name] = [price, quantity] messagebox.showinfo('Success', f"Product '{name}' added to inventory.")

def update\_product():

name = name\_entry.get().strip()

# check if product exists in inventory if name not in inventory\_df.index:

messagebox.showerror('Error', f"Product '{name}' does not exist in inventory.")

return

# get updated price and quantity values from user

try:

price = float(price\_entry.get()) quantity = int(quantity\_entry.get()) except ValueError:

messagebox.showerror('Error', 'Please enter valid price and quantity values.') return

# update values in inventory inventory\_df.loc[name, 'Price'] = price inventory\_df.loc[name, 'Quantity'] = quantity messagebox.showinfo('Success', f"Product '{name}' updated in inventory.")

def delete\_product():

name = name\_entry.get().strip()

# check if product exists if name not in inventory\_df.index:

messagebox.showerror('Error', f"Product '{name}' does not exist in inventory.")

return

# remove product from inventory inventory\_df.drop(name, inplace=True) messagebox.showinfo('Success', f"Product '{name}' deleted from inventory.")

def sell\_product():

name = name\_entry.get().strip()

quantity = int(quantity\_entry.get())

# check if product exists if name not in inventory\_df.index:

messagebox.showerror('Error', f"Product '{name}' does not exist in inventory.")

return

# check if enough quantity is available if inventory\_df.loc[name, 'Quantity'] < quantity:

messagebox.showerror('Error', f"Not enough quantity of '{name}' available in inventory.")

return

# update quantity in inventory inventory\_df.loc[name, 'Quantity'] -= quantity

# update sales data price = inventory\_df.loc[name, 'Price'] total = price \* quantity sales\_df.loc[len(sales\_df)] = [name, price, quantity, total]

messagebox.showinfo('Success', f"{quantity} units of '{name}' sold.")

def show\_inventory():

inventory\_str = str(inventory\_df) messagebox.showinfo('Inventory', inventory\_str)

def generate\_report():

report\_str = str(sales\_df) messagebox.showinfo('Sales Report', report\_str)

def save\_inventory():

inventory\_df.to\_csv(inventory\_file) messagebox.showinfo('Success', 'Inventory saved to file.') def save\_sales():

sales\_df.to\_csv(sales\_file) messagebox.showinfo('Success', 'Sales data saved to file.')

# create main window

root = tk.Tk() root.title('Grocery Store') root.configure(bg='white')

# create labels and entries for product details name\_label = tk.Label(root, text='Product Name:') name\_label.config(bg='white') name\_label.grid(row=0, column=0, padx=5, pady=5) name\_entry = tk.Entry(root) name\_entry.grid(row=0, column=1, padx=5, pady=5)

price\_label = tk.Label(root, text='Price (Rs):') price\_label.config(bg='white') price\_label.grid(row=1, column=0, padx=5, pady=5) price\_entry = tk.Entry(root) price\_entry.grid(row=1, column=1, padx=5, pady=5)

quantity\_label = tk.Label(root, text='Quantity (kgs):') quantity\_label.config(bg='white') quantity\_label.grid(row=2, column=0, padx=5, pady=5) quantity\_entry = tk.Entry(root) quantity\_entry.grid(row=2, column=1, padx=5, pady=5)

# create buttons for adding, updating, deleting, and selling products add\_button = tk.Button(root, text='Add a New

Product',highlightbackground='grey',borderwidth=3,command=add\_pr oduct) add\_button.grid(row=3, column=0, padx=5, pady=5)

update\_button = tk.Button(root, text='Update Exsisting Product',highlightbackground='grey',borderwidth=3, command=update\_product) update\_button.grid(row=3, column=1, padx=5, pady=5)

delete\_button = tk.Button(root, text='Delete Exsisting Product',highlightbackground='grey',borderwidth=3, command=delete\_product) delete\_button.grid(row=4, column=0, padx=5, pady=5) sell\_button = tk.Button(root, text='Sell Products', highlightbackground='grey',borderwidth=3,command=sell\_product) sell\_button.grid(row=4, column=1, padx=5, pady=5)

# create buttons for displaying inventory and generating sales report inventory\_button = tk.Button(root, text='Show Inventory', highlightbackground='grey',borderwidth=3,command=show\_inventory

)

inventory\_button.grid(row=5, column=0, padx=5, pady=5)

report\_button = tk.Button(root, text='Show Sales Report', highlightbackground='grey',borderwidth=3,command=generate\_report

)

report\_button.grid(row=5, column=1, padx=5, pady=5)

# create button for saving inventory and sales data to files save\_button = tk.Button(root, text='Save Inventory and Sales Data',highlightbackground='grey', command=lambda:[save\_inventory(), save\_sales()]) save\_button.grid(row=6, column=0, columnspan=2, padx=5, pady=5) # start main loop root.mainloop()

### 6.INPUT/OUTPUT

INPUT CAN BE GIVEN ONLY IN ONE WAY AS TO GIVE DATA IN THE MAIN INTERFACE AS SHOWN:



The script loads inventory and sales data from CSV files, and creates empty dataframes if the files do not exist. The inventory dataframe has columns for product name, price, and quantity, while the sales dataframe has columns for product name, price, quantity, and total

sales.

The GUI has labels and entry fields for entering product details such as name, price, and quantity. It also has buttons for adding a new product, updating an existing product, deleting an existing product, selling products, displaying the inventory, and generating a sales report.

When a new product is added, the script checks if the product already exists in the inventory dataframe. If the product already exists, an error message is displayed, and the product is not added. Otherwise, the product is added to the inventory dataframe, and a success message is displayed.

SUCCESS MESSAGE:



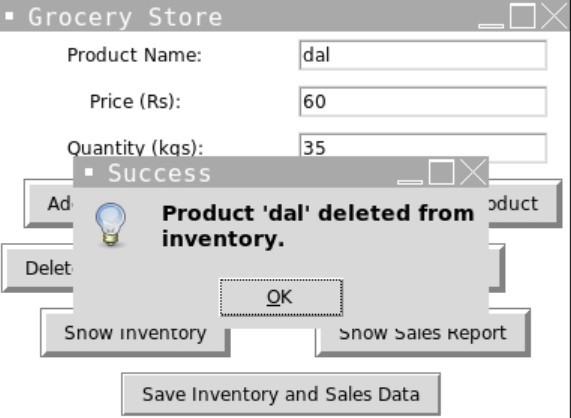
When an existing product is updated, the script checks if the product exists in the inventory dataframe. If the product does not exist, an error message is displayed, and the product is not updated. Otherwise, the price and quantity values for the existing product are updated, and a success message is displayed.

SUCCESS MESSAGE:



When an existing product is deleted, the script checks if the product exists in the inventory dataframe. If the product does not exist, an error message is displayed, and the product is not deleted. Otherwise, the product is removed from the inventory dataframe, and a success message is displayed.

SUCCESS MESSAGE:



When products are sold, the script checks if the product exists in the inventory dataframe, and if enough quantity is available. If the product does not exist or enough quantity is not available, an error message is displayed, and the sale is not processed. Otherwise, the quantity of the product is updated in the inventory dataframe, and a new row is added to the sales dataframe with the product name, price, quantity, and total sales.

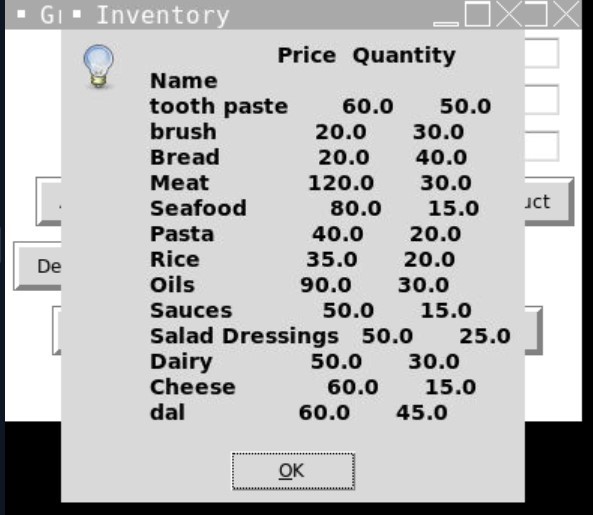
The script also provides functionality for displaying the inventory and generating a sales report. The inventory is displayed in a message box when the "Show Inventory" button is clicked, and the sales report is displayed in a message box when the "Generate Sales Report" button is clicked.

SUCCESS MESSAGE when sold:

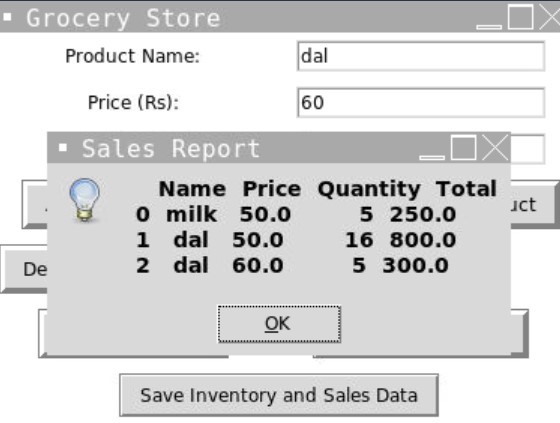


MESSAGE FOR SHOW

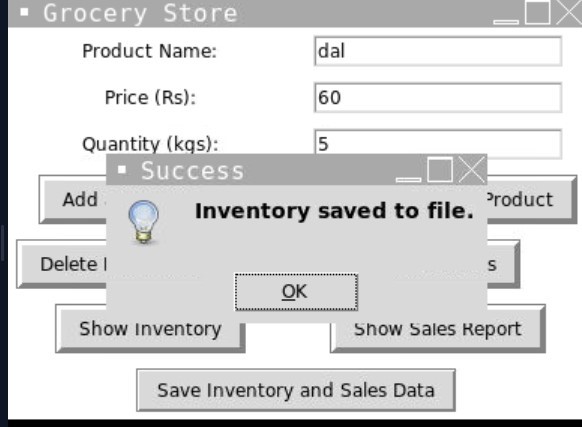
INVENTORY:



MESSAGE FOR SHOW SALES:



The inventory and sales dataframes can be saved to CSV files using the "Save Inventory" and "Save Sales Data" buttons, respectively.



### 7.SCOPE OF PROJECT

The scope of the project involves designing a user-friendly graphical user interface (GUI) that allows store managers to easily add, update, and delete products from their inventory, record sales data, and generate reports.

The system should be able to handle large amounts of data, perform calculations such as total sales, and display information in a clear and concise manner. Additionally, the system should have error handling capabilities to prevent data loss or corruption.

Overall, the project aims to provide grocery store owners with a comprehensive tool to manage their inventory and sales data, allowing them to make informed decisions based on the data generated by the system.

### 8.FUTURE IMPROVEMENTS

User authentication: Implement user authentication to ensure that only authorized personnel can access and modify the inventory and sales data.

Barcode scanner integration: Integrate a barcode scanner to quickly add and sell products without manually entering the product details.

Payment processing: Implement a payment processing system to facilitate payments for customers.

Discount and promotion management: Implement a discount and promotion management system to offer discounts and promotions to customers based on their purchase history or specific criteria.

Analytics and forecasting: Implement analytics and forecasting features to analyze the sales data and predict future trends.

Online ordering and delivery: Implement online ordering and delivery features to allow customers to place orders and have them delivered to their homes.

Real-time inventory updates: Implement real-time inventory updates to ensure that the inventory is always up-to-date and prevent over-selling.

Multi-store management: Implement multi-store management features to manage inventory and sales data for multiple store locations.

Integration with accounting software: Integrate with accounting software to automatically record sales data and generate financial reports.

8.CONCLUSION:

It is a functional grocery store inventory and sales management system using Python. The system allows for the addition, updating, and deletion of products from the inventory, selling of products, and generating reports on the inventory and sales data. The graphical user interface (GUI) makes it easy for users to interact with the system by entering the details of products and performing various tasks with the click of a button. Overall, your project is a good implementation of a grocery store inventory and sales management system, although additional features and error handling could be added to improve its functionality and usability.