

All Expenses						
Date		Category		Description		Amount
25-09-25		food	-		400.0	

```
import tkinter as tk
from tkinter import messagebox, ttk
import pandas as pd

# Store expenses
expenses = []

def add_expense():
    date = entry_date.get()
    category = entry_category.get()
    description = entry_description.get()
    amount = entry_amount.get()

    if not (date and category and description and amount):
        messagebox.showwarning("Input Error", "All fields are required!")
        return

    try:
        amount = float(amount)
    except ValueError:
        messagebox.showerror("Error", "Amount must be a number!")
        return

    expenses.append({
        "Date": date,
        "Category": category,
        "Description": description,
        "Amount": amount
    })

    messagebox.showinfo("Success", "Expense Added Successfully!")
    entry_date.delete(0, tk.END)
    entry_category.delete(0, tk.END)
    entry_description.delete(0, tk.END)
    entry_amount.delete(0, tk.END)
```

```
def view_expenses():
    if not expenses:
        messagebox.showinfo("Info", "No expenses recorded yet.")
        return

    view_window = tk.Toplevel(root)
    view_window.title("All Expenses")

    df = pd.DataFrame(expenses)

    tree = ttk.Treeview(view_window, columns=list(df.columns), show="headings")
    tree.pack(fill=tk.BOTH, expand=True)

    for col in df.columns:
        tree.heading(col, text=col)

    for _, row in df.iterrows():
        tree.insert("", tk.END, values=list(row))
```

```

def summary():
    if not expenses:
        messagebox.showinfo("Info", "No expenses to summarize.")
        return

    df = pd.DataFrame(expenses)
    total = df["Amount"].sum()
    cat_summary = df.groupby("Category")["Amount"].sum()

    summary_window = tk.Toplevel(root)
    summary_window.title("Expense Summary")

    tk.Label(summary_window, text=f"Total Expense: ₹{total:.2f}", font=("Arial", 12, "bold")).pack()

    tk.Label(summary_window, text="\nCategory-wise Spending:", font=("Arial", 11)).pack()

    for cat, amt in cat_summary.items():
        tk.Label(summary_window, text=f"{cat}: ₹{amt:.2f}").pack()

def export_excel():
    if not expenses:
        messagebox.showerror("Error", "No expenses to export.")
        return

    df = pd.DataFrame(expenses)
    df.to_excel("expenses.xlsx", index=False)
    messagebox.showinfo("Success", "Exported as expenses.xlsx")

```

```
# -----  
# GUI STARTS HERE  
# -----  
  
root = tk.Tk()  
root.title("Personal Finance Tracker")  
root.geometry("400x450")  
root.resizable(False, False)  
  
tk.Label(root, text="PERSONAL FINANCE TRACKER", font=("Arial", 14, "bold")).pack(pady=10)  
  
# --- Input Fields ---  
  
tk.Label(root, text="Date (DD-MM-YYYY)").pack()  
entry_date = tk.Entry(root, width=30)  
entry_date.pack()  
  
tk.Label(root, text="Category").pack()  
entry_category = tk.Entry(root, width=30)  
entry_category.pack()  
  
tk.Label(root, text="Description").pack()  
entry_description = tk.Entry(root, width=30)  
entry_description.pack()  
  
tk.Label(root, text="Amount (₹)").pack()  
entry_amount = tk.Entry(root, width=30)  
entry_amount.pack()  
  
# --- Buttons ---  
tk.Button(root, text="Add Expense", width=20, bg="lightgreen", command=add_expense).pack(pady=10)  
tk.Button(root, text="View All Expenses", width=20, command=view_expenses).pack(pady=5)  
tk.Button(root, text="View Summary", width=20, command=summary).pack(pady=5)  
tk.Button(root, text="Export to Excel", width=20, command=export_excel).pack(pady=5)  
tk.Button(root, text="Exit", width=20, bg="lightcoral", command=root.destroy).pack(pady=10)  
  
root.mainloop()
```



PERSONAL FINANCE TRACKER

Date (DD-MM-YYYY)

Category

Description

Amount (₹)

Add Expense

View All Expenses

View Summary

Export to Excel

Exit