Tkinter Social Media App - Step-by-Step Code Explanation

This project is a GUI-based Social Media Application created using Tkinter in Python. Users can post messages, attach images, like, edit, comment, and delete posts. The application stores posts in a local JSON file and displays them on a scrollable interface.

import tkinter as tk  
from tkinter import filedialog, messagebox  
from PIL import Image, ImageTk  
import json  
import os

Importing necessary modules: tkinter for GUI, PIL for image handling, json for storing data, and os for file operations.

class SocialMediaApp:  
 def \_\_init\_\_(self, master):

Defining the main class for the app that will handle UI and logic.

self.master = master  
 self.master.title("Social Media App")

Setting the title for the main window.

self.posts = {}  
 self.post\_id = 1  
 self.images = {}  
 self.image\_path = None

Initializing variables: posts dictionary, post ID tracker, image storage, and path.

tk.Label(master, text="Username:").pack()  
 self.user\_entry = tk.Entry(master)  
 self.user\_entry.pack()

Adding a label and entry box to accept username input from the user.

tk.Label(master, text="What's on your mind?").pack()  
 self.post\_entry = tk.Entry(master, width=50)  
 self.post\_entry.pack()

Adding a label and entry box for the user to write their post content.

self.attach\_btn = tk.Button(master, text="Attach Image", command=self.select\_image)  
 self.attach\_btn.pack()

Adding a button to allow the user to attach an image using a file dialog.

self.post\_btn = tk.Button(master, text="Post", command=self.create\_post)  
 self.post\_btn.pack(pady=5)

Adding a button to submit the post.

# Setting up scrollable frame for displaying posts

Creating a scrollable area using Canvas and Frame to hold and display posts.

# select\_image method to pick image from file dialog

Allows users to select an image file and save its path.

# create\_post method to save the post and update UI

Saves the new post with all fields and refreshes the post display.

# refresh\_posts method to reload posts in the scrollable frame

Clears existing widgets and reloads all posts from the data.

# display\_post method to show a single post with image, comments, and buttons

Displays each post along with attached image, buttons, and comment section.

# like\_post method to increase like count

Handles the logic for liking a post by incrementing the count.

# edit\_post method to modify the content of a post

Opens a new window with an entry to edit the content of a post.

# delete\_post method to remove a post

Deletes a post after confirmation from the user.

# add\_comment method to add a comment to the post

Adds a comment to a particular post and updates the interface.

# save\_data and load\_data to persist and retrieve posts using JSON file

Handles saving posts into a JSON file and loading them at startup.

# Running the App  
if \_\_name\_\_ == "\_\_main\_\_":  
 root = tk.Tk()  
 app = SocialMediaApp(root)  
 root.mainloop()

Starts the main application loop.