1. Project Overview (Nitzagram):

- My project is essentially a simulation of a social media app similar to Instagram. The main goal is to create an interface where you can view posts (text, images, and videos), interact with them (like, comment), and navigate between them.
- The app features posts of different types:
    - **Text Posts**: Users can write a description.
    - **Image posts**: Display images from the file system.
    - **Video posts**: Play videos in a loop or on demand.
    - **User Interactions**: Users can like and comment on posts.
- The **Pygame** library is used for creating the graphical interface, rendering the posts, and handling user inputs (like mouse clicks and keyboard events).
- **OpenCV (cv2)** is used for handling video playback.
- **NumPy** is used for array manipulation, particularly useful for video frame processing in OpenCV.

2. Directory Breakdown:

Let's break down the directory structure and explain the role of each folder and file:

---

a. Main Project Folder (Instagram or Nitzagram):

- This is where your main project files live.
- Typically, this directory contains:
    - **main.py**: The entry point of the application, where the core game loop is implemented, handling user events, displaying posts, etc.
    - **helpers.py**: Helper functions to make the code modular. Functions for handling user input, text conversion, drawing on the screen, etc.
    - **constants.py**: Defines constants like window dimensions, colors, fonts, positions, etc., used throughout the project.

---

b. Libraries and Dependencies:

- **pygame**: Used for handling the graphical user interface (GUI), input events (mouse and keyboard), and rendering images, text, and shapes. It's the main tool for rendering the posts and interactive buttons.
- **cv2 (OpenCV)**: Used for video processing. It provides functions for loading, resizing, playing, and manipulating videos. You're using it to display video posts and handle playback.
- **numpy**: Mainly used to manipulate the frames of the video. OpenCV often relies on NumPy for efficient array manipulation since images and videos can be represented as NumPy arrays.

c. File Structure:

Instagram Project (Root Folder)

```
|
|—— main.py           # Main script that runs the app
|—— helpers.py        # Helper functions (for handling comments, drawing, etc.)
|—— constants.py      # All the constant values like window size, colors, etc.
|
```

```
├── images/              # Folder containing images used in posts
│   ├── noa_kirel.jpg    # Example image used for an image post
│   └── ronaldo.jpg      # Another image used for an image post
│
├── videos/              # Folder containing video files
│   ├── video1.mp4       # Example video file used for a video post
│   └── video2.mp4       # Another video file
│
├── classes/             # Folder containing the main classes for posts, comments, etc.
│   ├── Post.py          # Base class for all post types
│   ├── ImagePost.py     # Class for image-based posts
│   ├── VideoPost.py     # Class for video-based posts
│   ├── TextPost.py      # Class for text-based posts
│   └── Comment.py       # Class for handling comments
│
└── README.md            # Project documentation
```

3. Explanation of Libraries and Files:

Pygame:

- **Main Purpose**: Handling graphical rendering and user input.
- **Key Features:**
  - Display images and text on the screen (rendering the posts).
  - Handle user interactions like mouse clicks and keyboard inputs (like, comment, navigation).
  - Control the game loop and screen updates.
- **Usage In Your Project:**
  - Render the background, posts, and user interface.
  - Display comments, likes, and text descriptions.
  - Handle interactions like button presses and navigation.

OpenCV (cv2):

- **Main Purpose**: Used to process and display video content.
- **Key Features:**
  - Read and manipulate video files.
  - Resize and rotate frames for video playback.
  - Convert video frames into a format that Pygame can use (via NumPy arrays).
- **Usage In Your Project:**
  - Play videos in posts.
  - Handle video frame-by-frame reading.
  - Resize videos to fit into your post's display area.

NumPy:

- **Main Purpose**: Provides high-performance array handling.
- **Key Features:**
  - Efficient handling of large datasets, like video frames.
  - Performs operations like resizing and manipulating pixel values.
  - Allows conversion of image data between OpenCV and Pygame.
- **Usage In Your Project:**
  - Process video frames as NumPy arrays.
  - Convert video frames to a format that Pygame can display.

---

4. Class Breakdown:

Post (Base Class):

- This is the parent class that all post types (Text, Image, Video) will inherit from.
- It contains basic information like the username, location, description, and like counter.
- The `display()` method is overridden in subclasses to customize the appearance of each type of post.

ImagePost (Child Class of Post):

- This class handles image-based posts.
- It loads and displays an image when the post is active.

VideoPost (Child Class of Post):

- This class handles video-based posts.
- It uses OpenCV to load, display, and loop videos. The video will stop and reset based on user input.

TextPost (Child Class of Post):

- This class handles text-based posts.
- It renders text descriptions from an array and allows for basic text customization like color.

Comment Class:

- The `Comment` class stores and displays user comments on posts.
- It allows users to interact with the post by commenting.

---

5. How the Code Works:

1. **Main Loop (main.py):**
   - Initializes Pygame and loads necessary resources (background, posts).
   - Displays the current post (image/video/text) based on the active post type.
   - Handles user interactions, like liking a post or adding a comment.
2. **Post Rendering:**
   - Depending on the post type, the `display()` method is called to show the content on the screen.
   - For image posts, the image is drawn. For video posts, OpenCV reads and displays video frames. For text posts, the text is rendered on the screen.
3. **User Interaction:**
   - Users can click to like, comment, or navigate to the next post.
   - Comments are read from user input and added to the post.

THE LEGENDARY CREATORS OF THIS 👍

FYODOR GROM & ELIAV BENAMIN