Task Notifier:

This Python script creates a desktop notification application using Tkinter. It allows users to schedule notifications with custom titles, messages, times, icons, and sounds. Here's a breakdown of what each part does:

1. Imports:

- tkinter, ttk, messagebox, filedialog: For creating the GUI and handling file dialogs and messages.
- json: To save and load notifications from a file.
- datetime, time: For scheduling notifications based on current time.
- threading: To run notification checks in a separate thread without freezing the GUI.
- os, platform: For managing files and checking the operating system.
- PIL (Pillow): For handling images, specifically converting PNGs to ICO format.
- winsound: For playing sound on Windows.
- win10toast and winotify, plyer: For cross-platform notifications.

2. Class NotifierApp:

This class defines the application and its functionality.

__init__Method:

- Creates the main window and sets the title and dimensions.
- Initializes paths for the default icon and sound.
- Checks if Windows notification libraries are available and sets the default notifier.
- Creates GUI components including a notification form and list.
- Starts a background thread to check for notifications.

Form and Sound Management:

- choose_sound(): Opens a file dialog to select a sound file.
- test_sound(): Plays the selected sound file (if on Windows).
- play_notification_sound(): Plays the notification sound when triggered.

Creating and Managing Notifications:

- create_form(): Defines the input form for creating a notification.
- create_notification(): Saves notification details (title, message, time, optional icon) and stores them in a list.
- send_notification(): Sends a notification using the available system (Win10 toast, Plyer) or falls back to a message box if neither library is available.

Icon Management:

- choose_icon(): Opens a dialog to select an icon, converts PNGs to ICO format if needed.
- Icons are encoded to base64 for storage and decoded during notifications to avoid handling files directly.

List and Storage Management:

- create_list(): Displays existing notifications in a tree view.
- refresh_list(): Reloads the list view when notifications are added, updated, or deleted.
- load_notifications() and save_notifications(): Load and save notifications to/from a JSON file.

Notification Scheduling:

- check_notifications(): Runs in a separate thread to check if the current time matches any notification times, and sends notifications if so. It checks every 30 seconds.

Updating and Deleting Notifications:

- update_notification(): Allows editing of an existing notification.
- delete_notification(): Deletes the selected notification.

3. Main Function:

- Starts the application by creating a Tk root window and initializing the NotifierApp class.

And the program file can be downloaded as an custom desktop app(exe) by using the command:

python -m PyInstaller –onefile –name “Task & Reminder Notifier” –icon=”ICO Image File Path “ “Program File path”

Note: The app will be running locally in your laptop.