**Extra Lab 1 – Load 10 Images in 4 Labels, Rotating Every Second**

**Objective:**

**Updated Code: Load 10 Images in 4 Labels, Rotating Every Second**

This version dynamically loads images from **4 folders**, displaying **one image per label at a time**. Every **second**, new images replace the existing ones, cycling through **10 images per folder**.

**Key Features**

* **Only 4 labels** – Images will rotate every second.
* **Loads from 4 folders, each with 10 images**.
* **Uses QTimer to update images in intervals**.
* **Uses QThread to load images asynchronously**.

**Updated Code**

import sys

import os

import time

from PySide6.QtWidgets import QApplication, QWidget, QLabel, QVBoxLayout, QPushButton, QFileDialog

from PySide6.QtGui import QPixmap

from PySide6.QtCore import Qt, QThread, Signal, QTimer

class ImageLoaderThread(QThread):

image\_loaded = Signal(int, QPixmap) # (Label Index, Loaded Image)

def \_\_init\_\_(self, label\_index, image\_path):

super().\_\_init\_\_()

self.label\_index = label\_index

self.image\_path = image\_path

def run(self):

"""Loads the image with a slight delay to simulate real loading."""

time.sleep(0.5) # Simulate loading time

pixmap = QPixmap(self.image\_path)

self.image\_loaded.emit(self.label\_index, pixmap)

class MultiFolderImageRotator(QWidget):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.setWindowTitle("Rotating Image Loader - PySide6")

self.setGeometry(200, 200, 400, 300)

# Layout

self.layout = QVBoxLayout()

# 4 QLabel widgets for images

self.labels = [QLabel(f"Label {i+1}") for i in range(4)]

for label in self.labels:

label.setAlignment(Qt.AlignCenter)

label.setFixedSize(100, 100)

label.setStyleSheet("border: 2px solid black; padding: 5px;")

self.layout.addWidget(label)

# Select Folders Button

self.select\_button = QPushButton("Select 4 Folders")

self.select\_button.clicked.connect(self.select\_folders)

self.layout.addWidget(self.select\_button)

# Start Rotation Button

self.start\_button = QPushButton("Start Image Rotation")

self.start\_button.clicked.connect(self.start\_image\_rotation)

self.layout.addWidget(self.start\_button)

self.start\_button.setEnabled(False)

self.setLayout(self.layout)

self.image\_folders = [] # Store selected folders

self.image\_paths = [[] for \_ in range(4)] # Store images per folder

self.current\_indices = [0] \* 4 # Track current image index per folder

self.threads = [] # Keep track of threads to prevent garbage collection

# Timer to update images every second

self.timer = QTimer(self)

self.timer.timeout.connect(self.update\_images)

def select\_folders(self):

"""Opens dialog to select 4 folders containing images."""

self.image\_folders.clear()

for i in range(4):

folder = QFileDialog.getExistingDirectory(self, f"Select Folder {i+1}")

if folder:

self.image\_folders.append(folder)

if len(self.image\_folders) == 4:

self.find\_images()

self.start\_button.setEnabled(True)

def find\_images(self):

"""Finds up to 10 images per folder."""

self.image\_paths = [[] for \_ in range(4)]

for i, folder in enumerate(self.image\_folders):

images = [os.path.join(folder, f) for f in os.listdir(folder)

if f.lower().endswith((".png", ".jpg", ".jpeg", ".bmp"))]

self.image\_paths[i] = images[:10] # Store first 10 images

def start\_image\_rotation(self):

"""Starts the timer to update images every second."""

self.current\_indices = [0] \* 4 # Reset index counters

self.timer.start(1000) # Update every 1 second

def update\_images(self):

"""Loads the next image for each label using threads."""

for i in range(4): # Iterate through labels

if self.image\_paths[i]: # Ensure images exist in folder

image\_index = self.current\_indices[i] % len(self.image\_paths[i])

image\_path = self.image\_paths[i][image\_index]

thread = ImageLoaderThread(i, image\_path)

thread.image\_loaded.connect(self.display\_image)

self.threads.append(thread)

thread.start()

self.current\_indices[i] += 1 # Move to the next image

def display\_image(self, label\_index, pixmap):

"""Updates the QLabel with the loaded image."""

self.labels[label\_index].setPixmap(pixmap)

self.labels[label\_index].setScaledContents(True)

if \_\_name\_\_ == "\_\_main\_\_":

app = QApplication(sys.argv)

window = MultiFolderImageRotator()

window.show()

sys.exit(app.exec())

**How It Works**

1. **Folder Selection**
   * User selects **4 folders**, each containing **10 images**.
   * The app collects the first **10 images per folder**.
2. **Image Rotation (Every 1 Second)**
   * QTimer triggers update\_images() **every second**.
   * Each label gets a **new image** from its corresponding folder.
   * The process repeats in a loop after 10 images.
3. **Threaded Image Loading**
   * Each image is loaded in **a separate QThread** to prevent UI lag.
   * Uses **signals** to update the QLabel when an image is ready.

**Expected Behavior**

* Click **"Select 4 Folders"** → Choose 4 different image folders.
* Click **"Start Image Rotation"** → Images start loading.
* Every **second**, the 4 labels display a **new image**.
* Once **all 10 images per folder** are displayed, the cycle **repeats**.