IMAGE IDENTIFIER

The Photography Platform is a Django-based web application that connects photographers and users by leveraging advanced facial recognition technology to help users find their event photos easily. This platform enables two main user types: photographers and regular users. Photographers upload event photos enriched with metadata and schedule/manage their events, while users upload personal photos to find matches from the photographers' collections using facial recognition models.

Core Features

User Features:

- Register and login as a regular user.
- Upload photos to find matching photos from event uploads.
- Advanced facial recognition using multiple AI models.
- View match results with associated confidence scores.
- Download matched photos.

Photographer Features:

- Register and login as a photographer.
- Upload photos associated with events, including essential metadata such as event name, date, and location.
- Manage events and photo collections seamlessly.
- Access a dashboard displaying statistics and analytics.
- Schedule, edit, and delete upcoming events and photo collections.

Data Models

- CustomUser: Extends Django's AbstractUser with a user_type field to differentiate photographers and regular users.
- Photo: Stores uploaded photos with associated metadata and facial recognition embeddings (e.g., event name, date, location, and the photographer who uploaded the photo).
- MatchedPhoto: Tracks matches between user-uploaded photos and photographer photos with timestamps.
- PhotographerProfile: Contains extended profile info for photographers, including business name, contact details, workload, and terms acceptance.

- ScheduleUpcomingEvents: Enables photographers to plan future events with details like event name, date, and venue.

URL Patterns and Views

- Authentication and main landing pages, including distinct login/logout paths for photographers and users.
- Separate dashboards for photographers (showing uploads, events, analytics) and users (showing profile and match results).
- Photographer photo management views (upload, event listing, photo addition, editing, and deletion).
- User-centric views for uploading photos to find matches, viewing matches, and downloading found photos.
- Dedicated error page when no face is detected on uploaded photos.

Facial Recognition System

- Integrates multiple AI facial recognition models: VGG-Face, FaceNet-512, ArcFace, and FaceNet-128.
- Supports multi-model processing for higher accuracy via consensus scoring across models.
- Processing flow:
    1. User uploads photo.
    2. Face detection and facial embeddings are extracted.
    3. Embeddings are compared against the database of photographer photos using selected models.
    4. Matches are scored and ranked.
    5. Results are presented with confidence scores.

Security and Access Control

- User type separation restricts access accordingly.
- Only photographers can manage their own photos, events, and collections.
- Input validation, image format checks, file size, and upload quantity limits protect against abusive uploads.
- Temporary files are cleaned up after processing.

Dashboard Analytics

- Photographers' dashboard displays total photo uploads, event counts, recent uploads, monthly trends, and upcoming scheduled events.

Technical Implementation Highlights

- Facial encodings are extracted using the DeepFace library, particularly with the ArcFace model example in use.
- Photo embeddings are stored as binary data (pickled embeddings) in the database fields.
- Multi-model consensus aggregates results from multiple models to improve matching reliability.
- AJAX support for real-time feedback during photo upload and matching.
- Session management is employed to store match results and processing statistics, allowing persistence across user interactions.

Installation and Setup

- Built using Django 4.0+, Pillow (PIL) for image handling, DeepFace library for facial recognition, TensorFlow or PyTorch backend for model computations.
- Proper MEDIA_ROOT and MEDIA_URL configuration required for file storage.
- Required Python packages include django, pillow, deepface, and tensorflow.

Usage Workflows

For Photographers:

1. Register with business details.
2. Access dashboard to manage photos and events.
3. Upload photos with event metadata.
4. Schedule and manage upcoming events.
5. Edit/delete events or photos as needed.

For Users:

1. Register as a regular user.
2. Upload a clear face photo on the matching page.
3. Select preferred AI model(s) and similarity threshold.
4. View ranked match results with confidence scores.
5. Download matched photos.

Error Handling

- Covers invalid image formats, face detection failures, processing errors, database issues, and file system problems.
- Users receive informative feedback and safe redirections where applicable.

Performance Considerations

- Facial recognition can be resource-intensive; optimizations like indexing and asynchronous background processing for large uploads are suggested.
- Automatic cleanups of temporary files help maintain system health.

This Photography Platform is a feature-rich, secure, and scalable Django web application that combines modern AI-powered facial recognition with a user-friendly interface to link users and photographers efficiently for event photo discovery and management.