**Overview**

Goal: Enable users to sign up, log in, upload wildlife images, detect animals using a Faster R-CNN model, and receive email alerts for detections.

**Mistakes in current code:**

1. Require video screenshots in a sequence or video data for the implementation of LSTM models. Independent screenshots cannot add to sequence of video for monitoring the attacks by animals
2. Unnecessary usage of hybrid deep learning neutral networks
3. Reducing code syntax by using streamlit dashboard and flask
4. Mask RCNN will help in finding out the regions where probability of objects is huge and help in object detection

**Components**

1. **Frontend:** Streamlit Dashboard

- User Authentication: Sign up and log in functionality.

- Image Upload: Interface for users to upload images.

- Animal Detection: Display detection results.

- Email Alerts: Notify users of specific animal detections.

2. **Backend:** Model Training and Inference

- Model Training: Fine-tune a pre-trained Faster R-CNN with a custom dataset.

- Inference: Load the trained model for image detection.

3. **Email Notification System:** Use SMTP (e.g., Gmail) for sending alerts.

**Data Flow**

1. **Sign-Up/Login:** Validate user credentials.

2. **Image Upload:** Process and temporarily store images.

3. **Animal Detection:** Run the model on uploaded images and display results.

4. **Email Alert:** Send notifications for detected animals.

**Technical Architecture**

- Frontend: Streamlit

- Backend: TensorFlow

- Email Service: SMTP (Gmail)

- Data Storage: Store user credentials (simple dictionary or database).

**Model Training Approach**

1. Data Preparation: Collect and convert images into TFRecord format.

2. Model Configuration: Modify pre-trained Faster R-CNN config for custom data.

3. Training: Train the model and save it for inference.

**User Interface**

- Main Menu: Options for sign-up, login, upload image, detect animals, log out.

- Sub-Pages: Include sign-up/login, image upload, and detection results.

**Security**

- Use hashed passwords and validate email inputs to enhance security.

**Deployment**

- Testing: Conduct unit and integration tests.

- Deployment: Host on platforms like Streamlit Sharing or Heroku.

**Future Enhancements**

- Implement user profiles, expand detection capabilities, and improve user management.

This concise design outlines the essential components and steps to build the wildlife detection system.