This project aims to build a Bird classification model using ResNet. With this model, we aim to accurately predict the 200 species of birds found in our dataset. The dataset used in this project is a directory that contains 200 different species of bird images, with each species grouped into a separate folder.

Preprocessing and preparing the images

As the data provided in the link contains an image folder and a list of classes, train, and test files I wrote a simple function to organize the images into train and test based on the list of train and test text files. Then I started to load images as the data to visualize and check their size and shape.

Python Libraries Used
The second step is choosing the main framework to design and develop our deep learning model.
For this project, I used Tensorflow and Keras to do that, but I also have great experience in Pytorch.

The third step is Image Data Handling and Preprocessing
To gain better performance in our model and prevent the overfitting problem I used ImageDataGenerator in Tensorflow for data augmentation. It can perform data augmentation techniques such as flipping, rotating, and zooming which help us our model see and train more images in many ways.

The 4th step is Model Building and Training

For building the model to train I used function API in Tensorflow. It allows you to create a model by simply adding layers one after another. However, we can use Sequential API to build our model, it also helps us build our model easily. After that, I added fully connected layers to our ResNet pretrained model as you mentioned in the assessment. Use dropout to reduce the overfitting of the model.we can use batch normalization also to help us to reduce overfiting. After all, I built the model and compiled it with Adam optimizer with 0.0001 learning rate and I chose Categorical_crossentropy for my loss function. I have trained the model so many times to find the best learning rate which is crucial in our training model, also I selected the 200 epochs for training because the ResNet model takes time to train the model.

The 5th step is evaluation(Important part please read it carefully I wrote the main producer here)

When the training was finished I checked the accuracy and loss to figure out how the model trained. Accuracy and loss in training data are good but I faced overfitting since validation accuracy and validation loss don't look good. The accuracy is high and validation accuracy is lower than it.
Then I tried to fine-tune the model with transfer learning and switched the trainable layer to True and trained again but unfortunately, the result was not much more different and I faced overfitting. I trained the model with more than 50 times change learning rate, epochs, and other parameters to gain a better model. I trained 2 ResNet models ResNet50 and ResNet101.

I decided to figure out why my training and model didn't look good then I tried the **EfficientNet** model. I started to train the model with **EfficientNetB0** and saw it trained much better than the ResNet model. As you know **EfficientNet** models are known for their high accuracy and efficiency.

Finally to avoid overfitting, I employed two techniques: model checkpoint and early stopping. The model checkpoint allowed us to save the best model during training, enabling us to resume training from the best model if the training process was interrupted. On the other hand, early stopping stopped the training process if the model's performance did not improve after a certain number of epochs, thereby preventing the model from overfitting to the training data.

After training our Resnet model on the train set, we evaluated its performance on a tes set that was not used during training . Our model achieved an accuracy of 85% in categorizing bird species, which is a high level of accuracy. Additionally, our model had a low loss of 0.463, indicating that it was able to generalize well to new data. But we should try to make it better since the validation accuracy is 40%.

The 6th step is visualizing the result

In this step, I use the matplotlib library to visualize the results I wrote the function called predict_pipline to see the result and compare the actual labels with the predicted ones. In this function, I wrote a preprocessing step for the image to ensure that they were suitable for input into the model for prediction.