Goal: Using image of a pistachio we need to find which type of Pistachio it is

Step1: Setting up the data:

```
Total_Images = glob.glob('..../*/*.jpg')
```

we have a total of 2148 files

So for Kirmizi → 1232

Siirt → 916

Step2: Prepare data with first getting everything together & then splitting

Train: 80% Val: 10% test: 10%

So that is about

Train: 1717 Val: 217 test: 214

- Step3: Lets create our training dataframe

```
train_df = pd.Series()
train_df['filename'] = train_image.map(lambda img_name: img_name.split("/")[-1])
train_df['ClassID'] = train_image_names.map(lambda img_name: img_name.split("/")[-2])
```

- Similarly do it for test_df

- Step4: Print a image of each category so that we can be sure of our directory path

- Step5: For this task we will use EfficientNet

```
from tensorflow.keras.applications import EfficientNetB0
model = EfficientNetB0(input_shape = (224,224,3), include_top = False, weights = 'imagenet')
```

fine tuning our model we will set last 15 layers as tr able

- EfficientNet B0
- ↓
- Flatten()
- Dropout() 0.5
- Dense 1

- Step6: Compile
  - → loss = "binary_crossentropy"
  - → optimizer = adam
  - → metrics : "accuracy"

- Step 7: We can also plot the model as

```
from tensorflow.keras.utils import plot_model
from IPython.display import Image
plot_model(model, to_file = 'model.png', show_shapes = True, show_layer_names = True)
Image(filename = 'model.png')
```

- Step8: To prevent from overfitting we will use EarlyStopping & to save model weights we will also use ModelCheckpoint

```
es = EarlyStopping(monitor='val_accuracy', mode='max', verbose=1, patience=20)
mc = ModelCheckpoint('model.h5', monitor='val_accuracy', mode='max', save_best_only=True)
```

- Step9: fit the model with
  - → validation_data
  - → epochs = 50
  - → verbose = 1
  - → callback with EarlyStopping & ModelCheckpoint

- Step10: Using History we can access