Student Name	SASIDHARAN S
Student Roll Number	2116190701193

Download the Dataset

In[1]:
```
import numpyas np
import tensorflowas tf
from tensorflow.kerasimport layers
from tensorflow.keras.modelsimport Sequential
import matplotlib.pyplotas plt
import os
```

In[2]:
```
batch_size= 16
```

Image Augmentation

In[3]:
```
data_aug= Sequential(
  [
    layers.RandomFlip("horizontal",input_shape=(180, 180, 3)),
    layers.RandomRotation(0.1),
    layers.RandomZoom(0.1),
  ]
)
```

In[5]:
```
os.listdir("C:\\Users\\Harini\\Flowers-Dataset")
```

Out[5]:
```
['flowers']
```

In[6]:
```
train_data= tf.keras.utils.image_dataset_from_directory(
  "C:\\Users\\Harini\\Flowers-Dataset",
  validation_split=0.25,
  subset="training",
  seed=120,
  image_size=(180, 180),
  batch_size=batch_size)
```

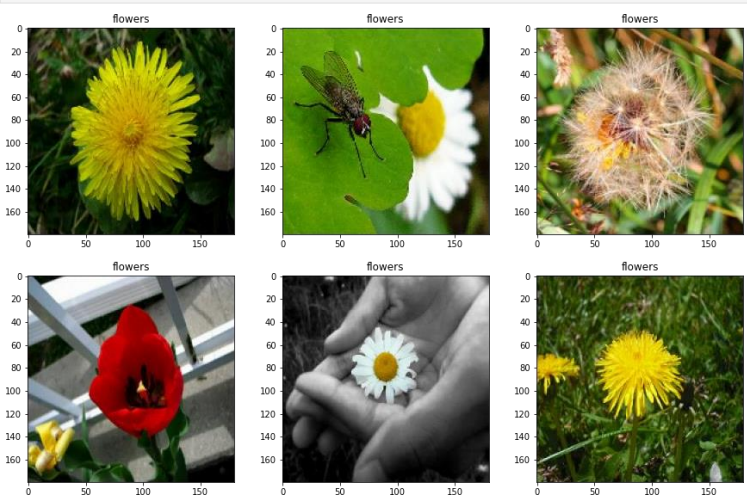
```
Found 4317 files belonging to 1 classes.
Using 3238 files for training.
```

In[7]:
```
val_data_set= tf.keras.utils.image_dataset_from_directory(
 "C:\\Users\\Harini\\Flowers-Dataset",
  validation_split=0.25,
  subset="validation",
  seed=120,
  image_size=(180, 180),
  batch_size=batch_size)
```

```
Found 4317 files belonging to 1 classes.
Using 1079 files for validation.
```

In[8]:
```
class_names= train_data.class_names
```

In[9]:
```
plt.figure(figsize=(15, 15))
for images, labels in train_data.take(1):
  for iin range(6):
    ax = plt.subplot(3, 3, i+ 1)
    plt.imshow(images[i].numpy().astype("uint8"))
    plt.title(class_names[labels[i]])
```



In[10]:
```
normalization_layer= layers.Rescaling(1./255)
```

In[11]:
```
dataset_normalized= train_data.map(lambda x, y: (normalization_layer(x), y))
image_batch, labels_batch= next(iter(dataset_normalized))
first_image= image_batch[0]
print(np.min(first_image), np.max(first_image))
```

```
0.0 1.0
```

Create Model

Add Layers (Convolution,MaxPooling,Flatten,Dense-(Hidden Layers),Output)

In[12]:
```
num_classes= len(class_names)

model = Sequential([
  data_aug,
  layers.Rescaling(1./255, input_shape=(180, 180, 3)),
  layers.Conv2D(16, 3, activation='relu'),
  layers.MaxPooling2D(), layers.Conv2D(32,
  3,activation='relu'), layers.Conv2D(32,
  3,activation='relu'),
  layers.MaxPooling2D(), layers.Conv2D(64,
  3, activation='relu'),
  layers.MaxPooling2D(), layers.Flatten(),
  layers.Dense(128, activation='relu'),
  layers.Dense(num_classes)
])
```

Compile The Model

compiling model with categorical cross entropy and adam optimizer

In[13]:
```
model.compile(optimizer='adam',
loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
metrics=['accuracy'])
```

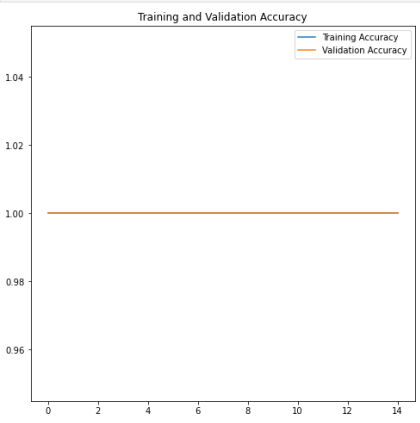
Fit The Model

In [14]:
```
epochs=15
history = model.fit(train_data,validation_data=val_data_set,epochs=epochs)
```

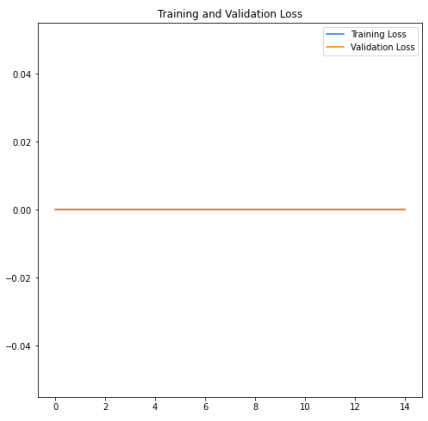
```
Epoch1/15
203/203 [==============================] - 33s 154ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch2/15
203/203 [==============================] - 28s 140ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch3/15
203/203 [==============================] - 29s 141ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch4/15
203/203 [==============================] - 29s 143ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch5/15
203/203 [==============================] - 28s 139ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch6/15
203/203 [==============================] - 29s 140ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch7/15
203/203 [==============================] - 28s 140ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch8/15
203/203 [==============================] - 29s 144ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch9/15
203/203 [==============================] - 28s 137ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch10/15
203/203 [==============================] - 28s 137ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch11/15
203/203 [==============================] - 28s 139ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch12/15
203/203 [==============================] - 29s 142ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch13/15
203/203 [==============================] - 28s 140ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch14/15
203/203 [==============================] - 29s 143ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
Epoch15/15
203/203 [==============================] - 29s 141ms/step - loss: 0.0000e+00 - accuracy: 1.0000 - val_loss: 0.0000e+00 - val_accuracy: 1.0000
```

In[15]:
```
epochs_range= range(epochs)

plt.figure(figsize=(8, 8))
plt.plot(epochs_range, history.history['accuracy'], label='Training Accuracy')
plt.plot(epochs_range, history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title('Training and Validation Accuracy')
plt.show()
```



In[16]:
```
plt.figure(figsize=(8, 8))
plt.plot(epochs_range, history.history['loss'], label='Training Loss')
plt.plot(epochs_range, history.history['val_loss'], label='Validation Loss')
plt.legend()
plt.title('Training and Validation Loss')
plt.show()
```



Save The Model

In[17]:
```
model.save("./flowers.h5")
```

In[18]:
```
model.load_weights('./flowers.h5')
```

Test The Model

In [19]:
```
from tensorflow.keras.preprocessingimport image
import numpyas np
```

In [20]:
```
img=image.load_img('C:\\Users\\Sai\\Flowers-Dataset\\flowers\\rose\\5172171681_5934378f08.jpg',target_size=(70,70))   img
```

Out[20]:

In [ ]: