In [ ]:

In [188…

```
fresh = len(os.listdir("/Users/alyaaljarallah/Desktop/Fruit_project/fruits_dataset/fresh"))
rotten = len(os.listdir("/Users/alyaaljarallah/Desktop/Fruit_project/fruits_dataset/rotten"))

print(f"Fresh: {fresh}, Rotten: {rotten}")
```

```
Fresh: 105, Rotten: 121
```

In [189…

```
dataset_path = "/Users/alyaaljarallah/Desktop/Fruit_project/fruits_dataset"

train_datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2, zoom_range=0.3,
    horizontal_flip=True,
    brightness_range=[0.6, 1.4]
)

val_datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2
)

train_generator = train_datagen.flow_from_directory( dataset_path,
target_size=(150, 150), batch_size=32, class_mode='binary', subset='training', shuffle=True)

val_generator = val_datagen.flow_from_directory( dataset_path,
target_size=(150, 150), batch_size=32, class_mode='binary', subset='validation', shuffle=False )
```

```
Found 181 images belonging to 2 classes.
Found 44 images belonging to 2 classes.
```

In [190…

```
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.models import Model
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.losses import BinaryCrossentropy
from tensorflow.keras.metrics import Precision, Recall
```

```
base_model = MobileNetV2( input_shape=(150, 150, 3), include_top=False, weights='imagenet'
)
base_model.trainable = False
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dropout(0.3)(x)
output = Dense(1, activation='sigmoid')(x)
model_mobilenet = Model(inputs=base_model.input, outputs=output)
model_mobilenet.compile(
optimizer=Adam(learning_rate=1e-4), loss=BinaryCrossentropy(label_smoothing=0.1),
metrics=['accuracy', Precision(name='precision'), Recall(name='recall')]
)
```

```
/var/folders/2n/f4mkx5h97l1dgwp_0486t5940000gn/T/ipykernel_99790/445761047.py:7: UserWarning: `input_shape` is undefi
ned or non-square, or `rows` is not in [96, 128, 160, 192, 224]. Weights for input shape (224, 224) will be loaded as
the default.
  base_model = MobileNetV2( input_shape=(150, 150, 3), include_top=False, weights='imagenet'
```

In [191…

```
from tensorflow.keras.callbacks import EarlyStopping
early_stop = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=True)
history_baseline = model_mobilenet.fit( train_generator,
validation_data=val_generator, epochs=20,callbacks=[early_stop] )
```

```
/Users/alyaaljarallah/opt/anaconda3/lib/python3.9/site-packages/keras/src/trainers/data_adapters/py_dataset_adapter.p
y:121: UserWarning: Your `PyDataset` class should call `super().__init__(**kwargs)` in its constructor. `**kwargs` ca
n include `workers`, `use_multiprocessing`, `max_queue_size`. Do not pass these arguments to `fit()`, as they will be
ignored.
  self._warn_if_super_not_called()
```

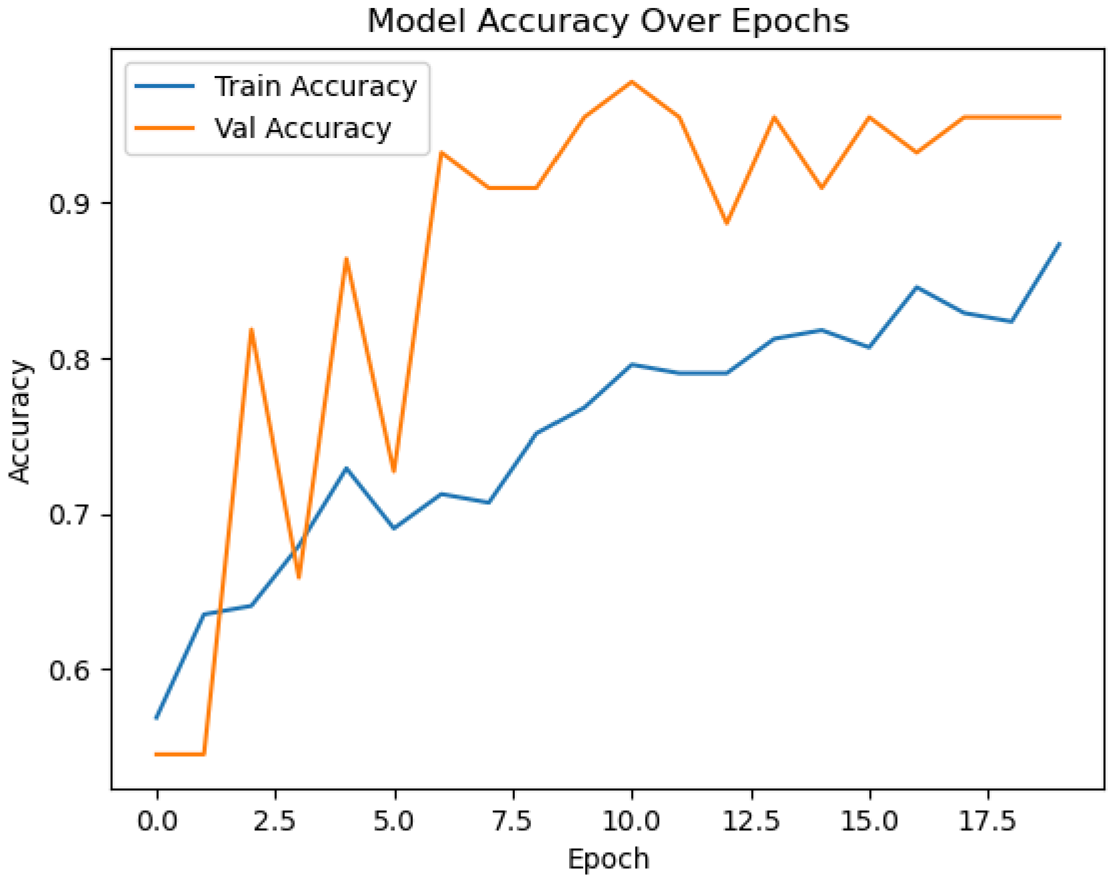
```
Epoch 1/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 23s 1s/step - accuracy: 0.4985 - loss: 0.9424 - precision: 0.5629 - recall: 0.7048 - val_acc
uracy: 0.4773 - val_loss: 0.7772 - val_precision: 0.5172 - val_recall: 0.6250
Epoch 2/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 666ms/step - accuracy: 0.4848 - loss: 0.9407 - precision: 0.5041 - recall: 0.7740 - val_a
ccuracy: 0.5000 - val_loss: 0.7624 - val_precision: 0.5357 - val_recall: 0.6250
Epoch 3/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 5s 899ms/step - accuracy: 0.5577 - loss: 0.8279 - precision: 0.5778 - recall: 0.7083 - val_a
ccuracy: 0.5000 - val_loss: 0.7529 - val_precision: 0.5357 - val_recall: 0.6250
Epoch 4/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 665ms/step - accuracy: 0.5238 - loss: 0.8949 - precision: 0.5807 - recall: 0.6701 - val_a
ccuracy: 0.6136 - val_loss: 0.7442 - val_precision: 0.6522 - val_recall: 0.6250
Epoch 5/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 662ms/step - accuracy: 0.5086 - loss: 0.8860 - precision: 0.5337 - recall: 0.6164 - val_a
ccuracy: 0.6591 - val_loss: 0.7384 - val_precision: 0.7143 - val_recall: 0.6250
Epoch 6/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 687ms/step - accuracy: 0.5481 - loss: 0.7622 - precision: 0.5899 - recall: 0.6189 - val_a
ccuracy: 0.6818 - val_loss: 0.7285 - val_precision: 0.7500 - val_recall: 0.6250
Epoch 7/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 5s 614ms/step - accuracy: 0.5517 - loss: 0.7562 - precision: 0.5924 - recall: 0.5761 - val_a
ccuracy: 0.6818 - val_loss: 0.7159 - val_precision: 0.7500 - val_recall: 0.6250
Epoch 8/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 723ms/step - accuracy: 0.5714 - loss: 0.7497 - precision: 0.5750 - recall: 0.6611 - val_a
ccuracy: 0.7045 - val_loss: 0.7051 - val_precision: 0.7895 - val_recall: 0.6250
Epoch 9/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 661ms/step - accuracy: 0.6290 - loss: 0.7140 - precision: 0.6623 - recall: 0.7064 - val_a
ccuracy: 0.7045 - val_loss: 0.6889 - val_precision: 0.7895 - val_recall: 0.6250
Epoch 10/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 634ms/step - accuracy: 0.6249 - loss: 0.6991 - precision: 0.6652 - recall: 0.6898 - val_a
ccuracy: 0.7045 - val_loss: 0.6730 - val_precision: 0.7895 - val_recall: 0.6250
Epoch 11/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 640ms/step - accuracy: 0.7063 - loss: 0.6320 - precision: 0.7287 - recall: 0.7840 - val_a
ccuracy: 0.7045 - val_loss: 0.6623 - val_precision: 0.7895 - val_recall: 0.6250
Epoch 12/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 643ms/step - accuracy: 0.6383 - loss: 0.6872 - precision: 0.6486 - recall: 0.6942 - val_a
ccuracy: 0.7045 - val_loss: 0.6545 - val_precision: 0.7895 - val_recall: 0.6250
Epoch 13/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 686ms/step - accuracy: 0.6367 - loss: 0.7142 - precision: 0.6761 - recall: 0.6545 - val_a
ccuracy: 0.7045 - val_loss: 0.6410 - val_precision: 0.7895 - val_recall: 0.6250
Epoch 14/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 658ms/step - accuracy: 0.5995 - loss: 0.7781 - precision: 0.6257 - recall: 0.6180 - val_a
ccuracy: 0.7045 - val_loss: 0.6266 - val_precision: 0.7895 - val_recall: 0.6250
Epoch 15/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 5s 774ms/step - accuracy: 0.6381 - loss: 0.6467 - precision: 0.6313 - recall: 0.6707 - val_a
```

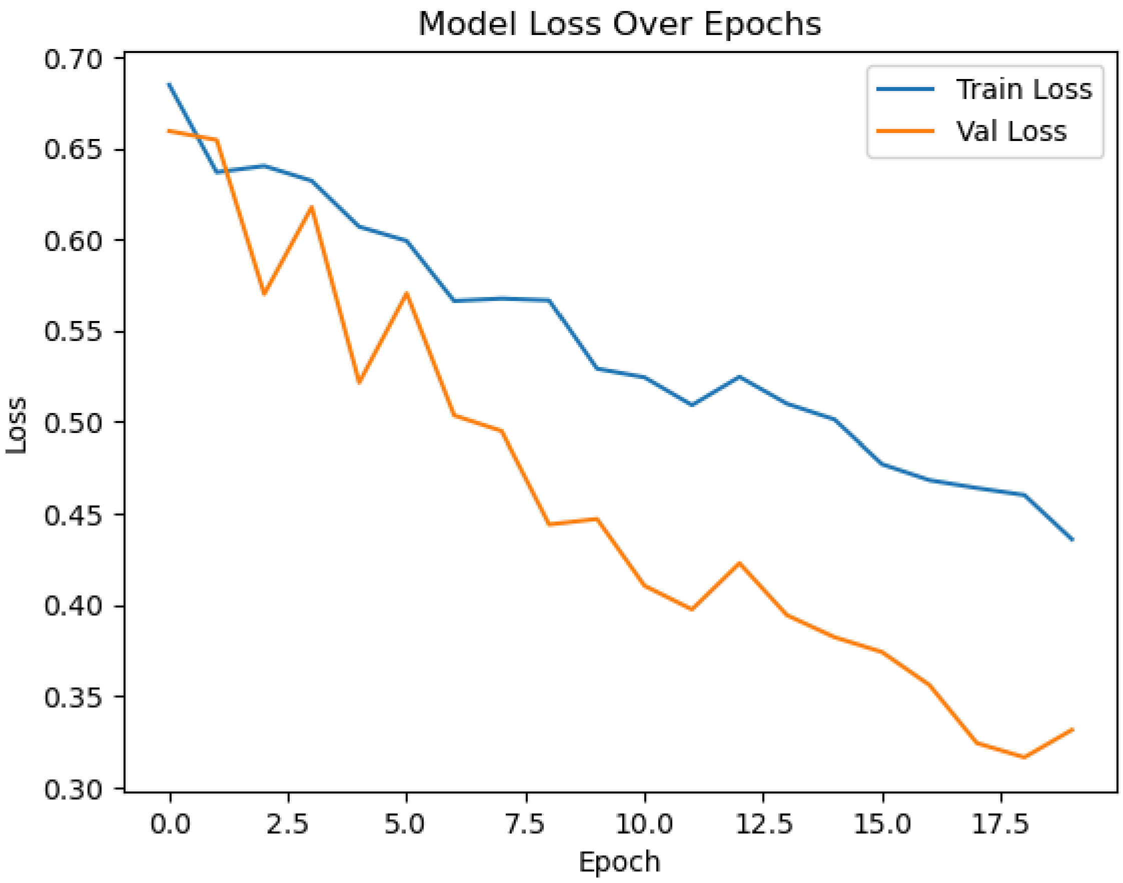
```
ccuracy: 0.7273 - val_loss: 0.6158 - val_precision: 0.8000 - val_recall: 0.6667
Epoch 16/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 632ms/step - accuracy: 0.6438 - loss: 0.6860 - precision: 0.6280 - recall: 0.7575 - val_a
ccuracy: 0.7273 - val_loss: 0.6069 - val_precision: 0.8000 - val_recall: 0.6667
Epoch 17/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 628ms/step - accuracy: 0.6916 - loss: 0.6782 - precision: 0.7242 - recall: 0.7250 - val_a
ccuracy: 0.7273 - val_loss: 0.5966 - val_precision: 0.8000 - val_recall: 0.6667
Epoch 18/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 578ms/step - accuracy: 0.6674 - loss: 0.6319 - precision: 0.6744 - recall: 0.6949 - val_a
ccuracy: 0.7273 - val_loss: 0.5860 - val_precision: 0.8000 - val_recall: 0.6667
Epoch 19/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 4s 624ms/step - accuracy: 0.7255 - loss: 0.5780 - precision: 0.7365 - recall: 0.7622 - val_a
ccuracy: 0.7273 - val_loss: 0.5783 - val_precision: 0.8000 - val_recall: 0.6667
Epoch 20/20
6/6 ━━━━━━━━━━━━━━━━━━━━ 3s 539ms/step - accuracy: 0.6738 - loss: 0.6444 - precision: 0.6991 - recall: 0.7318 - val_a
ccuracy: 0.7273 - val_loss: 0.5705 - val_precision: 0.8000 - val_recall: 0.6667
```

In [197…

```python
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report
val_generator.reset()
y_probs_m = model_mobilenet.predict(val_generator)
y_preds_m = (y_probs_m > 0.5).astype("int32").flatten()
y_true_m = val_generator.classes[:len(y_preds_m)]


# Compute
accuracy = accuracy_score(y_true_m, y_preds_m)
precision = precision_score(y_true_m, y_preds_m)
recall = recall_score(y_true_m, y_preds_m)
f1 = f1_score(y_true_m, y_preds_m)


# Print
print("MobileNetV2 Evaluation Metrics:")
print(f"Accuracy :  {accuracy:.4f}")
print(f"Precision:  {precision:.4f}")
print(f"Recall   :  {recall:.4f}")
print(f"F1 Score :  {f1:.4f}")


# Accuracy
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Val Accuracy')
plt.title('Model Accuracy Over Epochs')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

# Loss
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Val Loss')
plt.title('Model Loss Over Epochs')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

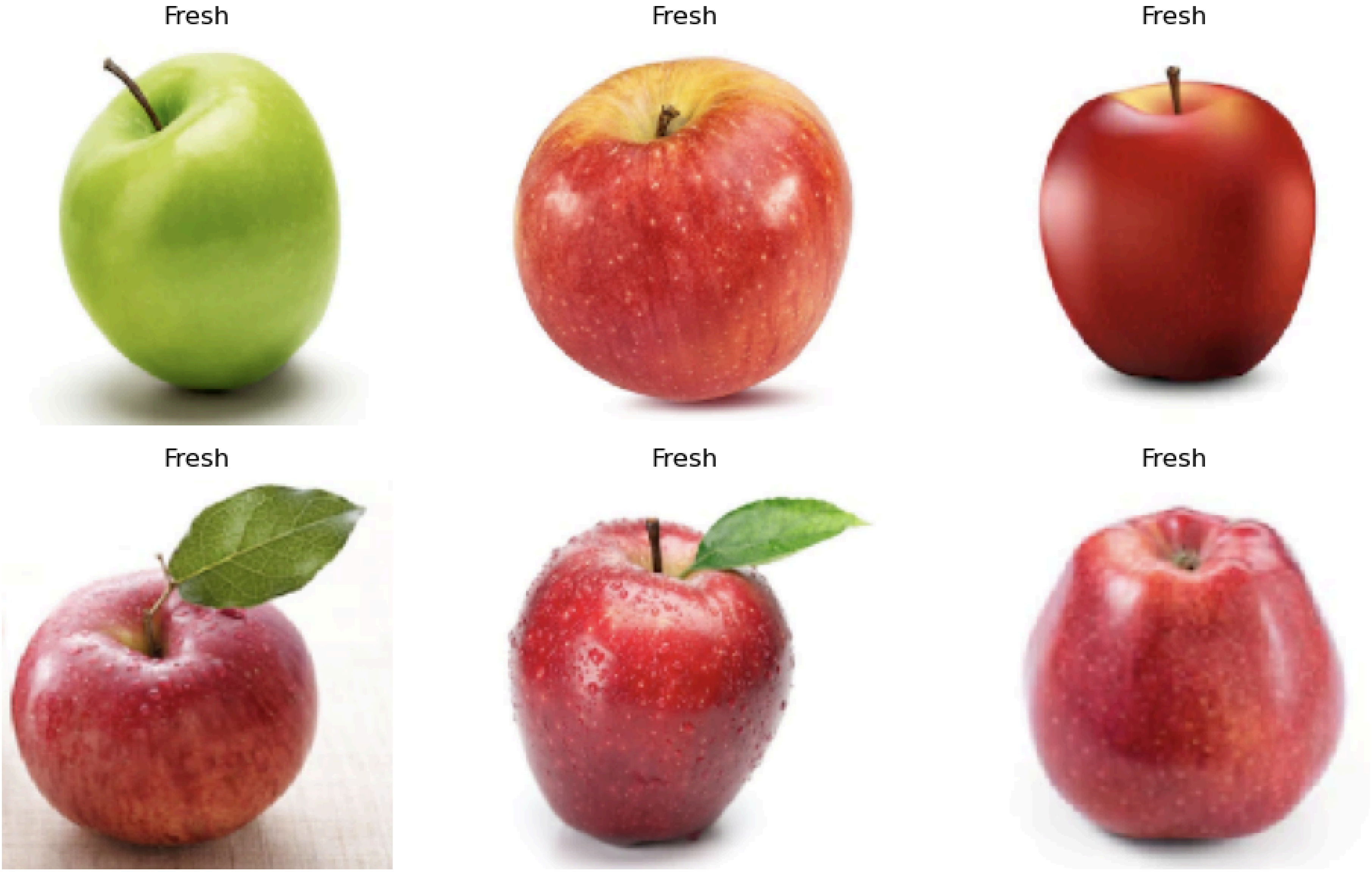
```
2/2 ━━━━━━━━━━━━━━━━━━━━ 3s 1s/step
MobileNetV2 Evaluation Metrics:
Accuracy :  0.7273
Precision:  0.8000
Recall   :  0.6667
F1 Score :  0.7273
```


Model Accuracy Over Epochs
Train Accuracy
Val Accuracy
Accuracy
0.6
0.7
0.8
0.9
Epoch
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5



In [198…

```
plt.figure(figsize=(10, 6))
for i in range(6):
    plt.subplot(2, 3, i + 1)
    plt.imshow(x_val[i])
    plt.title("Fresh" if y_val[i] == 0 else "Rotten")
    plt.axis('off')

plt.tight_layout()
plt.show()
```


Fresh
Fresh
Fresh
Fresh
Fresh
Fresh