In [1]:
```
import os
os.environ["CUDA_VISIBLE_DEVICES"] = "1"
```

In [2]:
```
import tensorflow as tf
gpus = tf.config.experimental.list_physical_devices('GPU')
for gpu in gpus:
    tf.config.experimental.set_memory_growth(gpu, True)
```

```
2025-10-26 17:22:30.955103: I tensorflow/core/util/port.cc:153] oneDNN custom ope
rations are on. You may see slightly different numerical results due to floating-
point round-off errors from different computation orders. To turn them off, set t
he environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-10-26 17:22:30.974909: E external/local_xla/xla/stream_executor/cuda/cuda_ff
t.cc:477] Unable to register cuFFT factory: Attempting to register factory for pl
ugin cuFFT when one has already been registered
WARNING: All log messages before absl::InitializeLog() is called are written to S
TDERR
E0000 00:00:1761474150.995019   44835 cuda_dnn.cc:8310] Unable to register cuDNN
factory: Attempting to register factory for plugin cuDNN when one has already bee
n registered
E0000 00:00:1761474151.001356   44835 cuda_blas.cc:1418] Unable to register cuBLA
S factory: Attempting to register factory for plugin cuBLAS when one has already
been registered
2025-10-26 17:22:31.022165: I tensorflow/core/platform/cpu_feature_guard.cc:210]
This TensorFlow binary is optimized to use available CPU instructions in performa
nce-critical operations.
To enable the following instructions: AVX2 AVX512F AVX512_VNNI AVX512_BF16 AVX512
_FP16 AVX_VNNI AMX_TILE AMX_INT8 AMX_BF16 FMA, in other operations, rebuild Tenso
rFlow with the appropriate compiler flags.
```

In [3]:
```
# IMPORTS & KONFIGURASI
import os
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import itertools
from sklearn.metrics import confusion_matrix, classification_report
import tensorflow as tf
from tensorflow.keras import layers, models, optimizers, callbacks

print("TensorFlow version:", tf.__version__)


BASE_DIR = "dataset"
TRAIN_DIR = os.path.join(BASE_DIR, "train")
VAL_DIR   = os.path.join(BASE_DIR, "validation")
TEST_DIR  = os.path.join(BASE_DIR, "test")

IMG_SIZE = 224
BATCH_SIZE = 16
EPOCHS = 20
AUTOTUNE = tf.data.AUTOTUNE
SEED = 42
```

```
TensorFlow version: 2.18.0
```

In [4]:
```
# Mengecek jumlah file per kelas
def count_images_in_dir(dirpath):
```

```
    counts = {}
    if not os.path.exists(dirpath):
        print(f"Directory not found: {dirpath}")
        return counts
    for cls in sorted(os.listdir(dirpath)):
        cls_path = os.path.join(dirpath, cls)
        if os.path.isdir(cls_path):
            counts[cls] = sum([1 for _ in os.listdir(cls_path) if os.path.isfile
    return counts

print("Train counts:", count_images_in_dir(TRAIN_DIR))
print("Validation counts:", count_images_in_dir(VAL_DIR))
print("Test counts:", count_images_in_dir(TEST_DIR))
```

```
Train counts: {'gudeg': 90, 'gulai': 80, 'rendang': 82, 'soto': 90}
Validation counts: {'gudeg': 10, 'gulai': 10, 'rendang': 10, 'soto': 10}
Test counts: {'gudeg': 20, 'gulai': 10, 'rendang': 10, 'soto': 10}
```

In [5]:

```
# LOAD DATASET
train_ds = tf.keras.utils.image_dataset_from_directory(
    TRAIN_DIR,
    labels="inferred",
    label_mode="int",
    image_size=(IMG_SIZE, IMG_SIZE),
    batch_size=BATCH_SIZE,
    shuffle=True,
    seed=SEED
)

val_ds = tf.keras.utils.image_dataset_from_directory(
    VAL_DIR,
    labels="inferred",
    label_mode="int",
    image_size=(IMG_SIZE, IMG_SIZE),
    batch_size=BATCH_SIZE,
    shuffle=False,
    seed=SEED
)

test_ds = tf.keras.utils.image_dataset_from_directory(
    TEST_DIR,
    labels="inferred",
    label_mode="int",
    image_size=(IMG_SIZE, IMG_SIZE),
    batch_size=BATCH_SIZE,
    shuffle=False
)

class_names = train_ds.class_names
num_classes = len(class_names)
print("Classes:", class_names)
```

```
Found 342 files belonging to 4 classes.
I0000 00:00:1761474157.319438   44835 gpu_device.cc:2022] Created device /job:loc
alhost/replica:0/task:0/device:GPU:0 with 25297 MB memory:  -> device: 0, name: N
VIDIA L40S, pci bus id: 0000:be:00.0, compute capability: 8.9
Found 40 files belonging to 4 classes.
Found 50 files belonging to 4 classes.
Classes: ['gudeg', 'gulai', 'rendang', 'soto']
```

In [6]:
```
# DATA AUGMENTATION & NORMALIZATION
from tensorflow.keras import layers

normalization_layer = layers.Rescaling(1./255)

data_augmentation = tf.keras.Sequential([
    layers.RandomFlip("horizontal"),
    layers.RandomRotation(0.12),
    layers.RandomZoom(0.08),
], name="data_augmentation")

def prepare_dataset(ds, training=False):
    ds = ds.map(lambda x,y: (normalization_layer(x), y), num_parallel_calls=AUTO
    if training:
        ds = ds.map(lambda x,y: (data_augmentation(x, training=True), y), num_pa
        ds = ds.shuffle(1000, seed=SEED)
    return ds.prefetch(buffer_size=AUTOTUNE)

train_ds_prepared = prepare_dataset(train_ds, training=True)
val_ds_prepared = prepare_dataset(val_ds, training=False)
test_ds_prepared = prepare_dataset(test_ds, training=False)

# Show sample images
plt.figure(figsize=(10,5))
for images, labels in train_ds.take(1):
    for i in range(6):
        ax = plt.subplot(2,3,i+1)
        img = images[i].numpy().astype("uint8")
        plt.imshow(img)
        plt.title(class_names[labels[i]])
        plt.axis("off")
plt.suptitle("Sampel")
plt.show()
```

```
2025-10-26 17:22:40.999749: I tensorflow/core/framework/local_rendezvous.cc:405]
Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```

In [7]:
```
# definisiin train
def plot_history(hist, title_prefix=""):
    df = pd.DataFrame(hist.history)
    df['epoch'] = range(1, len(df)+1)
    display(df[['epoch'] + [c for c in df.columns if 'accuracy' in c or 'loss' i
    # plot accuracy
    plt.figure(figsize=(12,4))
    plt.subplot(1,2,1)
    plt.plot(df['epoch'], df.get('accuracy'), label='train_acc')
    plt.plot(df['epoch'], df.get('val_accuracy'), label='val_acc')
    plt.xlabel('Epoch'); plt.ylabel('Accuracy'); plt.legend(); plt.title(title_p
    # plot loss
    plt.subplot(1,2,2)
    plt.plot(df['epoch'], df.get('loss'), label='train_loss')
    plt.plot(df['epoch'], df.get('val_loss'), label='val_loss')
    plt.xlabel('Epoch'); plt.ylabel('Loss'); plt.legend(); plt.title(title_prefi
    plt.show()
    return df
```

In [8]:
```
# ResNet50 (tanpa pretrained weights)
from tensorflow.keras.applications import ResNet50

def build_resnet50_custom(input_shape=(IMG_SIZE, IMG_SIZE, 3), num_classes=num_c
    base = ResNet50(include_top=False, weights=None, input_shape=input_shape, po
    x = base.output
    x = layers.Dense(256, activation='relu')(x)
    x = layers.Dropout(0.4)(x)
    outputs = layers.Dense(num_classes, activation='softmax')(x)
    model = models.Model(inputs=base.input, outputs=outputs, name="ResNet50_scra
    return model

resnet_model = build_resnet50_custom()
resnet_model.compile(optimizer=optimizers.Adam(learning_rate=1e-4), loss='sparse
resnet_model.summary()

checkpoint_cb_res = callbacks.ModelCheckpoint("resnet50_best.h5", save_best_only
earlystop_cb_res = callbacks.EarlyStopping(monitor='val_accuracy', patience=6, r

hist_resnet = resnet_model.fit(
    train_ds_prepared,
    validation_data=val_ds_prepared,
    epochs=EPOCHS,
    callbacks=[checkpoint_cb_res, earlystop_cb_res]
)

df_resnet = plot_history(hist_resnet, title_prefix="ResNet50 (scratch)")
df_resnet.to_csv("history_resnet50.csv", index=False)
resnet_model.save("resnet50_final.h5")
```

Model: "ResNet50_scratch"

Layer (type)	Output Shape	Param #	Connected to
input_layer_1 (InputLayer)	(None, 224, 224, 3)	0	-
conv1_pad (ZeroPadding2D)	(None, 230, 230, 3)	0	input_layer_1[0]…
conv1_conv (Conv2D)	(None, 112, 112, 64)	9,472	conv1_pad[0][0]
conv1_bn (BatchNormalizatio…	(None, 112, 112, 64)	256	conv1_conv[0][0]
conv1_relu (Activation)	(None, 112, 112, 64)	0	conv1_bn[0][0]
pool1_pad (ZeroPadding2D)	(None, 114, 114, 64)	0	conv1_relu[0][0]
pool1_pool (MaxPooling2D)	(None, 56, 56, 64)	0	pool1_pad[0][0]
conv2_block1_1_conv (Conv2D)	(None, 56, 56, 64)	4,160	pool1_pool[0][0]
conv2_block1_1_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block1_1_c…
conv2_block1_1_relu (Activation)	(None, 56, 56, 64)	0	conv2_block1_1_b…
conv2_block1_2_conv (Conv2D)	(None, 56, 56, 64)	36,928	conv2_block1_1_r…
conv2_block1_2_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block1_2_c…
conv2_block1_2_relu (Activation)	(None, 56, 56, 64)	0	conv2_block1_2_b…
conv2_block1_0_conv (Conv2D)	(None, 56, 56, 256)	16,640	pool1_pool[0][0]
conv2_block1_3_conv (Conv2D)	(None, 56, 56, 256)	16,640	conv2_block1_2_r…
conv2_block1_0_bn (BatchNormalizatio…	(None, 56, 56, 256)	1,024	conv2_block1_0_c…
conv2_block1_3_bn (BatchNormalizatio…	(None, 56, 56, 256)	1,024	conv2_block1_3_c…
conv2_block1_add (Add)	(None, 56, 56, 256)	0	conv2_block1_0_b… conv2_block1_3_b…
conv2_block1_out (Activation)	(None, 56, 56, 256)	0	conv2_block1_add…

conv2_block2_1_conv (Conv2D)	(None, 56, 56, 64)	16,448	conv2_block1_out…
conv2_block2_1_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block2_1_c…
conv2_block2_1_relu (Activation)	(None, 56, 56, 64)	0	conv2_block2_1_b…
conv2_block2_2_conv (Conv2D)	(None, 56, 56, 64)	36,928	conv2_block2_1_r…
conv2_block2_2_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block2_2_c…
conv2_block2_2_relu (Activation)	(None, 56, 56, 64)	0	conv2_block2_2_b…
conv2_block2_3_conv (Conv2D)	(None, 56, 56, 256)	16,640	conv2_block2_2_r…
conv2_block2_3_bn (BatchNormalizatio…	(None, 56, 56, 256)	1,024	conv2_block2_3_c…
conv2_block2_add (Add)	(None, 56, 56, 256)	0	conv2_block1_out… conv2_block2_3_b…
conv2_block2_out (Activation)	(None, 56, 56, 256)	0	conv2_block2_add…
conv2_block3_1_conv (Conv2D)	(None, 56, 56, 64)	16,448	conv2_block2_out…
conv2_block3_1_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block3_1_c…
conv2_block3_1_relu (Activation)	(None, 56, 56, 64)	0	conv2_block3_1_b…
conv2_block3_2_conv (Conv2D)	(None, 56, 56, 64)	36,928	conv2_block3_1_r…
conv2_block3_2_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block3_2_c…
conv2_block3_2_relu (Activation)	(None, 56, 56, 64)	0	conv2_block3_2_b…
conv2_block3_3_conv (Conv2D)	(None, 56, 56, 256)	16,640	conv2_block3_2_r…
conv2_block3_3_bn (BatchNormalizatio…	(None, 56, 56, 256)	1,024	conv2_block3_3_c…
conv2_block3_add (Add)	(None, 56, 56, 256)	0	conv2_block2_out… conv2_block3_3_b…
conv2_block3_out (Activation)	(None, 56, 56, 256)	0	conv2_block3_add…

conv3_block1_1_conv (Conv2D)	(None, 28, 28, 128)	32,896	conv2_block3_out…
conv3_block1_1_bn (BatchNormalizatio…	(None, 28, 28, 128)	512	conv3_block1_1_c…
conv3_block1_1_relu (Activation)	(None, 28, 28, 128)	0	conv3_block1_1_b…
conv3_block1_2_conv (Conv2D)	(None, 28, 28, 128)	147,584	conv3_block1_1_r…
conv3_block1_2_bn (BatchNormalizatio…	(None, 28, 28, 128)	512	conv3_block1_2_c…
conv3_block1_2_relu (Activation)	(None, 28, 28, 128)	0	conv3_block1_2_b…
conv3_block1_0_conv (Conv2D)	(None, 28, 28, 512)	131,584	conv2_block3_out…
conv3_block1_3_conv (Conv2D)	(None, 28, 28, 512)	66,048	conv3_block1_2_r…
conv3_block1_0_bn (BatchNormalizatio…	(None, 28, 28, 512)	2,048	conv3_block1_0_c…
conv3_block1_3_bn (BatchNormalizatio…	(None, 28, 28, 512)	2,048	conv3_block1_3_c…
conv3_block1_add (Add)	(None, 28, 28, 512)	0	conv3_block1_0_b… conv3_block1_3_b…
conv3_block1_out (Activation)	(None, 28, 28, 512)	0	conv3_block1_add…
conv3_block2_1_conv (Conv2D)	(None, 28, 28, 128)	65,664	conv3_block1_out…
conv3_block2_1_bn (BatchNormalizatio…	(None, 28, 28, 128)	512	conv3_block2_1_c…
conv3_block2_1_relu (Activation)	(None, 28, 28, 128)	0	conv3_block2_1_b…
conv3_block2_2_conv (Conv2D)	(None, 28, 28, 128)	147,584	conv3_block2_1_r…
conv3_block2_2_bn (BatchNormalizatio…	(None, 28, 28, 128)	512	conv3_block2_2_c…
conv3_block2_2_relu (Activation)	(None, 28, 28, 128)	0	conv3_block2_2_b…
conv3_block2_3_conv (Conv2D)	(None, 28, 28, 512)	66,048	conv3_block2_2_r…
conv3_block2_3_bn (BatchNormalizatio…	(None, 28, 28, 512)	2,048	conv3_block2_3_c…

conv3_block2_add (Add)	(None, 28, 28, 512)	0	conv3_block1_out… conv3_block2_3_b…
conv3_block2_out (Activation)	(None, 28, 28, 512)	0	conv3_block2_add…
conv3_block3_1_conv (Conv2D)	(None, 28, 28, 128)	65,664	conv3_block2_out…
conv3_block3_1_bn (BatchNormalizatio…	(None, 28, 28, 128)	512	conv3_block3_1_c…
conv3_block3_1_relu (Activation)	(None, 28, 28, 128)	0	conv3_block3_1_b…
conv3_block3_2_conv (Conv2D)	(None, 28, 28, 128)	147,584	conv3_block3_1_r…
conv3_block3_2_bn (BatchNormalizatio…	(None, 28, 28, 128)	512	conv3_block3_2_c…
conv3_block3_2_relu (Activation)	(None, 28, 28, 128)	0	conv3_block3_2_b…
conv3_block3_3_conv (Conv2D)	(None, 28, 28, 512)	66,048	conv3_block3_2_r…
conv3_block3_3_bn (BatchNormalizatio…	(None, 28, 28, 512)	2,048	conv3_block3_3_c…
conv3_block3_add (Add)	(None, 28, 28, 512)	0	conv3_block2_out… conv3_block3_3_b…
conv3_block3_out (Activation)	(None, 28, 28, 512)	0	conv3_block3_add…
conv3_block4_1_conv (Conv2D)	(None, 28, 28, 128)	65,664	conv3_block3_out…
conv3_block4_1_bn (BatchNormalizatio…	(None, 28, 28, 128)	512	conv3_block4_1_c…
conv3_block4_1_relu (Activation)	(None, 28, 28, 128)	0	conv3_block4_1_b…
conv3_block4_2_conv (Conv2D)	(None, 28, 28, 128)	147,584	conv3_block4_1_r…
conv3_block4_2_bn (BatchNormalizatio…	(None, 28, 28, 128)	512	conv3_block4_2_c…
conv3_block4_2_relu (Activation)	(None, 28, 28, 128)	0	conv3_block4_2_b…
conv3_block4_3_conv (Conv2D)	(None, 28, 28, 512)	66,048	conv3_block4_2_r…
conv3_block4_3_bn (BatchNormalizatio…	(None, 28, 28, 512)	2,048	conv3_block4_3_c…

conv3_block4_add (Add)	(None, 28, 28, 512)	0	conv3_block3_out… conv3_block4_3_b…
conv3_block4_out (Activation)	(None, 28, 28, 512)	0	conv3_block4_add…
conv4_block1_1_conv (Conv2D)	(None, 14, 14, 256)	131,328	conv3_block4_out…
conv4_block1_1_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block1_1_c…
conv4_block1_1_relu (Activation)	(None, 14, 14, 256)	0	conv4_block1_1_b…
conv4_block1_2_conv (Conv2D)	(None, 14, 14, 256)	590,080	conv4_block1_1_r…
conv4_block1_2_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block1_2_c…
conv4_block1_2_relu (Activation)	(None, 14, 14, 256)	0	conv4_block1_2_b…
conv4_block1_0_conv (Conv2D)	(None, 14, 14, 1024)	525,312	conv3_block4_out…
conv4_block1_3_conv (Conv2D)	(None, 14, 14, 1024)	263,168	conv4_block1_2_r…
conv4_block1_0_bn (BatchNormalizatio…	(None, 14, 14, 1024)	4,096	conv4_block1_0_c…
conv4_block1_3_bn (BatchNormalizatio…	(None, 14, 14, 1024)	4,096	conv4_block1_3_c…
conv4_block1_add (Add)	(None, 14, 14, 1024)	0	conv4_block1_0_b… conv4_block1_3_b…
conv4_block1_out (Activation)	(None, 14, 14, 1024)	0	conv4_block1_add…
conv4_block2_1_conv (Conv2D)	(None, 14, 14, 256)	262,400	conv4_block1_out…
conv4_block2_1_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block2_1_c…
conv4_block2_1_relu (Activation)	(None, 14, 14, 256)	0	conv4_block2_1_b…
conv4_block2_2_conv (Conv2D)	(None, 14, 14, 256)	590,080	conv4_block2_1_r…
conv4_block2_2_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block2_2_c…
conv4_block2_2_relu (Activation)	(None, 14, 14, 256)	0	conv4_block2_2_b…

conv4_block2_3_conv (Conv2D)	(None, 14, 14, 1024)	263,168	conv4_block2_2_r…
conv4_block2_3_bn (BatchNormalizatio…	(None, 14, 14, 1024)	4,096	conv4_block2_3_c…
conv4_block2_add (Add)	(None, 14, 14, 1024)	0	conv4_block1_out… conv4_block2_3_b…
conv4_block2_out (Activation)	(None, 14, 14, 1024)	0	conv4_block2_add…
conv4_block3_1_conv (Conv2D)	(None, 14, 14, 256)	262,400	conv4_block2_out…
conv4_block3_1_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block3_1_c…
conv4_block3_1_relu (Activation)	(None, 14, 14, 256)	0	conv4_block3_1_b…
conv4_block3_2_conv (Conv2D)	(None, 14, 14, 256)	590,080	conv4_block3_1_r…
conv4_block3_2_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block3_2_c…
conv4_block3_2_relu (Activation)	(None, 14, 14, 256)	0	conv4_block3_2_b…
conv4_block3_3_conv (Conv2D)	(None, 14, 14, 1024)	263,168	conv4_block3_2_r…
conv4_block3_3_bn (BatchNormalizatio…	(None, 14, 14, 1024)	4,096	conv4_block3_3_c…
conv4_block3_add (Add)	(None, 14, 14, 1024)	0	conv4_block2_out… conv4_block3_3_b…
conv4_block3_out (Activation)	(None, 14, 14, 1024)	0	conv4_block3_add…
conv4_block4_1_conv (Conv2D)	(None, 14, 14, 256)	262,400	conv4_block3_out…
conv4_block4_1_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block4_1_c…
conv4_block4_1_relu (Activation)	(None, 14, 14, 256)	0	conv4_block4_1_b…
conv4_block4_2_conv (Conv2D)	(None, 14, 14, 256)	590,080	conv4_block4_1_r…
conv4_block4_2_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block4_2_c…
conv4_block4_2_relu (Activation)	(None, 14, 14, 256)	0	conv4_block4_2_b…

conv4_block4_3_conv (Conv2D)	(None, 14, 14, 1024)	263,168	conv4_block4_2_r…
conv4_block4_3_bn (BatchNormalizatio…	(None, 14, 14, 1024)	4,096	conv4_block4_3_c…
conv4_block4_add (Add)	(None, 14, 14, 1024)	0	conv4_block3_out… conv4_block4_3_b…
conv4_block4_out (Activation)	(None, 14, 14, 1024)	0	conv4_block4_add…
conv4_block5_1_conv (Conv2D)	(None, 14, 14, 256)	262,400	conv4_block4_out…
conv4_block5_1_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block5_1_c…
conv4_block5_1_relu (Activation)	(None, 14, 14, 256)	0	conv4_block5_1_b…
conv4_block5_2_conv (Conv2D)	(None, 14, 14, 256)	590,080	conv4_block5_1_r…
conv4_block5_2_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block5_2_c…
conv4_block5_2_relu (Activation)	(None, 14, 14, 256)	0	conv4_block5_2_b…
conv4_block5_3_conv (Conv2D)	(None, 14, 14, 1024)	263,168	conv4_block5_2_r…
conv4_block5_3_bn (BatchNormalizatio…	(None, 14, 14, 1024)	4,096	conv4_block5_3_c…
conv4_block5_add (Add)	(None, 14, 14, 1024)	0	conv4_block4_out… conv4_block5_3_b…
conv4_block5_out (Activation)	(None, 14, 14, 1024)	0	conv4_block5_add…
conv4_block6_1_conv (Conv2D)	(None, 14, 14, 256)	262,400	conv4_block5_out…
conv4_block6_1_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block6_1_c…
conv4_block6_1_relu (Activation)	(None, 14, 14, 256)	0	conv4_block6_1_b…
conv4_block6_2_conv (Conv2D)	(None, 14, 14, 256)	590,080	conv4_block6_1_r…
conv4_block6_2_bn (BatchNormalizatio…	(None, 14, 14, 256)	1,024	conv4_block6_2_c…
conv4_block6_2_relu (Activation)	(None, 14, 14, 256)	0	conv4_block6_2_b…

conv4_block6_3_conv (Conv2D)	(None, 14, 14, 1024)	263,168	conv4_block6_2_r…
conv4_block6_3_bn (BatchNormalizatio…	(None, 14, 14, 1024)	4,096	conv4_block6_3_c…
conv4_block6_add (Add)	(None, 14, 14, 1024)	0	conv4_block5_out… conv4_block6_3_b…
conv4_block6_out (Activation)	(None, 14, 14, 1024)	0	conv4_block6_add…
conv5_block1_1_conv (Conv2D)	(None, 7, 7, 512)	524,800	conv4_block6_out…
conv5_block1_1_bn (BatchNormalizatio…	(None, 7, 7, 512)	2,048	conv5_block1_1_c…
conv5_block1_1_relu (Activation)	(None, 7, 7, 512)	0	conv5_block1_1_b…
conv5_block1_2_conv (Conv2D)	(None, 7, 7, 512)	2,359,808	conv5_block1_1_r…
conv5_block1_2_bn (BatchNormalizatio…	(None, 7, 7, 512)	2,048	conv5_block1_2_c…
conv5_block1_2_relu (Activation)	(None, 7, 7, 512)	0	conv5_block1_2_b…
conv5_block1_0_conv (Conv2D)	(None, 7, 7, 2048)	2,099,200	conv4_block6_out…
conv5_block1_3_conv (Conv2D)	(None, 7, 7, 2048)	1,050,624	conv5_block1_2_r…
conv5_block1_0_bn (BatchNormalizatio…	(None, 7, 7, 2048)	8,192	conv5_block1_0_c…
conv5_block1_3_bn (BatchNormalizatio…	(None, 7, 7, 2048)	8,192	conv5_block1_3_c…
conv5_block1_add (Add)	(None, 7, 7, 2048)	0	conv5_block1_0_b… conv5_block1_3_b…
conv5_block1_out (Activation)	(None, 7, 7, 2048)	0	conv5_block1_add…
conv5_block2_1_conv (Conv2D)	(None, 7, 7, 512)	1,049,088	conv5_block1_out…
conv5_block2_1_bn (BatchNormalizatio…	(None, 7, 7, 512)	2,048	conv5_block2_1_c…
conv5_block2_1_relu (Activation)	(None, 7, 7, 512)	0	conv5_block2_1_b…
conv5_block2_2_conv (Conv2D)	(None, 7, 7, 512)	2,359,808	conv5_block2_1_r…

conv5_block2_2_bn (BatchNormalizatio…	(None, 7, 7, 512)	2,048	conv5_block2_2_c…
conv5_block2_2_relu (Activation)	(None, 7, 7, 512)	0	conv5_block2_2_b…
conv5_block2_3_conv (Conv2D)	(None, 7, 7, 2048)	1,050,624	conv5_block2_2_r…
conv5_block2_3_bn (BatchNormalizatio…	(None, 7, 7, 2048)	8,192	conv5_block2_3_c…
conv5_block2_add (Add)	(None, 7, 7, 2048)	0	conv5_block1_out… conv5_block2_3_b…
conv5_block2_out (Activation)	(None, 7, 7, 2048)	0	conv5_block2_add…
conv5_block3_1_conv (Conv2D)	(None, 7, 7, 512)	1,049,088	conv5_block2_out…
conv5_block3_1_bn (BatchNormalizatio…	(None, 7, 7, 512)	2,048	conv5_block3_1_c…
conv5_block3_1_relu (Activation)	(None, 7, 7, 512)	0	conv5_block3_1_b…
conv5_block3_2_conv (Conv2D)	(None, 7, 7, 512)	2,359,808	conv5_block3_1_r…
conv5_block3_2_bn (BatchNormalizatio…	(None, 7, 7, 512)	2,048	conv5_block3_2_c…
conv5_block3_2_relu (Activation)	(None, 7, 7, 512)	0	conv5_block3_2_b…
conv5_block3_3_conv (Conv2D)	(None, 7, 7, 2048)	1,050,624	conv5_block3_2_r…
conv5_block3_3_bn (BatchNormalizatio…	(None, 7, 7, 2048)	8,192	conv5_block3_3_c…
conv5_block3_add (Add)	(None, 7, 7, 2048)	0	conv5_block2_out… conv5_block3_3_b…
conv5_block3_out (Activation)	(None, 7, 7, 2048)	0	conv5_block3_add…
avg_pool (GlobalAveragePool…	(None, 2048)	0	conv5_block3_out…
dense (Dense)	(None, 256)	524,544	avg_pool[0][0]
dropout (Dropout)	(None, 256)	0	dense[0][0]
dense_1 (Dense)	(None, 4)	1,028	dropout[0][0]

Total params: 24,113,284 (91.98 MB)

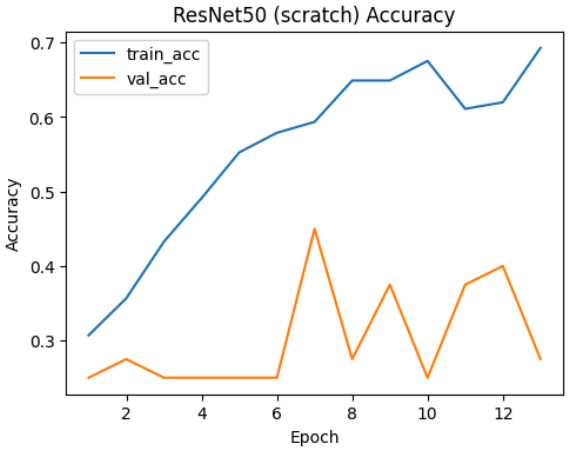
Trainable params: 24,060,164 (91.78 MB)

Non-trainable params: 53,120 (207.50 KB)

```
Epoch 1/20
WARNING: All log messages before absl::InitializeLog() is called are written to S
TDERR
I0000 00:00:1761474213.590941   45161 service.cc:148] XLA service 0x7f0420002280
initialized for platform CUDA (this does not guarantee that XLA will be used). De
vices:
I0000 00:00:1761474213.590971   45161 service.cc:156]   StreamExecutor device
(0): NVIDIA L40S, Compute Capability 8.9
2025-10-26 17:23:34.381166: I tensorflow/compiler/mlir/tensorflow/utils/dump_mlir
_util.cc:268] disabling MLIR crash reproducer, set env var `MLIR_CRASH_REPRODUCER
_DIRECTORY` to enable.
I0000 00:00:1761474217.590679   45161 cuda_dnn.cc:529] Loaded cuDNN version 91002
 3/22 ━━━━━━━━━━━━━━━━━━━━ 0s 46ms/step - accuracy: 0.3507 - loss: 1.7556
I0000 00:00:1761474232.974295   45161 device_compiler.h:188] Compiled cluster usi
ng XLA!  This line is logged at most once for the lifetime of the process.
12/22 ━━━━━━━━━━━━━━━━━━━━ 0s 47ms/step - accuracy: 0.3122 - loss: 1.9623
2025-10-26 17:23:59.316924: I external/local_xla/xla/stream_executor/cuda/cuda_as
m_compiler.cc:397] ptxas warning : Registers are spilled to local memory in funct
ion 'gemm_fusion_dot_16074_0', 8 bytes spill stores, 8 bytes spill loads

2025-10-26 17:24:14.192784: I external/local_xla/xla/stream_executor/cuda/cuda_as
m_compiler.cc:397] ptxas warning : Registers are spilled to local memory in funct
ion 'input_add_reduce_fusion_36', 416 bytes spill stores, 416 bytes spill loads

21/22 ━━━━━━━━━━━━━━━━━━━━ 1s 1s/step - accuracy: 0.3150 - loss: 1.9535
2025-10-26 17:24:21.557630: I external/local_xla/xla/stream_executor/cuda/cuda_as
m_compiler.cc:397] ptxas warning : Registers are spilled to local memory in funct
ion 'gemm_fusion_dot_1698_0', 8 bytes spill stores, 8 bytes spill loads

WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `ker
as.saving.save_model(model)`. This file format is considered legacy. We recommend
using instead the native Keras format, e.g. `model.save('my_model.keras')` or `ke
ras.saving.save_model(model, 'my_model.keras')`.
22/22 ━━━━━━━━━━━━━━━━━━━━ 73s 1s/step - accuracy: 0.3143 - loss: 1.9499 - val_ac
curacy: 0.2500 - val_loss: 1.3892
Epoch 2/20
21/22 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step - accuracy: 0.3198 - loss: 1.6514
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `ker
as.saving.save_model(model)`. This file format is considered legacy. We recommend
using instead the native Keras format, e.g. `model.save('my_model.keras')` or `ke
ras.saving.save_model(model, 'my_model.keras')`.
```

```
22/22 ━━━━━━━━━━━━━━━━━━━━ 3s 131ms/step - accuracy: 0.3230 - loss: 1.6371 - val_
accuracy: 0.2750 - val_loss: 1.3934
Epoch 3/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 39ms/step - accuracy: 0.4182 - loss: 1.4922 - val_a
ccuracy: 0.2500 - val_loss: 1.4392
Epoch 4/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 39ms/step - accuracy: 0.4587 - loss: 1.2061 - val_a
ccuracy: 0.2500 - val_loss: 1.6254
Epoch 5/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 40ms/step - accuracy: 0.5773 - loss: 0.9557 - val_a
ccuracy: 0.2500 - val_loss: 1.9654
Epoch 6/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 43ms/step - accuracy: 0.6007 - loss: 0.9440 - val_a
ccuracy: 0.2500 - val_loss: 2.1462
Epoch 7/20
21/22 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step - accuracy: 0.6157 - loss: 0.9004
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `ker
as.saving.save_model(model)`. This file format is considered legacy. We recommend
using instead the native Keras format, e.g. `model.save('my_model.keras')` or `ke
ras.saving.save_model(model, 'my_model.keras')`.
22/22 ━━━━━━━━━━━━━━━━━━━━ 3s 128ms/step - accuracy: 0.6138 - loss: 0.9028 - val_
accuracy: 0.4500 - val_loss: 1.3961
Epoch 8/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 36ms/step - accuracy: 0.6608 - loss: 0.7619 - val_a
ccuracy: 0.2750 - val_loss: 1.4512
Epoch 9/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 43ms/step - accuracy: 0.6425 - loss: 0.8006 - val_a
ccuracy: 0.3750 - val_loss: 2.0123
Epoch 10/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 40ms/step - accuracy: 0.6948 - loss: 0.7095 - val_a
ccuracy: 0.2500 - val_loss: 2.5896
Epoch 11/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.6097 - loss: 1.0463 - val_a
ccuracy: 0.3750 - val_loss: 2.1869
Epoch 12/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 48ms/step - accuracy: 0.6816 - loss: 0.8134 - val_a
ccuracy: 0.4000 - val_loss: 2.3752
Epoch 13/20
22/22 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.6727 - loss: 0.7906 - val_a
ccuracy: 0.2750 - val_loss: 3.3538
```

	epoch	accuracy	loss	val_accuracy	val_loss
0	1	0.307018	1.911539	0.250	1.389237
1	2	0.356725	1.487802	0.275	1.393383
2	3	0.432749	1.412195	0.250	1.439188
3	4	0.491228	1.137691	0.250	1.625364
4	5	0.552632	0.965986	0.250	1.965444
5	6	0.578947	0.992301	0.250	2.146234
6	7	0.593567	0.927367	0.450	1.396112
7	8	0.649123	0.816250	0.275	1.451221
8	9	0.649123	0.805432	0.375	2.012328
9	10	0.675439	0.793351	0.250	2.589563
10	11	0.611111	0.995137	0.375	2.186922
11	12	0.619883	0.901486	0.400	2.375246
12	13	0.692982	0.774684	0.275	3.353831




```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `ker
as.saving.save_model(model)`. This file format is considered legacy. We recommend
using instead the native Keras format, e.g. `model.save('my_model.keras')` or `ke
ras.saving.save_model(model, 'my_model.keras')`.
```

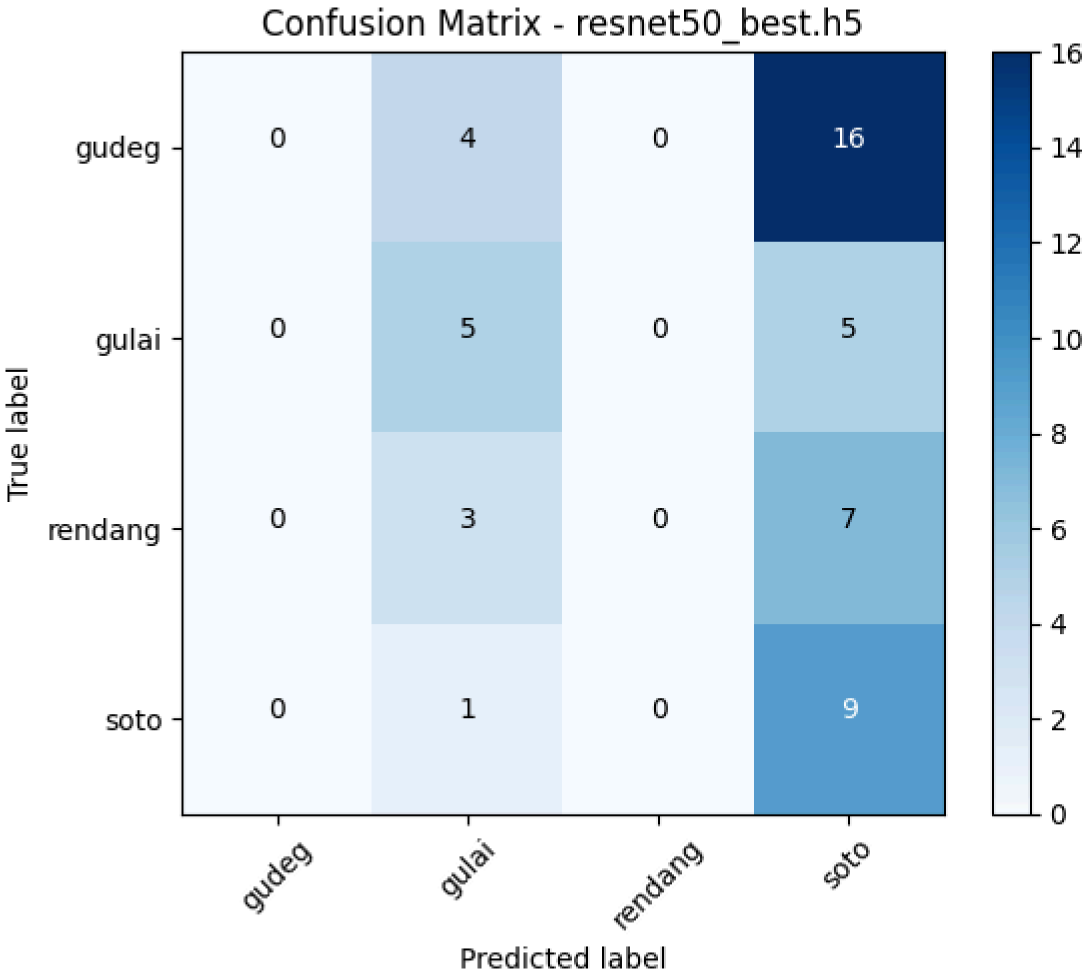
In [9]:
```python
# EVALUASI PADA TEST SET
def evaluate_and_confusion_matrix(model_path, ds, class_names):
    if not os.path.exists(model_path):
        print("Model file not found:", model_path)
        return None
    model = tf.keras.models.load_model(model_path)
    loss, acc = model.evaluate(ds)
    print(f"Model {model_path} -> loss: {loss:.4f}, acc: {acc:.4f}")
    y_true = []
    y_pred = []
    for images, labels in ds:
        probs = model.predict(images)
        preds = np.argmax(probs, axis=1)
        y_true.extend(labels.numpy().tolist())
        y_pred.extend(preds.tolist())
```

```
    cm = confusion_matrix(y_true, y_pred)
    print("Classification Report:\n", classification_report(y_true, y_pred, targ
    return cm
```

In [11]:
```
for mfile in ["resnet50_best.h5"]:
    if os.path.exists(mfile):
        cm = evaluate_and_confusion_matrix(mfile, test_ds_prepared, class_names)
        if cm is not None:
            plt.figure(figsize=(6,5))
            plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)
            plt.title(f'Confusion Matrix - {mfile}')
            plt.colorbar()
            tick_marks = np.arange(len(class_names))
            plt.xticks(tick_marks, class_names, rotation=45)
            plt.yticks(tick_marks, class_names)
            thresh = cm.max() / 2.
            for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])
                plt.text(j, i, cm[i, j],
                         horizontalalignment="center",
                         color="white" if cm[i, j] > thresh else "black")
            plt.ylabel('True label')
            plt.xlabel('Predicted label')
            plt.tight_layout()
            plt.show()
    else:
        print("File not found:", mfile)
```

```
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be b
uilt. `model.compile_metrics` will be empty until you train or evaluate the mode
l.
4/4 ━━━━━━━━━━━━━━━━━━━━ 5s 418ms/step - accuracy: 0.1974 - loss: 1.4404
Model resnet50_best.h5 -> loss: 1.4227, acc: 0.2800
1/1 ━━━━━━━━━━━━━━━━━━━━ 3s 3s/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 54ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step
Classification Report:
              precision    recall  f1-score   support

       gudeg       0.00      0.00      0.00        20
       gulai       0.38      0.50      0.43        10
     rendang       0.00      0.00      0.00        10
        soto       0.24      0.90      0.38        10

    accuracy                           0.28        50
   macro avg       0.16      0.35      0.20        50
weighted avg       0.13      0.28      0.16        50
```

```
/opt/tljh/user/envs/dltf/lib/python3.12/site-packages/sklearn/metrics/_classifica
tion.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/opt/tljh/user/envs/dltf/lib/python3.12/site-packages/sklearn/metrics/_classifica
tion.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/opt/tljh/user/envs/dltf/lib/python3.12/site-packages/sklearn/metrics/_classifica
tion.py:1565: UndefinedMetricWarning: Precision is ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```



In [12]:
```
MODEL_SAVE_PATH = "resnet50_best.h5"
resnet_model.save(MODEL_SAVE_PATH)
print(f"Model tersimpan di: {MODEL_SAVE_PATH}")
```

```
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `ker
as.saving.save_model(model)`. This file format is considered legacy. We recommend
using instead the native Keras format, e.g. `model.save('my_model.keras')` or `ke
ras.saving.save_model(model, 'my_model.keras')`.
Model tersimpan di: resnet50_best.h5
```

In [17]:
```
import matplotlib.pyplot as plt
import numpy as np
import random
from tqdm import tqdm
import mitdeeplearning as mdl
import tensorflow as tf
from tensorflow.keras import layers, models
import os
from pathlib import Path
```

In [18]:
```
from pathlib import Path
from collections import defaultdict

base_dir = Path("dataset")
splits = ["train", "validation", "test"]

for split in splits:
    split_dir = base_dir / split
    print(f"\n📁 {split_dir.name.upper()}")
    for cls in sorted(os.listdir(split_dir)):
        class_dir = split_dir / cls
        n = len([f for f in class_dir.iterdir() if f.is_file()])
        print(f"  {cls:<10}: {n} file")
```

```
📁 TRAIN
  .ipynb_checkpoints: 0 file
  gudeg     : 80 file
  gulai     : 80 file
  rendang   : 80 file
  soto      : 80 file

📁 VALIDATION
  .ipynb_checkpoints: 0 file
  gudeg     : 10 file
  gulai     : 10 file
  rendang   : 10 file
  soto      : 10 file

📁 TEST
  .ipynb_checkpoints: 0 file
  gudeg     : 10 file
  gulai     : 10 file
  rendang   : 10 file
  soto      : 10 file
```

In [19]:
```
train_dir = os.path.join(base_dir, "train")
test_dir  = os.path.join(base_dir, "test")
val_dir   = os.path.join(base_dir, "validation")

# Muat dataset dari folder
train_ds = tf.keras.utils.image_dataset_from_directory(
    train_dir,
    image_size=(224, 224),
    batch_size=32,
    shuffle=True
)

test_ds = tf.keras.utils.image_dataset_from_directory(
```

```
    test_dir,
    image_size=(224, 224),
    batch_size=10,
    shuffle=False
)

val_ds = tf.keras.utils.image_dataset_from_directory(
    val_dir,
    image_size=(224, 224),
    batch_size=10,
    shuffle=False
)


class_names = train_ds.class_names
print("Kelas:", class_names)
data_augmentation = tf.keras.Sequential([
    layers.RandomFlip("horizontal"),
    layers.RandomRotation(0.2),
    layers.RandomZoom(0.15),
    layers.RandomTranslation(0.1, 0.1),
    layers.RandomContrast(0.2),
    layers.RandomBrightness(factor=0.2),
])
train_ds = train_ds.map(lambda x, y: (data_augmentation(x, training=True)/255.0,
val_ds = val_ds.map(lambda x, y: (x/255.0, y))
test_ds = test_ds.map(lambda x, y: (x/255.0, y))
AUTOTUNE = tf.data.AUTOTUNE

train_ds = train_ds.cache().prefetch(AUTOTUNE)
val_ds   = val_ds.cache().prefetch(AUTOTUNE)
test_ds  = test_ds.cache().prefetch(AUTOTUNE)
```

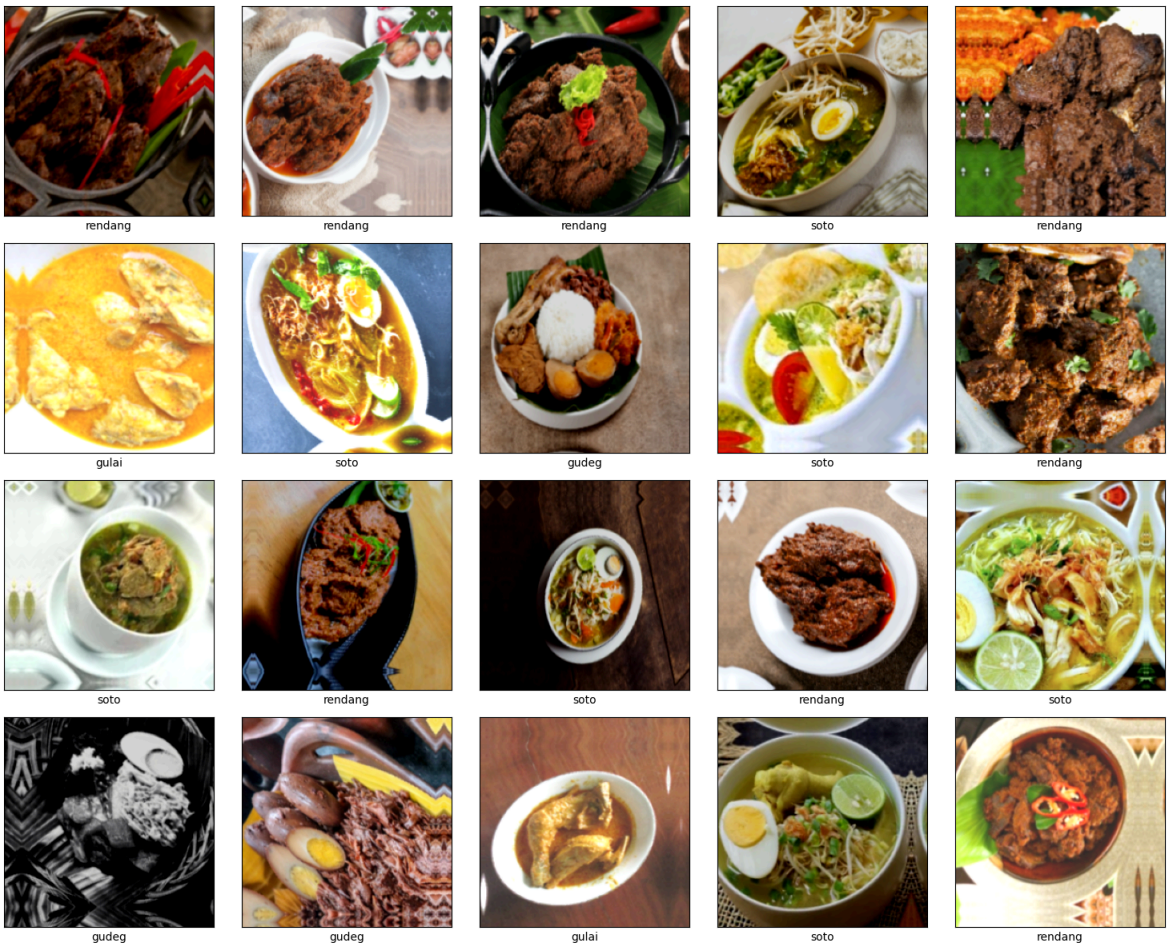
```
Found 322 files belonging to 4 classes.
Found 40 files belonging to 4 classes.
Found 40 files belonging to 4 classes.
Kelas: ['gudeg', 'gulai', 'rendang', 'soto']
```

In [20]:

```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf

def visualize(dataset, num_images=20):
    for images, labels in dataset.take(1):
        n = int(min(num_images, images.shape[0]))
        idxs = np.random.choice(images.shape[0], n, replace=False)

        # grid 4x5 untuk default 20; kalau n beda, bikin grid dinamis
        rows = 4 if n >= 16 else int(np.ceil(n/5))
        cols = 5 if n >= 5  else n

        plt.figure(figsize=(3*cols, 3*rows))
        for i, idx in enumerate(idxs):
            plt.subplot(rows, cols, i + 1)
            plt.xticks([]); plt.yticks([]); plt.grid(False)
            plt.imshow(images[idx].numpy())
            label_index = int(labels[idx].numpy())
            plt.xlabel(class_names[label_index])
        plt.tight_layout()
        plt.show()
```

```
def visualize_pred(dataset, model, num_images=20):
    for images, labels in dataset.take(1):
        preds = model(images, training=False)
        if isinstance(preds, (list, tuple)):
            preds = preds[0]
        preds = tf.convert_to_tensor(preds, dtype=tf.float32)
        probs = tf.nn.softmax(preds, axis=1)
        pred_classes = tf.argmax(probs, axis=1).numpy()
        conf_scores = tf.reduce_max(probs, axis=1).numpy()  # 0..1

        n = int(min(num_images, images.shape[0]))
        idxs = np.random.choice(images.shape[0], n, replace=False)

        rows = 4 if n >= 16 else int(np.ceil(n/5))
        cols = 5 if n >= 5  else n

        plt.figure(figsize=(3*cols, 3*rows))
        for i, idx in enumerate(idxs):
            plt.subplot(rows, cols, i + 1)
            plt.xticks([]); plt.yticks([]); plt.grid(False)
            plt.imshow(images[idx].numpy())
            true_label = class_names[int(labels[idx].numpy())]
            pred_label = class_names[int(pred_classes[idx])]
            confidence = conf_scores[idx] * 100.0
            plt.xlabel(f"True: {true_label}\nPred: {pred_label} ({confidence:.1f
        plt.tight_layout()
        plt.show()

visualize(train_ds, num_images=20)
```

```
2025-10-27 16:19:55.048509: W tensorflow/core/kernels/data/cache_dataset_ops.cc:9
14] The calling iterator did not fully read the dataset being cached. In order to
avoid unexpected truncation of the dataset, the partially cached contents of the
dataset  will be discarded. This can happen if you have an input pipeline similar
to `dataset.cache().take(k).repeat()`. You should use `dataset.take(k).cache().re
peat()` instead.
```



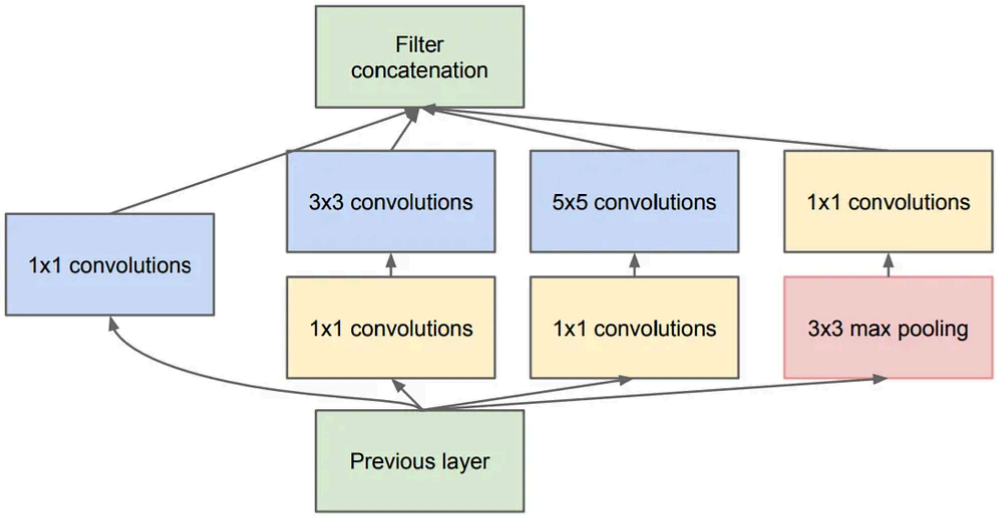

```
2025-10-27 16:19:55.954612: I tensorflow/core/framework/local_rendezvous.cc:405]
Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
```

Membangun Model dengan Architecture GoogLeNet

Untuk GoogLeNet kami menggunakan referensi dari https://www.geeksforgeeks.org/machine-learning/understanding-googlenet-model-cnn-architecture/ yang merupakan ringkasan dari paper "Going deeper with convolutions"

Inception block yang kami gunakan adalah Inception V1

In [21]:
```
from tensorflow.keras import Model, Input
```



(b) Inception module with dimension reductions

In [22]:
```
def inception_block(x, f1, f3r, f3, f5r, f5, fpp, name):
    #f1, f2, f4, dan seterusnya adalah filter
    b1 = layers.Conv2D(f1, kernel_size=(1, 1), padding="same", activation="relu"

    b2 = layers.Conv2D(f3r, kernel_size=(1, 1), padding="same", activation="relu
    b2 = layers.Conv2D(f3,  kernel_size=(3, 3), padding="same", activation="relu

    b3 = layers.Conv2D(f5r, kernel_size=(1, 1), padding="same", activation="relu
    b3 = layers.Conv2D(f5,  kernel_size=(5, 5), padding="same", activation="relu

    b4 = layers.MaxPooling2D(pool_size=(1, 1), strides=1, padding="same", name=f
    b4 = layers.Conv2D(fpp, kernel_size=(1, 1), padding="same", activation="relu

    return layers.Concatenate(name=f"{name}_concat")([b1,b2,b3,b4])
```

In [23]:
```
x = tf.random.normal((1, 28, 28, 192))  # batch 1
block = inception_block(x, 64, 96, 128, 16, 32, 32, "test_block")
print("Output shape:", block.shape)
```

```
Output shape: (1, 28, 28, 256)
```

In [24]:
```
def aux_classifier(x, num_classes, name):
    y = layers.AveragePooling2D(pool_size=(5,5), strides=3, name=f"{name}_avgpoo
    y = layers.Conv2D(128, kernel_size=(1,1), activation="relu", name=f"{name}_c
    y = layers.Flatten(name=f"{name}_flat")(y)
    y = layers.Dense(1024, activation="relu", name=f"{name}_fc")(y)
    y = layers.Dropout(0.7, name=f"{name}_dropout")(y)
    y = layers.Dense(num_classes, activation="softmax", name=f"{name}_pred")(y)
    return y
```

Membuat model utama dari Architecture GoogLeNet

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

In [25]:
```
class LocalResponseNorm(layers.Layer):
    def __init__(self, depth_radius=5, bias=1.0, alpha=1e-4, beta=0.75, **kwargs
        super().__init__(**kwargs)
        self.depth_radius, self.bias, self.alpha, self.beta = depth_radius, bias
    def call(self, inputs):
        return tf.nn.local_response_normalization(
            inputs, depth_radius=self.depth_radius, bias=self.bias,
            alpha=self.alpha, beta=self.beta
        )
def googlenet_stem_with_lrn(input_shape=(224,224,3)):
    inp = Input(shape=input_shape)
    x = layers.Conv2D(64, 7, strides=2, padding="same", activation="relu", name=
    x = LocalResponseNorm(name="lrn1")(x)
    x = layers.MaxPooling2D(3, strides=2, padding="same", name="pool1_3x3")(x)
    x = layers.Conv2D(64, 1, padding="same", activation="relu", name="conv2_1x1"
    x = layers.Conv2D(192, 3, padding="same", activation="relu", name="conv2_3x3
    x = LocalResponseNorm(name="lrn2")(x)
    x = layers.MaxPooling2D(3, strides=2, padding="same", name="pool2_3x3")(x)
    return inp, x
```

In [26]:
```
def build_googLeNet(input_shape=(224,224,3), num_classes=4, use_lrn=False):
    if use_lrn:
        inp, x = googlenet_stem_with_lrn(input_shape)
    else:
        inp = Input(shape=input_shape)
        x = layers.Conv2D(64, kernel_size=(7, 7), strides=2, padding="same", act

        x = layers.MaxPooling2D(pool_size=(3, 3), strides=2, padding="same", nam

        x = layers.Conv2D(64, kernel_size=(1, 1), padding="same", activation="re
        x = layers.Conv2D(192, kernel_size=(3, 3), padding="same", activation="r
        if use_lrn:
            x = tf.nn.local_response_normalization(x, depth_radius=5, bias=1, al
        x = layers.MaxPooling2D(pool_size=(3, 3), strides=2, padding="same", nam

    # ===== Inception 3 =====
    x = inception_block(x,  64,  96,128, 16, 32, 32, "inception_3a")
    x = inception_block(x, 128, 128,192, 32, 96, 64, "inception_3b")
    x = layers.MaxPooling2D(3, strides=2, padding="same", name="pool3_3x3")(x)
```

```
    # ===== Inception 4 + aux1 =====
    x = inception_block(x, 192,  96,208, 16, 48, 64, "inception_4a")
    aux1 = aux_classifier(x, num_classes, name="aux1")

    x = inception_block(x, 160, 112,224, 24, 64, 64, "inception_4b")
    x = inception_block(x, 128, 128,256, 24, 64, 64, "inception_4c")
    x = inception_block(x, 112, 144,288, 32, 64, 64, "inception_4d")
    aux2 = aux_classifier(x, num_classes, name="aux2")

    x = inception_block(x, 256, 160,320, 32,128,128, "inception_4e")
    x = layers.MaxPooling2D(pool_size=(3, 3), strides=2, padding="same", name="p

    # ===== Inception 5 + head utama =====
    x = inception_block(x, 256, 160,320, 32,128,128, "inception_5a")
    x = inception_block(x, 384, 192,384, 48,128,128, "inception_5b")

    x = layers.GlobalAveragePooling2D(name="global_avgpool")(x)
    x = layers.Dropout(0.4, name="dropout")(x)
    main = layers.Dense(num_classes, activation="softmax", name="predictions")(x

    return Model(inputs=inp, outputs=[main, aux1, aux2], name="GoogLeNet")
```

In [27]:
```
model = build_googLeNet(input_shape=(224,224,3), num_classes=len(class_names), u
model.summary()
```

Model: "GoogLeNet"

Layer (type)	Output Shape	Param #	Connected to
input_layer_3 (InputLayer)	(None, 224, 224, 3)	0	-
conv1_7x7 (Conv2D)	(None, 112, 112, 64)	9,472	input_layer_3[0]…
pool1_3x3 (MaxPooling2D)	(None, 56, 56, 64)	0	conv1_7x7[0][0]
conv2_1x1 (Conv2D)	(None, 56, 56, 64)	4,160	pool1_3x3[0][0]
conv2_3x3 (Conv2D)	(None, 56, 56, 192)	110,784	conv2_1x1[0][0]
pool2_3x3 (MaxPooling2D)	(None, 28, 28, 192)	0	conv2_3x3[0][0]
inception_3a_3x3_r… (Conv2D)	(None, 28, 28, 96)	18,528	pool2_3x3[0][0]
inception_3a_5x5_r… (Conv2D)	(None, 28, 28, 16)	3,088	pool2_3x3[0][0]
inception_3a_pool (MaxPooling2D)	(None, 28, 28, 192)	0	pool2_3x3[0][0]
inception_3a_1x1 (Conv2D)	(None, 28, 28, 64)	12,352	pool2_3x3[0][0]
inception_3a_3x3 (Conv2D)	(None, 28, 28, 128)	110,720	inception_3a_3x3…
inception_3a_5x5 (Conv2D)	(None, 28, 28, 32)	12,832	inception_3a_5x5…
inception_3a_pool_… (Conv2D)	(None, 28, 28, 32)	6,176	inception_3a_poo…
inception_3a_concat (Concatenate)	(None, 28, 28, 256)	0	inception_3a_1x1… inception_3a_3x3… inception_3a_5x5… inception_3a_poo…
inception_3b_3x3_r… (Conv2D)	(None, 28, 28, 128)	32,896	inception_3a_con…
inception_3b_5x5_r… (Conv2D)	(None, 28, 28, 32)	8,224	inception_3a_con…
inception_3b_pool (MaxPooling2D)	(None, 28, 28, 256)	0	inception_3a_con…
inception_3b_1x1 (Conv2D)	(None, 28, 28, 128)	32,896	inception_3a_con…
inception_3b_3x3	(None, 28, 28,	221,376	inception_3b_3x3…

(Conv2D)	192)		
inception_3b_5x5 (Conv2D)	(None, 28, 28, 96)	76,896	inception_3b_5x5…
inception_3b_pool_… (Conv2D)	(None, 28, 28, 64)	16,448	inception_3b_poo…
inception_3b_concat (Concatenate)	(None, 28, 28, 480)	0	inception_3b_1x1… inception_3b_3x3… inception_3b_5x5… inception_3b_poo…
pool3_3x3 (MaxPooling2D)	(None, 14, 14, 480)	0	inception_3b_con…
inception_4a_3x3_r… (Conv2D)	(None, 14, 14, 96)	46,176	pool3_3x3[0][0]
inception_4a_5x5_r… (Conv2D)	(None, 14, 14, 16)	7,696	pool3_3x3[0][0]
inception_4a_pool (MaxPooling2D)	(None, 14, 14, 480)	0	pool3_3x3[0][0]
inception_4a_1x1 (Conv2D)	(None, 14, 14, 192)	92,352	pool3_3x3[0][0]
inception_4a_3x3 (Conv2D)	(None, 14, 14, 208)	179,920	inception_4a_3x3…
inception_4a_5x5 (Conv2D)	(None, 14, 14, 48)	19,248	inception_4a_5x5…
inception_4a_pool_… (Conv2D)	(None, 14, 14, 64)	30,784	inception_4a_poo…
inception_4a_concat (Concatenate)	(None, 14, 14, 512)	0	inception_4a_1x1… inception_4a_3x3… inception_4a_5x5… inception_4a_poo…
inception_4b_3x3_r… (Conv2D)	(None, 14, 14, 112)	57,456	inception_4a_con…
inception_4b_5x5_r… (Conv2D)	(None, 14, 14, 24)	12,312	inception_4a_con…
inception_4b_pool (MaxPooling2D)	(None, 14, 14, 512)	0	inception_4a_con…
inception_4b_1x1 (Conv2D)	(None, 14, 14, 160)	82,080	inception_4a_con…
inception_4b_3x3 (Conv2D)	(None, 14, 14, 224)	226,016	inception_4b_3x3…
inception_4b_5x5 (Conv2D)	(None, 14, 14, 64)	38,464	inception_4b_5x5…

inception_4b_pool_… (Conv2D)	(None, 14, 14, 64)	32,832	inception_4b_poo…
inception_4b_concat (Concatenate)	(None, 14, 14, 512)	0	inception_4b_1x1… inception_4b_3x3… inception_4b_5x5… inception_4b_poo…
inception_4c_3x3_r… (Conv2D)	(None, 14, 14, 128)	65,664	inception_4b_con…
inception_4c_5x5_r… (Conv2D)	(None, 14, 14, 24)	12,312	inception_4b_con…
inception_4c_pool (MaxPooling2D)	(None, 14, 14, 512)	0	inception_4b_con…
inception_4c_1x1 (Conv2D)	(None, 14, 14, 128)	65,664	inception_4b_con…
inception_4c_3x3 (Conv2D)	(None, 14, 14, 256)	295,168	inception_4c_3x3…
inception_4c_5x5 (Conv2D)	(None, 14, 14, 64)	38,464	inception_4c_5x5…
inception_4c_pool_… (Conv2D)	(None, 14, 14, 64)	32,832	inception_4c_poo…
inception_4c_concat (Concatenate)	(None, 14, 14, 512)	0	inception_4c_1x1… inception_4c_3x3… inception_4c_5x5… inception_4c_poo…
inception_4d_3x3_r… (Conv2D)	(None, 14, 14, 144)	73,872	inception_4c_con…
inception_4d_5x5_r… (Conv2D)	(None, 14, 14, 32)	16,416	inception_4c_con…
inception_4d_pool (MaxPooling2D)	(None, 14, 14, 512)	0	inception_4c_con…
inception_4d_1x1 (Conv2D)	(None, 14, 14, 112)	57,456	inception_4c_con…
inception_4d_3x3 (Conv2D)	(None, 14, 14, 288)	373,536	inception_4d_3x3…
inception_4d_5x5 (Conv2D)	(None, 14, 14, 64)	51,264	inception_4d_5x5…
inception_4d_pool_… (Conv2D)	(None, 14, 14, 64)	32,832	inception_4d_poo…
inception_4d_concat (Concatenate)	(None, 14, 14, 528)	0	inception_4d_1x1… inception_4d_3x3… inception_4d_5x5… inception_4d_poo…

inception_4e_3x3_r… (Conv2D)	(None, 14, 14, 160)	84,640	inception_4d_con…
inception_4e_5x5_r… (Conv2D)	(None, 14, 14, 32)	16,928	inception_4d_con…
inception_4e_pool (MaxPooling2D)	(None, 14, 14, 528)	0	inception_4d_con…
inception_4e_1x1 (Conv2D)	(None, 14, 14, 256)	135,424	inception_4d_con…
inception_4e_3x3 (Conv2D)	(None, 14, 14, 320)	461,120	inception_4e_3x3…
inception_4e_5x5 (Conv2D)	(None, 14, 14, 128)	102,528	inception_4e_5x5…
inception_4e_pool_… (Conv2D)	(None, 14, 14, 128)	67,712	inception_4e_poo…
inception_4e_concat (Concatenate)	(None, 14, 14, 832)	0	inception_4e_1x1… inception_4e_3x3… inception_4e_5x5… inception_4e_poo…
pool4_3x3 (MaxPooling2D)	(None, 7, 7, 832)	0	inception_4e_con…
inception_5a_3x3_r… (Conv2D)	(None, 7, 7, 160)	133,280	pool4_3x3[0][0]
inception_5a_5x5_r… (Conv2D)	(None, 7, 7, 32)	26,656	pool4_3x3[0][0]
inception_5a_pool (MaxPooling2D)	(None, 7, 7, 832)	0	pool4_3x3[0][0]
inception_5a_1x1 (Conv2D)	(None, 7, 7, 256)	213,248	pool4_3x3[0][0]
inception_5a_3x3 (Conv2D)	(None, 7, 7, 320)	461,120	inception_5a_3x3…
inception_5a_5x5 (Conv2D)	(None, 7, 7, 128)	102,528	inception_5a_5x5…
inception_5a_pool_… (Conv2D)	(None, 7, 7, 128)	106,624	inception_5a_poo…
inception_5a_concat (Concatenate)	(None, 7, 7, 832)	0	inception_5a_1x1… inception_5a_3x3… inception_5a_5x5… inception_5a_poo…
inception_5b_3x3_r… (Conv2D)	(None, 7, 7, 192)	159,936	inception_5a_con…
inception_5b_5x5_r… (Conv2D)	(None, 7, 7, 48)	39,984	inception_5a_con…

inception_5b_pool (MaxPooling2D)	(None, 7, 7, 832)	0	inception_5a_con…
aux1_avgpool (AveragePooling2D)	(None, 4, 4, 512)	0	inception_4a_con…
aux2_avgpool (AveragePooling2D)	(None, 4, 4, 528)	0	inception_4d_con…
inception_5b_1x1 (Conv2D)	(None, 7, 7, 384)	319,872	inception_5a_con…
inception_5b_3x3 (Conv2D)	(None, 7, 7, 384)	663,936	inception_5b_3x3…
inception_5b_5x5 (Conv2D)	(None, 7, 7, 128)	153,728	inception_5b_5x5…
inception_5b_pool_… (Conv2D)	(None, 7, 7, 128)	106,624	inception_5b_poo…
aux1_conv (Conv2D)	(None, 4, 4, 128)	65,664	aux1_avgpool[0][…
aux2_conv (Conv2D)	(None, 4, 4, 128)	67,712	aux2_avgpool[0][…
inception_5b_concat (Concatenate)	(None, 7, 7, 1024)	0	inception_5b_1x1… inception_5b_3x3… inception_5b_5x5… inception_5b_poo…
aux1_flat (Flatten)	(None, 2048)	0	aux1_conv[0][0]
aux2_flat (Flatten)	(None, 2048)	0	aux2_conv[0][0]
global_avgpool (GlobalAveragePool…	(None, 1024)	0	inception_5b_con…
aux1_fc (Dense)	(None, 1024)	2,098,176	aux1_flat[0][0]
aux2_fc (Dense)	(None, 1024)	2,098,176	aux2_flat[0][0]
dropout (Dropout)	(None, 1024)	0	global_avgpool[0…
aux1_dropout (Dropout)	(None, 1024)	0	aux1_fc[0][0]
aux2_dropout (Dropout)	(None, 1024)	0	aux2_fc[0][0]
predictions (Dense)	(None, 4)	4,100	dropout[0][0]
aux1_pred (Dense)	(None, 4)	4,100	aux1_dropout[0][…
aux2_pred (Dense)	(None, 4)	4,100	aux2_dropout[0][…

Total params: 10,315,580 (39.35 MB)

Trainable params: 10,315,580 (39.35 MB)

Non-trainable params: 0 (0.00 B)

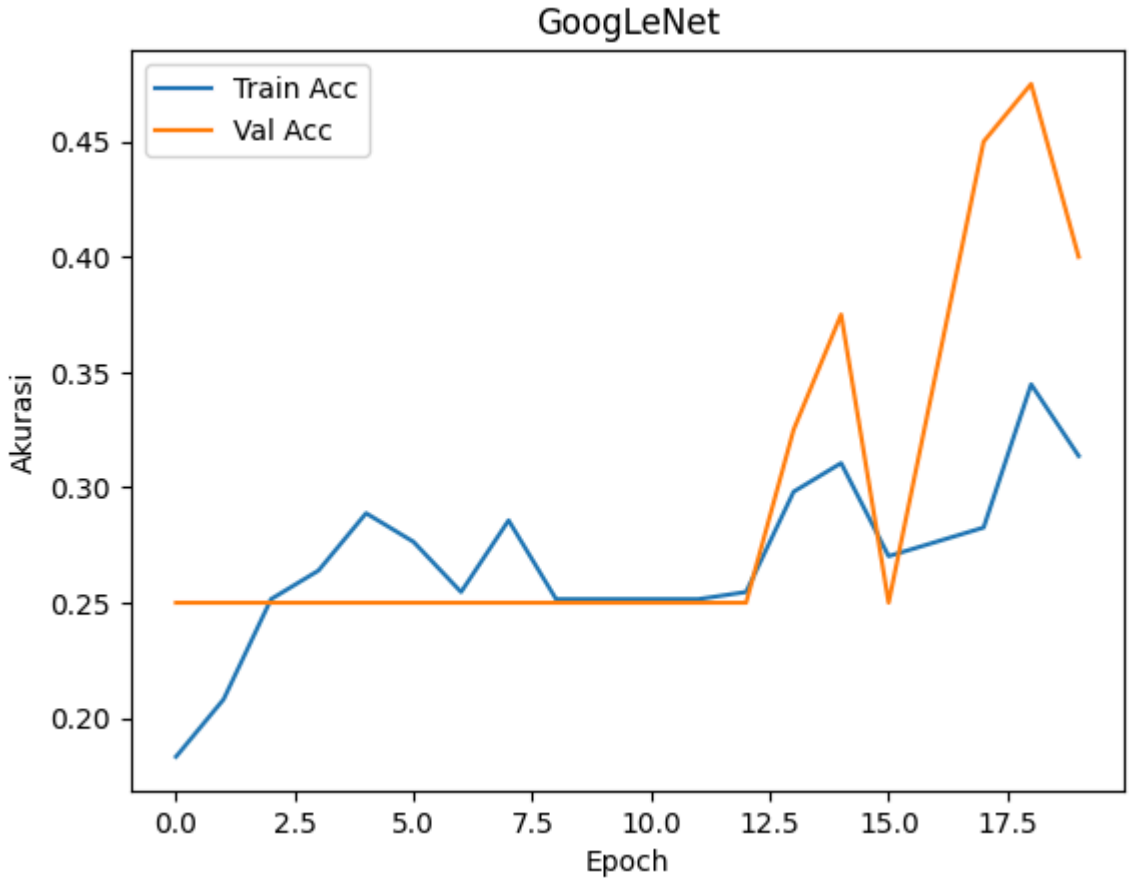
In [28]:
```
from tensorflow.keras.optimizers import Adam
# === Compile multi-output model dengan named losses ===
model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate=0.0005),
    loss={
        "predictions": "sparse_categorical_crossentropy",
        "aux1_pred": "sparse_categorical_crossentropy",
        "aux2_pred": "sparse_categorical_crossentropy",
    },
    loss_weights={
        "predictions": 1.0,
        "aux1_pred": 0.3,
        "aux2_pred": 0.3,
    },
    metrics={"predictions": ["accuracy"]},
)

# === Dataset multi-output ===
def triple_y(x, y):
    return x, (y, y, y)

train_input = train_ds.map(triple_y)
val_input   = val_ds.map(triple_y)
test_input  = test_ds.map(triple_y)

# === Training ===
history = model.fit(
    train_input,
    epochs=20,
    validation_data=val_input
)

# === Evaluasi ===
test_loss, test_acc, *_ = model.evaluate(test_input)
print("Akurasi :", test_acc)
```

```
Epoch 1/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 44s 2s/step - aux1_pred_loss: 1.3923 - aux2_pred_loss:
1.3895 - loss: 2.2242 - predictions_accuracy: 0.1694 - predictions_loss: 1.3895 -
val_aux1_pred_loss: 1.3855 - val_aux2_pred_loss: 1.3863 - val_loss: 2.2179 - val_
predictions_accuracy: 0.2500 - val_predictions_loss: 1.3863
Epoch 2/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 1s 52ms/step - aux1_pred_loss: 1.3854 - aux2_pred_los
s: 1.3857 - loss: 2.2177 - predictions_accuracy: 0.2167 - predictions_loss: 1.386
0 - val_aux1_pred_loss: 1.3835 - val_aux2_pred_loss: 1.3863 - val_loss: 2.2175 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3865
Epoch 3/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step - aux1_pred_loss: 1.3838 - aux2_pred_los
s: 1.3871 - loss: 2.2187 - predictions_accuracy: 0.2364 - predictions_loss: 1.386
5 - val_aux1_pred_loss: 1.3606 - val_aux2_pred_loss: 1.3862 - val_loss: 2.2110 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3869
Epoch 4/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 37ms/step - aux1_pred_loss: 1.3841 - aux2_pred_los
s: 1.3885 - loss: 2.2189 - predictions_accuracy: 0.2445 - predictions_loss: 1.386
7 - val_aux1_pred_loss: 1.3672 - val_aux2_pred_loss: 1.3875 - val_loss: 2.2140 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3876
Epoch 5/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 34ms/step - aux1_pred_loss: 1.3748 - aux2_pred_los
s: 1.3897 - loss: 2.2185 - predictions_accuracy: 0.2691 - predictions_loss: 1.387
8 - val_aux1_pred_loss: 1.3629 - val_aux2_pred_loss: 1.3854 - val_loss: 2.2123 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3878
Epoch 6/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - aux1_pred_loss: 1.3611 - aux2_pred_los
s: 1.3852 - loss: 2.2135 - predictions_accuracy: 0.2600 - predictions_loss: 1.387
3 - val_aux1_pred_loss: 1.2405 - val_aux2_pred_loss: 1.3268 - val_loss: 2.1325 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3623
Epoch 7/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step - aux1_pred_loss: 1.3073 - aux2_pred_los
s: 1.3474 - loss: 2.1630 - predictions_accuracy: 0.2536 - predictions_loss: 1.356
6 - val_aux1_pred_loss: 1.4895 - val_aux2_pred_loss: 2.4473 - val_loss: 4.4632 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 3.2822
Epoch 8/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step - aux1_pred_loss: 1.4540 - aux2_pred_los
s: 1.8183 - loss: 3.1086 - predictions_accuracy: 0.2989 - predictions_loss: 2.120
0 - val_aux1_pred_loss: 1.3943 - val_aux2_pred_loss: 1.3794 - val_loss: 2.2168 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3847
Epoch 9/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 39ms/step - aux1_pred_loss: 1.4354 - aux2_pred_los
s: 1.3906 - loss: 2.2362 - predictions_accuracy: 0.2405 - predictions_loss: 1.388
1 - val_aux1_pred_loss: 1.3682 - val_aux2_pred_loss: 1.3796 - val_loss: 2.2098 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3855
Epoch 10/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step - aux1_pred_loss: 1.3699 - aux2_pred_los
s: 1.3794 - loss: 2.2149 - predictions_accuracy: 0.2405 - predictions_loss: 1.389
1 - val_aux1_pred_loss: 1.3473 - val_aux2_pred_loss: 1.3694 - val_loss: 2.2007 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3857
Epoch 11/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step - aux1_pred_loss: 1.3691 - aux2_pred_los
s: 1.3797 - loss: 2.2145 - predictions_accuracy: 0.2405 - predictions_loss: 1.388
4 - val_aux1_pred_loss: 1.2859 - val_aux2_pred_loss: 1.3154 - val_loss: 2.1655 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3851
Epoch 12/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step - aux1_pred_loss: 1.3288 - aux2_pred_los
s: 1.3564 - loss: 2.1975 - predictions_accuracy: 0.2405 - predictions_loss: 1.390
7 - val_aux1_pred_loss: 1.2005 - val_aux2_pred_loss: 1.2492 - val_loss: 2.1170 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3821
```

```
Epoch 13/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - aux1_pred_loss: 1.2679 - aux2_pred_los
s: 1.3232 - loss: 2.1650 - predictions_accuracy: 0.2402 - predictions_loss: 1.385
4 - val_aux1_pred_loss: 1.1733 - val_aux2_pred_loss: 1.2899 - val_loss: 2.1196 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3806
Epoch 14/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 32ms/step - aux1_pred_loss: 1.2326 - aux2_pred_los
s: 1.2892 - loss: 2.1434 - predictions_accuracy: 0.2639 - predictions_loss: 1.380
7 - val_aux1_pred_loss: 1.1347 - val_aux2_pred_loss: 1.1798 - val_loss: 2.0189 -
val_predictions_accuracy: 0.3250 - val_predictions_loss: 1.3245
Epoch 15/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 40ms/step - aux1_pred_loss: 1.2305 - aux2_pred_los
s: 1.2392 - loss: 2.1153 - predictions_accuracy: 0.3105 - predictions_loss: 1.371
3 - val_aux1_pred_loss: 1.1735 - val_aux2_pred_loss: 1.2619 - val_loss: 2.0914 -
val_predictions_accuracy: 0.3750 - val_predictions_loss: 1.3608
Epoch 16/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 39ms/step - aux1_pred_loss: 1.1967 - aux2_pred_los
s: 1.2481 - loss: 2.1091 - predictions_accuracy: 0.2780 - predictions_loss: 1.369
4 - val_aux1_pred_loss: 1.0750 - val_aux2_pred_loss: 1.1425 - val_loss: 1.9755 -
val_predictions_accuracy: 0.2500 - val_predictions_loss: 1.3102
Epoch 17/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 39ms/step - aux1_pred_loss: 1.2003 - aux2_pred_los
s: 1.2289 - loss: 2.0845 - predictions_accuracy: 0.2721 - predictions_loss: 1.350
1 - val_aux1_pred_loss: 1.1113 - val_aux2_pred_loss: 1.2086 - val_loss: 2.0372 -
val_predictions_accuracy: 0.3500 - val_predictions_loss: 1.3412
Epoch 18/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - aux1_pred_loss: 1.2018 - aux2_pred_los
s: 1.2630 - loss: 2.1016 - predictions_accuracy: 0.2765 - predictions_loss: 1.357
6 - val_aux1_pred_loss: 1.1175 - val_aux2_pred_loss: 1.1393 - val_loss: 1.9957 -
val_predictions_accuracy: 0.4500 - val_predictions_loss: 1.3187
Epoch 19/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step - aux1_pred_loss: 1.1626 - aux2_pred_los
s: 1.2307 - loss: 2.0503 - predictions_accuracy: 0.3677 - predictions_loss: 1.328
4 - val_aux1_pred_loss: 1.0748 - val_aux2_pred_loss: 1.1042 - val_loss: 1.8736 -
val_predictions_accuracy: 0.4750 - val_predictions_loss: 1.2200
Epoch 20/20
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 35ms/step - aux1_pred_loss: 1.1341 - aux2_pred_los
s: 1.1643 - loss: 1.9942 - predictions_accuracy: 0.3386 - predictions_loss: 1.298
5 - val_aux1_pred_loss: 1.0845 - val_aux2_pred_loss: 1.1118 - val_loss: 1.9432 -
val_predictions_accuracy: 0.4000 - val_predictions_loss: 1.2843
4/4 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step - aux1_pred_loss: 1.0623 - aux2_pred_loss:
1.0653 - loss: 1.9444 - predictions_accuracy: 0.3667 - predictions_loss: 1.3061
Akurasi : 1.3133375644683838
```

In [29]:

```
plt.title("GoogLeNet")
plt.plot(history.history['predictions_accuracy'], label='Train Acc')
plt.plot(history.history['val_predictions_accuracy'], label='Val Acc')
plt.xlabel('Epoch')
plt.ylabel('Akurasi')
plt.legend()
plt.show()
```



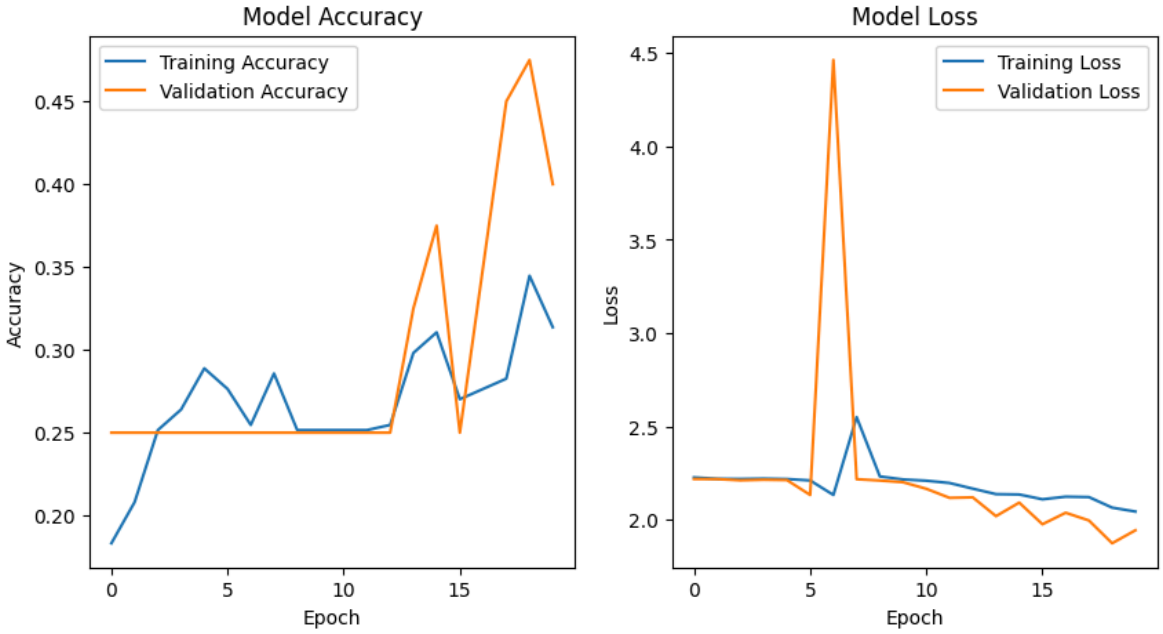
In [30]:
```
import pandas as pd
#Evaluasi
df_history = pd.DataFrame(history.history)
print(df_history.head())
plt.figure(figsize=(10,5))

# Accuracynya
plt.subplot(1, 2, 1)
plt.plot(df_history['predictions_accuracy'], label='Training Accuracy')
plt.plot(df_history['val_predictions_accuracy'], label='Validation Accuracy')
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()

# Lossnya
plt.subplot(1, 2, 2)
plt.plot(df_history['loss'], label='Training Loss')
plt.plot(df_history['val_loss'], label='Validation Loss')
plt.title('Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
```

	aux1_pred_loss	aux2_pred_loss	loss	predictions_accuracy	\
0	1.394224	1.391362	2.226406	0.183230	
1	1.384810	1.385456	2.219024	0.208075	
2	1.376246	1.383984	2.218354	0.251553	
3	1.395927	1.384061	2.219993	0.263975	
4	1.373280	1.384379	2.217795	0.288820	

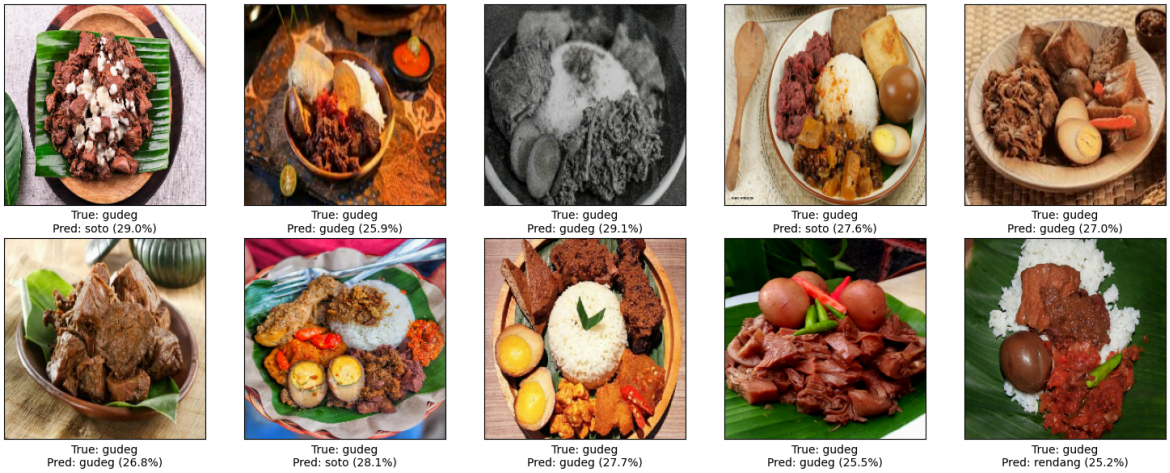
	predictions_loss	val_aux1_pred_loss	val_aux2_pred_loss	val_loss	\
0	1.389929	1.385544	1.386298	2.217877	
1	1.385688	1.383463	1.386342	2.217457	
2	1.384766	1.360582	1.386230	2.210959	
3	1.383335	1.367160	1.387502	2.214015	
4	1.382770	1.362902	1.385363	2.212282	

	val_predictions_accuracy	val_predictions_loss
0	0.25	1.386324
1	0.25	1.386516
2	0.25	1.386915
3	0.25	1.387617
4	0.25	1.387802

Out[30]: <matplotlib.legend.Legend at 0x7feb28338950>

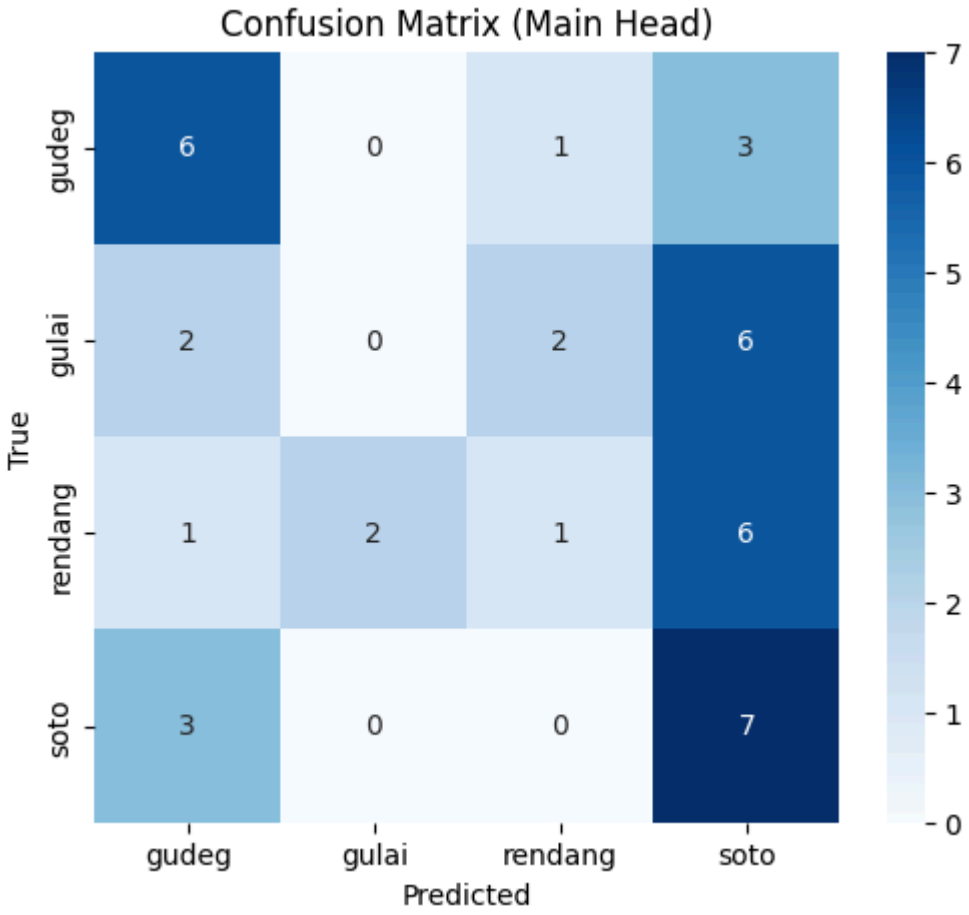


In [31]:
```
visualize_pred(test_ds, model, 20)
```



In [33]:
```
pred_main, pred_aux1, pred_aux2 = model.predict(test_input)

y_true = np.concatenate([y for x, y in test_ds], axis=0)
y_pred = np.argmax(pred_main, axis=1)
cm = confusion_matrix(y_true, y_pred)

plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix (Main Head)')
plt.show()
```

```
4/4 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step
```



In [35]:
```
#menyimpan model
model.save_weights("GoogLeNet_A_AlekSnack.weights.h5")
```

In [ ]:
```

```

In [2]:
```
import matplotlib.pyplot as plt
import numpy as np
import random
from tqdm import tqdm
import mitdeeplearning as mdl
import tensorflow as tf
from tensorflow.keras import layers, models
```

```
Gym has been unmaintained since 2022 and does not support NumPy 2.0 amongst other critical functionality.
Please upgrade to Gymnasium, the maintained drop-in replacement of Gym, or contact the authors of your software and request that they upgrade.
Users of this version of Gym should be able to simply replace 'import gym' with 'import gymnasium as gym' in the vast majority of cases.
See the migration guide at https://gymnasium.farama.org/introduction/migration_guide/ for additional information.
```

In [ ]:
```
import os

#sebelumnya tidak begini tapi minjam template Resnet
BASE_DIR = "dataset"
tr_dir = os.path.join(BASE_DIR, "train")
vl_dir   = os.path.join(BASE_DIR, "validation")
ts_dir  = os.path.join(BASE_DIR, "test")

input_dir = "input"

def load_dataset(directory, img_size=(224,224), batch_size=32, shuffle=True):
    ds = tf.keras.utils.image_dataset_from_directory(
        directory,
        image_size=img_size,
        batch_size=batch_size,
        shuffle=shuffle
    )
    return ds


train_ds = load_dataset(tr_dir, batch_size=90)
val_ds = load_dataset(vl_dir, batch_size=10, shuffle=False)
test_ds = load_dataset(ts_dir, batch_size=10, shuffle=False)

input_ds = load_dataset(input_dir, batch_size=90)

class_names = train_ds.class_names
print("Kelas:", class_names)

val_ds = val_ds.map(lambda x, y: (x/255.0, y))
test_ds = test_ds.map(lambda x, y: (x/255.0, y))
```

```
Found 332 files belonging to 4 classes.
Found 40 files belonging to 4 classes.
Found 40 files belonging to 4 classes.
Found 41 files belonging to 4 classes.
Kelas: ['gudeg', 'gulai', 'rendang', 'soto']
```

In [4]:
```
# Mengecek jumlah file per kelas meminjam template notebook Resnet lain
def count_images_in_dir(dirpath):
    counts = {}
    if not os.path.exists(dirpath):
```

```
        print(f"Directory not found: {dirpath}")
        return counts
    for cls in sorted(os.listdir(dirpath)):
        cls_path = os.path.join(dirpath, cls)
        if os.path.isdir(cls_path):
            counts[cls] = sum([1 for _ in os.listdir(cls_path) if os.path.isfile
    return counts

print("Train counts:", count_images_in_dir(tr_dir))
print("Validation counts:", count_images_in_dir(vl_dir))
print("Test counts:", count_images_in_dir(ts_dir))
```

```
Train counts: {'gudeg': 80, 'gulai': 80, 'rendang': 82, 'soto': 90}
Validation counts: {'gudeg': 10, 'gulai': 10, 'rendang': 10, 'soto': 10}
Test counts: {'gudeg': 10, 'gulai': 10, 'rendang': 10, 'soto': 10}
```

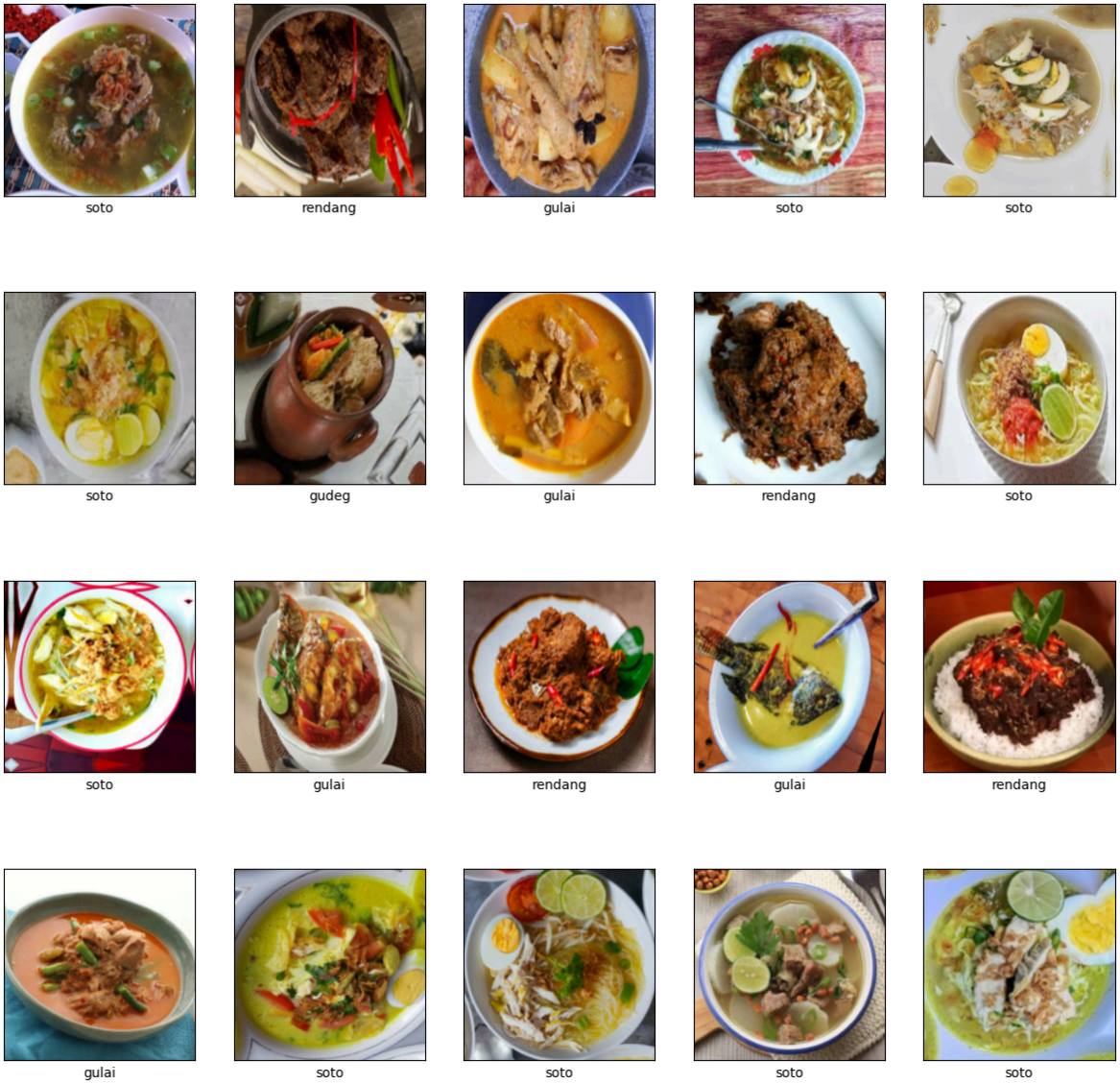
In [5]:
```
from tensorflow.keras import layers

data_augmentation = tf.keras.Sequential([ #untuk menangani masalah overfitting k
    layers.RandomFlip("horizontal_and_vertical"),
    layers.RandomRotation(0.1),
    layers.RandomZoom(0.1),
])

train_ds = train_ds.map(lambda x, y: (data_augmentation(x, training=True), y))
train_ds = train_ds.map(lambda x, y: (x/255.0, y))
```

In [6]:
```
def visualize(dataset, num_images=20):
    plt.figure(figsize=(15, 15))
    for images, labels in dataset.take(1):
        rand_i = np.random.choice(len(images), num_images, replace=False) #buat
        for i in range(len(rand_i)):
            idx = rand_i[i]
            plt.subplot(4, 5, i + 1)
            plt.xticks([])
            plt.yticks([])
            plt.grid(False)
            plt.imshow(images[idx].numpy())
            label_index = int(labels[idx].numpy())
            plt.xlabel(class_names[label_index])
    plt.show()

def visualize_pred(dataset,num_images=20):
    plt.figure(figsize=(15, 15))
    for images, labels in input_ds.take(1):
        predictions = model.predict(images)
        pred_classes = tf.argmax(predictions, axis=1)
        conf_scores = tf.reduce_max(tf.nn.softmax(predictions, axis=1), axis=1)

        rand_i = np.random.choice(images.shape[0], 20, replace=False)

        for i in range(len(rand_i)):
            idx = rand_i[i]
            plt.subplot(4, 5, i + 1)
            plt.xticks([])
            plt.yticks([])
            plt.grid(False)
            plt.imshow(images[idx].numpy().astype("uint8"))
            true_label = class_names[int(labels[idx])]
            pred_label = class_names[int(pred_classes[idx])]
```

```
            confidence = conf_scores[idx].numpy() * 100
            plt.xlabel(f"True: {true_label}\nPred: {pred_label} ({confidence:.1f

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

```
<Figure size 640x480 with 0 Axes>
```

In [7]:
```
#show isi dataset Train
visualize(train_ds)
```



In [8]:
```
model = models.Sequential([ # 3 Conv2D karena databasenya makanan
    layers.Conv2D(32, (3,3), activation='relu', input_shape=(224,224,3)),
    layers.MaxPooling2D(2,2),
    layers.Conv2D(64, (3,3), activation='relu'),
    layers.MaxPooling2D(2,2),
    layers.Dropout(0.3),
    layers.Conv2D(128, (3,3), activation='relu'),
    layers.MaxPooling2D(2,2),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(len(class_names), activation='softmax')
])

model.summary()
```

```
C:\Users\Pongo\AppData\Roaming\Python\Python312\site-packages\keras\src\layers\convolutional\base_conv.py:107: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 222, 222, 32)	896
max_pooling2d (MaxPooling2D)	(None, 111, 111, 32)	0
conv2d_1 (Conv2D)	(None, 109, 109, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 64)	0
dropout (Dropout)	(None, 54, 54, 64)	0
conv2d_2 (Conv2D)	(None, 52, 52, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 128)	0
flatten (Flatten)	(None, 86528)	0
dense (Dense)	(None, 128)	11,075,712
dropout_1 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 4)	516

Total params: 11,169,476 (42.61 MB)

Trainable params: 11,169,476 (42.61 MB)

Non-trainable params: 0 (0.00 B)

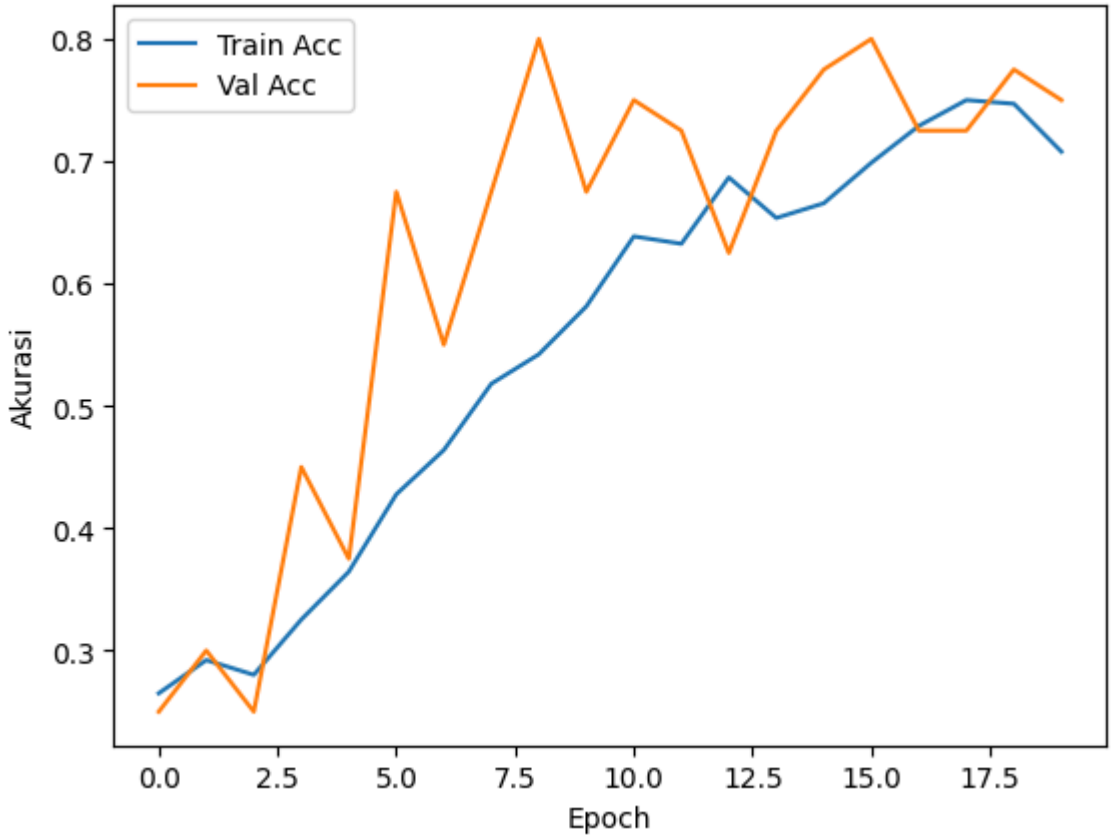
In [9]:
```
from tensorflow.keras.optimizers import Adam

optimizer = Adam(learning_rate=0.0005) #sebelumnya 0.001(default) tapi hasilnya

model.compile(
    optimizer=optimizer,
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']
)
```

In [10]:
```
#training modelnya
history = model.fit(
    train_ds,
    epochs=20,
    validation_data=val_ds
)

test_loss, test_acc = model.evaluate(test_ds)
print("Akurasi :", test_acc)

plt.plot(history.history['accuracy'], label='Train Acc')
```

```
plt.plot(history.history['val_accuracy'], label='Val Acc')
plt.xlabel('Epoch')
plt.ylabel('Akurasi')
plt.legend()
plt.show()
```

```
Epoch 1/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 31s 6s/step - accuracy: 0.2616 - loss: 3.8043 - val_accu
racy: 0.2500 - val_loss: 2.1731
Epoch 2/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 18s 4s/step - accuracy: 0.3128 - loss: 2.6175 - val_accu
racy: 0.3000 - val_loss: 1.3640
Epoch 3/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 18s 4s/step - accuracy: 0.2961 - loss: 1.3905 - val_accu
racy: 0.2500 - val_loss: 1.3787
Epoch 4/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 17s 4s/step - accuracy: 0.3220 - loss: 1.3877 - val_accu
racy: 0.4500 - val_loss: 1.3754
Epoch 5/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 18s 4s/step - accuracy: 0.3506 - loss: 1.3505 - val_accu
racy: 0.3750 - val_loss: 1.3561
Epoch 6/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 17s 4s/step - accuracy: 0.4281 - loss: 1.3098 - val_accu
racy: 0.6750 - val_loss: 1.2939
Epoch 7/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 17s 4s/step - accuracy: 0.4589 - loss: 1.2709 - val_accu
racy: 0.5500 - val_loss: 1.2322
Epoch 8/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 17s 4s/step - accuracy: 0.4932 - loss: 1.1787 - val_accu
racy: 0.6750 - val_loss: 1.0661
Epoch 9/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 18s 4s/step - accuracy: 0.5395 - loss: 1.1323 - val_accu
racy: 0.8000 - val_loss: 0.9848
Epoch 10/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 18s 4s/step - accuracy: 0.6085 - loss: 0.9981 - val_accu
racy: 0.6750 - val_loss: 1.0195
Epoch 11/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 17s 4s/step - accuracy: 0.6339 - loss: 0.9657 - val_accu
racy: 0.7500 - val_loss: 0.8628
Epoch 12/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 18s 4s/step - accuracy: 0.6460 - loss: 0.9357 - val_accu
racy: 0.7250 - val_loss: 0.8623
Epoch 13/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 17s 4s/step - accuracy: 0.6588 - loss: 0.8656 - val_accu
racy: 0.6250 - val_loss: 0.8350
Epoch 14/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 15s 3s/step - accuracy: 0.6489 - loss: 0.8639 - val_accu
racy: 0.7250 - val_loss: 0.7648
Epoch 15/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 5s 1s/step - accuracy: 0.6615 - loss: 0.7930 - val_accur
acy: 0.7750 - val_loss: 0.7362
Epoch 16/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 5s 1s/step - accuracy: 0.6999 - loss: 0.7537 - val_accur
acy: 0.8000 - val_loss: 0.7230
Epoch 17/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 5s 1s/step - accuracy: 0.7453 - loss: 0.6392 - val_accur
acy: 0.7250 - val_loss: 0.7197
Epoch 18/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 12s 3s/step - accuracy: 0.7459 - loss: 0.6837 - val_accu
racy: 0.7250 - val_loss: 0.6919
Epoch 19/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 7s 1s/step - accuracy: 0.7484 - loss: 0.6672 - val_accur
acy: 0.7750 - val_loss: 0.6829
Epoch 20/20
4/4 ━━━━━━━━━━━━━━━━━━━━ 17s 4s/step - accuracy: 0.7109 - loss: 0.6608 - val_accu
racy: 0.7500 - val_loss: 0.6642
```

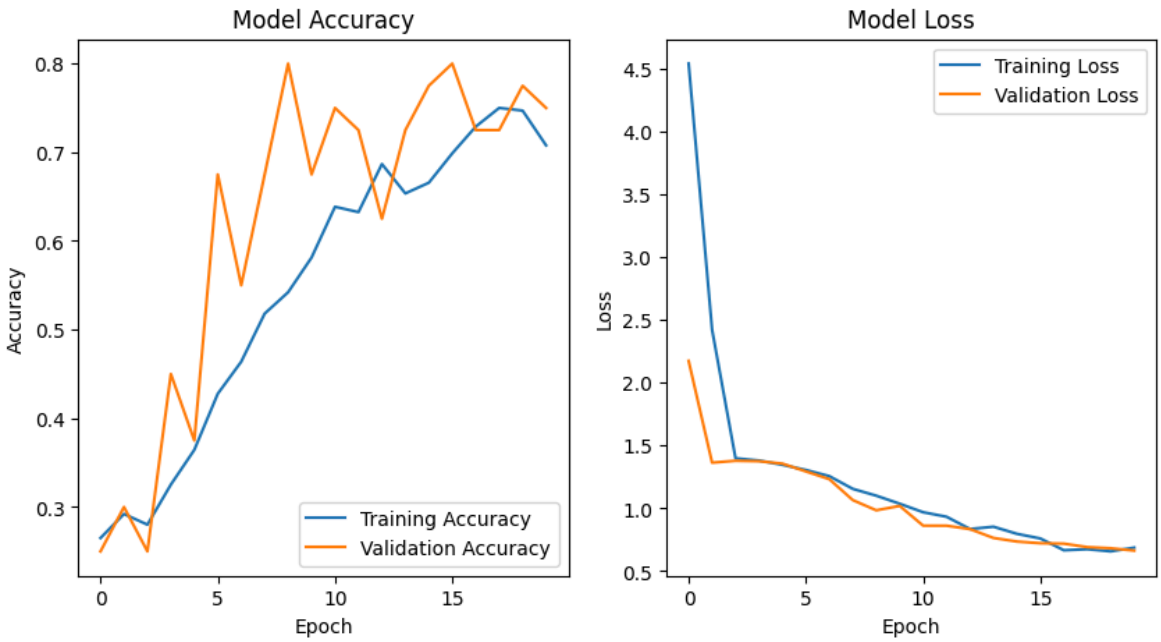
```
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 123ms/step - accuracy: 0.7600 - loss: 0.5487
Akurasi : 0.824999988079071
```



In [11]:
```
import pandas as pd
#Evaluasi
df_history = pd.DataFrame(history.history)
print(df_history.head())
plt.figure(figsize=(10,5))

# Accuracynya
plt.subplot(1, 2, 1)
plt.plot(df_history['accuracy'], label='Training Accuracy')
plt.plot(df_history['val_accuracy'], label='Validation Accuracy')
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()

# Lossnya
plt.subplot(1, 2, 2)
plt.plot(df_history['loss'], label='Training Loss')
plt.plot(df_history['val_loss'], label='Validation Loss')
plt.title('Model Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
```

```
   accuracy      loss  val_accuracy  val_loss
0  0.265060  4.538239         0.250  2.173126
1  0.292169  2.420999         0.300  1.364035
2  0.280120  1.398465         0.250  1.378685
3  0.325301  1.380896         0.450  1.375415
4  0.364458  1.346586         0.375  1.356057
```

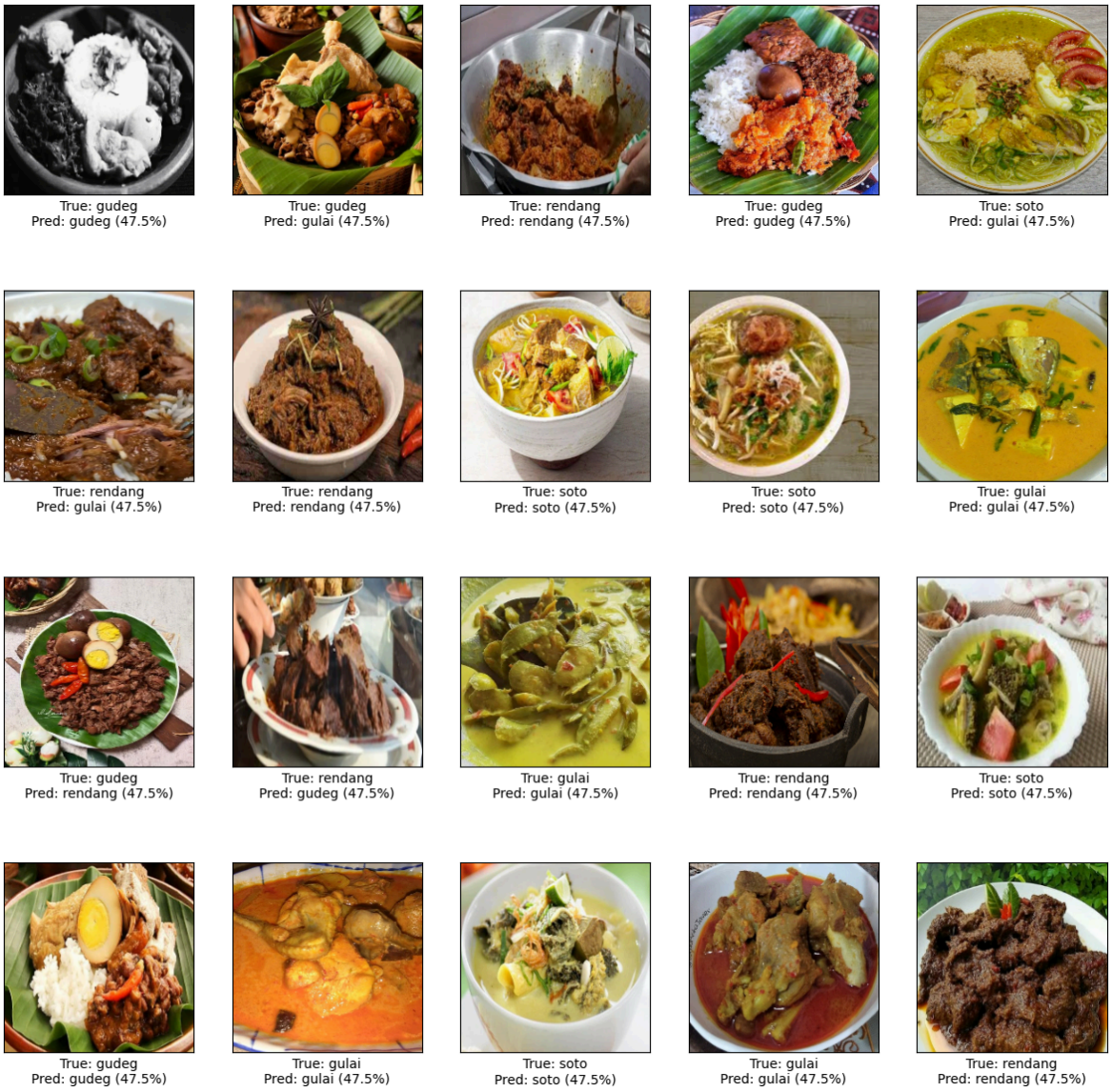
Out[11]: <matplotlib.legend.Legend at 0x23bd14db2c0>



Prediksi dari gambar di folder input

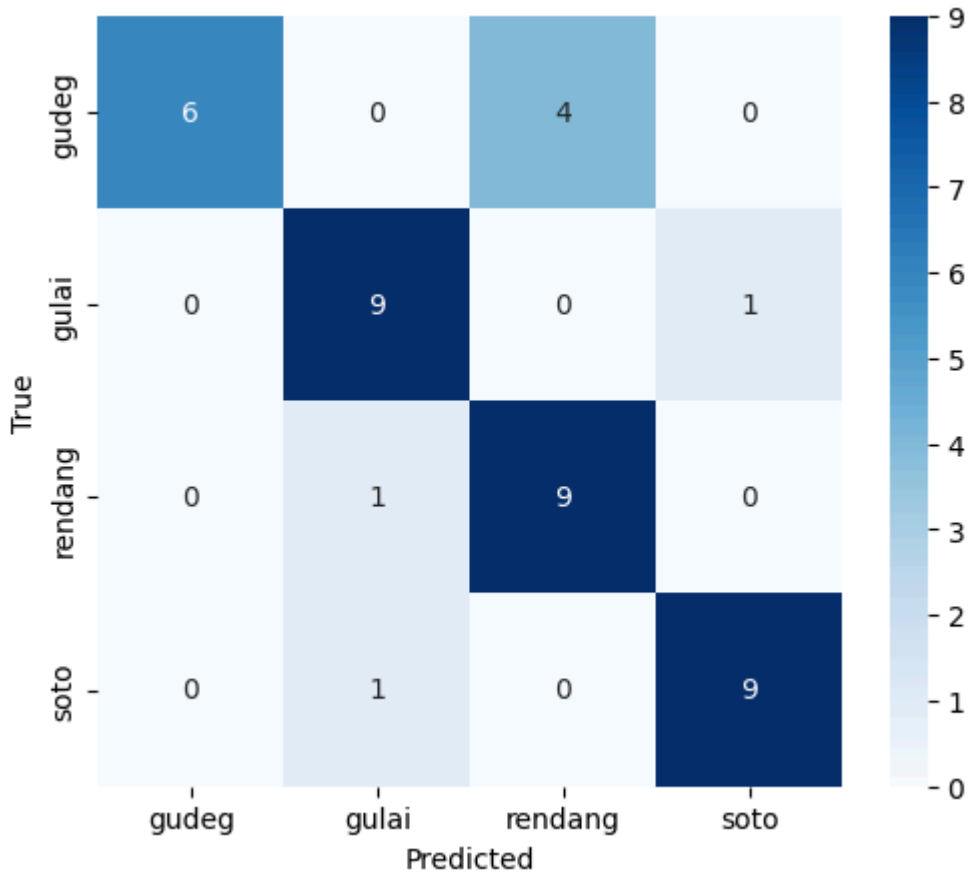
In [12]:
```
visualize_pred(input_ds)
```

```
2/2 ━━━━━━━━━━━━━━━━━━━━ 1s 344ms/step
```



In [13]:
```
import seaborn as sns
from sklearn.metrics import confusion_matrix
#Confusion Graph
y_true = np.concatenate([y for x, y in test_ds], axis=0)
y_pred = np.argmax(model.predict(test_ds), axis=1)

cm = confusion_matrix(y_true, y_pred)
plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```

```
4/4 ━━━━━━━━━━━━━━━━━━━━ 1s 110ms/step
```



In [17]:
```
#menyimpan model
model.save_weights('Custom_A_AlekSnack.weights.h5')
```

Python Code

```python
# streamlit_app.py
import streamlit as st
from PIL import Image
import numpy as np
import tensorflow as tf
import os
import pandas as pd

st.set_page_config(page_title="Prediksi Masakan Nusantara", layout="centered")

st.title("Prediksi Masakan Nusantara (gudeg | gulai | rendang | soto)")
st.write("Model: **ResNet50 (scratch)** — dilatih tanpa pretrained weights.")

# Load Model
MODEL_PATH = "resnet50_best.h5"

if not os.path.exists(MODEL_PATH):
    st.error(f"Model tidak ditemukan: {MODEL_PATH}. Pastikan file .h5 berada di folder yang sama.")
    st.stop()

@st.cache_resource
def load_model(path):
    return tf.keras.models.load_model(path)

model = load_model(MODEL_PATH)
class_names = ["gudeg", "gulai", "rendang", "soto"]

# Upload & Predict
uploaded_file = st.file_uploader("Upload gambar masakan (jpg/png)", type=["jpg","jpeg","png"])

if uploaded_file is not None:
    img = Image.open(uploaded_file).convert("RGB")
    st.image(img, caption="Gambar input", use_column_width=True)

    # preprocessing
    IMG_SIZE = model.input_shape[1] if model.input_shape and len(model.input_shape) > 1 else 224
    img_resized = img.resize((IMG_SIZE, IMG_SIZE))
    x = np.array(img_resized) / 255.0
    x = np.expand_dims(x, axis=0)

    # prediksi
    preds = model.predict(x)[0]
    top_idx = np.argmax(preds)

    st.markdown(f"### 🧑‍🍳 Prediksi: **{class_names[top_idx]}**")
    st.write(f"**Confidence:** {preds[top_idx]*100:.2f}%")

    # tampilkan 3 teratas
    top3 = np.argsort(preds)[-3:][::-1]
```

```
50        st.write("Top 3 kemungkinan:")
51        for i in top3:
52            st.write(f"- {class_names[i]} : {preds[i]*100:.2f}%")
53
54        # tampilkan grafik probabilitas
55        df = pd.DataFrame({
56            "class": class_names,
57            "probability": preds
58        }).sort_values(by="probability", ascending=False)
59        st.bar_chart(df.set_index("class"))
60
61    else:
62        st.info("Upload gambar untuk melihat hasil prediksi.")
63
64    st.markdown("---")
65    st.caption("© 2025 Prediksi Masakan Nusantara — Model: ResNet50 dari awal (tanpa pre
trained weights)")
66
```