LAMPIRAN

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.models import Sequential
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay, precision_score
import pandas as pd
```

In [2]:
```
IMAGE_SIZE = 256
BATCH_SIZE = 16
CHANNELS = 3
n_classes = 6

dataset = tf.keras.preprocessing.image_dataset_from_directory(
    "Penyakit Tanaman3",
    shuffle=True,
    image_size=(IMAGE_SIZE, IMAGE_SIZE),
    batch_size=BATCH_SIZE
)
```

```
Found 3734 files belonging to 6 classes.
```

In [3]:
```
class_names = dataset.class_names
print(f'Class names: {class_names}')
```

```
Class names: ['Cabai_Anthracnose', 'Cabai_Normal', 'Cabai_leaf_curl', 'Tomat_Normal',
'Tomat___Early_blight', 'Tomat___Late_blight']
```

In [4]:
```
count_per_class = {class_name: 0 for class_name in class_names}

for _, labels in dataset:
    for label in labels:
        count_per_class[class_names[label]] += 1

print(count_per_class)
```

```
{'Cabai_Anthracnose': 620, 'Cabai_Normal': 600, 'Cabai_leaf_curl': 621, 'Tomat_Normal
': 670, 'Tomat___Early_blight': 612, 'Tomat___Late_blight': 611}
```

In [5]:
```
plt.figure(figsize=(15, 15))
for images, labels in dataset.take(1):
    for i in range(16):
        ax = plt.subplot(4, 4, i + 1)
        plt.imshow(images[i].numpy().astype("uint8"))
        plt.title(f'{class_names[labels[i]]}')
        plt.axis("off")
plt.show()
```

[10]:
```python
def get_dataset_partitions_tf(ds, train_split=0.7, val_split=0.15, test_split=0.15, s
    ds_size = len(ds)
    if shuffle:
        ds = ds.shuffle(shuffle_size, seed=12)
    train_size = int(train_split * ds_size)
    val_size = int(val_split * ds_size)

    train_ds = ds.take(train_size)
    remaining_ds = ds.skip(train_size)
    val_ds = remaining_ds.take(val_size)
    test_ds = remaining_ds.skip(val_size)

    return train_ds, val_ds, test_ds

train_ds, val_ds, test_ds = get_dataset_partitions_tf(dataset)
```

In [11]:
```
AUTOTUNE = tf.data.AUTOTUNE
train_ds = train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)
test_ds = test_ds.cache().prefetch(buffer_size=AUTOTUNE)


resize_and_rescale = layers.Lambda(lambda x: x / 255.0)


data_augmentation = Sequential([
    layers.Lambda(lambda x: tf.image.random_flip_left_right(x)),
    layers.Lambda(lambda x: tf.image.random_flip_up_down(x)),
    layers.Lambda(lambda x: tf.image.random_brightness(x, max_delta=0.2)),
    layers.Lambda(lambda x: tf.image.random_contrast(x, 0.8, 1.2)),
])


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

In [12]:
```
model = Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(IMAGE_SIZE, IMAGE_SIZE,
    layers.MaxPooling2D(2, 2),

    layers.Conv2D(64, (3, 3), activation='relu',),
    layers.MaxPooling2D(2, 2),

    layers.Conv2D(128, (3, 3), activation='relu',),
    layers.MaxPooling2D(2, 2),

    layers.Conv2D(128, (3, 3), activation='relu',),
    layers.MaxPooling2D(2, 2),

    layers.Flatten(),
    layers.Dropout(0.5),

    layers.Dense(128, activation='relu',),
    layers.Dense(n_classes, activation='softmax')
])
```

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


model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 254, 254, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 127, 127, 32)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 125, 125, 64)      18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 62, 62, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 60, 60, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 30, 30, 128)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 28, 28, 128)       147584
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 14, 14, 128)       0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
dropout (Dropout)            (None, 25088)             0
_________________________________________________________________
dense (Dense)                (None, 128)               3211392
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 774
=================================================================
Total params: 3,452,998
Trainable params: 3,452,998
Non-trainable params: 0
_________________________________________________________________
```

In [14]:
```python
def preprocess_and_augment(image, label):
    image = resize_and_rescale(image)
    image = data_augmentation(image)
    return image, label


train_ds = train_ds.map(preprocess_and_augment, num_parallel_calls=AUTOTUNE)
val_ds = val_ds.map(lambda x, y: (resize_and_rescale(x), y), num_parallel_calls=AUTOT
test_ds = test_ds.map(lambda x, y: (resize_and_rescale(x), y), num_parallel_calls=AUT
```

In [15]:
```python
history = model.fit(
    train_ds,
    epochs=20,
    verbose=1,
    validation_data=val_ds,
    steps_per_epoch=len(train_ds),
    validation_steps=len(val_ds)
)
```

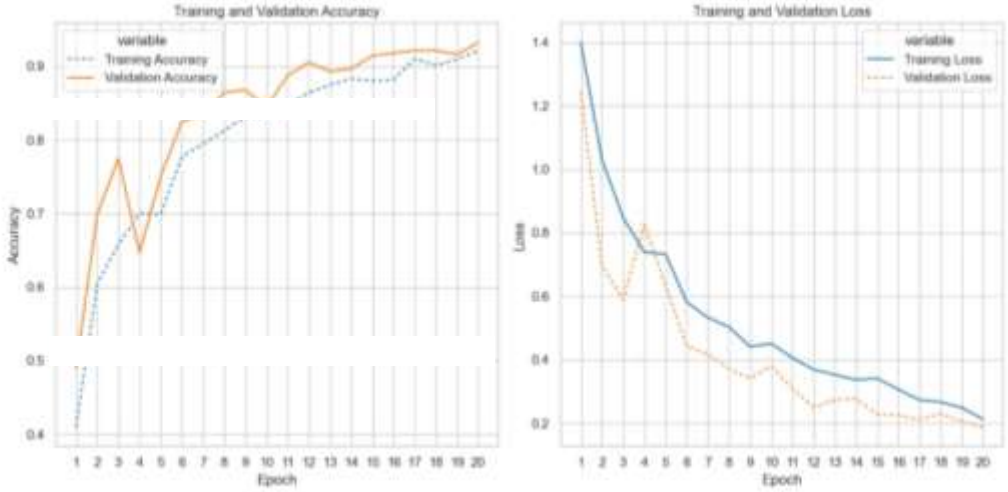
```
Epoch 1/20
163/163 [==============================] - 1157s 6s/step - loss: 1.6259 - accuracy: 0
.3063 - val_loss: 1.2437 - val_accuracy: 0.4893
Epoch 2/20
163/163 [==============================] - 701s 4s/step - loss: 1.1060 - accuracy: 0.
5839 - val_loss: 0.6995 - val_accuracy: 0.6982
Epoch 3/20
163/163 [==============================] - 622s 4s/step - loss: 0.8612 - accuracy: 0.
6589 - val_loss: 0.5875 - val_accuracy: 0.7750
Epoch 4/20
163/163 [==============================] - 545s 3s/step - loss: 0.7509 - accuracy: 0.
6978 - val_loss: 0.8270 - val_accuracy: 0.6464
Epoch 5/20
163/163 [==============================] - 554s 3s/step - loss: 0.7418 - accuracy: 0.
7000 - val_loss: 0.6360 - val_accuracy: 0.7500
Epoch 6/20
163/163 [==============================] - 551s 3s/step - loss: 0.6128 - accuracy: 0.
7701 - val_loss: 0.4423 - val_accuracy: 0.8250
Epoch 7/20
163/163 [==============================] - 548s 3s/step - loss: 0.5238 - accuracy: 0.
7985 - val_loss: 0.4174 - val_accuracy: 0.8339
Epoch 8/20
163/163 [==============================] - 551s 3s/step - loss: 0.4797 - accuracy: 0.
8210 - val_loss: 0.3718 - val_accuracy: 0.8643
Epoch 9/20
163/163 [==============================] - 550s 3s/step - loss: 0.4350 - accuracy: 0.
8296 - val_loss: 0.3413 - val_accuracy: 0.8679
Epoch 10/20
163/163 [==============================] - 527s 3s/step - loss: 0.4338 - accuracy: 0.
8329 - val_loss: 0.3795 - val_accuracy: 0.8446
Epoch 11/20
163/163 [==============================] - 451s 3s/step - loss: 0.3859 - accuracy: 0.
8510 - val_loss: 0.3089 - val_accuracy: 0.8875
Epoch 12/20
163/163 [==============================] - 384s 2s/step - loss: 0.3765 - accuracy: 0.
8665 - val_loss: 0.2496 - val_accuracy: 0.9054
Epoch 13/20
163/163 [==============================] - 373s 2s/step - loss: 0.3117 - accuracy: 0.
8860 - val_loss: 0.2728 - val_accuracy: 0.8929
Epoch 14/20
163/163 [==============================] - 383s 2s/step - loss: 0.3244 - accuracy: 0.
8799 - val_loss: 0.2771 - val_accuracy: 0.8964
Epoch 15/20
163/163 [==============================] - 374s 2s/step - loss: 0.3882 - accuracy: 0.
8648 - val_loss: 0.2269 - val_accuracy: 0.9143
Epoch 16/20
163/163 [==============================] - 374s 2s/step - loss: 0.2895 - accuracy: 0.
8853 - val_loss: 0.2249 - val_accuracy: 0.9179
Epoch 17/20
163/163 [==============================] - 377s 2s/step - loss: 0.2806 - accuracy: 0.
9134 - val_loss: 0.2116 - val_accuracy: 0.9214
Epoch 18/20
163/163 [==============================] - 378s 2s/step - loss: 0.2655 - accuracy: 0.
8931 - val_loss: 0.2289 - val_accuracy: 0.9214
Epoch 19/20
163/163 [==============================] - 374s 2s/step - loss: 0.2465 - accuracy: 0.
9050 - val_loss: 0.2049 - val_accuracy: 0.9161
Epoch 20/20
163/163 [==============================] - 374s 2s/step - loss: 0.1986 - accuracy: 0.
9236 - val_loss: 0.1876 - val_accuracy: 0.9321
```

In [16]:
```
scores = model.evaluate(test_ds)
```

```
36/36 [==============================] - 165s 655ms/step - loss: 0.2314 - accuracy:
0.9132
```

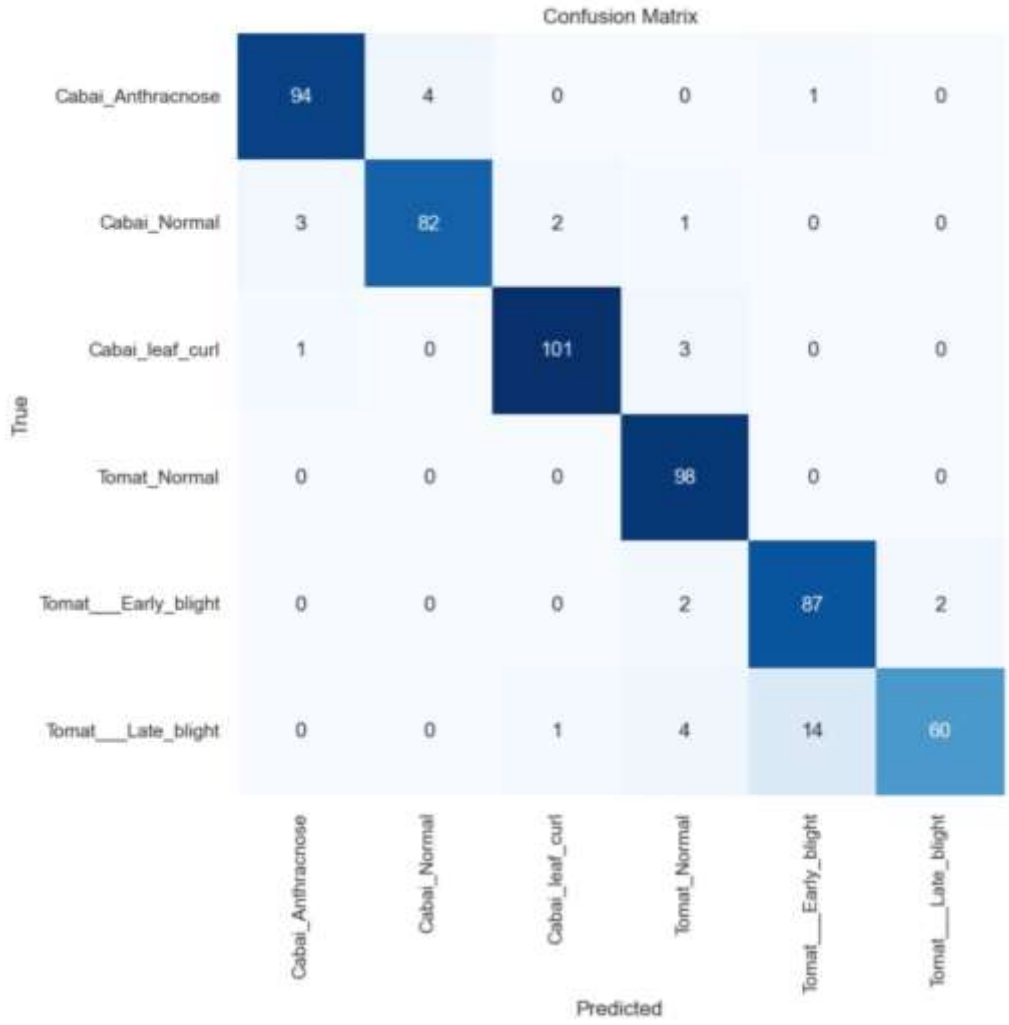
In [17]:
```
sns.set(style="whitegrid")
```

In [18]:
```
# Konversi history ke DataFrame
history_df = pd.DataFrame({
    'Epoch': range(1, len(history.history['loss']) + 1),
    'Training Accuracy': history.history['accuracy'],
    'Validation Accuracy': history.history['val_accuracy'],
    'Training Loss': history.history['loss'],
    'Validation Loss': history.history['val_loss']
})

# Pastikan Epoch dalam bentuk integer
history_df['Epoch'] = history_df['Epoch'].astype(int)

# Plot menggunakan DataFrame
plt.figure(figsize=(12, 6))

# Plot Accuracy
plt.subplot(1, 2, 1)
sns.lineplot(
    data=history_df.melt(id_vars='Epoch',
                         value_vars=['Training Accuracy', 'Validation Accuracy']),
    x='Epoch',
    y='value',
    hue='variable',
    style='variable',
    dashes=[(2,2), ()],
    palette="tab10"
)
plt.title('Training and Validation Accuracy')
plt.ylabel('Accuracy')
plt.xticks(history_df['Epoch'])

# Plot Loss
plt.subplot(1, 2, 2)
sns.lineplot(
    data=history_df.melt(id_vars='Epoch',
                         value_vars=['Training Loss', 'Validation Loss']),
    x='Epoch',
    y='value',
    hue='variable',
    style='variable',
    dashes=[(), (2,2)],
    palette="tab10"
)
plt.title('Training and Validation Loss')
plt.ylabel('Loss')
plt.xticks(history_df['Epoch'])

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



In [19]:
```
predictions = model.predict(val_ds)
predicted_classes = np.argmax(predictions, axis=1)


true_classes = np.concatenate([y for x, y in val_ds], axis=0)
```

In [20]:
```
cm = confusion_matrix(true_classes, predicted_classes)
cm_display = ConfusionMatrixDisplay(cm, display_labels=class_names)

plt.figure(figsize=(8, 8))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False,
            xticklabels=class_names, yticklabels=class_names)
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```



In [21]:
```
precision = precision_score(true_classes, predicted_classes, average=None)
recall = recall_score(true_classes, predicted_classes, average=None)
f1 = f1_score(true_classes, predicted_classes, average=None)
```

In [22]:
```
metrics_df = pd.DataFrame({
    'Class': class_names,
    'Precision': precision,
    'Recall': recall,
    'F1-Score': f1
})

print(metrics_df)
```

```
                Class  Precision    Recall  F1-Score
0    Cabai_Anthracnose   0.959184  0.949495  0.954315
1        Cabai_Normal    0.953488  0.931818  0.942529
2      Cabai_leaf_curl   0.971154  0.961905  0.966507
3        Tomat_Normal    0.907407  1.000000  0.951456
4  Tomat___Early_blight   0.852941  0.956044  0.901554
5   Tomat___Late_blight   0.967742  0.759494  0.851064
```

In [23]:
```
def predict(model, img):
    img_array = tf.keras.preprocessing.image.img_to_array(img)
    img_array = tf.expand_dims(img_array, 0)

    predictions = model.predict(img_array)

    predicted_class = class_names[np.argmax(predictions[0])]
    confidence = round(100 * (np.max(predictions[0])), 2)
    return predicted_class, confidence
```

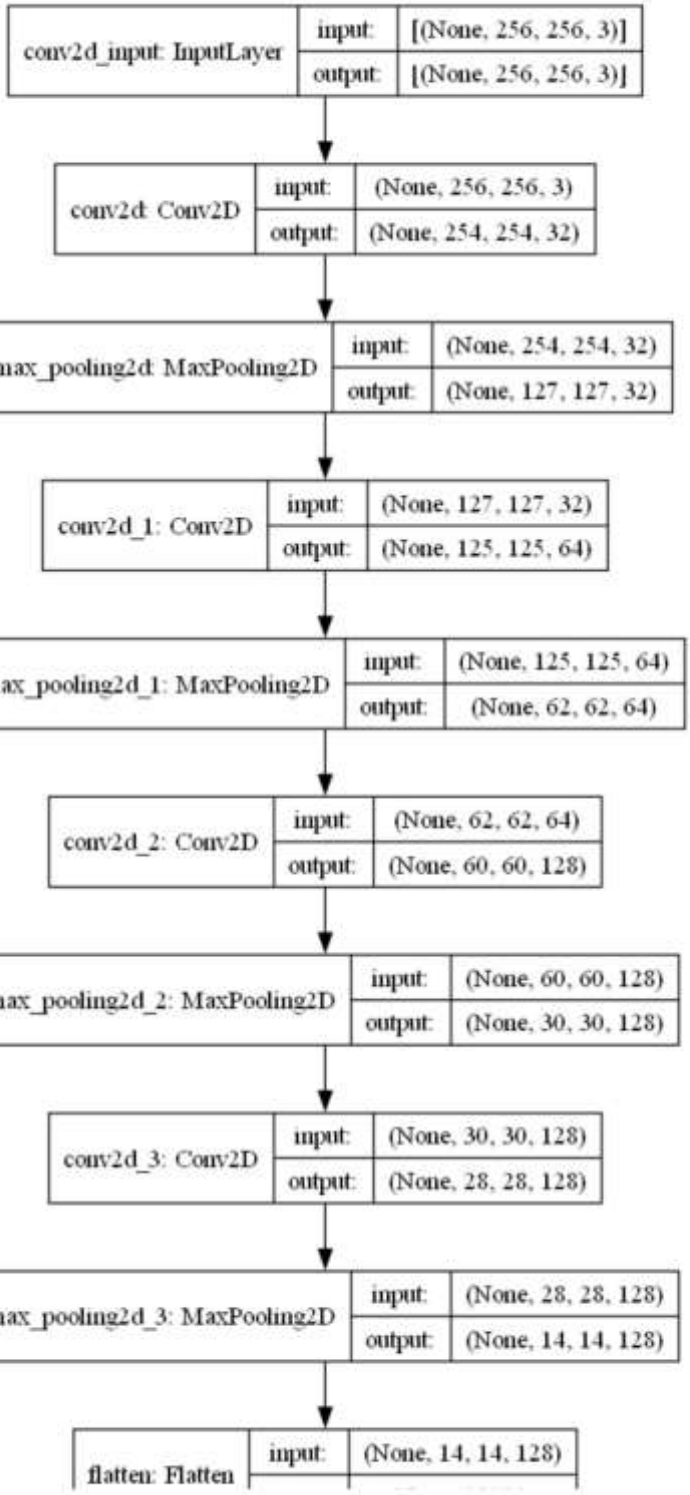
In [24]:
```
def predict(model, image):

    img_array = np.expand_dims(image, axis=0)

    predictions = model.predict(img_array)


    predicted_class_idx = np.argmax(predictions[0])
    predicted_class = class_names[predicted_class_idx]


    confidence = 100 * np.max(predictions[0])

    return predicted_class, confidence


plt.figure(figsize=(15, 15))
for images, labels in test_ds.take(1):
    for i in range(9):
        ax = plt.subplot(3, 3, i + 1)


        img = images[i].numpy()


        if img.max() <= 1.0:
            img = (img * 255).astype("uint8")


        plt.imshow(img)


        predicted_class, confidence = predict(model, images[i].numpy())
        actual_class = class_names[labels[i]]


        plt.title(f"Actual: {actual_class},\nPredicted: {predicted_class}\nConfidence
        plt.axis("off")

plt.show()
```

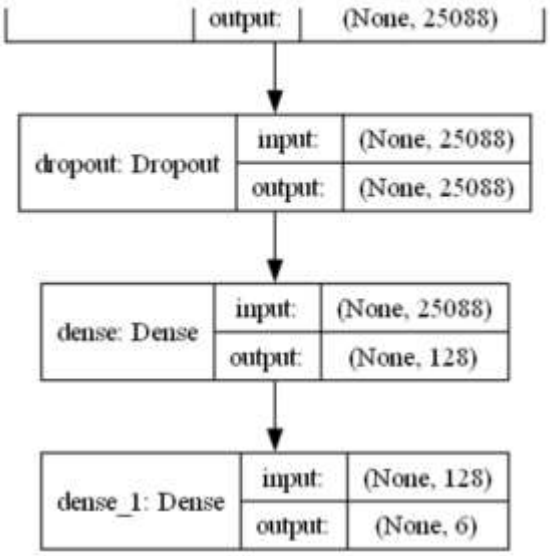
In [25]:
```
model.save("model_penyakit_tanaman17.h5")
```

In [26]:
```
import pydot
```

In [27]:
```
tf.keras.utils.plot_model(model, to_file='model_architecture.png', show_shapes=True,
```

Out[27]:

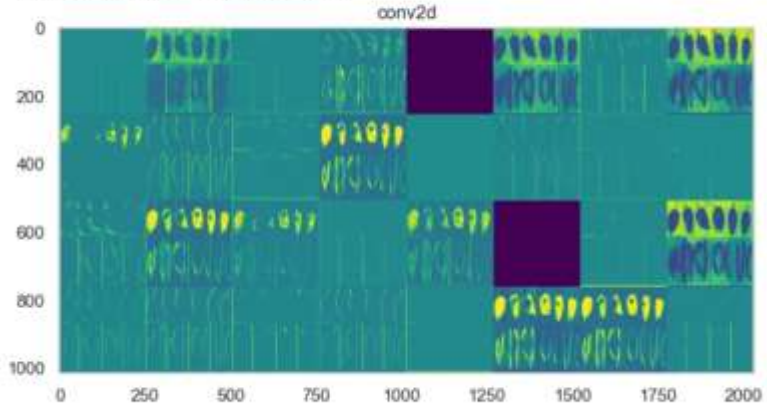

conv2d_input: InputLayer
input: [(None, 256, 256, 3)]
output: [(None, 256, 256, 3)]
conv2d: Conv2D
input: (None, 256, 256, 3)
output: (None, 254, 254, 32)
max_pooling2d: MaxPooling2D
input: (None, 254, 254, 32)
output: (None, 127, 127, 32)
conv2d_1: Conv2D
input: (None, 127, 127, 32)
output: (None, 125, 125, 64)
max_pooling2d_1: MaxPooling2D
input: (None, 125, 125, 64)
output: (None, 62, 62, 64)
conv2d_2: Conv2D
input: (None, 62, 62, 64)
output: (None, 60, 60, 128)
max_pooling2d_2: MaxPooling2D
input: (None, 60, 60, 128)
output: (None, 30, 30, 128)
conv2d_3: Conv2D
input: (None, 30, 30, 128)
output: (None, 28, 28, 128)
max_pooling2d_3: MaxPooling2D
input: (None, 28, 28, 128)
output: (None, 14, 14, 128)
flatten: Flatten
input: (None, 14, 14, 128)

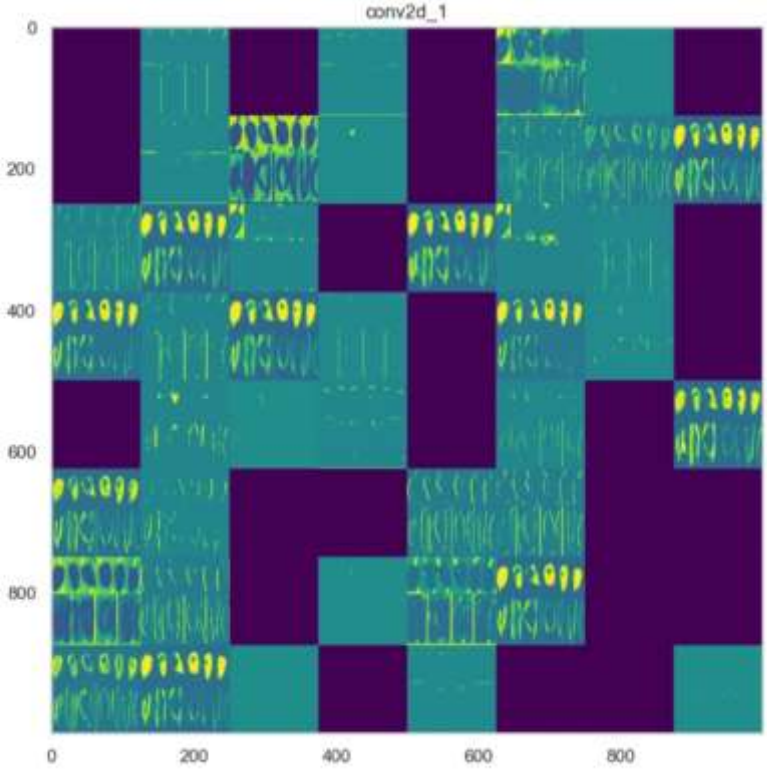


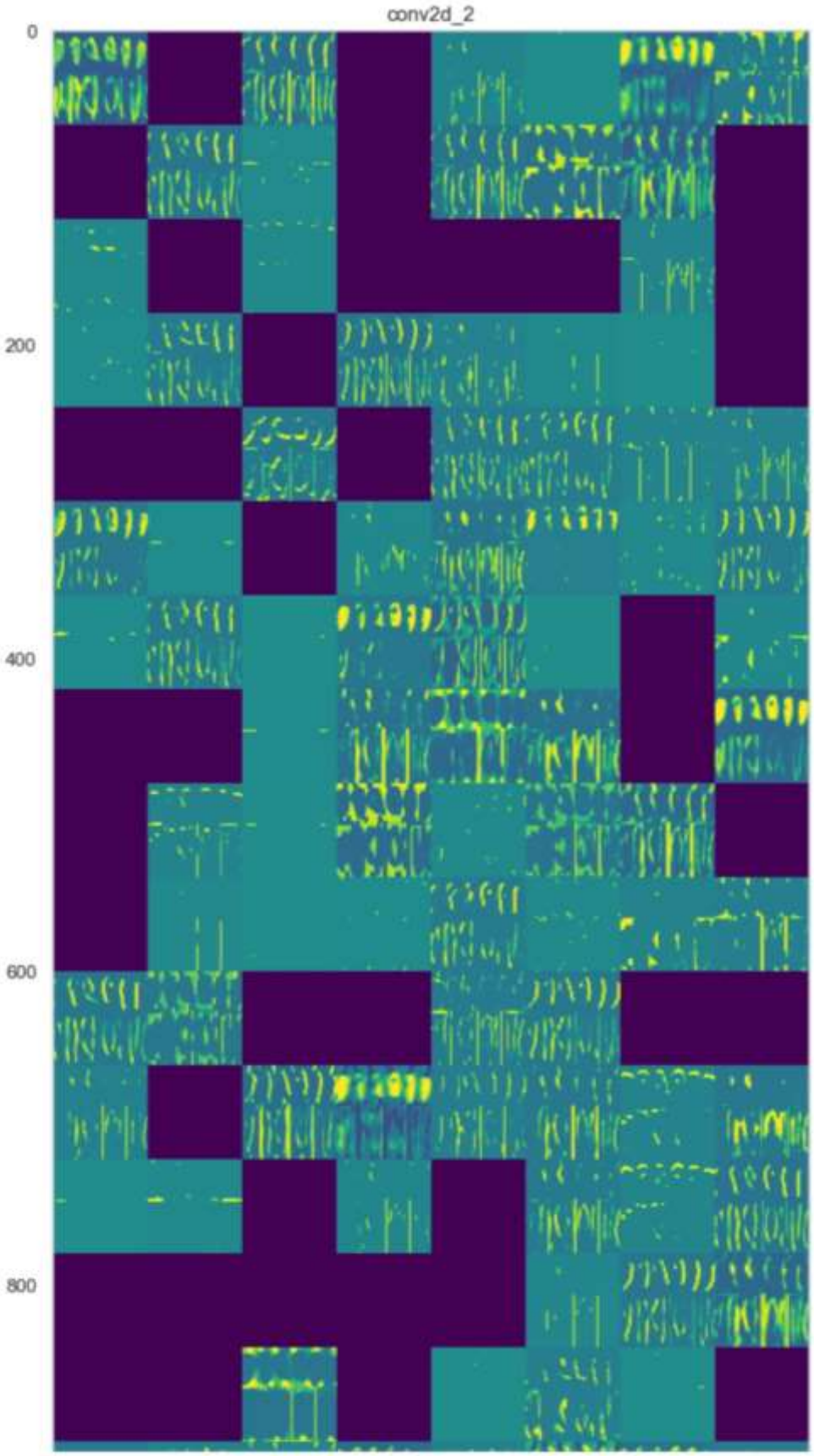
In [29]:

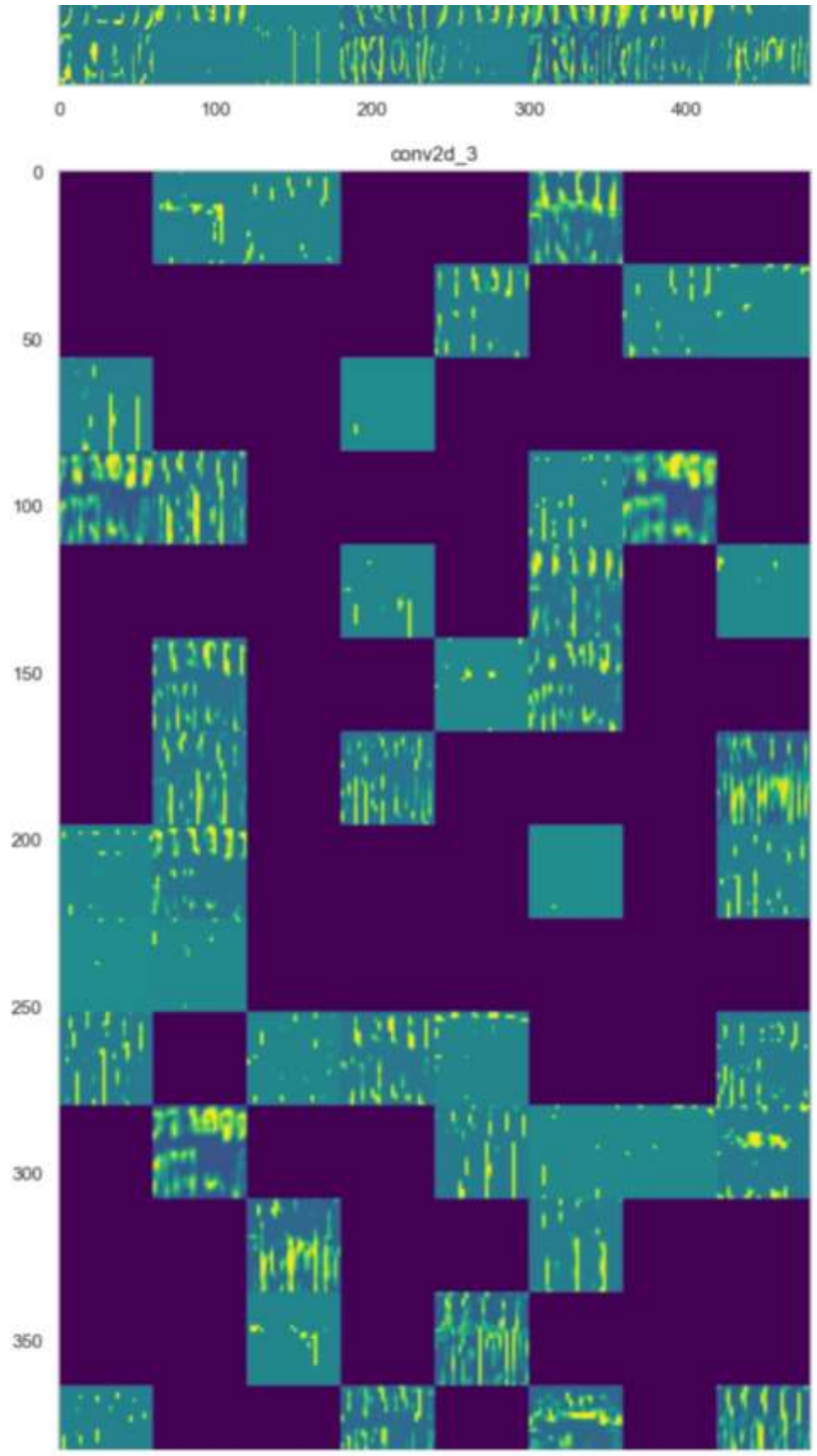
```python
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf

def visualize_activations(model, image, layer_names):
    layer_outputs = [model.get_layer(name).output for name in layer_names]
    activation_model = tf.keras.models.Model(inputs=model.input, outputs=layer_output

    img_array = np.expand_dims(image, axis=0)
    activations = activation_model.predict(img_array)

    for layer_name, layer_activation in zip(layer_names, activations):
        n_features = layer_activation.shape[-1]
        size = layer_activation.shape[1]
        n_cols = n_features // 8
        display_grid = np.zeros((size * n_cols, 8 * size))

        for col in range(n_cols):
            for row in range(8):
                channel_image = layer_activation[0, :, :, col * 8 + row]
                # Normalisasi aktivasi
                channel_image -= channel_image.mean()
                channel_image /= channel_image.std()
                channel_image *= 64
                channel_image += 128
                channel_image = np.clip(channel_image, 0, 255).astype('uint8')
                display_grid[col * size : (col + 1) * size, row * size : (row + 1) *

        scale = 1. / size
        plt.figure(figsize=(scale * display_grid.shape[1], scale * display_grid.shape
        plt.title(layer_name)
        plt.grid(False)
        plt.imshow(display_grid, aspect='auto', cmap='viridis')  # Gunakan skala warn
        plt.show()

# Cek nama lapisan model
for layer in model.layers:
    print(layer.name)

# Gunakan nama lapisan yang benar
layer_names = ['conv2d', 'conv2d_1', 'conv2d_2', 'conv2d_3']
for images, labels in test_ds.take(1):
```

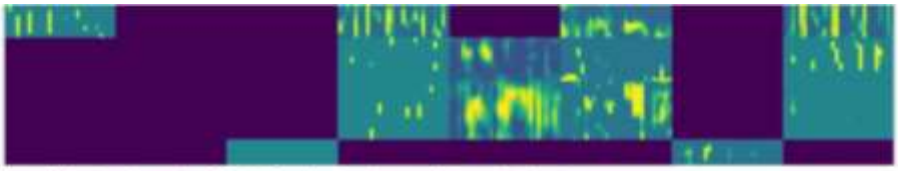
```
    break
conv2d_1
max_pooling2d_1
conv2d_2
max_pooling2d_2
conv2d_3
max_pooling2d_3
flatten
dropout
dense
dense_1
C:\Users\ASUS\AppData\Local\Temp\ipykernel_10472\1605610153.py:23: RuntimeWarning: in
valid value encountered in divide
  channel_image /= channel_image.std()
```






conv2d_2
0
200
400
600
800


0
100
200
300
400
conv2d_3
0
50
100
150
200
250
300
350



In [31]:
```
from ipywidgets import FileUpload, Button, VBox, Output
from IPython.display import display, clear_output
from PIL import Image
import io
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf


def visualize_activations(model, image, layer_names):
    layer_outputs = [model.get_layer(name).output for name in layer_names]
    activation_model = tf.keras.models.Model(inputs=model.input, outputs=layer_output

    img_array = np.expand_dims(image, axis=0)
    activations = activation_model.predict(img_array)

    for layer_name, layer_activation in zip(layer_names, activations):
        n_features = layer_activation.shape[-1]
        size = layer_activation.shape[1]
        n_cols = n_features // 8
        display_grid = np.zeros((size * n_cols, 8 * size))

        for col in range(n_cols):
            for row in range(8):
                channel_index = col * 8 + row
                if channel_index >= n_features:
                    break

                channel_image = layer_activation[0, :, :, channel_index]
                channel_image -= channel_image.mean()
                channel_image /= channel_image.std()
                channel_image *= 64
                channel_image += 128
                channel_image = np.clip(channel_image, 0, 255).astype('uint8')
                display_grid[col * size : (col + 1) * size,
                             row * size : (row + 1) * size] = channel_image

        scale = 1. / size
        plt.figure(figsize=(scale * display_grid.shape[1], scale * display_grid.shape
        plt.title(layer_name)
        plt.grid(False)
        plt.imshow(display_grid, aspect='auto', cmap='viridis')
        plt.show()


upload = FileUpload(accept='image/*', multiple=False)
predict_btn = Button(description='Predict & Visualize')
output = Output()


layer_names = ['conv2d', 'conv2d_1', 'conv2d_2', 'conv2d_3']

def on_predict_clicked(b):
    global last_img_array

    with output:
        clear_output()
        if not upload.value:
            print("Upload gambar terlebih dahulu")
            return
```

```
            return

        uploaded_file = next(iter(upload.value.values()))
        img = Image.open(io.BytesIO(uploaded_file['content']))
        img = img.resize((IMAGE_SIZE, IMAGE_SIZE))
        if img.mode != 'RGB':
            img = img.convert('RGB')

        img_array = np.array(img) / 255.0
        last_img_array = np.expand_dims(img_array, axis=0)

        prediction = model.predict(np.expand_dims(img_array, axis=0))
        predicted_class = class_names[np.argmax(prediction[0])]
        confidence = np.max(prediction[0]) * 100

        plt.figure(figsize=(6,6))
        plt.imshow(img)
        plt.title(f"Predicted: {predicted_class}\nConfidence: {confidence:.2f}%")
        plt.axis('off')
        plt.show()

        print("\nVisualisasi Filter:")
        visualize_activations(model, img_array, layer_names)

VBox(children=(FileUpload(value={}, accept='image/*', description='Upload'), Button(d
escription='Predict & Vis…
predict_btn.on_click(on_predict_clicked)
```

In [34]:
```
def display_input_pixel_values(image):

    print("Nilai Piksel Gambar Input:")
    print(image)

if 'last_img_array' in locals():

    display_input_pixel_values(last_img_array[0])
else:
    print("Gambar input belum tersedia. Pastikan untuk mengunggah gambar terlebih dah
```

```
Nilai Piksel Gambar Input:
[[[0.35294118 0.5254902  0.04313725]
  [0.2627451  0.42352941 0.00392157]
  [0.23137255 0.39215686 0.        ]
  ...
  [0.29803922 0.45882353 0.04313725]
  [0.24313725 0.38431373 0.01568627]
  [0.21960784 0.32941176 0.00392157]]

 [[0.32156863 0.47843137 0.01960784]
  [0.25490196 0.40392157 0.        ]
  [0.23137255 0.38823529 0.        ]
  ...
  [0.29803922 0.45490196 0.04705882]
  [0.24705882 0.38823529 0.01960784]
  [0.22352941 0.32941176 0.00784314]]

 [[0.31372549 0.45098039 0.02352941]
  [0.26666667 0.40392157 0.        ]
  [0.24705882 0.40392157 0.00392157]
  ...
  [0.28627451 0.43529412 0.03137255]
  [0.27058824 0.40392157 0.03529412]
  [0.25098039 0.35294118 0.03137255]]

 ...

 [[0.23921569 0.43137255 0.00392157]
  [0.23529412 0.43137255 0.        ]
  [0.23137255 0.42352941 0.        ]
  ...
  [0.52156863 0.7372549  0.12941176]
  [0.51764706 0.73333333 0.12941176]
  [0.5254902  0.74117647 0.13333333]]

 [[0.23921569 0.43137255 0.00392157]
  [0.23529412 0.43137255 0.        ]
  [0.23137255 0.42352941 0.        ]
  ...
  [0.51764706 0.7372549  0.13333333]
  [0.52156863 0.74117647 0.13333333]
  [0.52156863 0.7372549  0.13333333]]

 [[0.23921569 0.43137255 0.00392157]
  [0.23529412 0.43137255 0.        ]
  [0.23137255 0.42352941 0.        ]
  ...
  [0.52156863 0.73333333 0.13333333]
  [0.5254902  0.74117647 0.1372549 ]
  [0.51764706 0.73333333 0.12941176]]]
```

In [35]:
```
def display_layer_pixel_values(model, image, layer_names):

    layer_outputs = [model.get_layer(name).output for name in layer_names]
    activation_model = tf.keras.models.Model(inputs=model.input, outputs=layer_output

    img_array = np.expand_dims(image, axis=0)
    activations = activation_model.predict(img_array)

    for layer_name, layer_activation in zip(layer_names, activations):
        print(f"\nNilai Piksel dari Lapisan {layer_name}:")
        print(layer_activation[0])

if 'last_img_array' in locals():

    display_layer_pixel_values(model, last_img_array[0], layer_names)  # Menampilkan
else:
    print("Gambar input belum tersedia. Pastikan untuk mengunggah gambar terlebih dah
```

```
WARNING:tensorflow:6 out of the last 18 calls to <function Model.make_predict_functio
n.<locals>.predict_function at 0x0000028748F6C5E0> triggered tf.function retracing. T
racing is expensive and the excessive number of tracings could be due to (1) creating
@tf.function repeatedly in a loop, (2) passing tensors with different shapes, (3) pas
sing Python objects instead of tensors. For (1), please define your @tf.function outs
ide of the loop. For (2), @tf.function has experimental_relax_shapes=True option that
relaxes argument shapes that can avoid unnecessary retracing. For (3), please refer t
o https://www.tensorflow.org/guide/function#controlling_retracing and https://www.ten
sorflow.org/api_docs/python/tf/function for  more details.

Nilai Piksel dari Lapisan conv2d:
[[[0.         0.         0.         ... 0.07843214 0.12723565 0.        ]
  [0.         0.         0.         ... 0.05929527 0.13236126 0.        ]
  [0.         0.         0.         ... 0.0462931  0.1265166  0.        ]
  ...
  [0.         0.         0.         ... 0.07090998 0.15771806 0.        ]
  [0.         0.         0.         ... 0.06890035 0.13311158 0.        ]
  [0.         0.         0.         ... 0.06546569 0.1095067  0.        ]]

 [[0.         0.         0.         ... 0.08585804 0.13734621 0.        ]
  [0.         0.         0.         ... 0.06922228 0.1388487  0.        ]
  [0.         0.         0.         ... 0.05467549 0.12722987 0.        ]
  ...
  [0.         0.         0.         ... 0.07191974 0.16017951 0.        ]
  [0.         0.         0.         ... 0.07045607 0.13758746 0.        ]
  [0.         0.         0.         ... 0.06646451 0.11457299 0.        ]]

 [[0.         0.         0.         ... 0.09526077 0.1467449  0.        ]
  [0.         0.         0.         ... 0.08001351 0.14987414 0.        ]
  [0.         0.         0.         ... 0.06422183 0.13722503 0.        ]
  ...
  [0.         0.         0.         ... 0.07591417 0.16108851 0.        ]
  [0.         0.         0.         ... 0.07849728 0.1445017  0.        ]
  [0.         0.         0.         ... 0.0755935  0.12198657 0.        ]]

 ...
```

```
...

[[0.        0.        0.        ... 0.04129378 0.13966304 0.        ]
 [0.        0.        0.        ... 0.03969447 0.13500425 0.        ]
 [0.        0.        0.        ... 0.03593701 0.12803592 0.        ]
 ...
 [0.        0.        0.        ... 0.14463438 0.22886808 0.        ]
 [0.        0.        0.        ... 0.14269935 0.22660886 0.        ]
 [0.        0.        0.        ... 0.1424301  0.22586283 0.        ]]

[[0.        0.        0.        ... 0.04121295 0.14032587 0.        ]
 [0.        0.        0.        ... 0.04001098 0.1365628  0.        ]
 [0.        0.        0.        ... 0.03628977 0.12888902 0.        ]
 ...
 [0.        0.        0.        ... 0.14394939 0.22744946 0.        ]
 [0.        0.        0.        ... 0.14167    0.2273028  0.        ]
 [0.        0.        0.        ... 0.1420828  0.2273046  0.        ]]

[[0.        0.        0.        ... 0.04121295 0.14032587 0.        ]
 [0.        0.        0.        ... 0.04001098 0.1365628  0.        ]
 [0.        0.        0.        ... 0.03682302 0.12931733 0.        ]
 ...
 [0.        0.        0.        ... 0.14287643 0.22567064 0.        ]
 [0.        0.        0.        ... 0.14103658 0.22590262 0.        ]
 [0.        0.        0.        ... 0.14027786 0.22759871 0.        ]]]

Nilai Piksel dari Lapisan conv2d_1:
[[[0.        0.42704338 0.        ... 0.        0.        0.        ]
  [0.        0.45538583 0.        ... 0.        0.        0.        ]
  [0.        0.47075775 0.        ... 0.        0.        0.        ]
  ...
    [0.        0.21923238 0.        ... 0.        0.        0.        ]
    [0.        0.26459095 0.        ... 0.        0.        0.        ]
    [0.        0.32835904 0.        ... 0.        0.        0.        ]]

   [[0.        0.43583182 0.        ... 0.        0.        0.        ]
    [0.        0.46242574 0.        ... 0.        0.        0.        ]
    [0.        0.4756548  0.        ... 0.        0.        0.        ]
    ...
    [0.        0.17755371 0.        ... 0.        0.        0.        ]
    [0.        0.2211492  0.        ... 0.        0.        0.        ]
    [0.        0.28832406 0.        ... 0.        0.        0.        ]]

   [[0.        0.43970212 0.        ... 0.        0.        0.        ]
    [0.        0.46652806 0.        ... 0.        0.        0.        ]
    [0.        0.48139933 0.        ... 0.        0.        0.        ]
    ...
    [0.        0.16436653 0.        ... 0.        0.        0.        ]
    [0.        0.20258477 0.        ... 0.        0.        0.        ]
    [0.        0.2810264  0.        ... 0.        0.        0.        ]]

   ...

   [[0.        0.46250156 0.        ... 0.        0.        0.        ]
    [0.        0.4576697  0.        ... 0.        0.        0.        ]
    [0.        0.44526953 0.        ... 0.        0.        0.        ]
    ...
    [0.        0.6498414  0.        ... 0.        0.        0.        ]
    [0.        0.64322877 0.        ... 0.        0.        0.        ]
    [0.        0.6109681  0.        ... 0.        0.        0.        ]]

   [[0.        0.46006057 0.        ... 0.        0.        0.        ]
    [0.        0.45574686 0.        ... 0.        0.        0.        ]
    [0.        0.44824025 0.        ... 0.        0.        0.        ]
    ...
    [0.        0.6410495  0.        ... 0.        0.        0.        ]
    [0.        0.64080644 0.        ... 0.        0.        0.        ]
    [0.        0.61549425 0.        ... 0.        0.        0.        ]]

   [[0.        0.45896354 0.        ... 0.        0.        0.        ]
    [0.        0.4524893  0.        ... 0.        0.        0.        ]
    [0.        0.4480042  0.        ... 0.        0.        0.        ]
```

```
 ...
 [0.         0.62614906 0.         ... 0.         0.         0.        ]
 [0.         0.6276101  0.         ... 0.         0.         0.        ]
 [0.         0.6136365  0.         ... 0.         0.         0.        ]]]

Nilai Piksel dari Lapisan conv2d_2:
[[[0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  ...
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]]

 [[0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  ...
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]]

 [[0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]

  ...
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]]

 ...

 [[0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  ...
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]]

 [[0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  ...
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]]

 [[0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  ...
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]
  [0. 0. 0. ... 0. 0. 0.]]]
```

```
[[0.         0.         0.1327901  ... 0.         0.         0.        ]
 [0.         0.         0.08507631 ... 0.         0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]
 ...
 [0.         0.         0.41085976 ... 0.         0.         0.        ]
 [0.         0.         0.4167137  ... 0.         0.         0.        ]
 [0.         0.         0.4147722  ... 0.         0.         0.        ]]

[[0.         0.         0.05771968 ... 0.         0.         0.        ]
 [0.         0.         0.02009986 ... 0.         0.         0.        ]
 [0.         0.         0.         ... 0.         0.         0.        ]
 ...
 [0.         0.         0.40799367 ... 0.         0.         0.        ]
 [0.         0.         0.41860104 ... 0.         0.         0.        ]
 [0.         0.         0.42549193 ... 0.         0.         0.        ]]]
```

In [ ]:

APP.PY

```
import matplotlib

matplotlib.use('Agg')

from flask import Flask, render_template, request, jsonify

from werkzeug.utils import secure_filename

import pandas as pd

import numpy as np

import os

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Activation, Dropout, ReLU

from PIL import Image

from fungsi import make_model

import matplotlib.pyplot as plt

import io

import base64

import json

from flask_socketio import SocketIO, emit  # type: ignore

import cv2


app = Flask(__name__, static_url_path='/static')

app.config['MAX_CONTENT_LENGTH'] = 6 * 1024 * 1024

app.config['UPLOAD_EXTENSIONS'] = ['.jpg', '.JPG', '.jpeg', '.png']

app.config['UPLOAD_PATH'] = './static/images/uploads/'



socketio = SocketIO(app)


model = None

n_classes = 6

Class_Penyakit = ['Cabai Anthracnose', 'Cabai Normal', 'Cabai leaf curl', 'Tomat Normal', 'Tomat Early blight', 'Tomat Late Blight']


@app.route("/")

def beranda():

    return render_template('index.html')



@app.route("/api/deteksi", methods=['POST'])

def apiDeteksi():

    hasil_prediksi = '(none)'

    confidence_score = 0.0

    gambar_prediksi = '(none)'

    detail_proses = []

    grafik_data = {}

    logits = []

    probabilities = []


    uploaded_file = request.files['file']

    filename = secure_filename(uploaded_file.filename)


    if filename and os.path.splitext(filename)[1] in app.config['UPLOAD_EXTENSIONS']:

        gambar_prediksi = os.path.join(app.config['UPLOAD_PATH'], filename)

        uploaded_file.save(gambar_prediksi)


        try:

            test_image = Image.open(gambar_prediksi).convert("RGB").resize((256, 256))

            test_image_x = np.expand_dims(np.array(test_image), axis=0).astype('float32') / 255.0


            logits_model = tf.keras.Model(inputs=model.input, outputs=model.get_layer('logits').output)

            logits = logits_model.predict(test_image_x)[0].tolist()
```

```
        y_pred = model.predict(test_image_x)
        probabilities = y_pred[0].tolist()

        pred_class = np.argmax(y_pred)
        confidence_score =
round(float(np.max(y_pred)), 4)
        hasil_prediksi = Class_Penyakit[pred_class]

        detail_proses.append(f"Kelas dengan
probabilitas tertinggi: {hasil_prediksi} dengan
confidence {confidence_score * 100:.2f}%.")

        grafik_data = {
            'labels': Class_Penyakit,
            'datasets': [{
                'label': 'Probabilitas Prediksi',
                'data': probabilities,
                'backgroundColor': [
                    'rgba(255, 99, 132, 0.2)',
                    'rgba(54, 162, 235, 0.2)',
                    'rgba(255, 206, 86, 0.2)',
                    'rgba(75, 192, 192, 0.2)',
                    'rgba(153, 102, 255, 0.2)',
                    'rgba(255, 159, 64, 0.2)'
                ],
                'borderColor': [
                    'rgba(255, 99, 132, 1)',
                    'rgba(54, 162, 235, 1)',
                    'rgba(255, 206, 86, 1)',
                    'rgba(75, 192, 192, 1)',
                    'rgba(153, 102, 255, 1)',
                    'rgba(255, 159, 64, 1)'
                ],
                'borderWidth': 1
            }]
        }
    except Exception as e:
        hasil_prediksi = f'Error: {e}'

    return jsonify({
        "prediksi": hasil_prediksi,
        "confidence": confidence_score,
        "gambar_prediksi": gambar_prediksi,
        "detail_proses": detail_proses,
        "grafik_data": grafik_data,
        "logits": logits,
        "probabilities": probabilities
    })

    return jsonify({"error": "Invalid file or unsupported
format"})
@app.route('/process', methods=['POST'])
def process_image():
    uploaded_file = request.files['file']
    if uploaded_file.filename != '':
        image =
Image.open(uploaded_file.stream).convert('RGB')
        image = image.resize((256, 256))
        image_array =
tf.keras.preprocessing.image.img_to_array(image)
        image_array = np.expand_dims(image_array,
axis=0) / 255.0
        conv_layer_outputs = [layer.output for layer in
model.layers if 'conv2d' in layer.name]
        activation_model =
tf.keras.models.Model(inputs=model.input,
outputs=conv_layer_outputs)
        activations =
activation_model.predict(image_array)
        layer_names = [layer.name for layer in
model.layers if 'conv2d' in layer.name]
        num_layers = len(layer_names)

        fig, axes = plt.subplots(1, num_layers, figsize=(15
* num_layers, 15))
        if num_layers == 1:
```

```
            axes = [axes]
        for i, (activation, layer_name) in enumerate(zip(activations, layer_names)):
            ax = axes[i]
            mean_activation = np.mean(activation[0], axis=-1)
            ax.imshow(mean_activation, cmap='viridis')
            ax.set_title(layer_name, fontsize=10)
            ax.axis('off')
        buf = io.BytesIO()
        plt.tight_layout()
        plt.savefig(buf, format='png')
        buf.seek(0)
        encoded_image = base64.b64encode(buf.read()).decode('utf-8')
        buf.close()
        plt.close()
        return jsonify({"visualization": encoded_image})

    return jsonify({"error": "No file uploaded"})

@app.route('/get_kernels')
def get_kernels():
    kernels_data = []
    for layer in model.layers:
        if 'conv2d' in layer.name:
            weights = layer.get_weights()[0]
            kernels = []
            for i in range(weights.shape[-1]):
                kernel = weights[:, :, :, i].tolist()
                kernels.append(kernel)
            kernels_data.append({
                'layer_name': layer.name,
                'kernels': kernels
            })
    return jsonify(kernels_data)

@app.route('/get_activations', methods=['POST'])
def get_activations():
    uploaded_file = request.files['file']
    image = Image.open(uploaded_file.stream).convert('RGB')
    image = image.resize((256, 256))
    image_array = tf.keras.preprocessing.image.img_to_array(image)
    image_array = np.expand_dims(image_array, axis=0) / 255.0

    activation_model = tf.keras.Model(
        inputs=model.input,
        outputs=[layer.output for layer in model.layers if 'conv2d' in layer.name]
    )
    activations = activation_model.predict(image_array)

    activations_data = []
    for i, activation in enumerate(activations):
        activations_data.append({
            'layer_name': model.layers[i].name,
            'activation_values': activation[0].tolist()
        })
    return jsonify(activations_data)
@app.route('/realtime')
def realtime():
    return render_template('realtime.html')

@socketio.on('image')
def handle_image(data):

    image_array = np.frombuffer(data, np.uint8)
    image = cv2.imdecode(image_array, cv2.IMREAD_COLOR)
    image_resized = cv2.resize(image, (256, 256))
    image_normalized = image_resized.astype('float32') / 255.0
```

```
    image_input = np.expand_dims(image_normalized, axis=0)

    y_pred = model.predict(image_input)
    pred_class = np.argmax(y_pred)
    confidence_score = round(float(np.max(y_pred)), 4)
    hasil_prediksi = Class_Penyakit[pred_class]

    emit('result', {
        'prediksi': hasil_prediksi,
        'confidence': confidence_score
    })

#=[Main]==========================================

model = make_model(n_classes=n_classes)
model.load_weights("./model_penyakit_tanaman17.h5")

if __name__ == '__main__':
    socketio.run(app, host='0.0.0.0', debug=True)
```

Fungsi.py

```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, Activation

def make_model(n_classes):
    model = Sequential()

    # Conv Block 1
    model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(256, 256, 3)))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    # Conv Block 2
    model.add(Conv2D(64, (3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    # Conv Block 3
    model.add(Conv2D(128, (3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    # Conv Block 4
    model.add(Conv2D(128, (3, 3), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    # Fully Connected
    model.add(Flatten())
    model.add(Dropout(0.5))
    model.add(Dense(128, activation='relu'))

    # Output dengan nama 'logits'
    model.add(Dense(n_classes, name='logits'))
    model.add(Activation('softmax'))

    return model
```

Client_side.js

```
$(document).ready(function () {
  $(".navbar a, footer a[href='#halamanku']").on("click", function (event) {
    if (this.hash !== "") {
      event.preventDefault();
      var hash = this.hash;
      $("html, body").animate(
        {
          scrollTop: $(hash).offset().top,
        },
        900,
        function () {
```

```
        window.location.hash = hash;
      }
    );
  }
});

$(window).scroll(function () {
  $(".slideanim").each(function () {
    var pos = $(this).offset().top;
    var winTop = $(window).scrollTop();
    if (pos < winTop + 600) {
      $(this).addClass("slide");
    }
  });
});

let chartInstance = null;
const video = document.getElementById("video");
const canvas = document.getElementById("canvas");
const cameraButton =
document.getElementById("camera_button");
let res_gambar_prediksi = null;

$("#prediksi_submit").click(function (e) {
  e.preventDefault();
  predictImage();
});

function predictImage() {
  var file_data = $("#input_gambar").prop("files")[0];
  if (!file_data) {
    alert("Silakan pilih file gambar terlebih dahulu.");
    return;
  }

  var pics_data = new FormData();
  pics_data.append("file", file_data);

  $.ajax({
    url: "/api/deteksi",
    type: "POST",
    data: pics_data,
    processData: false,
    contentType: false,
    success: function (res) {
      handlePredictionSuccess(res, pics_data);
    },
    error: function (err) {
      console.log("Error during prediction: ", err);
      alert("Gagal melakukan prediksi.");
    },
  });
}

function display_visualizations(visualizations) {
  if (visualizations.visualization) {
    const img = document.createElement("img");
    img.src = "" +
visualizations.visualization;
    img.className = "zoomable";
    img.style.width = "100%";

document.getElementById("visualizations").innerHTM
L = "";

document.getElementById("visualizations").appendChi
ld(img);
  }
}

function handlePredictionSuccess(res, pics_data) {
  var res_data_prediksi = res["prediksi"];
  res_gambar_prediksi = res["gambar_prediksi"]; //
Gunakan variabel global
  var res_confidence = res["confidence"];
```

```
var detail_proses = res["detail_proses"];
var grafik_data = res["grafik_data"];
var logits = res["logits"];
var probabilities = res["probabilities"];

generate_prediksi(
  res_data_prediksi,
  res_gambar_prediksi,
  res_confidence,
  detail_proses,
  grafik_data,
  logits,
  probabilities
);

$.ajax({
  url: "/process",
  type: "POST",
  data: pics_data,
  processData: false,
  contentType: false,
  success: function (visualizations) {
    display_visualizations(visualizations);
  },
});

$.ajax({
  url: "/get_activations",
  type: "POST",
  data: pics_data,
  processData: false,
  contentType: false,
  success: function (activations) {
    display_activation_details(activations);
  },
});
  }

  function generate_prediksi(
    data_prediksi,
    image_prediksi,
    confidence,
    detail_proses,
    grafik_data,
    logits,
    probabilities
  ) {
    var str = "";
    if (image_prediksi == "(none)") {
      str +=
        "<h3>Hasil Prediksi </h3><br><h4>Silahkan masukkan file gambar (.jpg)</h4>";
    } else {
      str += `<h3>Hasil Prediksi</h3><br>
            <img src="${image_prediksi}" class="zoomable" width="300" height = "200">`;

      if (confidence < 0.2) {
        str += "<h4>Maaf Gambar tidak dapat dikenali</h4>";
      } else {
        str += `<h3>${data_prediksi}</h3>
              <h4>Confidence: ${(confidence * 100).toFixed(2)}%</h4>`;
      }
    }

    $("#hasil_prediksi").html(str);
    update_detail_proses(detail_proses);
    update_chart(grafik_data);
    display_kernel_visualization();
    display_math_calculation(logits, probabilities);

    if (data_prediksi !== "Hasil Prediksi") {
```

```
    $("#detail_penyakit").prop("disabled", false);


    $("#detail_penyakit").data("prediksi",
data_prediksi);
   } else {
    $("#detail_penyakit").prop("disabled", true);
   }
 }


 $("#detail_penyakit").click(function () {
  var penyakit = $(this).data("prediksi");
  showPenyakitDetail(penyakit);
 });


 function showPenyakitDetail(penyakit) {
  var modal = document.getElementById("myModal");
  var span =
document.getElementsByClassName("close")[0];


  var penyakitTitle =
document.getElementById("penyakit-title");
  var penyakitDescription =
document.getElementById("penyakit-description");
  var penyakitImage =
document.getElementById("penyakit-image");


  if (res_gambar_prediksi) {
   penyakitImage.src = res_gambar_prediksi;
  } else {
   penyakitImage.src = "";
   console.error("res_gambar_prediksi tidak
didefinisikan");
  }


  switch (penyakit) {
   case "Cabai Anthracnose":
    penyakitTitle.textContent = "Cabai Anthracnose";
    penyakitDescription.textContent =
      "Penyakit Cabai Anthracnose disebabkan oleh
jamur Colletotrichum capsici. Gejala awalnya muncul
sebagai bercak kecil berwarna cokelat pada daun,
batang, dan buah. Seiring perkembangan penyakit,
bercak tersebut akan membesar dan berubah menjadi
bercak hitam yang mengeras, yang dapat menyebabkan
buah jatuh sebelum matang. Untuk mengendalikan
penyakit ini, penting untuk menggunakan bibit yang
bebas dari penyakit, melakukan rotasi tanaman, serta
menjaga kelembapan tanah agar tidak berlebihan.
Penggunaan fungisida juga dapat diterapkan sebagai
langkah pencegahan.";
    break;
   case "Cabai Normal":
    penyakitTitle.textContent = "Cabai Sehat";
    penyakitDescription.textContent =
     "Cabai Anda tampak sehat dan tidak
menunjukkan gejala penyakit. Daun hijau cerah dan
batang yang kuat menunjukkan bahwa tanaman dalam
kondisi baik. Untuk mempertahankan kondisi ini,
pemeliharaan yang baik sangat penting, termasuk
penyiraman yang tepat, pemupukan yang seimbang, dan
pengendalian hama secara rutin.";
    break;
   case "Cabai Leaf Curl":
    penyakitTitle.textContent = "Cabai Leaf Curl";
    penyakitDescription.textContent =
     "Penyakit Cabai Leaf Curl disebabkan oleh virus
seperti Cucumber mosaic virus (CMV) dan Tomato
yellow leaf curl virus (TYLCV). Gejala yang muncul
meliputi daun yang menjadi kecil, menguning, dan
menggulung ke atas, yang mengindikasikan bahwa
pertumbuhan tanaman terhambat. Untuk mengendalikan
penyakit ini, penting untuk menggunakan bibit yang
bebas virus, mengendalikan serangga vektor seperti
kutu daun, dan menghapus tanaman yang terinfeksi
untuk mencegah penyebaran lebih lanjut.";
    break;
   case "Tomat Normal":
    penyakitTitle.textContent = "Tomat Sehat";
    penyakitDescription.textContent =
     "Tomat Anda tampak sehat dan tidak
menunjukkan gejala penyakit. Daun hijau cerah dan
batang yang kuat menunjukkan bahwa tanaman dalam
kondisi baik. Untuk menjaga kesehatan tanaman tomat,
pemeliharaan yang baik, termasuk penyiraman yang
tepat dan pengendalian hama, sangat diperlukan.";
    break;
   case "Tomat Early Blight":
    penyakitTitle.textContent = "Tomat Early blight";
```

```
        penyakitDescription.textContent =

          "Penyakit Tomat Early Blight disebabkan oleh jamur Alternaria solani. Gejala awalnya ditandai dengan bercak kecil berwarna cokelat pada daun, seringkali dengan tepi kuning. Bercak ini dapat berkembang menjadi bercak hitam yang mengeras dan dapat muncul pada batang serta buah, menyebabkan pertumbuhan yang tidak sehat. Pengendalian penyakit ini dapat dilakukan dengan menggunakan bibit yang bebas penyakit, melakukan rotasi tanaman, serta menjaga kelembapan tanah agar tidak berlebihan.";

        break;

      case "Tomat Late Blight":

        penyakitTitle.textContent = "Tomat Late Blight";

        penyakitDescription.textContent =

          "Penyakit Tomat Late Blight disebabkan oleh jamur Phytophthora infestans. Gejala awalnya berupa bercak hijau kekuningan pada daun, yang kemudian berubah menjadi bercak hitam. Daun yang terinfeksi dapat layu dan jatuh. Untuk mengendalikan penyakit ini, penting untuk menggunakan bibit yang bebas penyakit, menjaga sirkulasi udara yang baik di antara tanaman, dan menghindari penyiraman yang berlebihan.";

        break;

      default:

        penyakitTitle.textContent = "Penyakit Tidak Dikenali";

        penyakitDescription.textContent =

          "Penyakit yang Anda alami tidak dikenali oleh sistem.";

        break;

    }


    modal.style.display = "block";


    window.onclick = function (event) {

      if (event.target == modal) {

        modal.style.display = "none";

      }

    };


    span.onclick = function () {

      modal.style.display = "none";

    };

  }


  function update_detail_proses(detail_proses) {

    var detailStr = "<h3>Detail Proses</h3><ul>";

    detail_proses.forEach(function (detail) {

      detailStr += `<li>${detail}</li>`;

    });

    detailStr += "</ul>";

    $("#proses_prediksi").html(detailStr);

  }


  function update_chart(grafik_data) {

    var ctx =
document.getElementById("grafik_prediksi").getConte
xt("2d");

    if (chartInstance) chartInstance.destroy();


    chartInstance = new Chart(ctx, {

      type: "bar",

      data: grafik_data,

      options: {

        scales: { y: { beginAtZero: true } },

      },

    });

  }


  function display_kernel_visualization() {

    $.ajax({

      url: "/get_kernels",

      type: "GET",

      success: function (kernels) {

        console.log("Kernels data:", kernels);

        let html = "<h3>Visualisasi Kernel CNN</h3>";

        kernels.forEach((layer) => {

          html += `<div class="layer-kernels mb-4">

                    <h4>${layer.layer_name}</h4>
```

```
                    <div class="kernel-grid">`;

          if (Array.isArray(layer.kernels)) {
            layer.kernels.forEach((kernel, kernelIdx) => {
              html += `<div class="kernel-container">
                                <h5>Kernel ${kernelIdx +
1}</h5>`;

              if (
                Array.isArray(kernel) &&
                Array.isArray(kernel[0]) &&
                Array.isArray(kernel[0][0])
              ) {
                kernel[0].forEach((channel) => {
                  html += '<div class="kernel-channel">';
                  if (Array.isArray(channel)) {
                    channel.forEach((row) => {
                      html += '<div class="kernel-row">';
                      if (Array.isArray(row)) {
                        row.forEach((value) => {
                          const color =
                            value > 0
                              ? `rgba(0,255,0,${Math.abs(value)})`
                              :
`rgba(255,0,0,${Math.abs(value)})`;
                          html += `<span class="kernel-value"
                                                title="Nilai:
${value.toFixed(
                                                  4
                                                )}"

style="background-color:${color}">${value.toFixed(
                            2
                          )}</span>`;
                        });
                      }
                      html += "</div>";
                    });
                  }
                  html += "</div>";
                });
              }
              html += "</div>";
            });
          }
          html += "</div></div>";
        });
        $("#kernel-visualization").html(html);
      },
      error: function (err) {
        console.error("Error fetching kernels:", err);
        $("#kernel-visualization").html(
          "<p>Gagal memuat visualisasi kernel.</p>"
        );
      },
    });
  }


  function display_math_calculation(logits,
probabilities) {
    let html = "<h3>Perhitungan Matematika
Probabilitas</h3>";
    html += '<div class="math-calculation">';

    html += '<h4>Logits:</h4><div class="logits">';
    logits.forEach((logit, i) => {
      html += `<div>z${i} = ${logit.toFixed(4)}</div>`;
    });
    html += "</div>";

    html += "<h4>Perhitungan Softmax:</h4>";
    const expLogits = logits.map((l) => Math.exp(l));
    const sumExp = expLogits.reduce((a, b) => a + b, 0);

    expLogits.forEach((exp, i) => {
```

```
    html += `
      <div class="softmax-step">
        <div>$$P(Kelas_{${i}}) =
\\frac{e^{z_{${i}}}}{\\sum e^{z}}$$</div>
        <div>$$=
\\frac{e^{${logits[i].toFixed(4)}}}{${expLogits
      .map((e) =>
`e^{${logits[expLogits.indexOf(e)].toFixed(4)}}`)
      .join(" + ")}}$$</div>
        <div>$$=
\\frac{${exp.toFixed(4)}}{${sumExp.toFixed(
      4
    )}} = ${probabilities[i].toFixed(4)}$$</div>
      </div>`;
  });
  html += "</div></div>";

  $("#math-calculation").html(html);
  MathJax.Hub.Queue(["Typeset", MathJax.Hub]);
}

$(document).on("click", ".zoomable", function () {
  new Viewer(this, {
    navbar: false,
    toolbar: {
      zoomIn: 1,
      zoomOut: 1,
      reset: 1,
    },
  }).show();
});

navigator.mediaDevices
  .getUserMedia({ video: true })
  .then((stream) => {
    video.srcObject = stream;
    video.play();
  })
  .catch((error) => {
    console.error("Error accessing camera:", error);
    alert("Gagal mengakses kamera.");
  });

cameraButton.addEventListener("click", function () {
  const context = canvas.getContext("2d");
  context.drawImage(video, 0, 0, canvas.width,
canvas.height);

  const imageData =
canvas.toDataURL("image/jpeg");
  const pics_data = new FormData();
  pics_data.append("file", dataURLtoBlob(imageData),
generateFileName());

  $.ajax({
    url: "/api/deteksi",
    type: "POST",
    data: pics_data,
    processData: false,
    contentType: false,
    success: function (res) {
      handlePredictionSuccess(res, pics_data);
    },
  });
});

function dataURLtoBlob(dataURL) {
  const byteString = atob(dataURL.split(",")[1]);
  const mimeString =
dataURL.split(",")[0].split(":")[1].split(";")[0];
  const ab = new ArrayBuffer(byteString.length);
  const ia = new Uint8Array(ab);
  for (let i = 0; i < byteString.length; i++) {
    ia[i] = byteString.charCodeAt(i);
  }
  return new Blob([ab], { type: mimeString });
```

```
  }


  function generateFileName() {

    const now = new Date();

    return
`capture_${now.getFullYear()}${(now.getMonth() +
1).toString().padStart(2,
"0")}${now.getDate().toString().padStart(2,
"0")}_${now.getHours().toString().padStart(2,
"0")}${now.getMinutes().toString().padStart(2,
"0")}${now.getSeconds().toString().padStart(2,
"0")}.jpg`;

  }

});
```

Navbar.js

```
const menuIcon = document.querySelector("#menu-
icon");

const navbar = document.querySelector(".navbar");

const navbg = document.querySelector(".nav-bg");

menuIcon.addEventListener("click", () => {

  menuIcon.classList.toggle("bx-x");

  navbar.classList.toggle("active");

  navbg.classList.toggle("active");

});
```

Index.html

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta name="viewport" content="width=device-
width, initial-scale=1.0" />

    <link

      rel="stylesheet"


href="https://cdn.jsdelivr.net/npm/boxicons@latest/css/
boxicons.min.css"

    />

    <link rel="stylesheet" href="static/css/styles.css" />


    <!-- Library Tambahan -->

    <script src="{{ url_for('static',
filename='js/jquery_3.6.0.min.js') }}"></script>

    <script
src="https://cdn.jsdelivr.net/npm/chart.js"></script>

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/mathjax/2.7.
7/MathJax.js?config=TeX-MML-
AM_CHTML"></script>

    <link

      rel="stylesheet"


href="https://cdnjs.cloudflare.com/ajax/libs/viewerjs/1.
10.5/viewer.min.css"/>

    <title>Deteksi Penyakit Tanaman Cabai dan
Tomat</title>

    <style>

      /* CSS Tambahan untuk Fitur Baru */

      .kernel-container {

        background: rgba(255, 255, 255, 0.1);

        backdrop-filter: blur(10px);

        border-radius: 10px;

        padding: 15px;

        margin: 15px 0;

        box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

      }

      .kernel-value {

        display: inline-block;

        width: 40px;

        height: 40px;

        text-align: center;

        line-height: 40px;

        margin: 2px;

        font-size: 8px;

        border: 1px solid rgba(255, 255, 255, 0.3);

        border-radius: 5px;

        background: rgba(255, 255, 255, 0.15);

        transition: all 0.3s ease;

      }
```

```
.kernel-value:hover {
  transform: scale(1.5);
  z-index: 100;
  box-shadow: 0 2px 8px rgba(0, 0, 0, 0.2);
}

.activation-grid {
  max-height: 400px;
  overflow-y: auto;
  background: rgba(255, 255, 255, 0.1);
  border-radius: 10px;
  padding: 15px;
  margin: 15px 0;
}

.activation-value {
  display: inline-block;
  width: 60px;
  height: 30px;
  line-height: 30px;
  text-align: center;
  margin: 2px;
  font-size: 10px;
  background: rgba(255, 255, 255, 0.1);
  border-radius: 3px;
}

.math-calculation {
  background: rgba(255, 255, 255, 0.1);
  backdrop-filter: blur(5px);
  border-radius: 10px;
  padding: 20px;
  margin: 20px 0;
  font-family: "Courier New", monospace;
}

    .zoomable {
      cursor: zoom-in;
      transition: transform 0.3s;
      border-radius: 10px;
      border: 2px solid rgba(255, 255, 255, 0.2);
    }

    .zoomable:hover {
      transform: scale(1.03);
    }

    .layer-title {
      color: #fff;
      text-shadow: 0 2px 4px rgba(0, 0, 0, 0.3);
      margin: 15px 0;
    }
  </style>
</head>

<body>
  <header class="header">
    <div class="logo">
      <img src="static/images/umpar.png" alt="" />
      <p>UNIVERSITAS MUHAMMADIYAH
PAREPARE</p>
    </div>
    <i class="bx bx-menu" id="menu-icon"></i>
    <nav class="navbar">
      <a href="#Dashboard">Home</a>
      <a href="#Aplikasi">Deteksi</a>
      <a href="/realtime">Deteksi Real-Time</a>
      <a href="#Card">Contact</a>
    </nav>
  </header>

  <div class="nav-bg"></div>
```

```
<section id="Dashboard" class="dash_container">
  <div class="konten__glass">
    <div class="dash__data">
      <h3 class="dash_title">SELAMAT DATANG</h3>
      <span class="dash_capt">
        Tanamanmu adalah bagian penting dari keseharian, dan Aplikasi ini
        membantu menjaga mereka tetap sehat. Dengan teknologi CNN
        (Convolutional Neural Networks), Kamu dapat mendeteksi tanda-tanda
        penyakit pada cabai dan tomat dengan mudah dan cepat.
      </span>
    </div>
    <div class="img_group">
      <img
        src="static/images/tct.png"
        alt=""
        class="img_konten2 zoomable"
      />
      <img src="static/images/kd3.png" alt="" class="img_konten zoomable" />
    </div>
    <div>
      <a href="#Aplikasi" class="dash__button1">MULAI</a>
    </div>
  </div>
</section>

<section id="Aplikasi" class="app_container">
  <div class="konten__glass">
    <div class="dash__data">
      <h3 class="dash_title">Mulai Deteksi</h3>
      <img
        src="{{ url_for('static', filename='images/deteksi.png') }}"
        width="300"
        class="zoomable"
      />
      <p class="dash_capt">
        Upload gambar bagian daun atau buah untuk mulai deteksi
      </p>
    </div>

    <div class="col-sm-6">
      <div id="hasil_prediksi" align="center"></div>

      <form id="uploadForm" method="POST" enctype="multipart/form-data">
        <input
          type="file"
          id="input_gambar"
          name="file"
          accept="image/*"
          style="margin: 15px 0"
        />
      </form>

      <div>
        <button type="submit" id="prediksi_submit" class="dash__button">
          PREDIKSI
        </button>
        <button
          type="button"
          id="detail_penyakit"
          class="dash__button"
          style="margin-left: 10px"
        >
          Detail Penyakit
        </button>
```

```
        </div>
        <div style="margin-top: 20px">
          <button id="camera_button"
class="dash__button">
            Ambil Gambar
          </button>
        </div>
        <video id="video" width="300" height="200"
autoplay></video>
        <canvas id="canvas" style="display:
none"></canvas>
       </div>
     </div>
     <!-- Visualisasi Tambahan -->
     <!-- <div id="kernel-visualization"></div>

     <div id="activation-details">
       <h3 class="layer-title">Aktivasi Layer</h3>
     </div> -->
     <h3 class="layer-title" align="center">Visualisasi
Model</h3>

     <div id="visualizations" align="center"></div>
     <div>
       <div id="math-calculation"></div>
     </div>

     <div class="grafik-container">
       <h3 class="layer-title">Grafik Probabilitas</h3>
       <canvas id="grafik_prediksi"></canvas>
     </div>
   </section>
   <div class="konten__glass" style="margin-top:
30px">
     <div id="proses_prediksi"></div>
   </div>
   <div id="myModal" class="modal">
     <div class="modal-content">
       <span class="close">&times;</span>
       <h2 id="penyakit-title" class="center-text"></h2>
       <img
         id="penyakit-image"
         class="center-image zoomable"
         src=""
         alt="Gambar Penyakit"
         width="200"
       />
       <p id="penyakit-description" class="center-
text"></p>
     </div>
   </div>

   <footer>
     <section id="Card" class="card">
       <!-- ... (bagian card tetap sama) ... -->
     </section>
   </footer>

   <!-- Script Tambahan -->
   <script
src="https://cdnjs.cloudflare.com/ajax/libs/viewerjs/1.1
0.5/viewer.min.js"></script>
   <script src="{{ url_for('static',
filename='js/client_side.js') }}"></script>
   <script src="static/js/n.js"></script>

   <script>
     // Inisialisasi Viewer.js untuk semua gambar dengan
class zoomable

document.querySelectorAll(".zoomable").forEach((ele
ment) => {
       new Viewer(element, {
         navbar: false,
         toolbar: {
           zoomIn: 1,
           zoomOut: 1,
```

```
          reset: 1,
        },
      });
    });
  </script>
</body>
<footer>
  <section id="Card" class="card">
    <div class="card__circle card__circle1"></div>
    <div class="card__circle card__circle2"></div>

    <div class="card__container bd-container">
      <div class="card__glass">
        <img src="static/images/profil.jpg" alt="" class="card__img" />

        <div class="card__data">
          <h3 class="card__title">Ahmad Selao, S.Tp., M.Sc</h3>
          <span class="card__profession">Pembimbing I</span>
        </div>
      </div>

      <div class="card__glass">
        <img src="static/images/profil.jpg" alt="" class="card__img" />

        <div class="card__data">
          <h3 class="card__title">Deris Pakiding</h3>
          <span class="card__profession">Mahasiswa</span>
        </div>

        <div class="card__social">
          <a

href="https://www.instagram.com/dderpa_07/profilecard/?igsh=MXZhb2xjcTRldHBueg=="
            class="card__link"
            target="_blank"
            ><i class="bx bxl-instagram"></i
          ></a>
          <a
            href="https://www.linkedin.com/in/deris-pakiding-7575de02"
            class="card__link"
            ><i class="bx bxl-linkedin"></i
          ></a>
        </div>
      </div>

      <div class="card__glass">
        <img src="static/images/profil.jpg" alt="" class="card__img" />

        <div class="card__data">
          <h3 class="card__title">Wahyuddin, S.Kom., M.Kom.</h3>
          <span class="card__profession">Pembimbing II</span>
        </div>
      </div>
    </div>
  </section>
</footer>
</html>
```

Realtime.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Deteksi Real-Time</title>
    <style>
```

```
    #video {
      border: 1px solid black;
    }
    #result {
      margin-top: 20px;
      font-size: 1.2em;
    }
  </style>
 </head>
 <body>
  <h1>Deteksi Real-Time</h1>
  <video id="video" width="640" height="480" autoplay></video>
  <div id="result"></div>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/socket.io/4.0.1/socket.io.js"></script>
  <script>
    const video = document.getElementById("video");
    const socket = io();

    // Mengakses webcam
    navigator.mediaDevices
      .getUserMedia({ video: true })
      .then((stream) => {
        video.srcObject = stream;
        video.play();
        detectFrame();
      })
      .catch((err) => {
        console.error("Error accessing webcam:", err);
      });

    function detectFrame() {
      const canvas = document.createElement("canvas");
      canvas.width = 256;
      canvas.height = 256;
      const context = canvas.getContext("2d");
      context.drawImage(video, 0, 0, canvas.width, canvas.height);
      const imageData = canvas.toDataURL("image/jpeg");
      const byteString = atob(imageData.split(",")[1]);
      const arrayBuffer = new Uint8Array(byteString.length);
      for (let i = 0; i < byteString.length; i++) {
        arrayBuffer[i] = byteString.charCodeAt(i);
      }

      // Kirim data gambar ke server
      socket.emit("image", arrayBuffer.buffer);

      requestAnimationFrame(detectFrame);
    }

    // Menerima hasil deteksi dari server
    socket.on("result", function (data) {
      document.getElementById("result").innerHTML = `<h3>Prediksi: ${
        data.prediksi
      }</h3>
            <h4>Confidence: ${(data.confidence * 100).toFixed(2)}%</h4>`;
    });
  </script>
 </body>
</html>
```

Style.css

```
/*========== GOOGLE FONTS ==========*/
@import url("https://fonts.googleapis.com/css2?family=Montserrat:wght@400;500;600&display=swap");

/*========== VARIABLES CSS ==========*/
```

```
:root {
  /*========== Colors ==========*/
  --first-color: #160797;
  --second-color: #20a2c0;
  --body-color: #dfe9f2;
  --title-color: #1c1c22;
  --text-color: #58576b;

  /*========== Font and typography ==========*/
  --body-font: "Montserrat", sans-serif;
  --normal-font-size: 0.938rem;
  --h3-font-size: 1.125rem;
  --small-font-size: 0.75rem;
}

@media screen and (min-width: 968px) {
  :root {
    --normal-font-size: 1rem;
    --h3-font-size: 1.25rem;
    --small-font-size: 0.813rem;
  }
}

*,
::before,
::after {
  box-sizing: border-box;
}

body {
  margin: 0;
  padding: 0;
  font-family: var(--body-font);
  font-size: var(--normal-font-size);
}

h3 {
  margin: 0;
}

a {
  text-decoration: none;
}

img {
  max-width: 100%;
  height: auto;
}
#visualization {
  width: 100%;
}
.bd-container {
  max-width: 968px;
  width: calc(100% - 3rem);
  margin-left: 1.5rem;
  margin-right: 1.5rem;
}

@import
url("https://fonts.googleapis.com/css2?family=Poppins:
wght@300;400;500;600;700;800;900&display=swap");
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: "Poppins", sans-serif;
}
.header {
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  padding: 5px 100px;
```

```
  background: rgba(255, 255, 255, 0.1);
  display: flex;
  justify-content: space-between;
  align-items: center;
  backdrop-filter: blur(10px);
  border-bottom: 2px solid rgba(255, 255, 255, 0.2);
  z-index: 100;
}
.header::before {
  content: "";
  position: absolute;
  top: 0;
  left: -100%;
  width: 100%;
  height: 100%;
  background: linear-gradient(
    90deg,
    transparent,
    rgba(255, 255, 255, 0.4),
    transparent
  );
  transition: 0.5s;
}
.header:hover::before {
  left: 100%;
}
.header p {
  color: #fff;
  font-size: 15px;
  text-decoration: none;
  font-weight: 600;
  margin-left: 10px;
}
.logo {
  width: 50px;
  cursor: default;
  display: flex;
  /* align-items: center; */
}
.navbar a {
  color: #fff;
  font-size: 18px;
  text-decoration: none;
  margin-left: 10px;
  transition: 0.3s;
}
.navbar a:hover {
  color: #f00;
}
#menu-icon {
  font-size: 36px;
  color: #fff;
  display: none;
}
/* BREAKPOINTS */
/* @media (max-width: 992px) {
  .header {
    padding: 0.5rem 4%;
  }
} */

@media (max-width: 768px) {
  #menu-icon {
    display: block;
  }
  .navbar {
    position: absolute;
    top: 100%;
    left: 0;
    width: 100%;
    padding: 0.5rem 4%;
    display: none;
```

```
  }
  .navbar.active {
    display: block;
  }
  .navbar a {
    display: block;
    margin: 1.5rem 0;
  }
  .nav-bg {
    position: absolute;
    top: 79px;
    left: 0;
    width: 100%;
    height: 295px;
    background: rgba(255, 255, 255, 0.1);
    border-bottom: 2px solid rgba(255, 255, 255, 0.2);
    backdrop-filter: blur(10px);
    z-index: 99;
    display: none;
  }
  .nav-bg.active {
    display: block;
  }
}
/*========== Konten Glass==========*/
.dash_container {
  position: relative;
  overflow: hidden;
  padding-top: 40px;
  /* padding: 3rem 0; */
  background-color: var(--body-color);
}

.konten__glass {
  position: relative;
  overflow: hidden;
  padding: 1.5rem;
  background: linear-gradient(
    130deg,
    rgba(251, 251, 254, 0.6),
    rgba(251, 251, 254, 0.2)
  );
  box-shadow: inset 2px 2px 1px rgba(251, 251, 254,
0.3),
    inset -2px -2px 1px rgba(251, 251, 254, 0.2);
  border-radius: 1.5rem;
  backdrop-filter: blur(10px);
  max-width: 990px;
  width: 95%;
  margin: 30px auto;
  display: flex;
  align-items: center;
  justify-content: space-between;
  flex-direction: row; /* Untuk desktop/layar besar */
}

.dash__data {
  flex-grow: 1;
  max-width: 45%;
  margin-right: 20px;
  text-align: left; /* Default teks rata kiri */
}
.dash_title {
  font-size: 70px;
  margin-right: 20px;
  padding-bottom: 10px;
  width: auto; /* Memungkinkan lebar otomatis sesuai
konten */
  max-width: none; /* Menghilangkan batas lebar
maksimal */
  line-height: 0.8; /* Mengatur tinggi garis agar terlihat
lebih proporsional */
}
```

```
.img_group {
  display: flex;
  justify-content: flex-start;
  align-items: center;
}

.img_konten {
  width: 290px !important;
  flex-shrink: 0;
}

.img_konten2 {
  width: 150px !important;
  flex-shrink: 0;
}

/* Media query untuk layar kecil (mobile) */
@media (max-width: 768px) {
  .konten__glass {
    flex-direction: column; /* Mengubah susunan menjadi kolom pada layar kecil */
    text-align: center; /* Mengatur teks menjadi rata tengah */
  }

  .img_group {
    justify-content: center; /* Menempatkan gambar di tengah */
    margin-bottom: 20px; /* Beri jarak antara gambar dan teks di bawahnya */
  }

  .dash__data {
    max-width: 100%; /* Teks akan memakan lebar penuh */
    margin-right: 0; /* Hilangkan margin kanan pada teks */
    position: center;
  }

  #grafik_prediksi {
    max-width: 100%; /* Pastikan canvas menggunakan lebar penuh dari kontainer */
    height: auto; /* Tinggi otomatis untuk menjaga rasio */
  }
}

.dash__button,
.dash__button1 {
  /* position: absolute; */
  align-items: center;
  bottom: 15px;
  left: 50%;
  transform: translateX(-50%);
  background: linear-gradient(
    130deg,
    rgba(251, 251, 254, 0.9),
    rgba(251, 251, 254, 0.2)
  );
  padding: 0.75rem 1.5rem;
  border-radius: 0.5rem;
  color: var(--title-color);
  font-weight: 500;
  transition: transform 0.4s ease, background 0.4s ease; /* Tambahkan transisi untuk transform dan background */
}

.dash__button1 {
  position: absolute;
}

.dash__button:hover,
.dash__button1:hover {
  background: linear-gradient(
    to bottom left,
    /* Gradient bergerak dari kanan atas ke kiri bawah */
    rgba(255, 255, 255, 0.8),
```

```
    /* Warna cahaya putih dari kanan atas */ rgba(3, 255, 36, 0.878)
      /* Warna hijau di kiri bawah */
  );
  transform: translateX(-50%) scale(1.1); /* Hanya gunakan scale tanpa mengubah padding */
  box-shadow: 0 8px 15px rgba(0, 0, 0, 0.3); /* Efek bayangan untuk tampilan yang lebih dramatis */
}
/* ==========Aplikasi Glass ========*/
.app_container {
  position: relative;
  overflow: hidden;
  padding: 2rem 0;
  background-color: var(--body-color);
}

.konten__glass {
  position: relative;
  overflow: hidden;
  padding: 1.5rem;
  background: linear-gradient(
    130deg,
    rgba(251, 251, 254, 0.6),
    rgba(251, 251, 254, 0.2)
  );
  box-shadow: inset 2px 2px 1px rgba(251, 251, 254, 0.3),
    inset -2px -2px 1px rgba(251, 251, 254, 0.2);
  border-radius: 1.5rem;
  backdrop-filter: blur(10px);
  max-width: 990px;
  width: 95%;
  margin: 30px auto;
  display: flex;
  align-items: center;
  justify-content: space-between;
  flex-direction: row; /* Untuk desktop/layar besar */
}

.dash__data {
  flex-grow: 1;
  max-width: 45%;
  margin-right: 20px;
  text-align: left; /* Default teks rata kiri */
}
.dash_title {
  font-size: 70px;
  margin-right: 20px;
  padding-bottom: 10px;
  width: auto; /* Memungkinkan lebar otomatis sesuai konten */
  max-width: none; /* Menghilangkan batas lebar maksimal */
  line-height: 0.8; /* Mengatur tinggi garis agar terlihat lebih proporsional */
}

.img_group {
  display: flex;
  justify-content: flex-start;
  align-items: center;
}

.img_konten {
  width: 290px !important;
  flex-shrink: 0;
}

.img_konten2 {
  width: 150px !important;
  flex-shrink: 0;
}

#hasil_prediksi,
```

```
#uploadForm,
#prediksi_submit {
  margin-bottom: 15px; /* Memberikan jarak antar elemen */
}

.col-sm-6 {
  display: flex;
  flex-direction: column;
  align-items: center; /* Menyejajarkan elemen di tengah secara horizontal */
}

/* Media query untuk layar kecil (mobile) */
@media (max-width: 768px) {
  .konten__glass {
    flex-direction: column; /* Mengubah susunan menjadi kolom pada layar kecil */
    text-align: center; /* Mengatur teks menjadi rata tengah */
  }

  .img_group {
    justify-content: center; /* Menempatkan gambar di tengah */
    margin-bottom: 20px; /* Beri jarak antara gambar dan teks di bawahnya */
  }

  .dash__data {
    max-width: 100%; /* Teks akan memakan lebar penuh */
    margin-right: 0; /* Hilangkan margin kanan pada teks */
    position: center;
  }
}
.dash__button {
  /* position: absolute; */
  bottom: 15px;
  left: 50%;
  transform: translateX(-50%);
  background: linear-gradient(
    130deg,
    rgba(251, 251, 254, 0.9),
    rgba(251, 251, 254, 0.2)
  );
  padding: 0.75rem 1.5rem;
  border-radius: 0.5rem;
  color: var(--title-color);
  font-weight: 500;
  transition: transform 0.4s ease, background 0.4s ease; /* Tambahkan transisi untuk transform dan background */
}

.dash__button:hover {
  background: linear-gradient(
    to bottom left,
    /* Gradient bergerak dari kanan atas ke kiri bawah */
      rgba(255, 255, 255, 0.8),
    /* Warna cahaya putih dari kanan atas */ rgba(3, 255, 36, 0.878)
      /* Warna hijau di kiri bawah */
  );
  transform: translateX(-50%) scale(1.1); /* Hanya gunakan scale tanpa mengubah padding */
  box-shadow: 0 8px 15px rgba(0, 0, 0, 0.3); /* Efek bayangan untuk tampilan yang lebih dramatis */
}

/*========== CARD GLASS ==========*/
.card {
  position: relative;
  overflow: hidden;
  padding: 2rem 0;
  background-color: var(--body-color);
}
```

```
.card__container {
  display: grid;
  gap: 1rem;
}

.card__glass {
  position: relative;
  overflow: hidden;
  text-align: center;
  padding: 1.5rem;
  background: linear-gradient(
    130deg,
    rgba(251, 251, 254, 0.6),
    rgba(251, 251, 254, 0.2)
  );
  box-shadow: inset 2px 2px 1px rgba(251, 251, 254,
0.3) inset -2px -2px 1px
    rgba(251, 251, 254, 0.2);
  border-radius: 1.5rem;
  backdrop-filter: blur(10px);
}

.card__img {
  width: 80px;
  height: 80px;
  border-radius: 50%;
  border: 2px solid #f4f4fb;
  margin-bottom: 1rem;
}

.card__data {
  margin-bottom: 1rem;
}

.card__title {
  font-size: var(--h3-font-size);
  color: var(--title-color);
  font-weight: 600;
  margin-bottom: 0.25rem;
}

.card__profession {
  font-size: var(--small-font-size);
  color: var(--text-color);
  font-weight: 500;
}

.card__social {
  position: absolute;
  top: 50%;
  left: 10%;
  transform: translateY(-70%);
}

.card__link {
  display: block;
  font-size: 1.35rem;
  color: var(--title-color);
  margin: 1rem 0;
  transform: translateX(-5rem);
}

.card__link:nth-child(1) {
  transition: 0.2s;
}
.card__link:nth-child(2) {
  transition: 0.5s;
}
.card__link:nth-child(3) {
  transition: 0.7s;
}
```

```
.card__glass:hover .card__link {
  transform: translateX(-1.5rem);
}

.card__circle {
  position: absolute;
  width: 250px;
  height: 250px;
  background: linear-gradient(
    130deg,
    rgba(162, 5, 5, 0.8),
    rgba(215, 63, 63, 0.2)
  );
  border-radius: 50%;
}

.card__circle1 {
  top: 20%;
  left: -20%;
}

.card__circle2 {
  bottom: -5%;
  right: -25%;
  background: linear-gradient(
    130deg,
    rgba(29, 242, 5, 0.8),
    rgba(255, 255, 255, 0.2)
  );
}

/*========== MEDIA QUERIES ==========*/
@media screen and (min-width: 568px) {
  .card__container {
    grid-template-columns: repeat(2, 1fr);
  }
}

@media screen and (min-width: 768px) {
  .bd-container {
    margin-left: auto;
    margin-right: auto;
  }

  .card {
    padding: 0;
  }

  .card__container {
    height: 100vh;
    grid-template-columns: repeat(3, 1fr);
    align-content: center;
  }

  .card__circle1 {
    left: 5%;
    top: 12%;
  }

  .card__circle2 {
    right: 8%;
    bottom: 15%;
  }
}

/*========== MODAL ==========*/
.modal {
  display: none; /* Hidden by default */
  position: fixed; /* Stay in place */
  z-index: 1000; /* Sit on top */
  left: 0;
```

```
  top: 0;
  width: 100%; /* Full width */
  height: 100%; /* Full height */
  overflow: auto; /* Enable scroll if needed */
  background-color: rgb(0, 0, 0); /* Fallback color */
  background-color: rgba(0, 0, 0, 0.4); /* Black w/ opacity */
}

.modal-content {
  background-color: #fefefe;
  margin: 15% auto; /* 15% from the top and centered */
  padding: 20px;
  border: 1px solid #888;
  width: 80%; /* Could be more or less, depending on screen size */
  border-radius: 1rem;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
}
.modal-content {
  position: relative;
  background-color: #fefefe;
  margin: 15% auto;
  padding: 20px;
  border: 1px solid #888;
  width: 80%;
  text-align: center;
}

.center-text {
  text-align: center;
  margin: 10px 0;
}

.center-image {
  display: block;
  margin: 20px auto;
  border-radius: 50%;
  width: 150px;
  height: 150px;
}

.close {
  color: #aaa;
  float: right;
  font-size: 28px;
  font-weight: bold;
  cursor: pointer;
}

.close:hover,
.close:focus {
  color: black;
  text-decoration: none;
  cursor: pointer;
}
#grafik_prediksi {
  width: 70% !important; /* Pastikan canvas menggunakan lebar penuh dari kontainer */
  height: auto !important; /* Tinggi otomatis untuk menjaga rasio */
}

.activation-grid {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
  gap: 1rem;
  padding: 1rem;
  background: rgba(255, 255, 255, 0.05);
  border-radius: 8px;
}
```

```
.activation-channel {
  background: rgba(255, 255, 255, 0.1);
  padding: 1rem;
  border-radius: 8px;
}

.activation-values {
  font-family: monospace;
  word-break: break-all;
}

.activation-value {
  display: inline-block;
  margin: 2px;
  padding: 2px 5px;
  background: rgba(0, 0, 0, 0.3);
  border-radius: 3px;
  font-size: 0.8em;
}

.kernel-grid {
  display: grid;
  grid-template-columns: repeat(auto-fill,
minmax(200px, 1fr));
  gap: 1rem;
  padding: 1rem;
  background: rgba(255, 255, 255, 0.05);
  border-radius: 8px;
}
.kernel-container {
  background: rgba(255, 255, 255, 0.1);
  padding: 1rem;
  border-radius: 8px;
}

.kernel-channel {
  font-family: monospace;
  word-break: break-all;
}

.kernel-value {
  display: inline-block;
  margin: 2px;
  padding: 2px 5px;
  background: rgba(0, 0, 0, 0.3);
  border-radius: 3px;
  font-size: 0.8em;
}
```