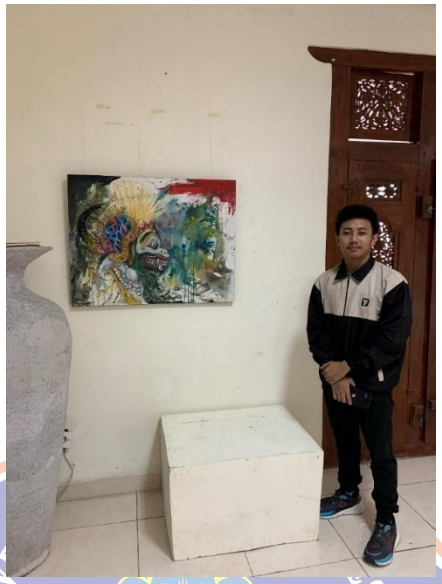
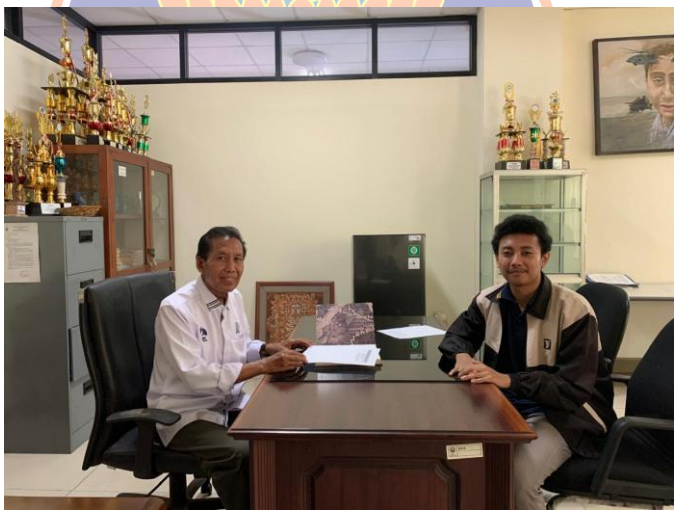
LAMPIRAN

Lampiran 1. Dokumentasi Dan Wawancara



Dokumentasi pameran seni milik program studi seni rupa di Fakultas Bahasa dan Seni, Undiksha.



Dokumentasi wawancara bersama bapak Dr. I Nyoman Sila, M.Hum. Dosen Program Studi Seni Rupa dan Wakil Dekan III Fakultas Bahasa dan Seni.

Lampiran 2. Kode Pembagian Data

```
train_df, temp_df = train_test_split(
    df,
    test_size=0.4,
    stratify=df['labels'],
    random_state=42
)


val_df, test_df = train_test_split(
    temp_df,
    test_size=0.5,
    stratify=temp_df['labels'],
    random_state=42
)
```

Lampiran 3. Kode CNN

```
def build_pure_CNN(input_shape, num_classes):
    inputs = Input(shape=input_shape)

    # Blok 1 sampai 4
    x = Conv2D(32, (3,3), activation='relu', padding='same',
kernel_initializer='he_normal')(inputs)
    x = BatchNormalization()(x)
    x = MaxPooling2D(2)(x)

    x = Conv2D(64, (3,3), activation='relu', padding='same',
kernel_initializer='he_normal')(x)
    x = BatchNormalization()(x)
    x = MaxPooling2D(2)(x)

    x = Conv2D(128, (3,3), activation='relu', padding='same',
kernel_initializer='he_normal')(x)
    x = BatchNormalization()(x)
    x = MaxPooling2D(2)(x)

    x = Conv2D(256, (3,3), activation='relu', padding='same',
kernel_initializer='he_normal')(x)
    x = BatchNormalization()(x)
    x = MaxPooling2D(2)(x)

    x = GlobalAveragePooling2D()(x)

    # Classifier Head
    x = Dense(256, activation='relu',
kernel_regularizer=l2(0.001), kernel_initializer='he_normal')(x)
    x = BatchNormalization()(x)
    x = Dropout(0.5)(x)

    x = Dense(128, activation='relu',
kernel_regularizer=l2(0.001), kernel_initializer='he_normal')(x)
    x = BatchNormalization()(x)
    x = Dropout(0.3)(x)
```

```
    outputs = Dense(num_classes, activation='softmax')(x)
    return Model(inputs, outputs)

# Inisialisasi Model
model = build_pure_CNN(img_shape, class_count)

# Compile
model.compile(
    optimizer=Adam(learning_rate=1e-4),
    loss=CategoricalCrossentropy(),
    metrics=['accuracy']
)

# Training
early_stopping = EarlyStopping(monitor='val_loss', patience=7,
restore_best_weights=True, verbose=1)

print("\n--- Memulai Training CNN ---\n")
history = model.fit(
    train_gen,
    validation_data=val_gen,
    epochs=50,
    callbacks=[early_stopping],
    verbose=1
)
```

Lampiran 4. Kode ResNet-50

```
    def build_optimized_resnet(num_classes):
    base_model = ResNet50(weights='imagenet', include_top=False,
input_shape=(224, 224, 3))


    # Fine-tuning: Unfreeze 15 layer
    base_model.trainable = True
    for layer in base_model.layers[:-15]:
        layer.trainable = False


    # Head Model
    inputs = Input(shape=(224, 224, 3))
    x = base_model(inputs)
    x = GlobalAveragePooling2D()(x)


    # Layer Dense 1
    x = BatchNormalization()(x)
    x = Dense(256, activation='relu',
kernel_regularizer=l2(0.001))(x)
    x = Dropout(0.4)(x)


    # Layer Dense 2
    x = BatchNormalization()(x)
    x = Dense(128, activation='relu',
kernel_regularizer=l2(0.001))(x)
    x = Dropout(0.3)(x)


    outputs = Dense(num_classes, activation='softmax')(x)


    model = Model(inputs, outputs)
    return model


# Pembuatan Model
class_count = len(train_gen.class_indices)
model = build_optimized_resnet(class_count)
```

```
# Compile Model
model.compile(
    optimizer=Adam(learning_rate=1e-4),
    loss=CategoricalCrossentropy(),
    metrics=['accuracy']
)

early_stopping = EarlyStopping(
    monitor='val_loss',
    patience=7,
    restore_best_weights=True,
    verbose=1
)

# Proses Training
history = model.fit(
    train_gen,
    validation_data=val_gen,
    epochs=50,
    callbacks=[early_stopping],
    verbose=1
)
```