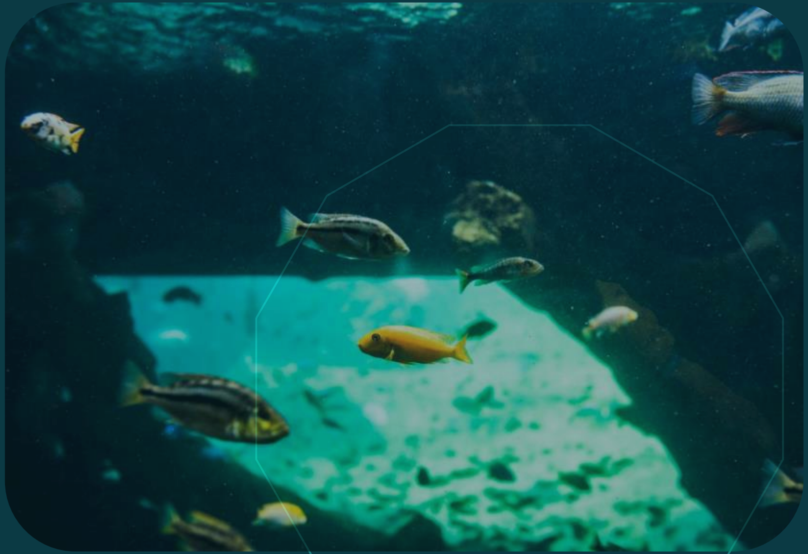
SMART AQUA SPECIES DETECTION

TEAM NAME: CHOOSEN ONES



ABOUT THE PROJECT:

Smart automated species identification and classification using AI & ML model (CNN).

TEAM MEMBER DETAILS:

1. Kanishka P
2. Vinothkumar V

QUICK GLANCE OF PROBLEM:

01

Difficulties in mannual monitoring

During active season landing of fish may be overwhelming for a manual monitoring.

02

Taking que

At the same time, taking que of the fish landing is difficult for researchers and administrators.

03

Errors and wrong extrapolations in manual monitoring

Manual reporting is limited with sample size, often too small, that lead to high level of errors and prone to wrong extrapolation for meaningful fish stock assessment.

MISSION STATEMENT

Computer algorithams and hardware systems can be replaced instead of manual monitoring mode, which enhance the accuracy and large scale marine monitorization and classification can be done.

TARGETING UNITED NATIONS SUSTAINABLE DEVELOPMENT GOALS:

❑ Life below water.

❑ Industry, Innovation and Infrastructure.

HUMAN VS AI

HUMAN

- Continuous monitorization is not possible.
- Less referance data.
- Low accuracy

AI

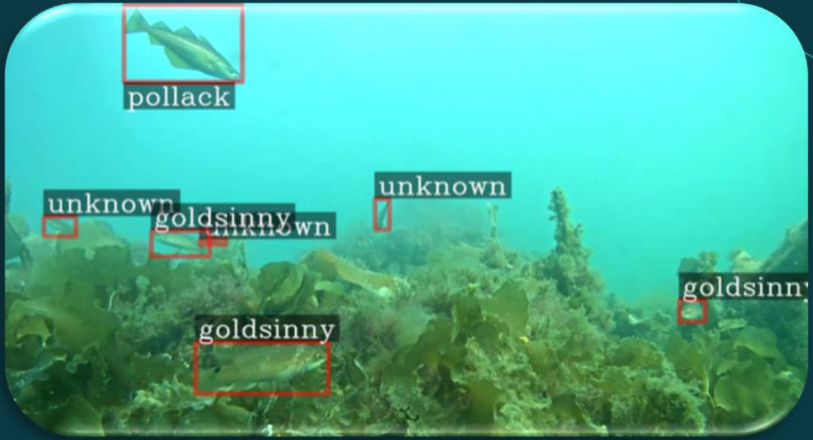
- Continuous monitorization is highly possible.
- Large reference data.
- High accuracy

PAINS TO BE RELIEVED:

- High man power
- Capital loss
- Extinction of rare species

GAINS TO BE CREATED:

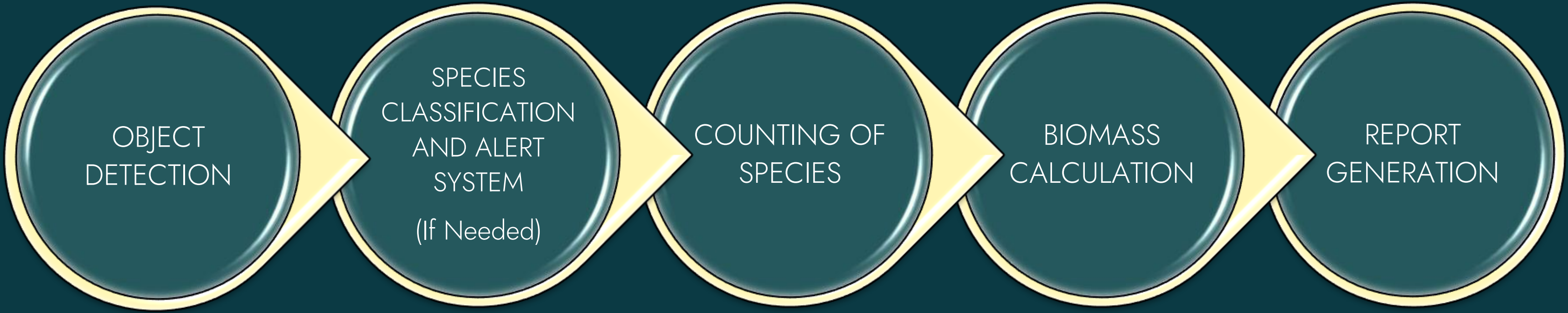
- Minimal manual intervention
- Time optimization
- Conservation of endangered species



PROPOSED SOLUTION:

- Input data recognition (Image).
- Processing and detection with the help of datasets.
- Classification and counting of species.
- Biomass calculation using the above results.
- Alert message (If only detects endangered ones).

WORKING (FLOW DIAGRAM):



OBJECT DETECTION

SPECIES CLASSIFICATION AND ALERT SYSTEM

(If Needed)

COUNTING OF SPECIES

BIOMASS CALCULATION

REPORT GENERATION

A VIDEO IS WORTH A THOUSAND PICTURES

TECHNOLOGICAL STACK:

Software

1 Image recognition (Python)

2 Detection and classification (CNN)

3 Cloud computing

4 Mobile app and web development

NOVELTY:

✓ Complete report generation

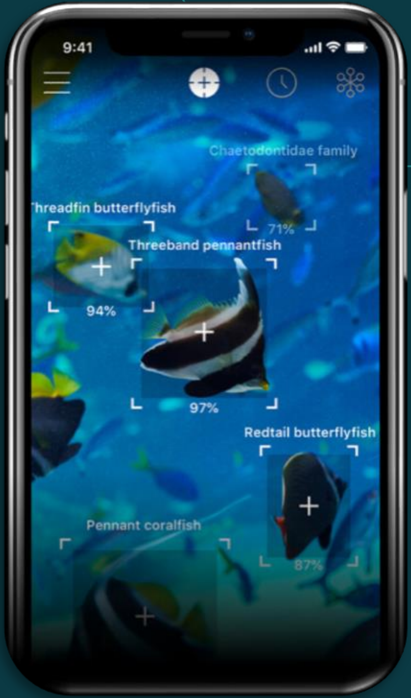
1. Classification
2. Counting
3. Biomass

✓ Alert message system.

✓ The cost is low compared to existing solutions.

APPLICATIONS:

- Cameras installed on the harbour and fishing boats.
- Integration with mobile application or web to get access for everyone.



CHALLENGES / LIMITATIONS:

- Simulation time.
- Unidentification of the unknown species.
- Lack of Realtime dataset.
- Internet connectivity for data transfer.
- Lighting conditions.
- Clarity of the input data.

WORK SIZE:

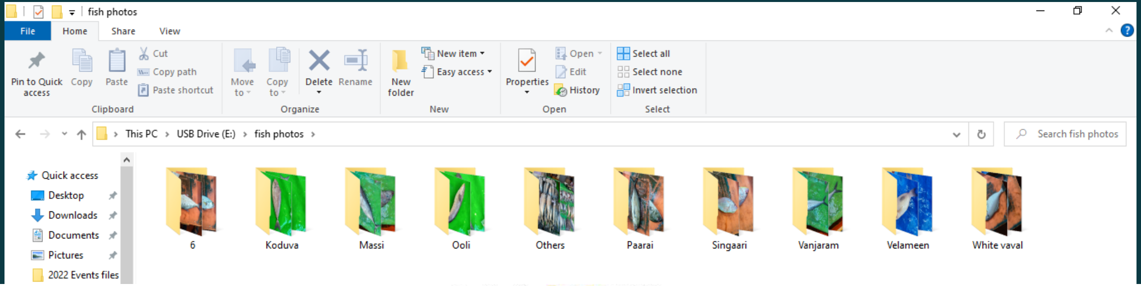
50% Dataset Collection

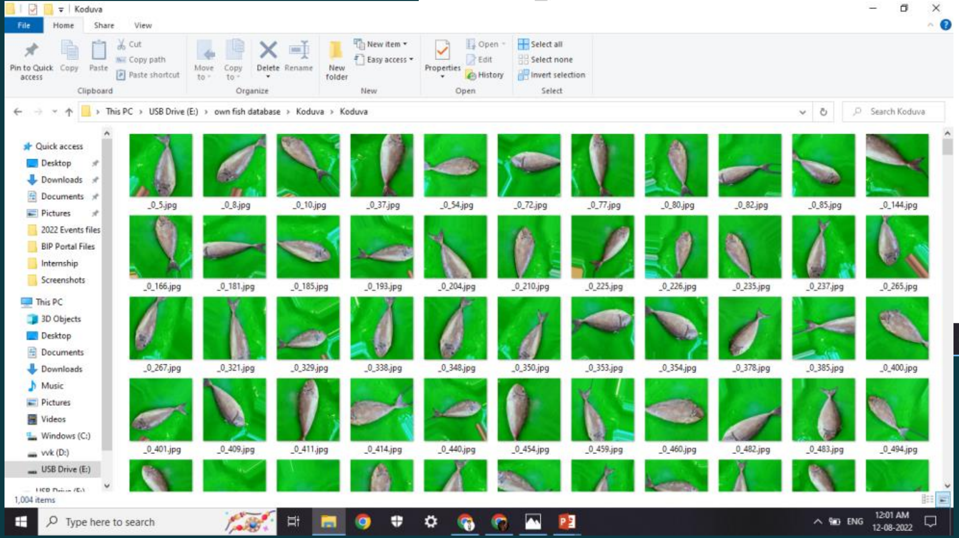
30% Code Building

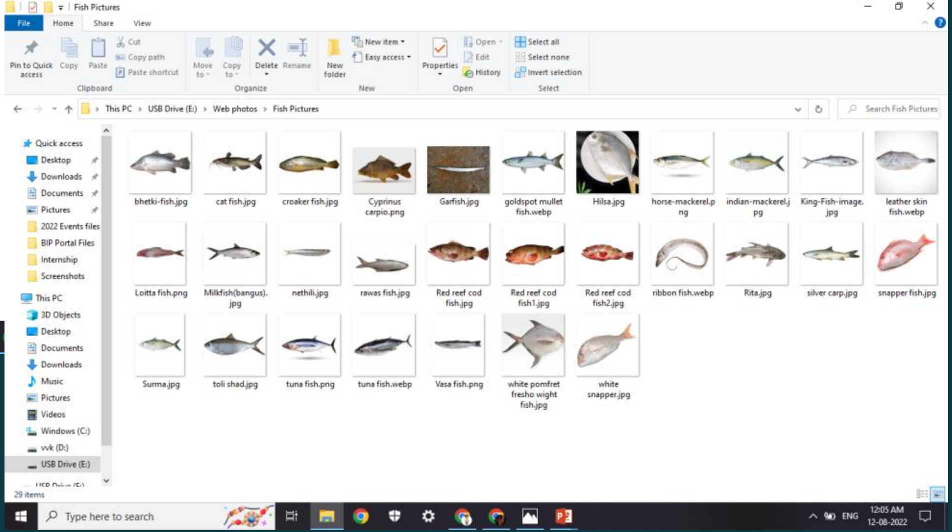
10% Training

10% Testing

DATASET COLLECTION:

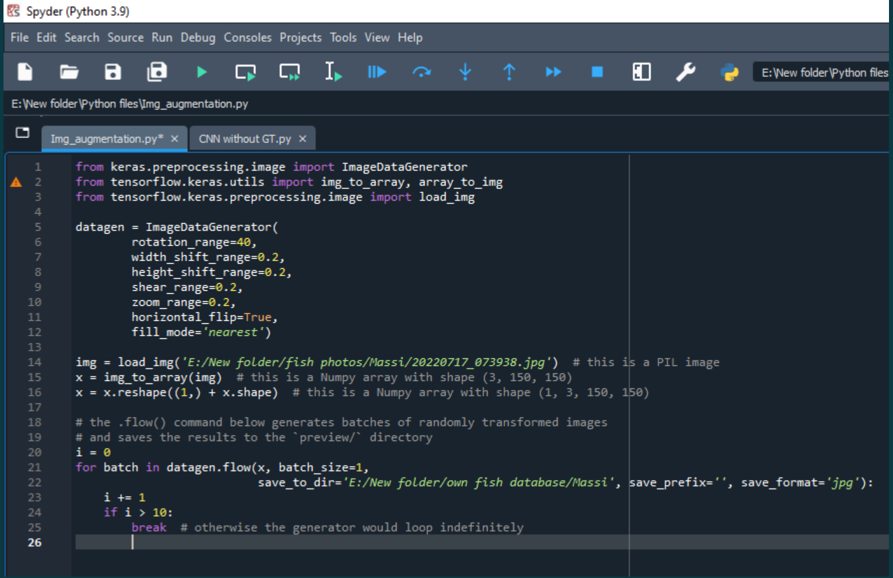




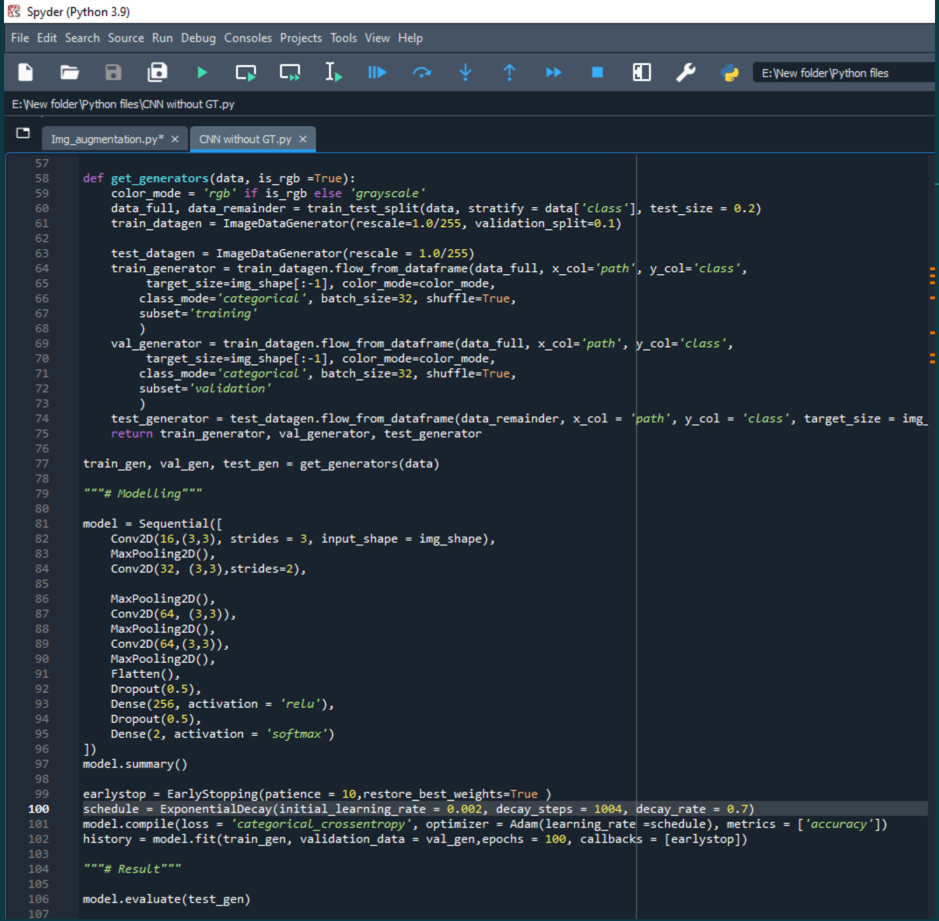


STEP ON FIELD FOR DATASETS: (UKKADAYAM FISH MARKET)

CODE BUILDING:

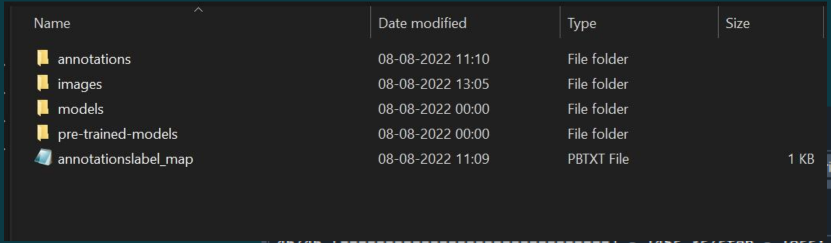

Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
E:\New folder\Python files
E:\New folder\Python files\Img_augmentation.py
Img_augmentation.py*
CNN without GT.py

```python
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.utils import img_to_array, array_to_img
from tensorflow.keras.preprocessing.image import load_img

datagen = ImageDataGenerator(
        rotation_range=40,
        width_shift_range=0.2,
        height_shift_range=0.2,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True,
        fill_mode='nearest')

img = load_img('E:/New folder/fish photos/Massi/20220717_073938.jpg')  # this is a PIL image
x = img_to_array(img)  # this is a Numpy array with shape (3, 150, 150)
x = x.reshape((1,) + x.shape)  # this is a Numpy array with shape (1, 3, 150, 150)

# the .flow() command below generates batches of randomly transformed images
# and saves the results to the `preview/` directory
i = 0
for batch in datagen.flow(x, batch_size=1,
                          save_to_dir='E:/New folder/own fish database/Massi', save_prefix='', save_format='jpg'):
    i += 1
    if i > 10:
        break  # otherwise the generator would loop indefinitely

```

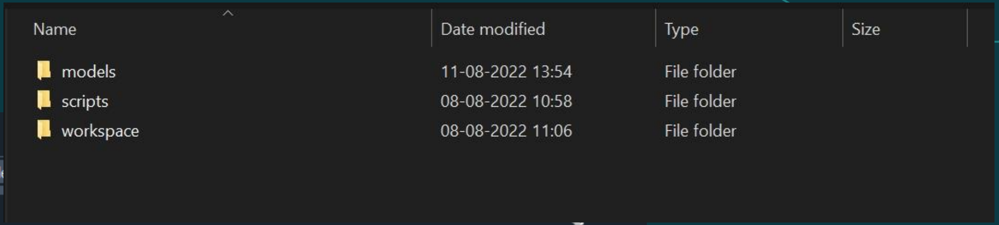

Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
E:\New folder\Python files
E:\New folder\Python files\CNN without GT.py
Img_augmentation.py*
CNN without GT.py

```python

def get_generators(data, is_rgb =True):
    color_mode = 'rgb' if is_rgb else 'grayscale'
    data_full, data_remainder = train_test_split(data, stratify = data['class'], test_size = 0.2)
    train_datagen = ImageDataGenerator(rescale=1.0/255, validation_split=0.1)

    test_datagen = ImageDataGenerator(rescale = 1.0/255)
    train_generator = train_datagen.flow_from_dataframe(data_full, x_col='path', y_col='class',
         target_size=img_shape[:-1], color_mode=color_mode,
        class_mode='categorical', batch_size=32, shuffle=True,
        subset='training'
        )
    val_generator = train_datagen.flow_from_dataframe(data_full, x_col='path', y_col='class',
         target_size=img_shape[:-1], color_mode=color_mode,
        class_mode='categorical', batch_size=32, shuffle=True,
        subset='validation'
        )
    test_generator = test_datagen.flow_from_dataframe(data_remainder, x_col = 'path', y_col = 'class', target_size = img_
    return train_generator, val_generator, test_generator

train_gen, val_gen, test_gen = get_generators(data)

"""# Modelling"""

model = Sequential([
    Conv2D(16,(3,3), strides = 3, input_shape = img_shape),
    MaxPooling2D(),
    Conv2D(32, (3,3),strides=2),

    MaxPooling2D(),
    Conv2D(64, (3,3)),
    MaxPooling2D(),
    Conv2D(64,(3,3)),
    MaxPooling2D(),
    Flatten(),
    Dropout(0.5),
    Dense(256, activation = 'relu'),
    Dropout(0.5),
    Dense(2, activation = 'softmax')
])
model.summary()

earlystop = EarlyStopping(patience = 10,restore_best_weights=True )
schedule = ExponentialDecay(initial_learning_rate = 0.002, decay_steps = 1004, decay_rate = 0.7)
model.compile(loss = 'categorical_crossentropy', optimizer = Adam(learning_rate =schedule), metrics = ['accuracy'])
history = model.fit(train_gen, validation_data = val_gen,epochs = 100, callbacks = [earlystop])

"""# Result"""

model.evaluate(test_gen)

```

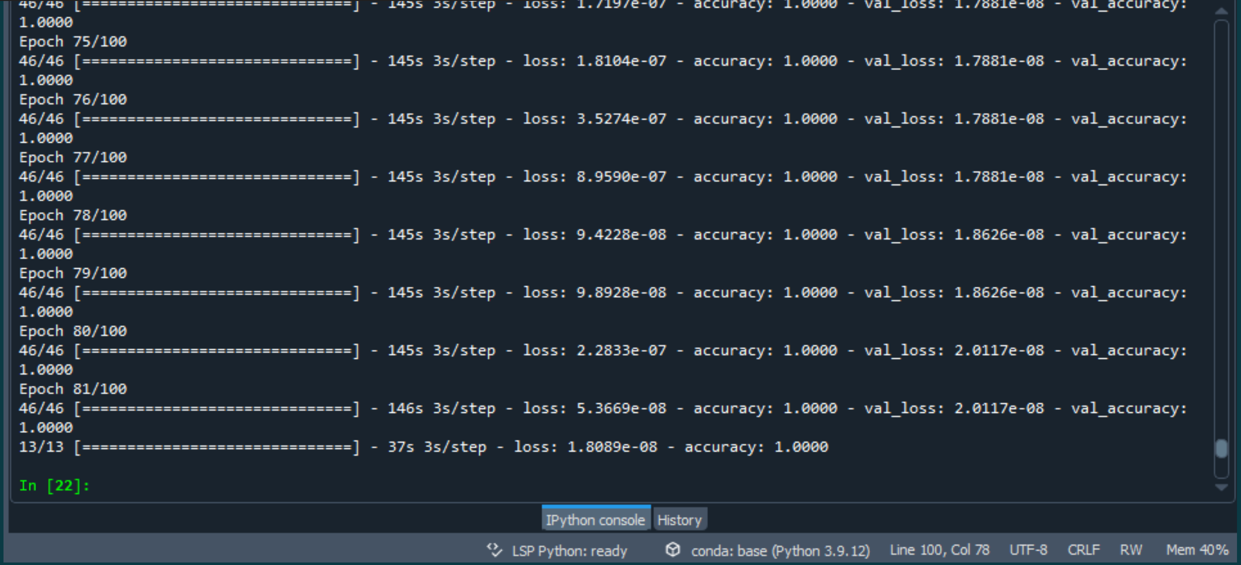
TRAINING AND TESTING:



Name	Date modified	Type	Size
annotations	08-08-2022 11:10	File folder	
images	08-08-2022 13:05	File folder	
models	08-08-2022 00:00	File folder	
pre-trained-models	08-08-2022 00:00	File folder	
annotationslabel_map	08-08-2022 11:09	PBTXT File	1 KB



Name	Date modified	Type	Size
models	11-08-2022 13:54	File folder	
scripts	08-08-2022 10:58	File folder	
workspace	08-08-2022 11:06	File folder	



```
46/46 [==============================] - 145s 3s/step - loss: 1.7197e-07 - accuracy: 1.0000 - val_loss: 1.7881e-08 - val_accuracy:
1.0000
Epoch 75/100
46/46 [==============================] - 145s 3s/step - loss: 1.8104e-07 - accuracy: 1.0000 - val_loss: 1.7881e-08 - val_accuracy:
1.0000
Epoch 76/100
46/46 [==============================] - 145s 3s/step - loss: 3.5274e-07 - accuracy: 1.0000 - val_loss: 1.7881e-08 - val_accuracy:
1.0000
Epoch 77/100
46/46 [==============================] - 145s 3s/step - loss: 8.9590e-07 - accuracy: 1.0000 - val_loss: 1.7881e-08 - val_accuracy:
1.0000
Epoch 78/100
46/46 [==============================] - 145s 3s/step - loss: 9.4228e-08 - accuracy: 1.0000 - val_loss: 1.8626e-08 - val_accuracy:
1.0000
Epoch 79/100
46/46 [==============================] - 145s 3s/step - loss: 9.8928e-08 - accuracy: 1.0000 - val_loss: 1.8626e-08 - val_accuracy:
1.0000
Epoch 80/100
46/46 [==============================] - 145s 3s/step - loss: 2.2833e-07 - accuracy: 1.0000 - val_loss: 2.0117e-08 - val_accuracy:
1.0000
Epoch 81/100
46/46 [==============================] - 146s 3s/step - loss: 5.3669e-08 - accuracy: 1.0000 - val_loss: 2.0117e-08 - val_accuracy:
1.0000
13/13 [==============================] - 37s 3s/step - loss: 1.8089e-08 - accuracy: 1.0000

In [22]:
```

MARKET SUSTAINABILITY:

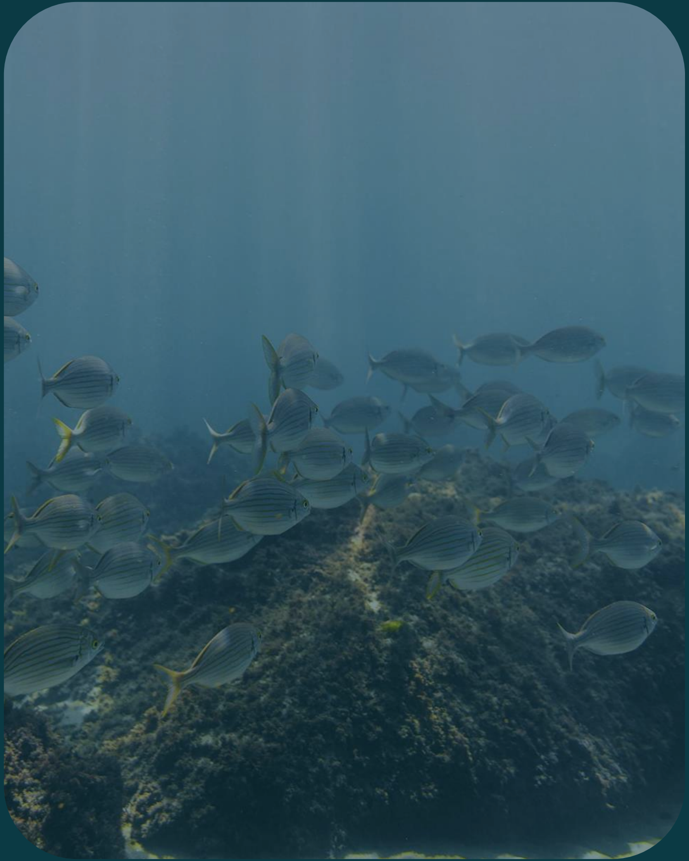
- **Market opportunities:** Fishermen, Government or private organizations, contractors.
- **Marketing plan:** Effective results with low price services or subscription.
- **Profit plan:** Small profit with large number of users or customers.
- **Maintenance Plan:** Effective maintenance of the services.

EYES ON:

Perfection on work

Errorless Simulation results

Giving affordable solution



THANK YOU