

## Final Project Documentation

Date	19 July 2024
Team ID	SWTID1720110092
Project Title	Beneath the Waves: Unravelling Coral Mysteries Through Deep Learning
Maximum Marks	10 Marks

### Project Initialization and Planning Phase

#### Define Problem Statements: Beneath The Waves: Unraveling Coral Mysteries Through Deep Learning:

The project "Beneath the Waves" addresses the challenges faced in coral reef monitoring and conservation by leveraging deep learning techniques to provide real-time, accurate, and comprehensive data. Coral reefs are crucial for marine biodiversity but are under threat from climate change, pollution, and human activities. Traditional methods for monitoring reef health, assessing biodiversity, and evaluating environmental impacts are labor-intensive and lack precision. By using advanced algorithms and neural networks, Beneath the Waves aims to identify signs of coral bleaching, disease, and other stressors, automate the identification and classification of marine species, and analyze the impact of human activities. This approach enables researchers and conservationists to take timely actions, develop effective conservation strategies, and foster a deeper understanding of coral ecosystems, ultimately contributing to their preservation and sustainability.



Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A marine biologist and conservationist	Monitor the health of coral reefs in real-time	I struggle with the time-consuming and labor-intensive traditional	They lack the precision needed for timely intervention	Frustrated and concerned about the deteriorating health of Coral reeves



## Project Initialization and Planning Phase

Date	19 July 2024
Team ID	SWTID1720110092
Project Title	Beneath the Waves: Unraveling Coral Mysteries Through Deep Learning
Maximum Marks	3 Marks

### Project Proposal (Proposed Solution) template

The proposal report aims to revolutionize coral reef conservation using deep learning algorithms, enhancing understanding and preservation efforts. It addresses the challenges of manual monitoring and assessment, promising more accurate data collection, timely interventions, and informed conservation strategies. Key features include advanced deep learning models for coral health monitoring and biodiversity assessment.

Project Overview	
Objective	The primary objective is to leverage deep learning techniques to advance coral reef conservation efforts, improving monitoring and assessment capabilities for enhanced preservation outcomes.
Scope	The project encompasses the development and implementation of deep learning models tailored for coral health monitoring and biodiversity assessment, aiming to provide researchers and conservationists with robust tools for data-driven decision-making.
Problem Statement	
Description	Current methods for coral reef monitoring and assessment rely heavily on manual processes, which are time-consuming, labor-intensive, and prone to errors. These inefficiencies hinder effective conservation efforts and pose significant challenges to understanding and mitigating threats to coral ecosystems.
Impact	Addressing these challenges with advanced deep learning algorithms will lead to more accurate and efficient monitoring, enabling timely interventions and informed conservation strategies. This will ultimately contribute to the preservation and sustainability of coral reefs worldwide.

Proposed Solution	
Approach	The project proposes the development and deployment of deep learning models capable of analyzing underwater imagery to detect signs of coral bleaching, disease, and other stressors. Additionally, the models will automate the identification and classification of marine species, facilitating comprehensive biodiversity assessments.
Key Features	<ul style="list-style-type: none"> <li>• Implementation of deep learning-based models for real-time coral health monitoring.</li> <li>• Automation of biodiversity assessment through advanced image recognition and classification techniques.</li> <li>• Integration of data-driven insights to inform conservation policies and strategies.</li> </ul>

## Resource Requirements

Resource Type	Description	Specification/Allocation
<b>Hardware</b>		
Computing Resources	CPU/GPU specifications, number of cores	2 x NVIDIA V100 GPUs
Memory	RAM specifications	RAM: 8 GB
Storage	Disk space for data, models, and logs	Disk space: 1 TB SSD
<b>Software</b>		
Frameworks	Python frameworks	Python frameworks: Flask
Libraries	Additional libraries	Additional libraries: scikit-learn, pandas, numpy, matplotlib, seaborn
Development Environment	IDE, version control	IDE: Jupyter Notebook, PyCharm
<b>Data</b>		
Data	Source, size, format	Kaggle dataset, UCI dataset



## Initial Project Planning Template

### Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create a product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members	Sprint Start Date	Sprint End Date (Planned)
Sprint-1	Data Collection and Preprocessing	SL-3	Understanding & loading data	2	Low	Mohammed Saif	2024/07/09	2024/07/16
Sprint-1	Data Collection and Preprocessing	SL-4	Data cleaning	1	High	Mohammed Saif	2024/07/09	2024/07/16
Sprint-1	Data Collection and Preprocessing	SL-5	EDA (Exploratory Data Analysis)	2	Medium	Akshita Gupta	2024/07/09	2024/07/16
Sprint-2	Model Development	SL-8	Training the model	2	Medium	Akshita Gupta	2024/07/16	2024/07/23
Sprint-2	Model Development	SL-9	Evaluating the model	1	High	Aditya Karthikeyan	2024/07/16	2024/07/23

<b>Sprint</b>	<b>Functional Requirement (Epic)</b>	<b>User Story Number</b>	<b>User Story / Task</b>	<b>Story Points</b>	<b>Priority</b>	<b>Team Members</b>	<b>Sprint Start Date</b>	<b>Sprint End Date (Planned)</b>
Sprint-2	Model Tuning and Testing	SL-13	Model tuning	1	High	Aditya Karthikeyan	2024/07/16	2024/07/23
Sprint-2	Model Tuning and Testing	SL-14	Model testing	2	Low	Arpan Chatterjee	2024/07/16	2024/07/23
Sprint-3	Web Integration and Deployment	SL-16	Building HTML templates	2	High	Arpan Chatterjee	2024/07/23	2024/07/30
Sprint-3	Web Integration and Deployment	SL-17	Local deployment	1	Medium	Aditya Karthikeyan	2024/07/23	2024/07/30
Sprint-4	Project Report	SL-20	Report	1	Medium	Mohammed Saif	2024/07/30	2024/08/13

## Data Collection Plan & Raw Data Sources Identification Template

Elevate your data strategy with the Data Collection plan and the Raw Data Sources report, ensuring meticulous data curation and integrity for informed decision-making in every analysis and decision-making endeavor.

### Data Collection Plan Template

Section	Description
Project Overview	The coral image classification project aims to develop an AI model capable of accurately distinguishing between healthy and bleached coral reefs. This model will contribute to conservation efforts by providing insights into coral health based on image analysis.
Data Collection Plan	The dataset will be collected from publicly available sources on Kaggle, specifically focusing on images of healthy and bleached coral reefs.
Raw Data Sources Identified	Train and Test dataset from Kaggle contains images categorized into healthy and bleached corals used for training a machine learning model.



### Raw Data Sources Template

Source Name	Description	Location/URL	Format	Size	Access Permissions
Train Dataset	Dataset containing training images of healthy and bleached coral reefs for machine learning model development.	<a href="https://www.kaggle.com/datasets/gauravduttakiit/corals-image-classification/data?select=train">https://www.kaggle.com/datasets/gauravduttakiit/corals-image-classification/data?select=train</a>	JPEG images	85.29 kB	Public
Test Dataset	Dataset containing test images of healthy and bleached coral reefs for model evaluation.	<a href="https://www.kaggle.com/datasets/gauravduttakiit/corals-image-classification/data?select=test">https://www.kaggle.com/datasets/gauravduttakiit/corals-image-classification/data?select=test</a>	JPEG images	21.42 kB	Public

## Data Collection and Preprocessing Phase

### Data Quality Report Template

The Data Quality Report Template will summarize data quality issues from the selected source, including severity levels and resolution plans. It will aid in systematically identifying and rectifying data discrepancies.

Data Source	Data Quality Issue	Severity	Resolution Plan
Dataset	Image Quality Variation	Moderate	Implement data augmentation techniques (e.g., rotation, shifting, zooming) to increase dataset diversity and model robustness.
Dataset	Class Imbalance	Moderate	Utilize techniques such as oversampling minority classes or adjusting class weights during model training to address imbalance issues.
Dataset	Inconsistent Metadata	Low	Develop scripts to standardize and validate metadata fields across the dataset to ensure consistency in data representation.
Dataset	Labeling Errors	High	Conduct thorough manual review and correction of labels, and

			implement automated validation checks to minimize future errors.
--	--	--	--

## Data Collection and Preprocessing Phase

### Preprocessing Template

The images will be preprocessed by resizing, normalizing, augmenting, denoising, adjusting contrast, detecting edges, converting color space, cropping, batch normalizing, and whitening data. These steps will enhance data quality, promote model generalization, and improve convergence during neural network training, ensuring robust and efficient performance across various computer vision tasks.

Section	Description
Data Overview	The data consists of images of coral reefs collected from various sources, labeled with categories related to reef health and environmental conditions.
Resizing	Resize images to a specified target size (224x224 pixels) using TensorFlow's ImageDataGenerator.
Normalization	Normalize pixel values to a specific range (0 to 1) using <code>rescale=1./255</code> in ImageDataGenerator.
Data Augmentation	Apply augmentation techniques such as rotation, shifting, zooming, and flipping to increase dataset diversity and model robustness..
Denoising	Not applicable for image data in this context.
Edge Detection	Not applicable for image data in this context.

Color Space Conversion	Not applicable for image data in this context.
Image Cropping	Not applicable as the images are already resized to a fixed size.
Batch Normalization	Apply batch normalization to the input of each layer in the neural network to improve training efficiency and convergence.
<b>Data Preprocessing Code Screenshots</b>	
Loading Data	<pre>import pandas as pd train_df = pd.read_csv('D:/train_updated.csv').sample(200) test_df = pd.read_csv('D:/test_updated.csv').sample(50)</pre>
Resizing	<pre>train_datagen = ImageDataGenerator(     rescale=1./255,     rotation_range=20,     width_shift_range=0.1,     height_shift_range=0.1,     shear_range=0.1,     zoom_range=0.1,     horizontal_flip=True,     fill_mode='nearest' )</pre>
Normalization	<pre>test_datagen = ImageDataGenerator(rescale=1./255)</pre>
Data Augmentation	<pre>train_generator = train_datagen.flow_from_dataframe(     dataframe=train_df,     x_col='local_filename',     y_col='label',     target_size=IMG_SIZE,     batch_size=BATCH_SIZE,     class_mode='categorical' )</pre>
Batch Normalization	<pre>from tensorflow.keras.layers import BatchNormalization x = BatchNormalization()(x)  </pre>

## Model Development Phase Template

### Model Selection Report

In the model selection report for future deep learning and computer vision projects, various architectures, such as CNNs or RNNs, will be evaluated. Factors such as performance, complexity, and computational requirements will be considered to determine the most suitable model for the task at hand.

### Model Selection Report:

Model	Description
Main Model	<ul style="list-style-type: none"><li>• <b>EfficientNetB0 Backbone:</b><ul style="list-style-type: none"><li>• <b>Architecture:</b> EfficientNetB0 is a convolutional neural network (CNN) that balances network depth, width, and resolution for efficient performance.</li><li>• <b>Pre-trained Weights:</b> The model uses pre-trained weights from ImageNet, which helps leverage learned features from a large, diverse dataset.</li><li>• <b>Excluding Top Layers:</b> The top (fully connected) layers are excluded (<code>include_top=False</code>) to allow custom layers to be added for the specific classification task.</li></ul></li><li>• <b>Fine-Tuning:</b></li></ul>

	<ul style="list-style-type: none"> <li>• <b>Partial Freezing:</b> The first 100 layers of EfficientNetB0 are frozen to retain their pre-trained weights, while the remaining layers are trainable to adapt to the new dataset.</li> <li>• <b>Custom Layers:</b> <ul style="list-style-type: none"> <li>• <b>Input Layer:</b> Defines the shape of the input images.</li> <li>• <b>Base Model:</b> The EfficientNetB0 base model processes the input images.</li> <li>• <b>Global Average Pooling:</b> Reduces the spatial dimensions of the feature maps, summarizing the presence of features.</li> <li>• <b>Dropout Layers:</b> Adds regularization by randomly setting a fraction of input units to 0 at each update during training to prevent overfitting.</li> </ul> </li> </ul> <p><b>Dense Layer:</b> Fully connected layer with 128 units and ReLU activation.</p> <ul style="list-style-type: none"> <li>• <b>Another Dropout Layer:</b> Adds further regularization.</li> <li>• <b>Output Layer:</b> Dense layer with 2 units (for binary classification) and softmax activation to output class probabilities.</li> </ul> <li>• <b>Model Compilation:</b> <ul style="list-style-type: none"> <li>• <b>Optimizer:</b> Adam optimizer with a learning rate of 0.0001, known for its adaptive learning rate capabilities.</li> <li>• <b>Loss Function:</b> Categorical cross-entropy, suitable for multi-class classification tasks.</li> <li>• <b>Metrics:</b> Accuracy, to monitor the fraction of correctly classified samples.</li> </ul> </li> <li>• <b>Data Augmentation and Generators:</b> <ul style="list-style-type: none"> <li>• <b>ImageDataGenerator:</b> Used for data augmentation to improve model generalization by creating variations of the training images.</li> <li>• <b>Data Generators:</b> Generate batches of tensor image data with real-time data augmentation for training and testing.</li> </ul> </li> <li>• <b>Model Training:</b></li>
--	--

- **Training:** The model is trained for a specified number of epochs with training and validation data generators.
- **Model Evaluation and Saving:**
  - **Saving the Model:** The trained model is saved to a file.
  - **Evaluation:** The model's performance is evaluated on the test set, and test accuracy is printed.



## Model Development Phase

### Initial Model Training Code, Model Validation and Evaluation Report

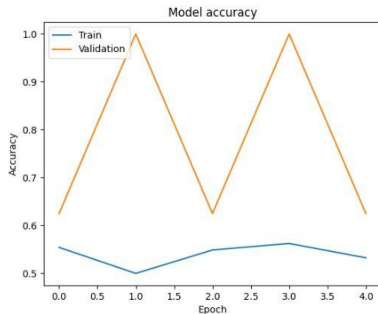
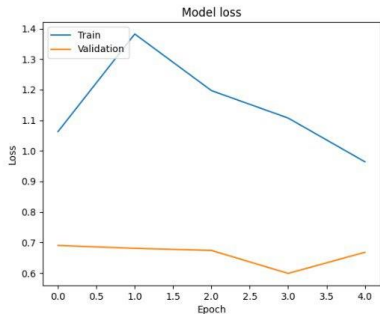
The initial model training code will be showcased in the future through a screenshot. The model validation and evaluation report will include a summary and training and validation performance metrics for multiple models, presented through respective screenshots.

### Initial Model Training Code (5 marks):

Paste the screenshot of the model training code

### Model Validation and Evaluation Report (5 marks):

Model	Summary	Training and Validation Performance Metrics
Model 1	<b>Neural network summarized code</b>	<pre> import tensorflow as tf  from tensorflow.keras.applications import EfficientNetB0  from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Dropout, BatchNormalization  from tensorflow.keras.models import Model  from tensorflow.keras.optimizers import Adam   # Define the model IMG_SIZE = (224, 224)  base_model = EfficientNetB0(weights='imagenet', include_top=False, input_shape=(*IMG_SIZE, 3))  base_model.trainable = True  fine_tune_at = 100  for layer in base_model.layers[:fine_tune_at]: </pre>

		<pre> layer.trainable = False  inputs = tf.keras.Input(shape=(*IMG_SIZE, 3)) x = base_model(inputs, training=True) x = GlobalAveragePooling2D()(x) x = BatchNormalization()(x) x = Dropout(0.3)(x) x = Dense(128, activation='relu')(x) x = BatchNormalization()(x) x = Dropout(0.3)(x) outputs = Dense(2, activation='softmax')(x)  model = Model(inputs, outputs) model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])  # Print the model summary model.summary() </pre>
	<p><b>Screenshot of the neural network summary</b></p>	<div>   </div>
	<p><b>Training and Validation Code</b></p>	<pre> import pandas as pd  from tensorflow.keras.preprocessing.image import ImageDataGenerator  # Load the data </pre>

```

train_df = pd.read_csv('D:/train_updated.csv').sample(200)
test_df = pd.read_csv('D:/test_updated.csv').sample(50)

# Data augmentation and preprocessing
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    zoom_range=0.1,
    horizontal_flip=True,
    fill_mode='nearest'
)

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_dataframe(
    dataframe=train_df,
    x_col='local_filename',
    y_col='label',
    target_size=IMG_SIZE,
    batch_size=16,
    class_mode='categorical'
)

test_generator = test_datagen.flow_from_dataframe(
    dataframe=test_df,
    x_col='local_filename',

```

```
y_col='label',
target_size=IMG_SIZE,
batch_size=16,
class_mode='categorical',
shuffle=False
)

# Train the model
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // 16,
    epochs=5,
    validation_data=test_generator,
    validation_steps=test_generator.samples // 16
)

# Save the model
model.save('coral_classification_model.h5')

# Output training and validation performance metrics
import matplotlib.pyplot as plt

# Plot training & validation accuracy values
plt.figure(figsize=(14, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
```

```
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')

# Plot training & validation loss values
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')

# Save the plot as a file
plt.savefig('training_validation_metrics.png')

# Show the plots
plt.show()
```

## Model Optimization and Tuning Phase Template

Date	19 July 2024
Team ID	SWTID1720110092
Project Title	Beneath the Waves: Unraveling Coral Mysteries Through Deep Learning
Maximum Marks	10 Marks

### Model Optimization and Tuning Phase

The Model Optimization and Tuning Phase involves refining neural network models for peak performance. It includes optimized model code, fine-tuning hyperparameters, comparing performance metrics, and justifying the final model selection for enhanced predictive accuracy and efficiency.

### Hyperparameter Tuning Documentation (8 Marks):

Model	Tuned Hyperparameters
Model 1	<p><b>IMG_SIZE:</b> Defines the target size for input images.</p> <pre>IMG_SIZE = (224, 224)</pre> <p><b>BATCH_SIZE:</b> Specifies the number of samples per gradient update.</p> <p><b>EPOCHS:</b> Number of times the entire training dataset is passed through the model.</p> <pre>BATCH_SIZE = 16 EPOCHS = 5</pre> <p><b>train_datagen and test_datagen:</b> Configures the image data generators for training and testing, including data augmentation parameters</p>

```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    zoom_range=0.1,
    horizontal_flip=True,
    fill_mode='nearest'
)

test_datagen = ImageDataGenerator(rescale=1./255)
```

**fine\_tune\_at:** Number of layers to freeze during fine-tuning.

```
fine_tune_at = 100
```

**Dropout rates:** Fraction of the input units to drop for the dropout layers.

```
x = Dropout(0.3)(x)
```

**Learning rate:** Learning rate for the Adam optimizer.

```
model = Model(inputs, outputs)
model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])
```

### Final Model Selection Justification (2 Marks):

Final Model	Reasoning
Model 1	<b>EfficientNetB0 Backbone:</b> EfficientNetB0 is known for its efficient architecture, providing a good balance between accuracy and

	<p>computational efficiency. It is a strong feature extractor due to its pre-trained weights on the ImageNet dataset.</p> <p><b>Data Augmentation:</b> The use of ImageDataGenerator with various augmentation techniques helps to generalize the model better by simulating real-world variations in the training images.</p> <p><b>Fine-Tuning:</b> By setting fine_tune_at = 100, the model partially fine-tunes the EfficientNetB0 backbone.</p> <p><b>Dropout Layers:</b> Dropout layers are used to prevent overfitting by randomly dropping a fraction of the units during training. This makes the model more robust and improves generalization.</p> <p><b>Reduced Complexity:</b> By reducing the dropout rates and the data augmentation parameters, the model converges faster, which is beneficial when working with smaller subsets of the data.</p> <p><b>Model Compilation:</b> The model is compiled with the Adam optimizer, a popular choice for its adaptive learning rate properties, along with categorical cross-entropy loss and accuracy as the metric.</p> <p><b>Training and Validation Strategy:</b> The training and validation strategy ensures that the model is evaluated on unseen data, providing a good indication of its generalization capability.</p>
--	---

## Output Screenshots:



## Beneath The Waves: Unraveling Coral Mysteries Through Deep Learning

### Project Overview

Beneath the Waves is an ambitious project that employs deep learning techniques to unravel mysteries surrounding coral reefs. By leveraging advanced algorithms and neural networks, this initiative aims to analyze and understand various aspects of coral ecosystems, contributing to conservation efforts and ecological research.

### Scenario 1: Coral Health Monitoring

Beneath the Waves can be used to monitor the health of coral reefs in real-time. Deep learning algorithms can analyze underwater imagery to identify signs of coral bleaching, disease, or other stressors.

### Scenario 2: Biodiversity Assessment

The project can contribute to biodiversity assessment by utilizing deep learning models to identify and classify different species of marine life around coral reefs.

### Scenario 3: Environmental Impact Assessment

Beneath the Waves can play a crucial role in assessing the environmental impact of human activities on coral reefs.

Upload Image

### Upload an Image

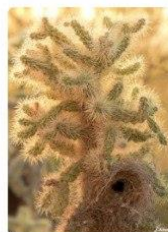
Choose File 5379411267...af45f5\_o.jpg

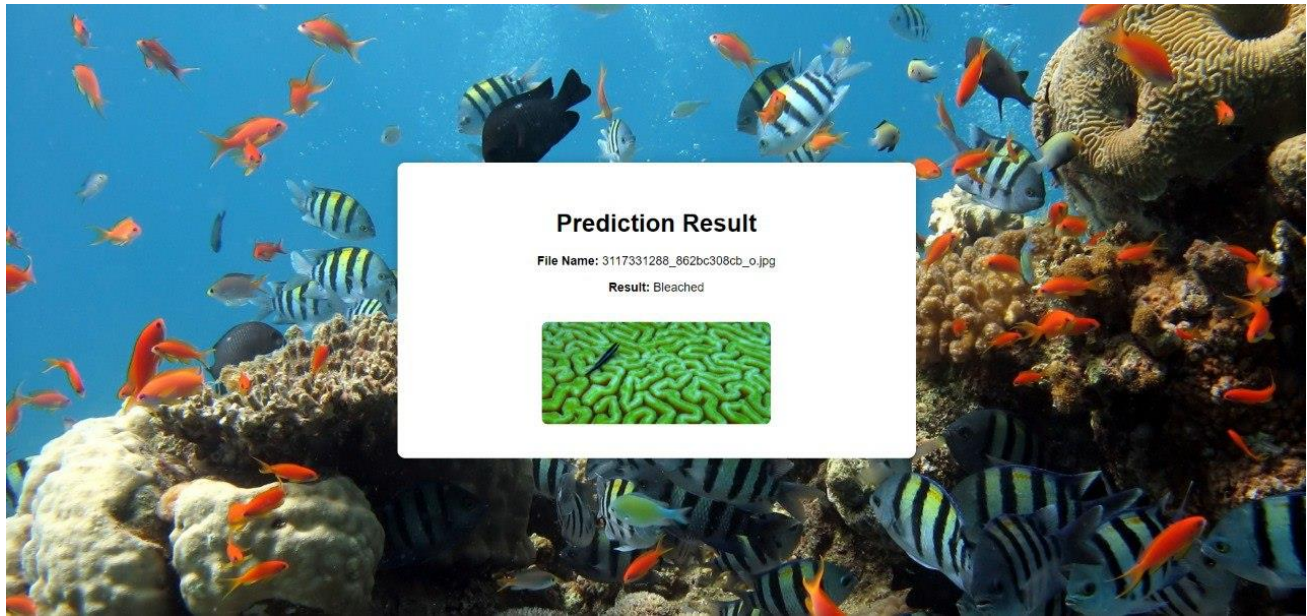
Upload

### Prediction Result

File Name: 5379411267\_fc71af45f5\_o.jpg

Result: Healthy





Found 200 validated image filenames belonging to 2 classes.  
Found 50 validated image filenames belonging to 2 classes.  
Model: "functional\_1"

Layer (type)	Output Shape	Param #
input_layer_1 (InputLayer)	(None, 224, 224, 3)	0
efficientnetb0 (Functional)	(None, 7, 7, 1280)	4,049,571
global_average_pooling2d (GlobalAveragePooling2D)	(None, 1280)	0
batch_normalization (BatchNormalization)	(None, 1280)	5,120
dropout (Dropout)	(None, 1280)	0
dense (Dense)	(None, 128)	163,968
batch_normalization_1 (BatchNormalization)	(None, 128)	512
dropout_1 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 2)	258

Total params: 4,219,429 (16.10 MB)  
Trainable params: 4,007,390 (15.29 MB)  
Non-trainable params: 212,039 (828.28 KB)  
Epoch 1/5



## Advantages & Disadvantages

### Advantages:

1. Scientific Insight: Enhanced understanding of coral ecosystems.
2. Conservation Efforts: Data aids in reef protection and restoration.
3. Educational Value: Increases public awareness and knowledge.
4. Biodiversity Preservation: Supports marine biodiversity conservation.
5. Technological Advancement: Utilizes cutting-edge technology for marine research.

### Disadvantages:

1. High Costs: Expensive equipment and fieldwork.
2. Environmental Impact: Potential disruption to marine life.
3. Data Complexity: Challenges in data collection and analysis.
4. Funding Dependency: Reliant on grants and sponsorships.
5. Accessibility Issues: Limited access to remote reef locations.

## 8. Conclusion

"Beneath Waves: Unravelling the Coral Mysteries" provides valuable insights into coral ecosystems, fostering conservation and educational efforts. While there are challenges, the project's potential to advance scientific knowledge and contribute to marine biodiversity preservation is significant.

## 9. Future Scope

1. Advanced Research: Further exploration of coral responses to climate change.
2. Innovative Technologies: Development of new tools for underwater research.
3. Global Collaboration: Expanding partnerships with international research institutions.
4. Policy Influence: Informing and shaping marine conservation policies.
5. Public Engagement: Increasing community involvement and awareness programs.

## Appendix:

### Source Code:

```
import os
import pandas as pd
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D,
Dropout, BatchNormalization
```

```
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam

tf.random.set_seed(42)

IMG_SIZE = (224, 224)
BATCH_SIZE = 16
EPOCHS = 5

# Load the data
train_df = pd.read_csv('D:/train_updated.csv').sample(200)
test_df = pd.read_csv('D:/test_updated.csv').sample(50)

# Data augmentation and preprocessing
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    shear_range=0.1,
    zoom_range=0.1,
    horizontal_flip=True,
    fill_mode='nearest'
)

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_dataframe(
    dataframe=train_df,
    x_col='local_filename',
    y_col='label',
    target_size=IMG_SIZE,
    batch_size=BATCH_SIZE,
    class_mode='categorical'
)

test_generator = test_datagen.flow_from_dataframe(
    dataframe=test_df,
    x_col='local_filename',
    y_col='label',
    target_size=IMG_SIZE,
    batch_size=BATCH_SIZE,
    class_mode='categorical',
    shuffle=False
)

# Load the base model
base_model = ResNet50(weights='imagenet', include_top=False,
input_shape=(*IMG_SIZE, 3))

base_model.trainable = True

fine_tune_at = 100
for layer in base_model.layers[:fine_tune_at]:
```

```
layer.trainable = False

# Build the model
inputs = tf.keras.Input(shape=(*IMG_SIZE, 3))
x = base_model(inputs, training=True)
x = GlobalAveragePooling2D()(x)
x = BatchNormalization()(x)
x = Dropout(0.3)(x)
x = Dense(128, activation='relu')(x)
x = BatchNormalization()(x)
x = Dropout(0.3)(x)
outputs = Dense(2, activation='softmax')(x)

model = Model(inputs, outputs)
model.compile(optimizer=Adam(learning_rate=0.0001),
              loss='categorical_crossentropy', metrics=['accuracy'])

model.summary()

# Train the model
history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // BATCH_SIZE,
    epochs=EPOCHS,
    validation_data=test_generator,
    validation_steps=test_generator.samples // BATCH_SIZE
)

# Save the model
model.save('resnet50_coral_classification_model.h5')

# Evaluate the model
test_loss, test_accuracy = model.evaluate(test_generator)
print(f"Test accuracy: {test_accuracy:.4f}")

# Output training and validation performance metrics
import matplotlib.pyplot as plt

# Plot training & validation accuracy values
plt.figure(figsize=(14, 5))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')

# Plot training & validation loss values
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
```

```
plt.xlabel('Epoch')  
plt.legend(['Train', 'Validation'], loc='upper left')  
  
plt.show()
```

Github Repo:-

[https://github.com/MohammedSaifAkkiwat/Project\\_Beneath\\_the\\_Waves](https://github.com/MohammedSaifAkkiwat/Project_Beneath_the_Waves)

Demo :-

[https://drive.google.com/file/d/1loryqmFCXY\\_0T0Osle5i4OQ5-w1xEQq2/view?usp=sharing](https://drive.google.com/file/d/1loryqmFCXY_0T0Osle5i4OQ5-w1xEQq2/view?usp=sharing)