In [144]:

```
import os
import matplotlib.pyplot as plt
import tensorflow as tf
```

In [145]:

```
dataset_dir = os.path.join(os.getcwd(), 'Downloads')

dataset_train_dir = os.path.join(dataset_dir, 'Train')
dataset_train_minecraft_len= len(os.listdir(os.path.join(dataset_train_dir, 'Mine')))
dataset_train_Among_len= len(os.listdir(os.path.join(dataset_train_dir, 'Among')))


dataset_validation_dir = os.path.join(dataset_dir, 'Validation')
dataset_validation_minecraft_len= len(os.listdir(os.path.join(dataset_validation_dir, 'M
dataset_validation_Among_len= len(os.listdir(os.path.join(dataset_validation_dir, 'Among


print('Train Mine: %s' % dataset_train_minecraft_len)
print('Validation Mine: %s' % dataset_validation_minecraft_len)

print('Train Among Us: %s' % dataset_train_Among_len)
print('Validation Among Us: %s' % dataset_validation_Among_len)
```

```
Train Mine: 473
Validation Mine: 456
Train Among Us: 505
Validation Among Us: 495
```

In [146]:

```
image_width = 160
image_height = 160
image_color_channel = 3
image_color_channel_size = 255
image_size = (image_width, image_height)
image_shape = image_size + (image_color_channel,)

batch_size = 32
epochs = 20
learning_rate = 0.0001

class_names = ['among', 'mine']
```

In [147]:

```
dataset_train = tf.keras.preprocessing.image_dataset_from_directory(
dataset_train_dir,
image_size = image_size,
batch_size = batch_size,
shuffle = True
)
```

```
Found 978 files belonging to 2 classes.
```

In [148]:

```
dataset_validation = tf.keras.preprocessing.image_dataset_from_directory(
    dataset_validation_dir,
    image_size = image_size,
    batch_size = batch_size,
    shuffle = True
)
```

```
Found 951 files belonging to 2 classes.
```

In [149]:

```
dataset_validation_cardinality = tf.data.experimental.cardinality(dataset_validation)
dataset_validation_batches = dataset_validation_cardinality // 5

dataset_test = dataset_validation.take(dataset_validation_batches)
dataset_validation = dataset_validation.skip(dataset_validation_batches)

print('Validation Dataset Cardinality: %d' % tf.data.experimental.cardinality(dataset_va
print('Test Dataset Cardinality: %d' % tf.data.experimental.cardinality(dataset_test))
```

```
Validation Dataset Cardinality: 24
Test Dataset Cardinality: 6
```

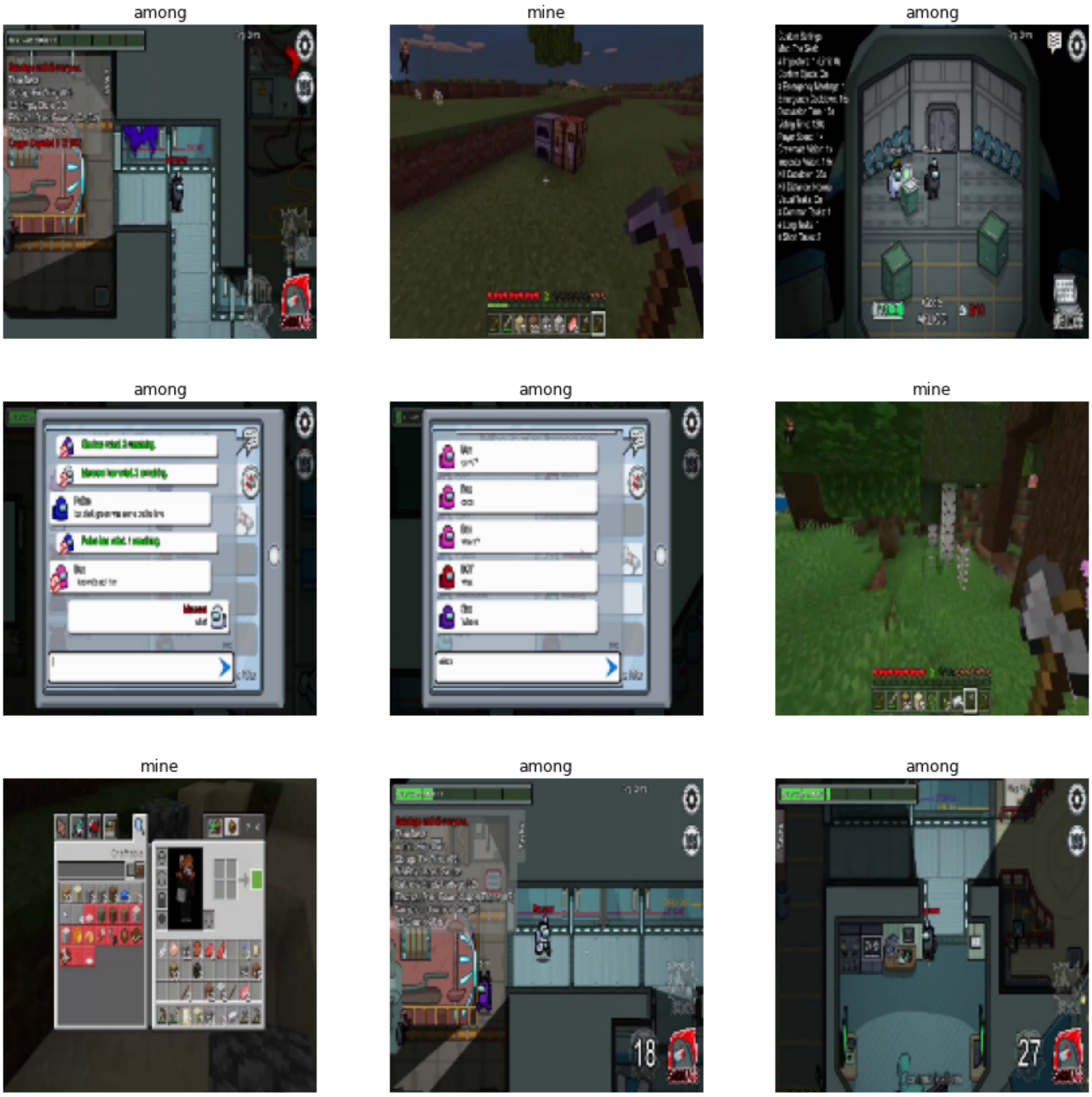
In [150]:

```
def plot_dataset(dataset):

    plt.gcf().clear()
    plt.figure(figsize = (15, 15))

    for features, labels in dataset.take(1):

        for i in range(9):

            plt.subplot(3, 3, i + 1)
            plt.axis('off')

            plt.imshow(features[i].numpy().astype('uint8'))
            plt.title(class_names[labels[i]])
```

In [151]:

```
plot_dataset(dataset_train)
```

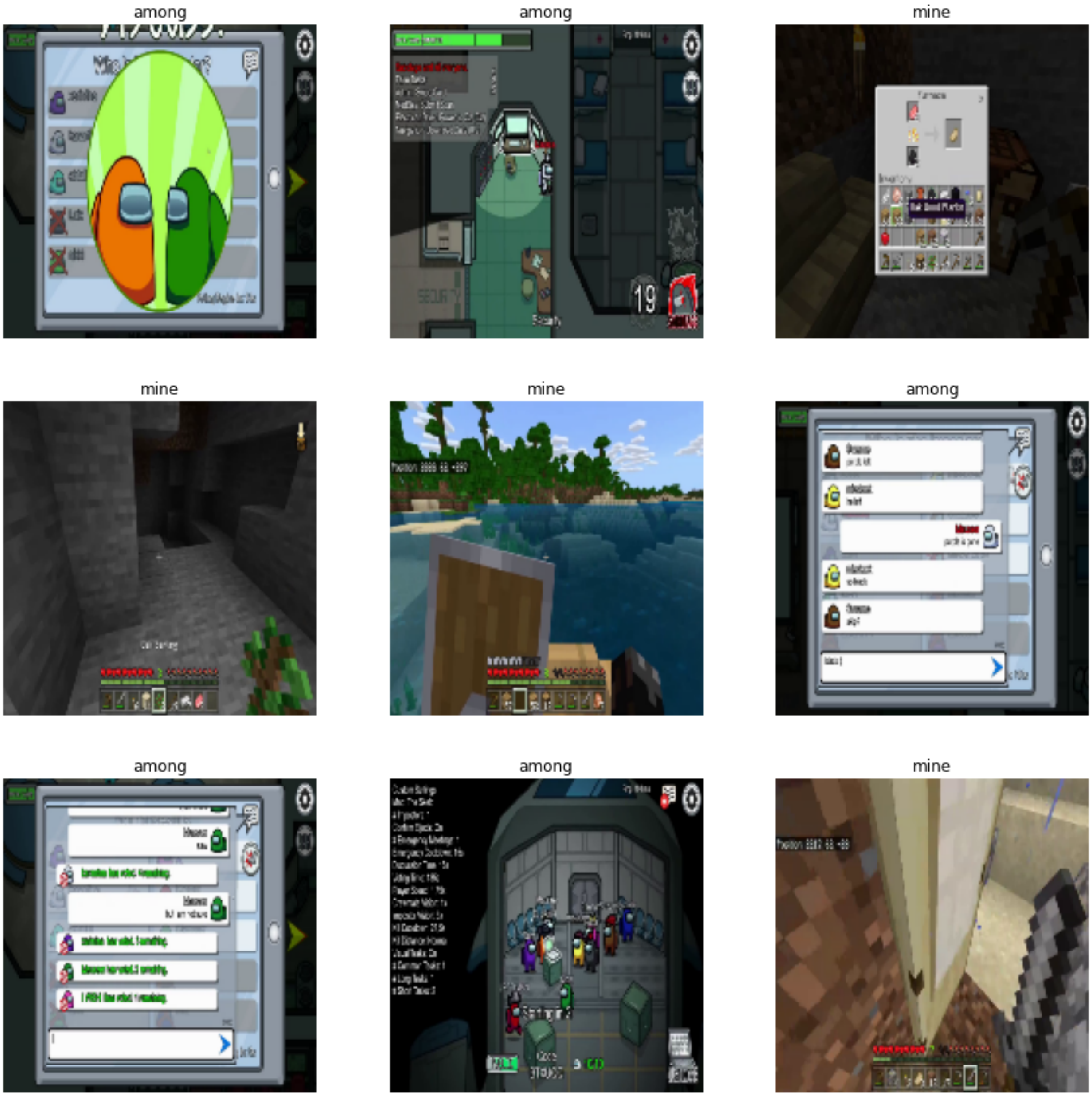
```
<Figure size 432x288 with 0 Axes>
```



In [152]:

```
plot_dataset(dataset_validation)
```

```
<Figure size 432x288 with 0 Axes>
```



among among mine

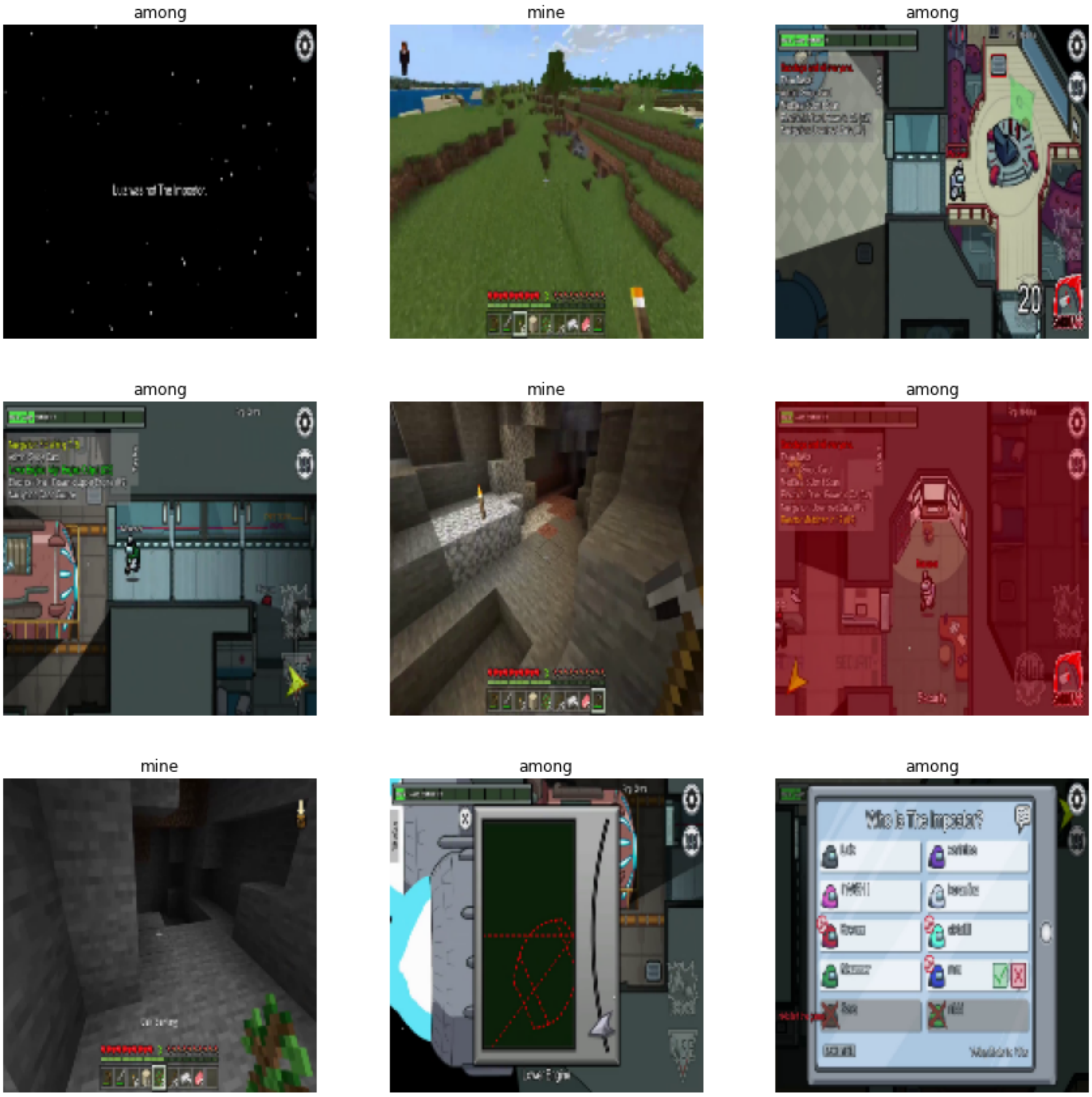
mine mine among

among among mine

In [153]:

```
plot_dataset(dataset_test)
```

```
<Figure size 432x288 with 0 Axes>
```



In [154]:

```
data_augmentation = tf.keras.models.Sequential([
    tf.keras.layers.experimental.preprocessing.RandomFlip('horizontal'),
    tf.keras.layers.experimental.preprocessing.RandomRotation(0.2),
    tf.keras.layers.experimental.preprocessing.RandomZoom(0.2)
])
```

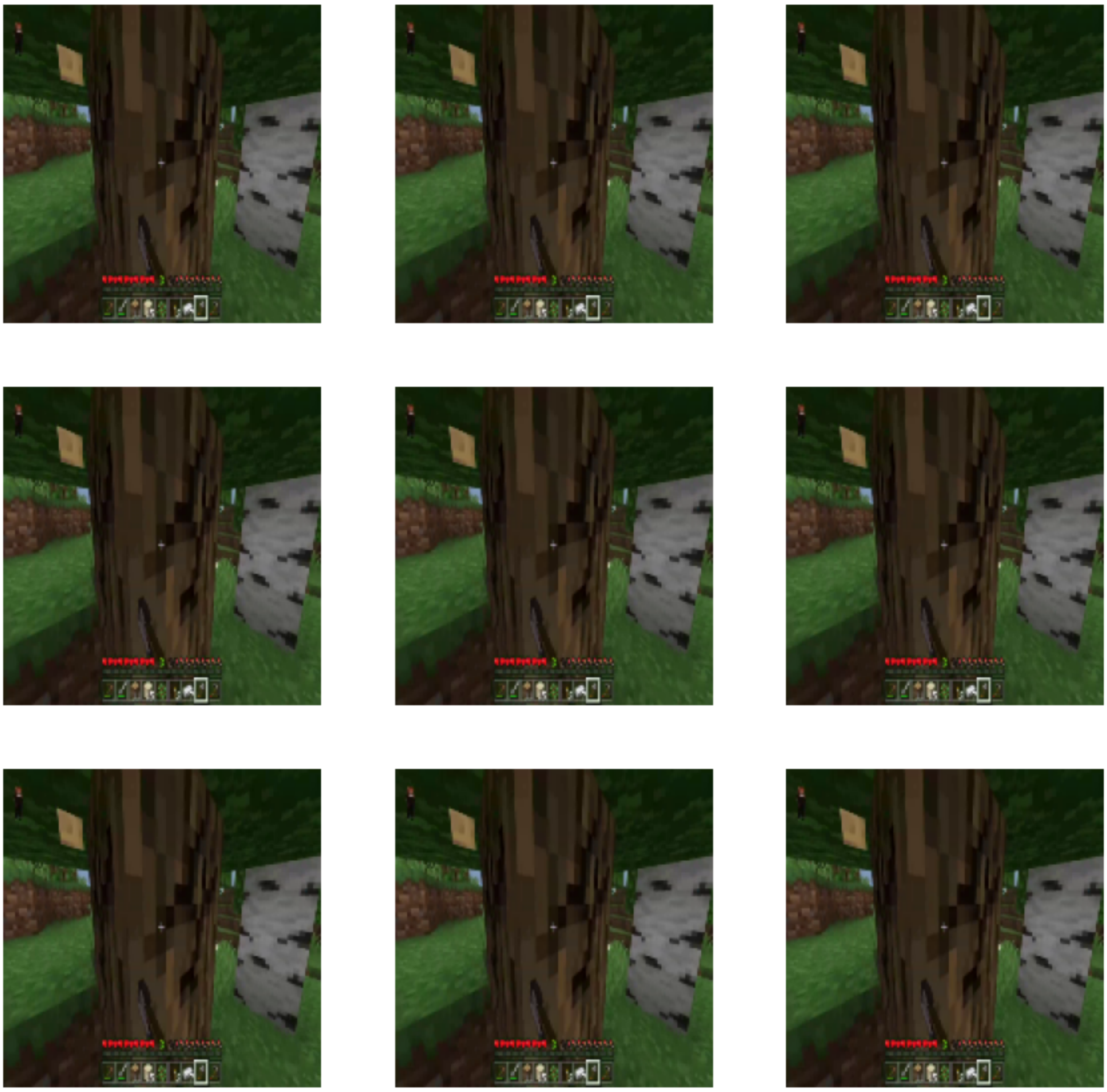
In [155]:

```
def plot_dataset_data_augmentation(dataset):

    plt.gcf().clear()
    plt.figure(figsize = (15, 15))

    for features, _ in dataset.take(1):

        feature = features[0]

        for i in range(9):

            feature_data_augmentation = data_augmentation(tf.expand_dims(feature, 0))

            plt.subplot(3, 3, i + 1)
            plt.axis('off')

            plt.imshow(feature_data_augmentation[0] / image_color_channel_size)
```

In [156]:

```
plot_dataset_data_augmentation(dataset_train)
```

```
<Figure size 432x288 with 0 Axes>
```



In [157]:

```
model_transfer_learning = tf.keras.applications.MobileNetV2(
    input_shape = image_shape,
    include_top = False,
    weights = 'imagenet'
)

model_transfer_learning.trainable = False

model_transfer_learning.summary()
```

```
Model: "mobilenetv2_1.00_160"
________________________________________________________________________
__________________________
 Layer (type)                   Output Shape         Param #     Connect
ed to
========================================================================
==========================
 input_2 (InputLayer)           [(None, 160, 160, 3  0           []
                                )]

 Conv1 (Conv2D)                 (None, 80, 80, 32)   864         ['input
_2[0][0]']

 bn_Conv1 (BatchNormalization)  (None, 80, 80, 32)   128         ['Conv1
[0][0]']

 Conv1_relu (ReLU)              (None, 80, 80, 32)   0           ['bn_Co
nv1[0][0]']

 expanded_conv_depthwise (Depth (None, 80, 80, 32)   288         ['Conv1
```

In [158]:

```
model = tf.keras.models.Sequential([
    tf.keras.layers.experimental.preprocessing.Rescaling(
        1. / image_color_channel_size,
        input_shape = image_shape
    ),
    data_augmentation,
    model_transfer_learning,
    tf.keras.layers.GlobalAveragePooling2D(),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(1, activation = 'sigmoid')
])

model.compile(
    optimizer=tf.keras.optimizers.Adam(learning_rate = learning_rate),
    loss = tf.keras.losses.BinaryCrossentropy(),
    metrics = ['accuracy']
)

model.summary()
```

```
Model: "sequential_13"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 rescaling_12 (Rescaling)    (None, 160, 160, 3)       0         
                                                                 
 sequential_12 (Sequential)  (None, 160, 160, 3)       0         
                                                                 
 mobilenetv2_1.00_160 (Funct  (None, 5, 5, 1280)       2257984   
 ional)                                                          
                                                                 
 global_average_pooling2d_1   (None, 1280)             0         
 (GlobalAveragePooling2D)                                        
                                                                 
 dropout_1 (Dropout)         (None, 1280)              0         
                                                                 
 dense_21 (Dense)            (None, 1)                 1281      
                                                                 
=================================================================
Total params: 2,259,265
Trainable params: 1,281
Non-trainable params: 2,257,984
_________________________________________________________________
```

In [159]:

```
history = model.fit(
    dataset_train,
    validation_data = dataset_validation,
    epochs = epochs
)
```

```
Epoch 1/20
31/31 [==============================] - 37s 1s/step - loss: 0.7957 - accu
racy: 0.5164 - val_loss: 0.7763 - val_accuracy: 0.5415
Epoch 2/20
31/31 [==============================] - 32s 1s/step - loss: 0.6422 - accu
racy: 0.6360 - val_loss: 0.6806 - val_accuracy: 0.6324
Epoch 3/20
31/31 [==============================] - 31s 1s/step - loss: 0.5222 - accu
racy: 0.7331 - val_loss: 0.5923 - val_accuracy: 0.7022
Epoch 4/20
31/31 [==============================] - 32s 1s/step - loss: 0.4805 - accu
racy: 0.7873 - val_loss: 0.5163 - val_accuracy: 0.7589
Epoch 5/20
31/31 [==============================] - 31s 1s/step - loss: 0.4089 - accu
racy: 0.8384 - val_loss: 0.4666 - val_accuracy: 0.7918
Epoch 6/20
31/31 [==============================] - 32s 1s/step - loss: 0.3763 - accu
racy: 0.8569 - val_loss: 0.4220 - val_accuracy: 0.8182
Epoch 7/20
31/31 [==============================] - 32s 1s/step - loss: 0.3377 - accu
racy: 0.8773 - val_loss: 0.3841 - val_accuracy: 0.8498
Epoch 8/20
31/31 [==============================] - 35s 1s/step - loss: 0.2921 - accu
racy: 0.9090 - val_loss: 0.3514 - val_accuracy: 0.8814
Epoch 9/20
31/31 [==============================] - 32s 1s/step - loss: 0.2837 - accu
racy: 0.9080 - val_loss: 0.3208 - val_accuracy: 0.9025
Epoch 10/20
31/31 [==============================] - 33s 1s/step - loss: 0.2553 - accu
racy: 0.9366 - val_loss: 0.2932 - val_accuracy: 0.9275
Epoch 11/20
31/31 [==============================] - 36s 1s/step - loss: 0.2336 - accu
racy: 0.9427 - val_loss: 0.2728 - val_accuracy: 0.9368
Epoch 12/20
31/31 [==============================] - 34s 1s/step - loss: 0.2223 - accu
racy: 0.9448 - val_loss: 0.2573 - val_accuracy: 0.9407
Epoch 13/20
31/31 [==============================] - 34s 1s/step - loss: 0.2062 - accu
racy: 0.9540 - val_loss: 0.2396 - val_accuracy: 0.9513
Epoch 14/20
31/31 [==============================] - 34s 1s/step - loss: 0.1952 - accu
racy: 0.9550 - val_loss: 0.2337 - val_accuracy: 0.9486
Epoch 15/20
31/31 [==============================] - 36s 1s/step - loss: 0.1717 - accu
racy: 0.9755 - val_loss: 0.2181 - val_accuracy: 0.9526
Epoch 16/20
31/31 [==============================] - 35s 1s/step - loss: 0.1610 - accu
racy: 0.9714 - val_loss: 0.2063 - val_accuracy: 0.9539
Epoch 17/20
31/31 [==============================] - 34s 1s/step - loss: 0.1586 - accu
racy: 0.9642 - val_loss: 0.1962 - val_accuracy: 0.9592
Epoch 18/20
31/31 [==============================] - 34s 1s/step - loss: 0.1492 - accu
```

```
racy: 0.9765 - val_loss: 0.1861 - val_accuracy: 0.9592
Epoch 19/20
31/31 [==============================] - 33s 1s/step - loss: 0.1415 - accu
racy: 0.9785 - val_loss: 0.1755 - val_accuracy: 0.9592
Epoch 20/20
31/31 [==============================] - 34s 1s/step - loss: 0.1337 - accu
racy: 0.9847 - val_loss: 0.1711 - val_accuracy: 0.9631
```

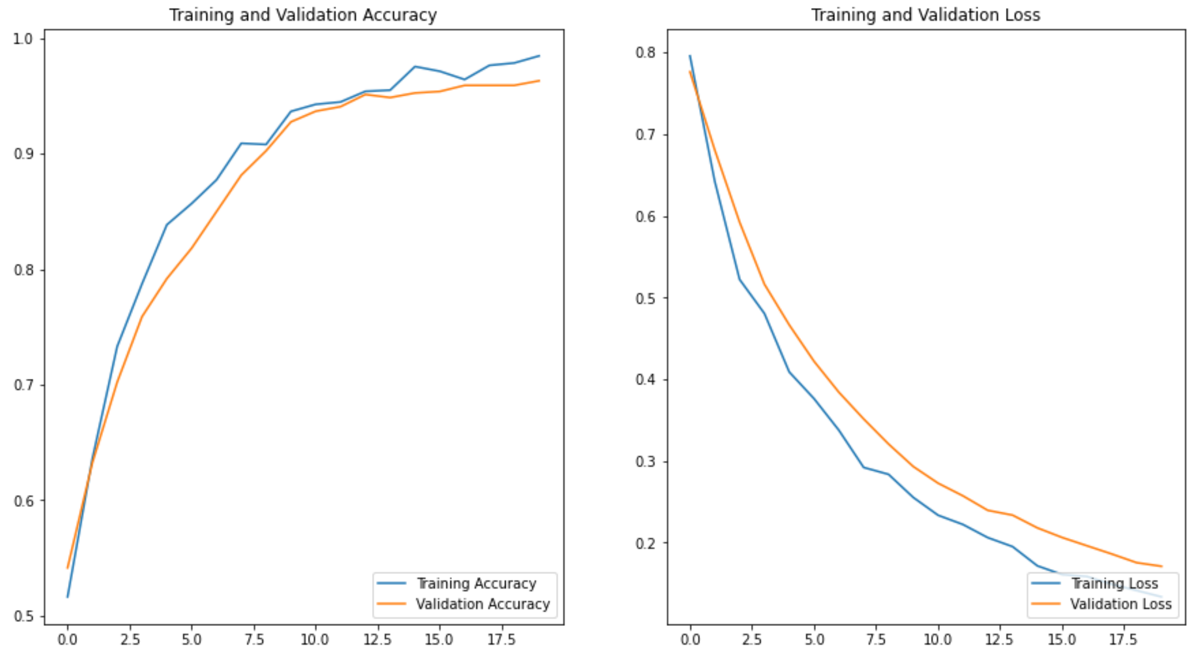
In [160]:

```
def plot_model():

    accuracy = history.history['accuracy']
    val_accuracy = history.history['val_accuracy']

    loss = history.history['loss']
    val_loss = history.history['val_loss']

    epochs_range = range(epochs)

    plt.gcf().clear()
    plt.figure(figsize = (15, 8))

    plt.subplot(1, 2, 1)
    plt.title('Training and Validation Accuracy')
    plt.plot(epochs_range, accuracy, label = 'Training Accuracy')
    plt.plot(epochs_range, val_accuracy, label = 'Validation Accuracy')
    plt.legend(loc = 'lower right')

    plt.subplot(1, 2, 2)
    plt.title('Training and Validation Loss')
    plt.plot(epochs_range, loss, label = 'Training Loss')
    plt.plot(epochs_range, val_loss, label = 'Validation Loss')
    plt.legend(loc = 'lower right')

    plt.show()
```

In [161]:

```
plot_model()
```

```
<Figure size 432x288 with 0 Axes>
```



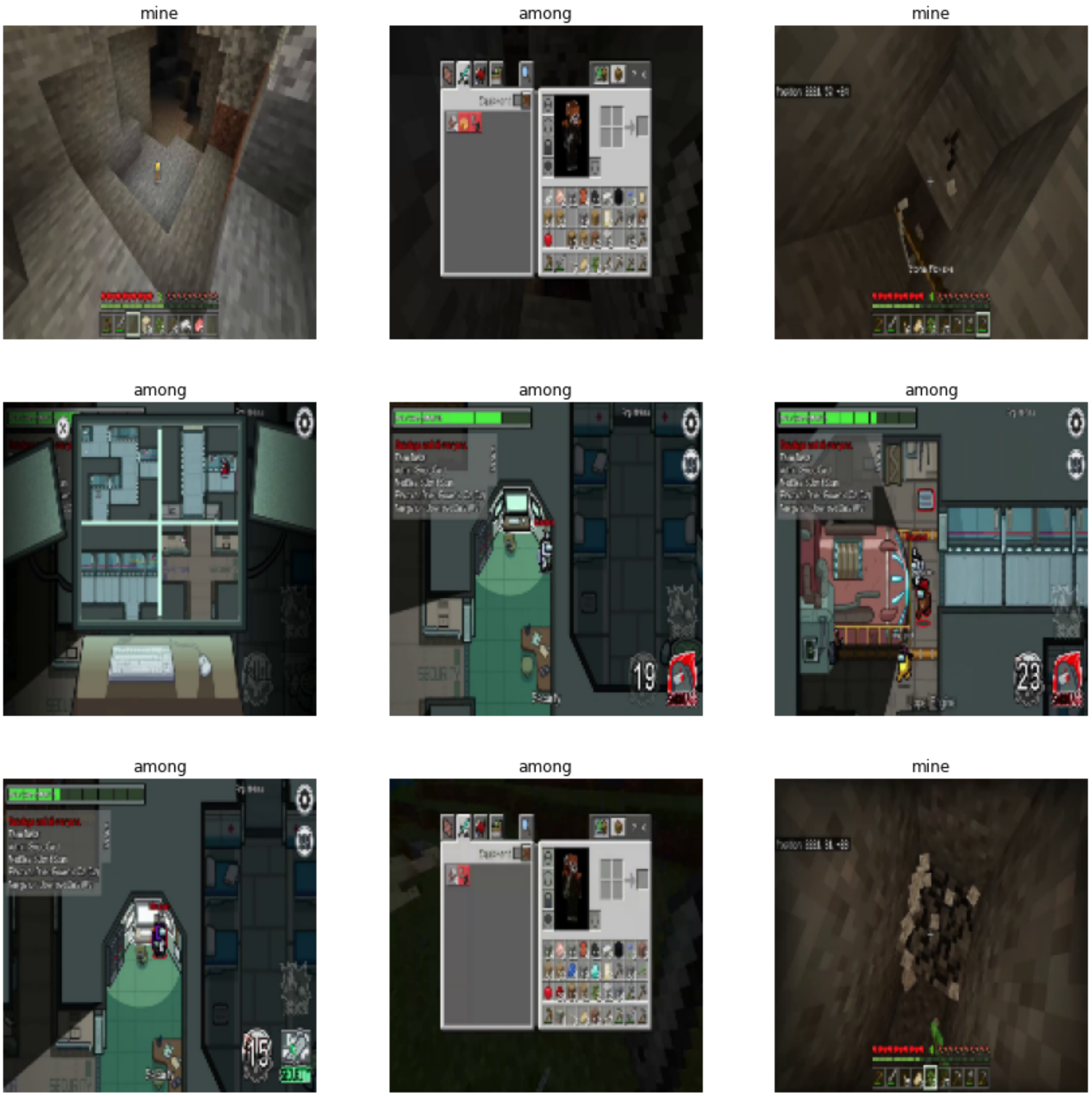
In [162]:

```
def plot_dataset_predictions(dataset):

    features, labels = dataset.as_numpy_iterator().next()

    predictions = model.predict_on_batch(features).flatten()
    predictions = tf.where(predictions < 0.5, 0, 1)

    print('Labels:     %s' % labels)
    print('Predictions: %s' %predictions.numpy())

    plt.gcf().clear()
    plt.figure(figsize = (15, 15))

    for i in range(9):

        plt.subplot(3, 3, i + 1)
        plt.axis('off')

        plt.imshow(features[i].astype('uint8'))
        plt.title(class_names[predictions[i]])
```

In [163]:

```
plot_dataset_predictions(dataset_test)
```

```
Labels:     [1 1 1 0 0 0 0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 0 0 1 0 1 0 1 0 0 1
0]
Predictions: [1 0 1 0 0 0 0 0 1 0 0 0 0 0 1 1 1 1 1 1 1 0 0 1 0 0 0 1 0 0
1 0]

<Figure size 432x288 with 0 Axes>
```



mine

among

mine

among

among

among

among

among

mine

In [164]:

```
model.save('path/to/model')
```

```
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_o
p, _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compil
ed_convolution_op, _jit_compiled_convolution_op while saving (showing 5 of
52). These functions will not be directly callable after loading.
```

```
INFO:tensorflow:Assets written to: path/to/model\assets
```

```
INFO:tensorflow:Assets written to: path/to/model\assets
```

In [ ]:

```

```