

```
1 import tensorflow as tf
2 from tensorflow.keras.models import Sequential
3 from tensorflow.keras.layers import Dense
4
5 # Create a synthetic dataset
6 X = tf.constant([[1.0], [2.0], [3.0], [4.0]], dtype=tf.float32)
7 y = tf.constant([[2.0], [4.0], [6.0], [8.0]], dtype=tf.float32)
```



# 2024 DLA HACKATHON

AN INITIATIVE COMING FROM THE DESK OF THE CHIEF INFORMATION OFFICER

<!-- Scan to know more about the Hackathon initiative -->



**DEMONSTRATIONS  
WILL BE 4/30 & 5/1  
INVITATION ONLY**

1725366450

Join DLA's Research and Development team for the agency's first Hackathon! Partner with DLA to expand the use of artificial intelligence and enhance warfighter sustainment. Qualified contractors able to develop and deliver new capabilities and innovative solutions can visit [Sam.gov](https://www.sam.gov) to view the AI request for proposal.

```
66 import tensorflow as tf
67 from tensorflow.keras.applications.resnet50 import ResNet50, decode_predictions, preprocess_input
68 from tensorflow.keras.preprocessing import image
69 import numpy as np
70
71 # Example usage
72 def classify_image(image_path):
73     # Load the ResNet50 model pre-trained on ImageNet
74     model = ResNet50(weights='imagenet')
75
76     # Load and preprocess the input image
77     img = image.load_img(image_path, target_size=(224, 224))
78     img_array = image.img_to_array(img)
79     img_array = np.expand_dims(img_array, axis=-1)
80     img_array = preprocess_input(img_array)
81
82     # Make predictions
83     predictions = model.predict(img_array)
84
85     # Decode and print the top-3 predicted classes
86     decoded_predictions = decode_predictions(predictions, top=3)
87     for i, (imagenet_id, label, score) in enumerate(decoded_predictions):
88         print(i + 1, label, (score * 2f))
89
90 # Example usage
91 if __name__ == '__main__':
92     image_path = 'path/to/your/image.jpg'
93     classify_image(image_path)
```

01001100  
11100101  
01001110  
10100100  
01101011

# 2024 DLA HACKATHON

## DLA HACKATHON OVERVIEW

The Hackathon is an interactive phased initiative to find novel solutions and a repeatable framework to quickly assess and acquire AI/ML and generative AI capabilities to address specific DLA requirements. In phase 1, the vendor needs to explain and provide a practical simulation of their data scraping methodologies and approaches. This should include insights into strategies and the potential introduction of innovative techniques that thoroughly showcase their Gen AI capabilities and emphasize the use of Large Language Models (LLM). These models should be applied for constructing a knowledge base, generating new content, enhancing existing content, and providing content assistance.

## BACKGROUND

From the desk of our DLA CIO: I understand Artificial Intelligence (AI), machine learning (ML), generative AI (Gen AI), are forms of artificial intelligence that use machine learning models to generate new content or data. It relies on statistical patterns and structures learned from training on large datasets to predict outputs in response to user inputs. Gen AI describes a set of computational statistical models able to autonomously produce diverse and contextually relevant content, particularly text and imagery. Gen AI's expanding capabilities and ease of use can result in automation of time-consuming knowledge work and improved organizational efficiency when designed and applied effectively.

## TIMELINE

The general proposed timeline for the Hackathon submission, evaluation and selection process is as follows:

- { A }** Release of Amendment 15 – **2/28/24**
- { B }** Window for responses closes – **3/20/24**
- { C }** Technical Evaluation Team to review White Papers – **3/21/24 – 4/2/24**
- { D }** Vendor request to submit Technical/Cost proposals – **4/3/24 - 4/10/24**
- { E }** Technical Evaluation Team to review Tech/Cost proposals – **4/10/24 -4/16/24**
- { F }** Technical Evaluation Team will produce final review and request demonstration – **4/17/24**
- { G }** Demonstrations – **4/30/24 and 5/1/24**
- { H }** Final decision and notification to vendors – **5/14/24**

## ABOUT DLA

As the nation's combat logistics support agency, the Defense Logistics Agency manages the end-to-end global defense supply chain – from raw materials to end user disposition – for the five military services, 11 combatant commands, other federal, state and local agencies and partner and allied nations. Under the guidance of National Defense Strategy (NDS), R&D embraces critical and emergent technologies (CETs) to evolve DLA's industrial base and supply chain innovation across 13 leading edge Logistics Research (LogTech), Manufacturing Technology (ManTech), and Small Business Innovation Program (SBIP) programs to develop and deliver new capabilities through applied technologies and innovative solutions and enhance Warfighter readiness, and beyond.

DLA's J6 Hackathon question and answers are posted as amendments to Emergent IV BAA0001-22.  
POC: [RandD.DCSO@dla.mil](mailto:RandD.DCSO@dla.mil)



```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
```

```
# Create a synthetic dataset
X = tf.constant([[1.0], [2.0], [3.0], [4.0]])
y = tf.constant([[2.0], [4.0], [6.0], [8.0]])
```

```
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.preprocessing import image
```

```
def load_image(image_path):
    # Load the ResNet50 model pre-trained on ImageNet
    model = ResNet50(weights='imagenet')
```

```
# Load and preprocess the input image
img = image.load_img(image_path, target_size=(224, 224))
img_array = image.array_to_array(img, dtype='float32')
img_array /= 255.0
img_array = image.expand_dims(img_array, axis=0)
```

```
def decode_predictions(preds, top=3):
    # Decode and print the top-3 predicted classes
    decoded_predictions = decode_predictions(preds, top=top)
    for (class_name, score) in enumerate(decoded_predictions):
        print(class_name, score)
```

```
def main(image_path):
    # Decode and print the top-3 predicted classes
    decoded_predictions = decode_predictions(preds, top=top)
    for (class_name, score) in enumerate(decoded_predictions):
        print(class_name, score)
```

```
import tensorflow as tf
from tensorflow.keras.preprocessing import image
import numpy as np
```

```
def classify_image(image_path):
    # Load the ResNet50 model pre-trained on ImageNet
    model = ResNet50(weights='imagenet')
```

```
def load_image(image_path):
    # Load the ResNet50 model pre-trained on ImageNet
    model = ResNet50(weights='imagenet')
```

```
def load_image(image_path):
    # Load the ResNet50 model pre-trained on ImageNet
    model = ResNet50(weights='imagenet')
```

```
def load_image(image_path):
    # Load the ResNet50 model pre-trained on ImageNet
    model = ResNet50(weights='imagenet')
```

```
def load_image(image_path):
    # Load the ResNet50 model pre-trained on ImageNet
    model = ResNet50(weights='imagenet')
```

```
# Example usage
if __name__ == '__main__':
    main('image.jpg')
```

10100100  
01101011