**Task # 1: Use the ML model to identify an object:**

**Program:**

**import** UIKit

**import** CoreML

**import** Vision

**class** ViewController: UIViewController, UIImagePickerControllerDelegate, UINavigationControllerDelegate {

**@IBOutlet** **weak** **var** shareBtn: UIButton!

**@IBOutlet** **weak** **var** libraryBtn: UIBarButtonItem!

**@IBOutlet** **weak** **var** cameraBtn: UIBarButtonItem!

**@IBOutlet** **weak** **var** category: UISegmentedControl!

**@IBOutlet** **weak** **var** myImage: UIImageView!

**let** imagePicker = UIImagePickerController()

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

        imagePicker.delegate = **self**

        imagePicker.allowsEditing = **false**

        shareBtn.isHidden = **true**

    }

**@IBAction** **func** myCamera(**\_** sender: UIBarButtonItem) {

        imagePicker.sourceType = .camera

        present(imagePicker, animated: **true**, completion: **nil**)

    }

**@IBAction** **func** accessPhotoLibrary(**\_** sender: UIBarButtonItem) {

        imagePicker.sourceType = .photoLibrary

        present(imagePicker, animated: **true**, completion: **nil**)

    }

**func** imagePickerController(**\_** picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [UIImagePickerController.InfoKey : **Any**]) {

**let** info = convertFromUIImagePickerControllerInfoKeyDictionary(info)

        cameraBtn.isEnabled = **false**

        libraryBtn.isEnabled = **false**

**if** **let** userImage = info[convertFromUIImagePickerControllerInfoKey(UIImagePickerController.InfoKey.originalImage)] **as**? UIImage {

            myImage.image = userImage

**guard** **let** ciImage = CIImage(image: userImage) **else** {

                fatalError("Could not convert uiImage to ciImage")

            }

            detect(image: ciImage)

        }

        imagePicker.dismiss(animated: **true**, completion: **nil**)

    }

**func** detect(image: CIImage) {

**guard** **let** model = **try**? VNCoreMLModel(for: Inceptionv3().model) **else** {

            fatalError("Loading coreML model failed ")

        }

**let** request = VNCoreMLRequest(model: model) { (request, error) **in**

**guard** **let** results = request.results **as**? [VNClassificationObservation] **else** {

                fatalError("Model failed to process image")

            }

            print(results)

            DispatchQueue.main.async {

**self**.cameraBtn.isEnabled = **true**

**self**.libraryBtn.isEnabled = **true**

**self**.shareBtn.isHidden = **false**

            }

**if** **let** firstResult = results.first {

**if** firstResult.identifier.contains("cliff") {

                    DispatchQueue.main.async {

**self**.navigationItem.title = "Correct!"

**self**.navigationController?.navigationBar.barTintColor = UIColor.green

                    }

                } **else** {

                    DispatchQueue.main.async {

**self**.navigationItem.title = "Incorrect!"

**self**.navigationController?.navigationBar.barTintColor = UIColor.red

                    }

                }

            }

        }

**let** handler = VNImageRequestHandler(ciImage: image)

**do** {

**try** handler.perform([request])

        } **catch** {

            print(error)

        }

    }

**Output:**

