NAME: JIGAR MAHETA

ID: n01416437

IOS ASSIGNMENT-3

Link for GIT :

Models:

Models for Image:

img:-

//

// img.swift

// Created by user202348 on 10/25/21.

// Copyright © 2021 user202348. All rights reserved.

import Foundation

class img

{

var imageName:String=""

var imageUrl:String=""

init(a:String,b:String) {

imageName=a

imageUrl=b

}

}

imgManager:

//

// imgManager.swift

// Created by user202348 on 10/25/21.

// Copyright © 2021 user202348. All rights reserved.

import Foundation

class imgManager

{

private var imageList = [image]()

//appending new image in the list

func addNewImage(i:image)

{

imageList.append(i)

}

//get all the added image.

func getImageList()->[image]{

return imageList

}

}

imageViewController:

//

// imageViewController.swift

// Created by user202348 on 10/25/21.

// Copyright © 2021 user202348. All rights reserved.

import UIKit

protocol AddingPictureProtocol {

func controllerDidFinishWithNewPicture(p:image)

func controllerDidCancel()

}

class imageViewController: UIViewController {

override func viewDidLoad() {

super.viewDidLoad()

}

var delegate:AddingPictureProtocol?

@IBOutlet weak var imageName: UITextField!

@IBOutlet weak var imageUrl: UITextField!

//adding imgae when button is pressedgg

@IBAction func BtnAddImageClicked(\_ sender: Any) {

if let imgName = imageName.text{

if let imgUrl = imageUrl.text{

if !imgName.isEmpty && !imgUrl.isEmpty{

let newImage = image(a: imgName, b: imgUrl)

delegate?.controllerDidFinishWithNewPicture(p: newImage)

dismiss(animated: true, completion: nil)

}

}

}

}

@IBAction func CancelClicked(\_ sender: Any) {

delegate?.controllerDidCancel()

dismiss(animated: true, completion: nil)

}

}

ViewController:

//

// ViewController.swift

// Created by user202348 on 10/25/21.

// Copyright © 2021 user202348. All rights reserved.

import UIKit

class ViewController: UIViewController, UIPickerViewDelegate, UIPickerViewDataSource, AddingPictureProtocol{

@IBOutlet weak var imgView: UIImageView!

@IBOutlet weak var imgName: UIPickerView!

var manager:imageManager = imageManager()

override func viewDidLoad() {

super.viewDidLoad()

// Do any additional setup after loading the view.

}

func numberOfComponents(in pickerView: UIPickerView) -> Int {

return 1

}

func pickerView(\_ pickerView: UIPickerView, numberOfRowsInComponent component: Int) -> Int {

return manager.getImageList().count

}

func pickerView(\_ pickerView: UIPickerView, titleForRow row: Int, forComponent component: Int) -> String? {

return manager.getImageList()[row].imageName

}

func pickerView(\_ pickerView: UIPickerView, didSelectRow row: Int, inComponent component: Int) {

let queue = DispatchQueue.init(label: "1")

queue.async {

let data = try? Data(contentsOf: URL(string: self.manager.getImageList()[row].imageUrl)!)

DispatchQueue.main.async {

self.imgView.image = UIImage(data: data!)

}

}

}

func controllerDidFinishWithNewPicture(p: image) {

manager.addNewImage(i: p)

imgName.reloadAllComponents()

}

func controllerDidCancel() {

}

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

let apvc = segue.destination as!imageViewController

apvc.delegate = self

}

}

MainView:

<?xml version="1.0" encoding="UTF-8"?>

<document type="com.apple.InterfaceBuilder3.CocoaTouch.Storyboard.XIB" version="3.0" toolsVersion="17701" targetRuntime="iOS.CocoaTouch" propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES" useSafeAreas="YES" colorMatched="YES" initialViewController="OF5-jz-82P">

<device id="retina6\_1" orientation="portrait" appearance="light"/>

<dependencies>

<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin" version="17703"/>

<capability name="Safe area layout guides" minToolsVersion="9.0"/>

<capability name="System colors in document resources" minToolsVersion="11.0"/>

<capability name="documents saved in the Xcode 8 format" minToolsVersion="8.0"/>

</dependencies>

<scenes>

<!--Image Viewer-->

<scene sceneID="tne-QT-ifu">

<objects>

<viewController id="BYZ-38-t0r" customClass="ViewController" customModule="Image\_Viewer\_Application" customModuleProvider="target" sceneMemberID="viewController">

<view key="view" contentMode="scaleToFill" id="8bC-Xf-vdC">

<rect key="frame" x="0.0" y="0.0" width="414" height="896"/>

<autoresizingMask key="autoresizingMask" widthSizable="YES" heightSizable="YES"/>

<subviews>

<pickerView contentMode="scaleToFill" fixedFrame="YES" translatesAutoresizingMaskIntoConstraints="NO" id="mYN-GX-wyC">

<rect key="frame" x="0.0" y="659" width="414" height="162"/>

<autoresizingMask key="autoresizingMask" widthSizable="YES" flexibleMaxY="YES"/>

<connections>

<outlet property="dataSource" destination="BYZ-38-t0r" id="LT1-6d-8zX"/>

<outlet property="delegate" destination="BYZ-38-t0r" id="acU-64-upr"/>

</connections>

</pickerView>

<imageView clipsSubviews="YES" userInteractionEnabled="NO" contentMode="scaleAspectFit" horizontalHuggingPriority="251" verticalHuggingPriority="251" fixedFrame="YES" translatesAutoresizingMaskIntoConstraints="NO" id="XW4-kV-C4l">

<rect key="frame" x="1" y="159" width="414" height="331"/>

<autoresizingMask key="autoresizingMask" flexibleMaxX="YES" flexibleMaxY="YES"/>

<color key="tintColor" white="1" alpha="1" colorSpace="custom" customColorSpace="genericGamma22GrayColorSpace"/>

</imageView>

</subviews>

<viewLayoutGuide key="safeArea" id="6Tk-OE-BBY"/>

<color key="backgroundColor" systemColor="systemBackgroundColor"/>

</view>

<navigationItem key="navigationItem" title="Image Viewer" id="hjb-0F-Mw9">

<barButtonItem key="rightBarButtonItem" systemItem="add" id="5Rp-8X-arR">

<connections>

<segue destination="JXz-Xm-80b" kind="presentation" modalPresentationStyle="fullScreen" id="lwL-pK-gyS"/>

</connections>

</barButtonItem>

</navigationItem>

<connections>

<outlet property="imgName" destination="mYN-GX-wyC" id="M9a-Ev-Yhk"/>

<outlet property="imgView" destination="XW4-kV-C4l" id="vg4-dq-gud"/>

</connections>

</viewController>

<placeholder placeholderIdentifier="IBFirstResponder" id="dkx-z0-nzr" sceneMemberID="firstResponder"/>

</objects>

<point key="canvasLocation" x="928.98550724637687" y="93.75"/>

</scene>

<!--Add New Image-->

<scene sceneID="STc-2J-2z8">

<objects>

<viewController id="JXz-Xm-80b" customClass="imageViewController" customModule="Image\_Viewer\_Application" customModuleProvider="target" sceneMemberID="viewController">

<view key="view" contentMode="scaleToFill" id="g4K-vd-oL9">

<rect key="frame" x="0.0" y="0.0" width="414" height="896"/>

<autoresizingMask key="autoresizingMask" widthSizable="YES" heightSizable="YES"/>

<subviews>

<textField opaque="NO" contentMode="scaleToFill" fixedFrame="YES" contentHorizontalAlignment="left" contentVerticalAlignment="center" borderStyle="roundedRect" placeholder="Enter Image Title" textAlignment="center" minimumFontSize="17" translatesAutoresizingMaskIntoConstraints="NO" id="gWN-uf-SDd">

<rect key="frame" x="20" y="256" width="374" height="34"/>

<autoresizingMask key="autoresizingMask" flexibleMaxX="YES" flexibleMaxY="YES"/>

<fontDescription key="fontDescription" type="system" pointSize="14"/>

<textInputTraits key="textInputTraits"/>

</textField>

<textField opaque="NO" contentMode="scaleToFill" fixedFrame="YES" contentHorizontalAlignment="left" contentVerticalAlignment="center" borderStyle="roundedRect" placeholder="Enter a new image URL" textAlignment="center" minimumFontSize="17" translatesAutoresizingMaskIntoConstraints="NO" id="FWm-Jt-Tgx">

<rect key="frame" x="20" y="325" width="374" height="34"/>

<autoresizingMask key="autoresizingMask" flexibleMaxX="YES" flexibleMaxY="YES"/>

<fontDescription key="fontDescription" type="system" pointSize="14"/>

<textInputTraits key="textInputTraits"/>

</textField>

<button opaque="NO" contentMode="scaleToFill" fixedFrame="YES" contentHorizontalAlignment="center" contentVerticalAlignment="center" buttonType="system" lineBreakMode="middleTruncation" translatesAutoresizingMaskIntoConstraints="NO" id="AL0-bf-Vd8">

<rect key="frame" x="95" y="433" width="71" height="41"/>

<autoresizingMask key="autoresizingMask" flexibleMaxX="YES" flexibleMaxY="YES"/>

<state key="normal" title="Cancel"/>

<connections>

<action selector="CancelClicked:" destination="JXz-Xm-80b" eventType="touchUpInside" id="Wkq-Wz-VZi"/>

</connections>

</button>

<button opaque="NO" contentMode="scaleToFill" fixedFrame="YES" contentHorizontalAlignment="center" contentVerticalAlignment="center" buttonType="system" lineBreakMode="middleTruncation" translatesAutoresizingMaskIntoConstraints="NO" id="Jws-1P-IEy">

<rect key="frame" x="226" y="433" width="71" height="41"/>

<autoresizingMask key="autoresizingMask" flexibleMaxX="YES" flexibleMaxY="YES"/>

<state key="normal" title="Add"/>

<connections>

<action selector="BtnAddImageClicked:" destination="JXz-Xm-80b" eventType="touchUpInside" id="4nW-cz-nTn"/>

</connections>

</button>

<navigationBar contentMode="scaleToFill" fixedFrame="YES" translatesAutoresizingMaskIntoConstraints="NO" id="qW0-L3-iqP">

<rect key="frame" x="0.0" y="36" width="414" height="44"/>

<autoresizingMask key="autoresizingMask" widthSizable="YES" flexibleMaxY="YES"/>

<items>

<navigationItem title="Add New Image" id="PWW-VU-3aW"/>

</items>

</navigationBar>

</subviews>

<viewLayoutGuide key="safeArea" id="ZMF-Mr-FBU"/>

<color key="backgroundColor" systemColor="systemBackgroundColor"/>

</view>

<navigationItem key="navigationItem" title="Add New Image" id="18W-6G-OLS"/>

<connections>

<outlet property="imageName" destination="gWN-uf-SDd" id="ken-Cf-Mbf"/>

<outlet property="imageUrl" destination="FWm-Jt-Tgx" id="Yjr-Ap-dAd"/>

</connections>

</viewController>

<placeholder placeholderIdentifier="IBFirstResponder" id="gJa-Rn-CRb" userLabel="First Responder" customClass="UIResponder" sceneMemberID="firstResponder"/>

</objects>

<point key="canvasLocation" x="1786" y="94"/>

</scene>

<!--Navigation Controller-->

<scene sceneID="DaH-rq-ANX">

<objects>

<navigationController automaticallyAdjustsScrollViewInsets="NO" id="OF5-jz-82P" sceneMemberID="viewController">

<toolbarItems/>

<navigationBar key="navigationBar" contentMode="scaleToFill" id="qsw-5I-7Ds">

<rect key="frame" x="0.0" y="44" width="414" height="44"/>

<autoresizingMask key="autoresizingMask"/>

</navigationBar>

<nil name="viewControllers"/>

<connections>

<segue destination="BYZ-38-t0r" kind="relationship" relationship="rootViewController" id="1RB-Am-xIG"/>

</connections>

</navigationController>

<placeholder placeholderIdentifier="IBFirstResponder" id="Dva-vC-Iyp" userLabel="First Responder" customClass="UIResponder" sceneMemberID="firstResponder"/>

</objects>

<point key="canvasLocation" x="18.840579710144929" y="93.75"/>

</scene>

</scenes>

<resources>

<systemColor name="systemBackgroundColor">

<color white="1" alpha="1" colorSpace="custom" customColorSpace="genericGamma22GrayColorSpace"/>

</systemColor>

</resources>

</document>

MainScreen

<?xml version="1.0" encoding="UTF-8" standalone="no"?>

<document type="com.apple.InterfaceBuilder3.CocoaTouch.Storyboard.XIB" version="3.0" toolsVersion="13122.16" targetRuntime="iOS.CocoaTouch" propertyAccessControl="none" useAutolayout="YES" launchScreen="YES" useTraitCollections="YES" useSafeAreas="YES" colorMatched="YES" initialViewController="01J-lp-oVM">

<dependencies>

<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin" version="13104.12"/>

<capability name="Safe area layout guides" minToolsVersion="9.0"/>

<capability name="documents saved in the Xcode 8 format" minToolsVersion="8.0"/>

</dependencies>

<scenes>

<!--View Controller-->

<scene sceneID="EHf-IW-A2E">

<objects>

<viewController id="01J-lp-oVM" sceneMemberID="viewController">

<view key="view" contentMode="scaleToFill" id="Ze5-6b-2t3">

<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>

<autoresizingMask key="autoresizingMask" widthSizable="YES" heightSizable="YES"/>

<color key="backgroundColor" xcode11CocoaTouchSystemColor="systemBackgroundColor" cocoaTouchSystemColor="whiteColor"/>

<viewLayoutGuide key="safeArea" id="6Tk-OE-BBY"/>

</view>

</viewController>

<placeholder placeholderIdentifier="IBFirstResponder" id="iYj-Kq-Ea1" userLabel="First Responder" sceneMemberID="firstResponder"/>

</objects>

<point key="canvasLocation" x="53" y="375"/>

</scene>

</scenes>

</document>

AppDeligate:

//

// AppDelegate.swift

// Created by user202348 on 10/25/21.

// Copyright © 2021 user202348. All rights reserved.

import UIKit

@main

class AppDelegate: UIResponder, UIApplicationDelegate {

func application(\_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {

// Override point for customization after application launch.

return true

}

// MARK: UISceneSession Lifecycle

func application(\_ application: UIApplication, configurationForConnecting connectingSceneSession: UISceneSession, options: UIScene.ConnectionOptions) -> UISceneConfiguration {

// Called when a new scene session is being created.

// Use this method to select a configuration to create the new scene with.

return UISceneConfiguration(name: "Default Configuration", sessionRole: connectingSceneSession.role)

}

func application(\_ application: UIApplication, didDiscardSceneSessions sceneSessions: Set<UISceneSession>) {

// Called when the user discards a scene session.

// If any sessions were discarded while the application was not running, this will be called shortly after application:didFinishLaunchingWithOptions.

// Use this method to release any resources that were specific to the discarded scenes, as they will not return.

}

}

SceneDeligate:

//

// SceneDelegate.swift

// Created by user202348 on 10/25/21.

// Copyright © 2021 user202348. All rights reserved.

import UIKit

class SceneDelegate: UIResponder, UIWindowSceneDelegate {

var window: UIWindow?

func scene(\_ scene: UIScene, willConnectTo session: UISceneSession, options connectionOptions: UIScene.ConnectionOptions) {

// Use this method to optionally configure and attach the UIWindow `window` to the provided UIWindowScene `scene`.

// If using a storyboard, the `window` property will automatically be initialized and attached to the scene.

// This delegate does not imply the connecting scene or session are new (see `application:configurationForConnectingSceneSession` instead).

guard let \_ = (scene as? UIWindowScene) else { return }

}

func sceneDidDisconnect(\_ scene: UIScene) {

// Called as the scene is being released by the system.

// This occurs shortly after the scene enters the background, or when its session is discarded.

// Release any resources associated with this scene that can be re-created the next time the scene connects.

// The scene may re-connect later, as its session was not necessarily discarded (see `application:didDiscardSceneSessions` instead).

}

func sceneDidBecomeActive(\_ scene: UIScene) {

// Called when the scene has moved from an inactive state to an active state.

// Use this method to restart any tasks that were paused (or not yet started) when the scene was inactive.

}

func sceneWillResignActive(\_ scene: UIScene) {

// Called when the scene will move from an active state to an inactive state.

// This may occur due to temporary interruptions (ex. an incoming phone call).

}

func sceneWillEnterForeground(\_ scene: UIScene) {

// Called as the scene transitions from the background to the foreground.

// Use this method to undo the changes made on entering the background.

}

func sceneDidEnterBackground(\_ scene: UIScene) {

// Called as the scene transitions from the foreground to the background.

// Use this method to save data, release shared resources, and store enough scene-specific state information

// to restore the scene back to its current state.

}

}

UITests:

//

// Created by user202348 on 10/25/21.

// Copyright © 2021 user202348. All rights reserved.

import XCTest

class Assignment3UITests: XCTestCase {

override func setUpWithError() throws {

// Put setup code here. This method is called before the invocation of each test method in the class.

// In UI tests it is usually best to stop immediately when a failure occurs.

continueAfterFailure = false

// In UI tests it’s important to set the initial state - such as interface orientation - required for your tests before they run. The setUp method is a good place to do this.

}

override func tearDownWithError() throws {

// Put teardown code here. This method is called after the invocation of each test method in the class.

}

func testExample() throws {

// UI tests must launch the application that they test.

let app = XCUIApplication()

app.launch()

// Use recording to get started writing UI tests.

// Use XCTAssert and related functions to verify your tests produce the correct results.

}

func testLaunchPerformance() throws {

if #available(macOS 10.15, iOS 13.0, tvOS 13.0, \*) {

// This measures how long it takes to launch your application.

measure(metrics: [XCTApplicationLaunchMetric()]) {

XCUIApplication().launch()

}

}

}

}

Tests:

//

//

import XCTest

@testable import Assignment3

class Assignment3Tests: XCTestCase {

override func setUpWithError() throws {

// Put setup code here. This method is called before the invocation of each test method in the class.

}

override func tearDownWithError() throws {

// Put teardown code here. This method is called after the invocation of each test method in the class.

}

func testExample() throws {

// This is an example of a functional test case.

// Use XCTAssert and related functions to verify your tests produce the correct results.

}

func testPerformanceExample() throws {

// This is an example of a performance test case.

self.measure {

// Put the code you want to measure the time of here.

}

}

}