//

// ViewController.swift

// MyVideoApp

//

// Created by Charles Konkol on 2016-05-06.

// Copyright (c) 2016 Rock Valley College. All rights reserved.

// Green highlights: Comments,

// Yellow highlights: code changes/additions

import UIKit

//1) Imports

import MobileCoreServices

import AVFoundation

import CoreData

import CoreMedia

import AVKit

//2 Add to ViewController, UIImagePickerControllerDelegate, UINavigationControllerDelegate

class ViewController: UIViewController, UIImagePickerControllerDelegate, UINavigationControllerDelegate{

//3 Add variables

var moviePlayer:AVPlayerViewController = AVPlayerViewController()

var vidlink:String!

//4) Add variable contactdb (used from UITableView

var videodb:NSManagedObject!

//5) Add ManagedObject Data Context

let managedObjectContext =

(UIApplication.sharedApplication().delegate

as! AppDelegate).managedObjectContext

//6 Create Outlet & Action for btnRecord

//Outlet

@IBOutlet weak var btnRecord: UIButton!

//Action

@IBAction func btnRecord(sender: AnyObject) {

//Code for func btnRecord

if txtName.text == ""

{

let alert = UIAlertController(title: "Name Required", message: "Please add name for video", preferredStyle: UIAlertControllerStyle.Alert)

alert.addAction(UIAlertAction(title: "OK", style: UIAlertActionStyle.Default, handler: nil))

self.presentViewController(alert, animated: true, completion: nil)

}

else

{

RecordVideo()

}

}

//7 Create Outlet Action for btnSave

//Outlet

@IBOutlet weak var btnSave: UIBarButtonItem!

//Action

@IBAction func btnSave(sender: AnyObject) {

//Code for func btnSave

if (videodb != nil) {

// Update existing device

videodb.setValue(txtName.text, forKey: "name")

} else {

// Create a new device

let entityDescription =

NSEntityDescription.entityForName("Video",

inManagedObjectContext: managedObjectContext)

let photod = Video(entity: entityDescription!,

insertIntoManagedObjectContext: managedObjectContext)

photod.name = txtName.text!

photod.datestamp = txtDate.text!

print("asdadadad: " + vidlink)

photod.link = vidlink

}

do {

try managedObjectContext.save()

self.dismissViewControllerAnimated(false, completion: nil)

} catch let error1 as NSError {

print(error1)

}

}

//8 Create Outlet & Action for btnPlay

//Outlet

@IBOutlet weak var btnPlay: UIButton!

//Action

@IBAction func btnPlay(sender: AnyObject) {

//9 Add Code for func btnPlay

let movieURL = NSURL.fileURLWithPath(vidlink!)

let player = AVPlayer(URL:movieURL)

let playerController = AVPlayerViewController()

playerController.player = player

self.addChildViewController(playerController)

self.view.addSubview(playerController.view)

playerController.view.frame = self.view.frame

player.play()

}

//10 Create Outlet for txtName

@IBOutlet weak var txtName: UITextField!

//11 Create Outlet for txtDate

@IBOutlet weak var txtDate: UITextField!

//12 Create Action for btnBack

//Action

@IBAction func btnBack(sender: AnyObject) {

self.dismissViewControllerAnimated(false, completion: nil)

}

override func viewDidLoad() {

super.viewDidLoad()

//13 Code to check if record selected

if (videodb != nil) {

txtName.text = videodb.valueForKey("name") as? String

txtDate.text = videodb.valueForKey("datestamp") as? String

print(videodb.valueForKey("datestamp") as! String)

vidlink = videodb.valueForKey("link") as! String

self.btnSave.title = "Update"

btnSave.enabled = true

btnRecord.hidden=true

} else {

// Create a new device

let date = NSDate()

let formatter = NSDateFormatter()

formatter.timeStyle = .ShortStyle

formatter.dateStyle = .ShortStyle

formatter.stringFromDate(date)

print(formatter.stringFromDate(date))

txtDate.text = formatter.stringFromDate(date)

txtName.becomeFirstResponder()

btnPlay.hidden=true

btnSave.enabled = false

}

}

//14 Add func playerDidFinishPlaying

func playerDidFinishPlaying(note: NSNotification) {

print("Video Finished")

}

//15 Add Record Function

func RecordVideo()

{

if UIImagePickerController.isSourceTypeAvailable(UIImagePickerControllerSourceType.Camera) {

print("captureVideoPressed and camera available.")

let imagePicker = UIImagePickerController()

imagePicker.delegate = self

imagePicker.sourceType = .Camera;

imagePicker.mediaTypes = [kUTTypeMovie as String]

imagePicker.allowsEditing = false

imagePicker.showsCameraControls = true

self.presentViewController(imagePicker, animated: true, completion: nil)

}

else {

print("Camera not available.")

}

}

override func didReceiveMemoryWarning() {

super.didReceiveMemoryWarning()

// Dispose of any resources that can be recreated.

}

//16 Add func imagePickerController for when recording is finished

func imagePickerController(picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [String : AnyObject]) {

//Random #

let myVar: Int = Int(rand())

let tempImage = info[UIImagePickerControllerMediaURL] as! NSURL!

let paths = NSSearchPathForDirectoriesInDomains(.DocumentDirectory, .UserDomainMask, true)[0]

let name = txtName.text! + "\(myVar)" + ".MOV"

let filePathToWrite = "\(paths)/\(name)"

let MovieData:NSData = NSData(contentsOfURL: tempImage)!

MovieData.writeToFile(filePathToWrite, atomically: true)

let pathString = tempImage.relativePath

vidlink = filePathToWrite

print("Video Save Link: " + vidlink)

UISaveVideoAtPathToSavedPhotosAlbum(pathString!, self, nil, nil)

btnSave.enabled = true

self.dismissViewControllerAnimated(true, completion: {})

}

//17 Add Next 4 Functions to complete recording

func moviePlayerDidFinishPlaying(notification: NSNotification) {

self.dismissViewControllerAnimated(true, completion: nil)

}

func videoEditorControllerDidCancel(editor: UIVideoEditorController) {

print("User cancelled")

self.dismissViewControllerAnimated(true, completion: nil)

}

func videoEditorController(editor: UIVideoEditorController, didSaveEditedVideoToPath editedVideoPath: String) {

print("editedVideoPath: " + editedVideoPath)

self.dismissViewControllerAnimated(true, completion: nil)

}

func videoEditorController(editor: UIVideoEditorController, didFailWithError error: NSError) {

self.dismissViewControllerAnimated(true, completion: nil)

}

}