# Video Upload/Download Plug

## 1. Introduction

This is video upload/download plug will explore how to pick up or shoot new video using camera. Selected video upload on server using multipart with name and category. Also showing uploaded video progress and preview that video. We will also show show all uploaded video in list. We also generate thumbnail image of selected video.
This plug will explore download video feature like pause/resume with progress and proper icons. You can view and delete downloaded video. Added network capability with pull to refresh. This plug persist download if network fluctuate and app terminate.

## 2. Pre-requisites:

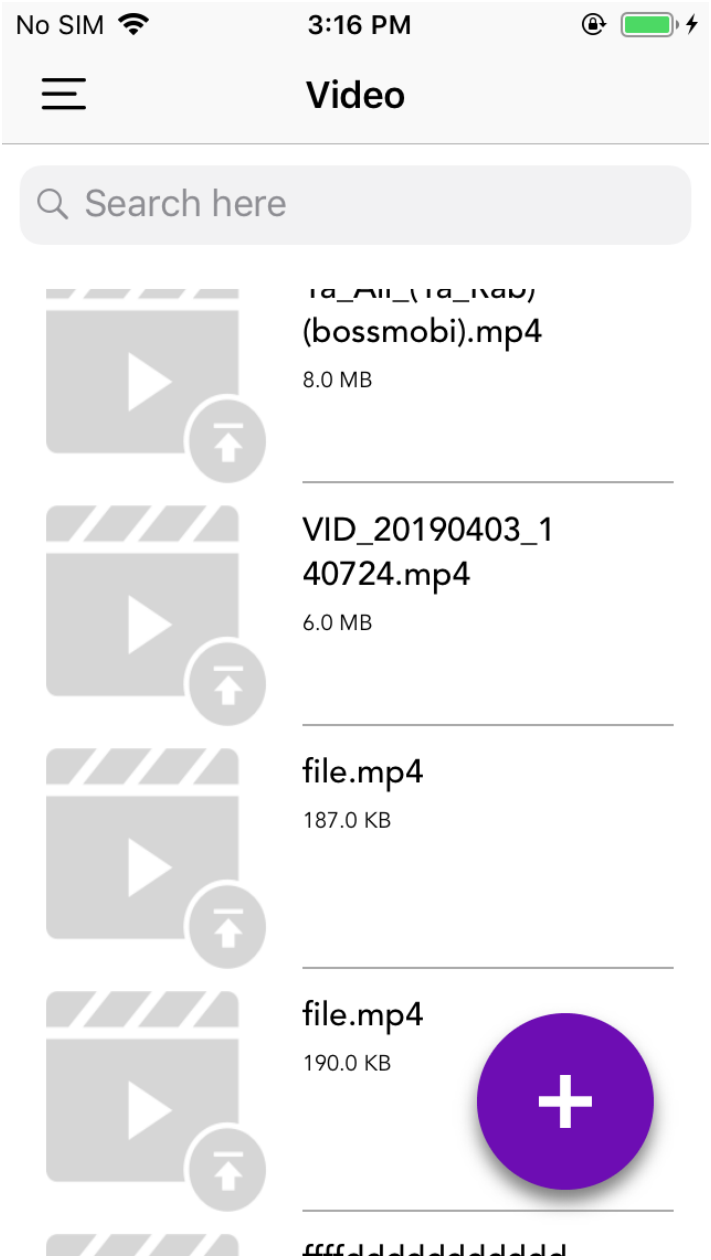
Language: Swift 4.0 and above
IDE: Xcode 9 and above

## 3. Features list

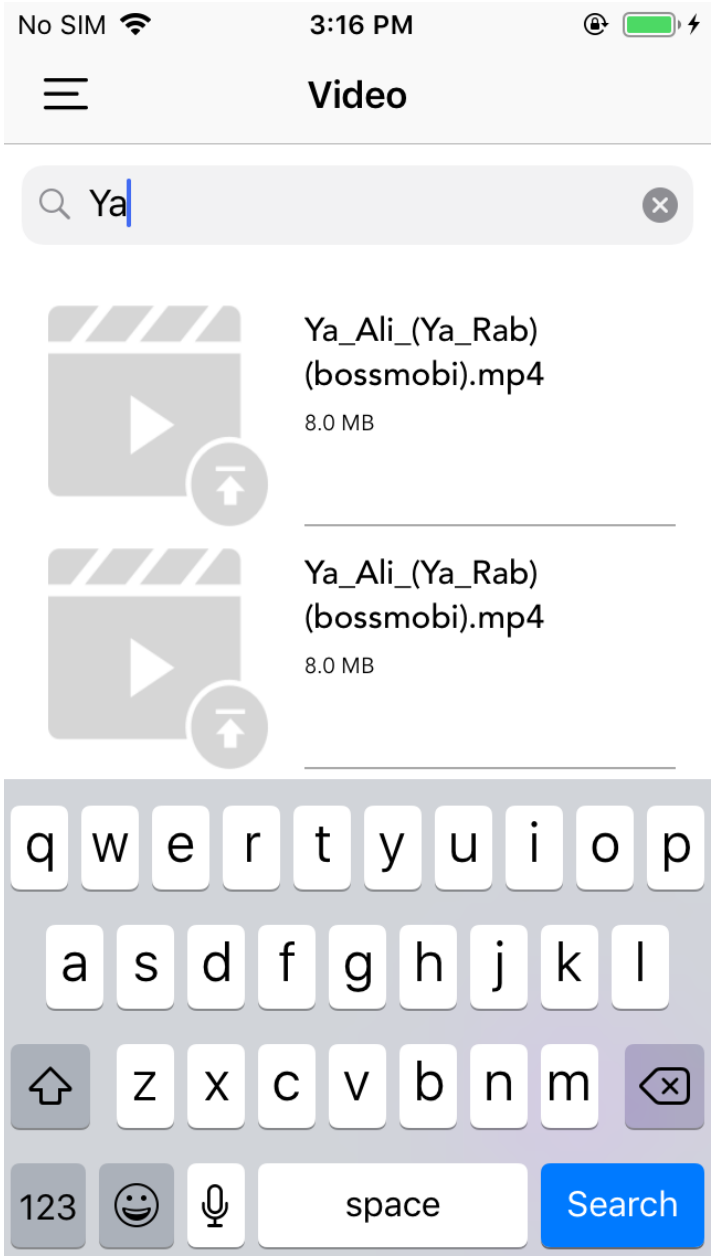
1. Pick up video from gallery
2. Shoot video using camera
3. Upload video with name & category
4. Uploading progress
5. Generate Thumbnail image
6. Showing uploaded video in list
7. Pull to refresh
8. Download video with progress
9. Download video with download percentages
10. Pause downloaded video
11. Resume at given progress
12. Delete downloaded video
13. Pause downloading if network goes off
14. Pause downloading if app kills
15. Persist downloading with progress and percentages

## 4. Screenshots

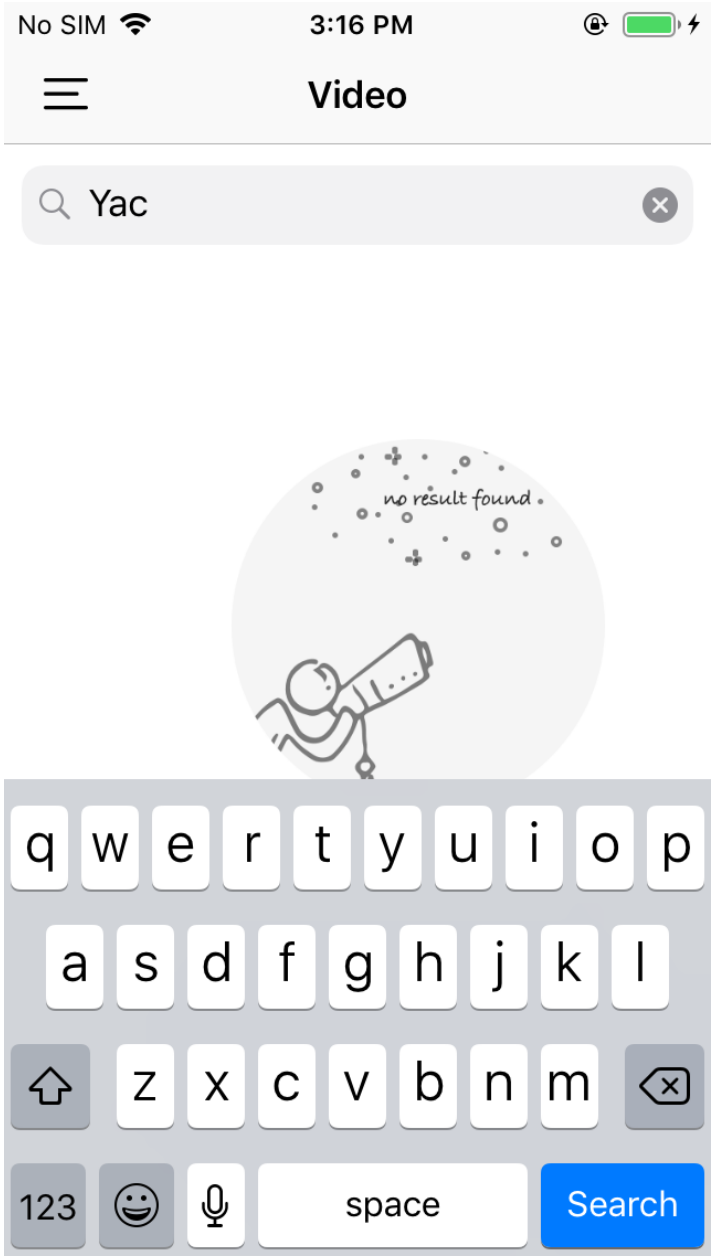
**Description:** Default list of uploaded video from server.



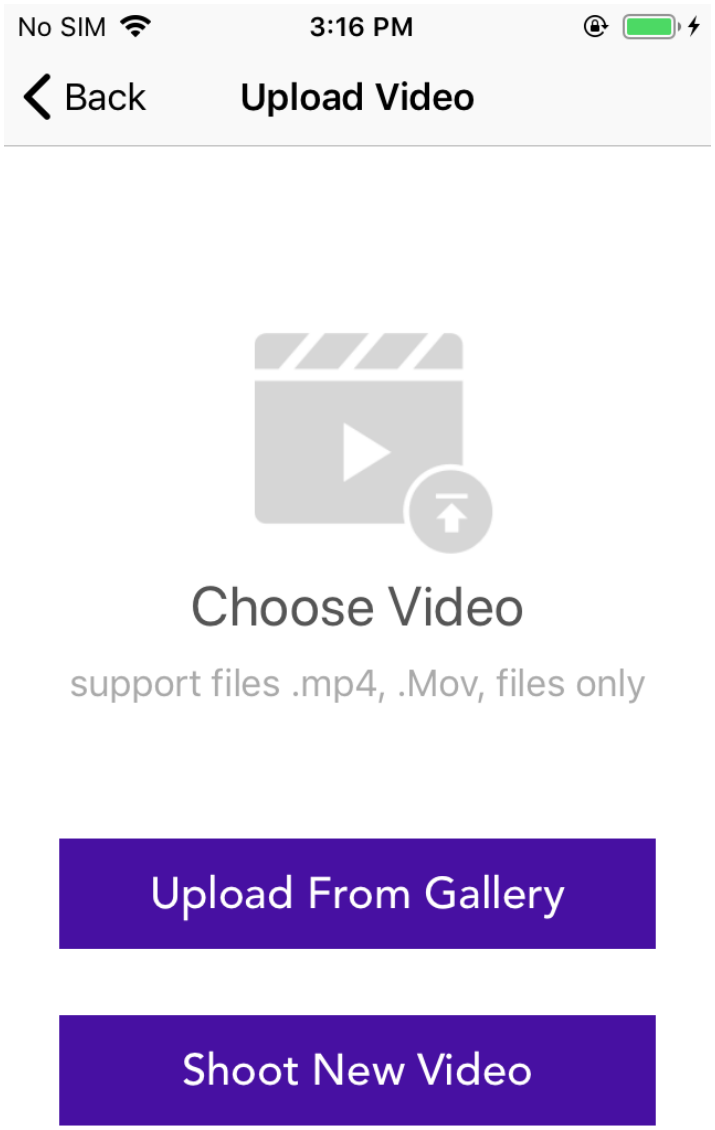
**Description:** Showing result of local search.



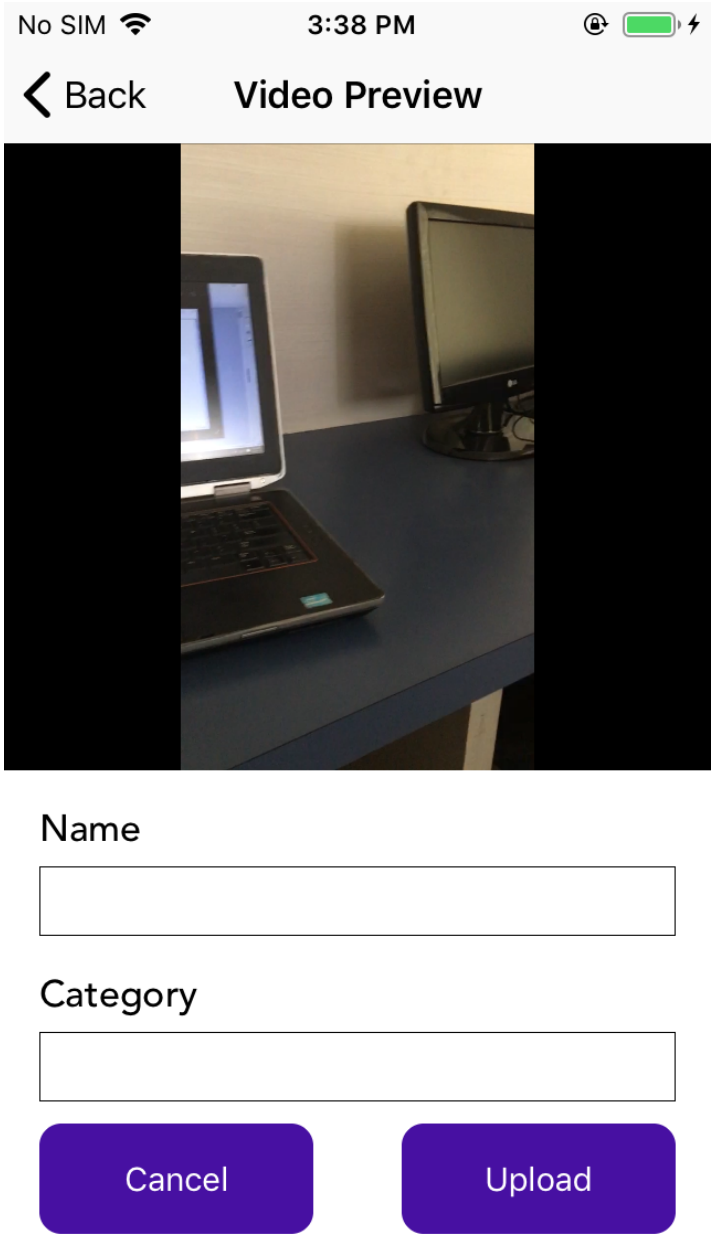
**Description:** Showing No search result found message view.



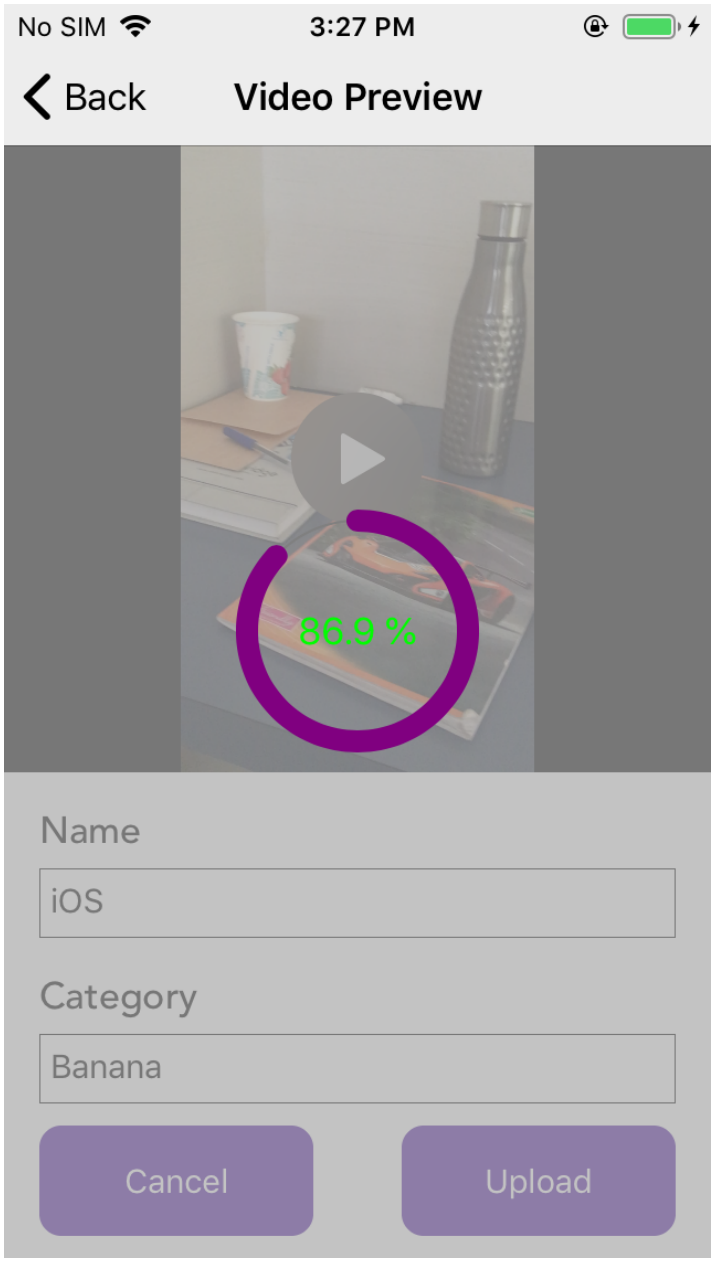
**Description:** Showing video picker options.



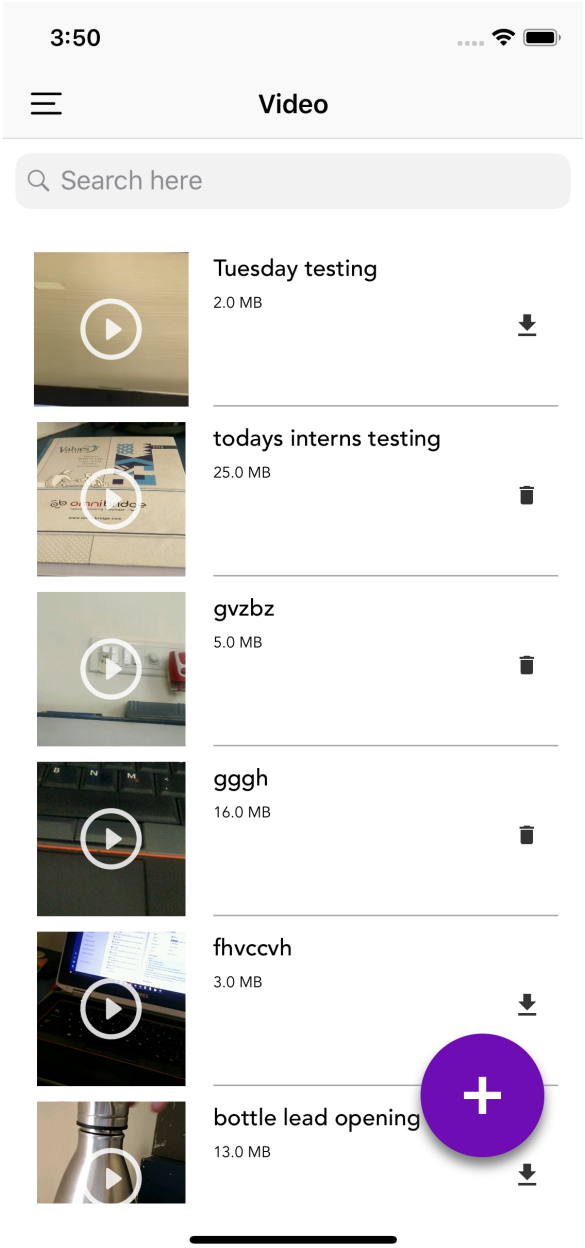
**Description:** Showing video preview page.



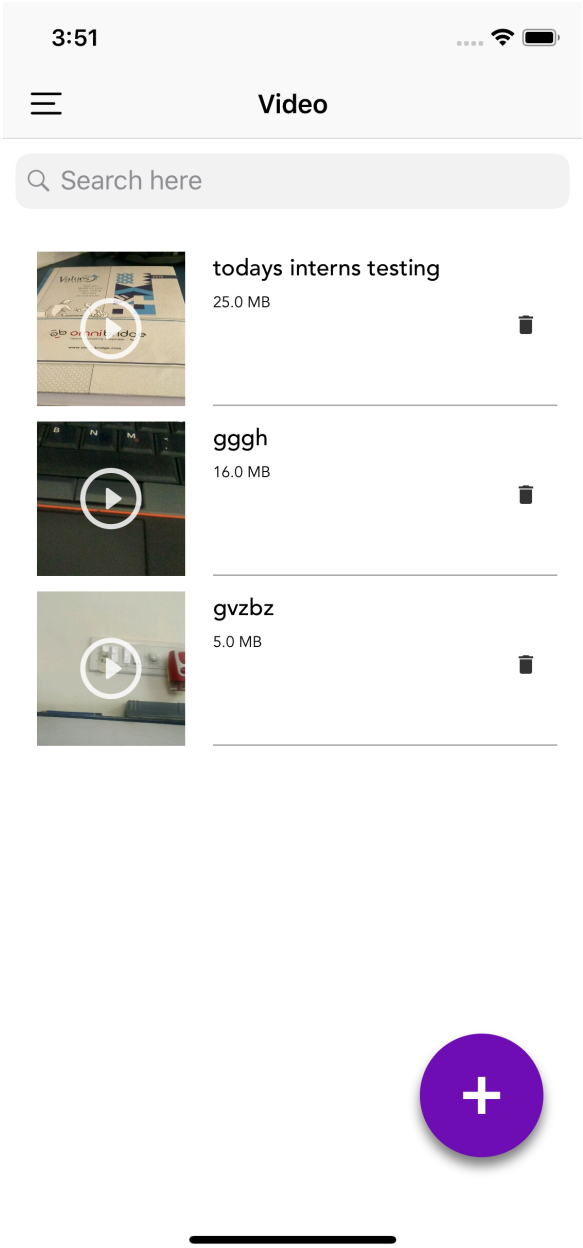
**Description:** Showing video  uploading progress.



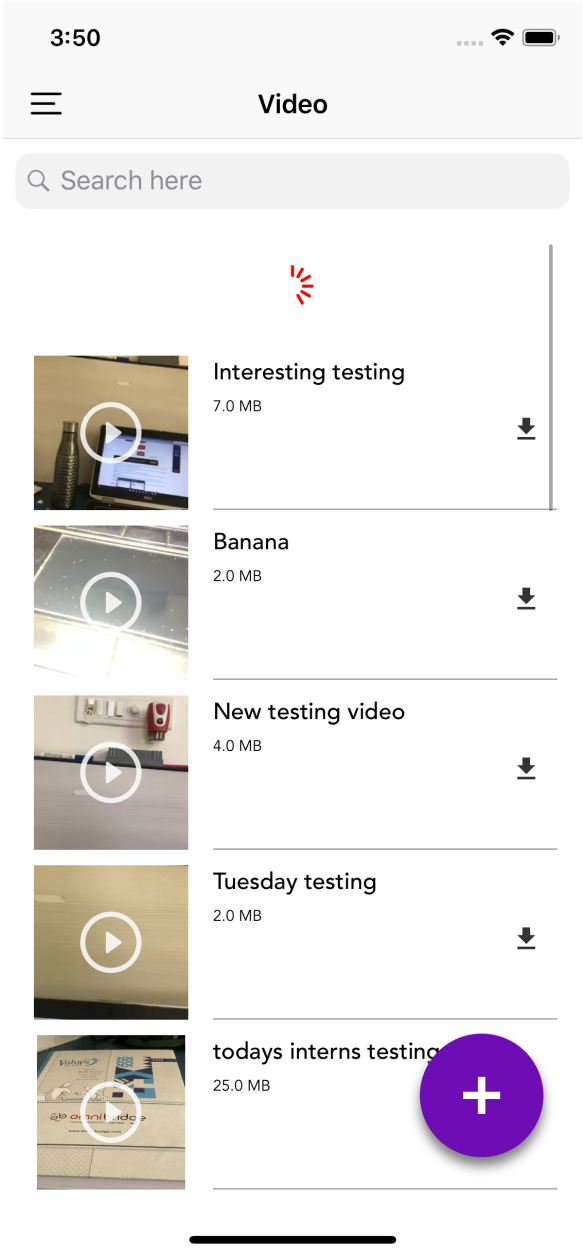
**Description:** Showing both downloaded and server records.



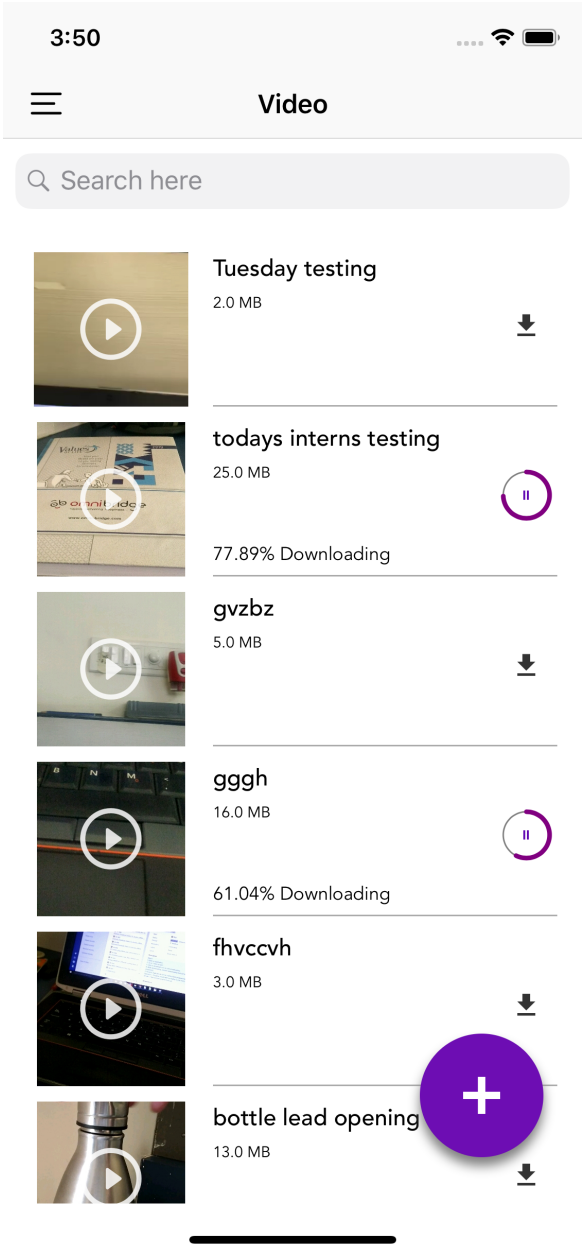
**Description:** Showing only downloaded records when no connection.



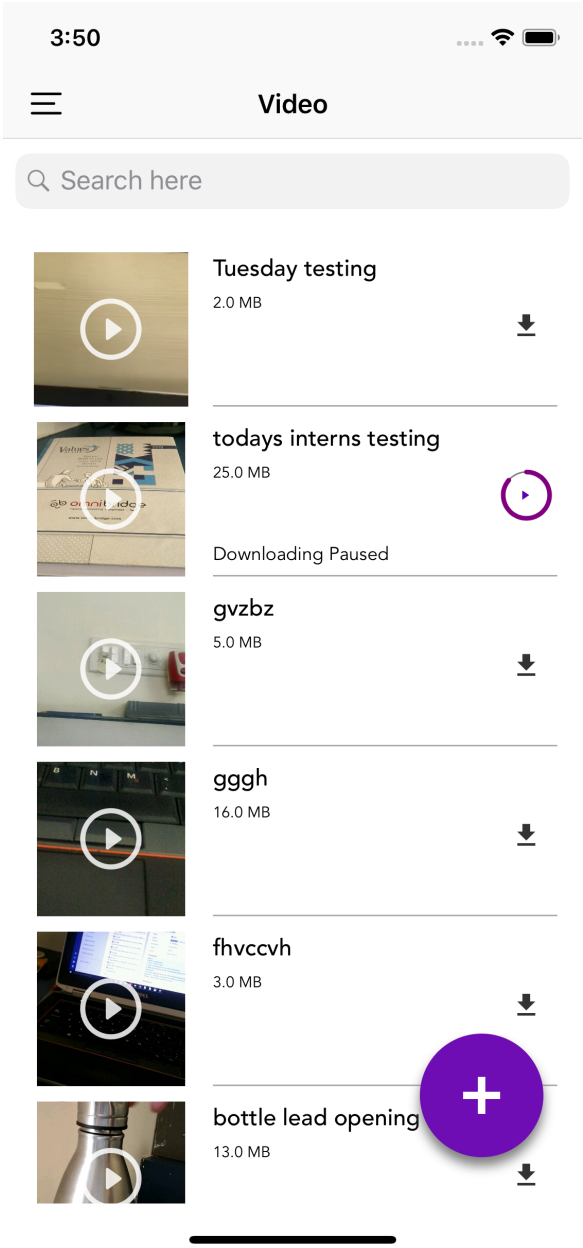
**Description:** Showing UI Pull to refresh.



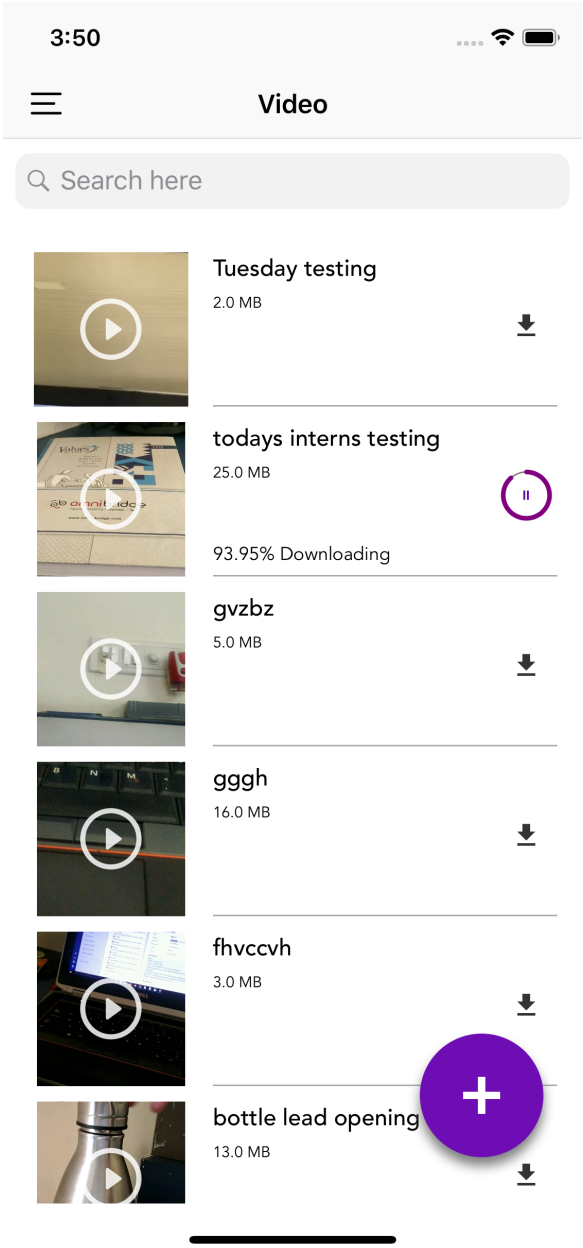
**Description:** Showing status and state of downloading video.



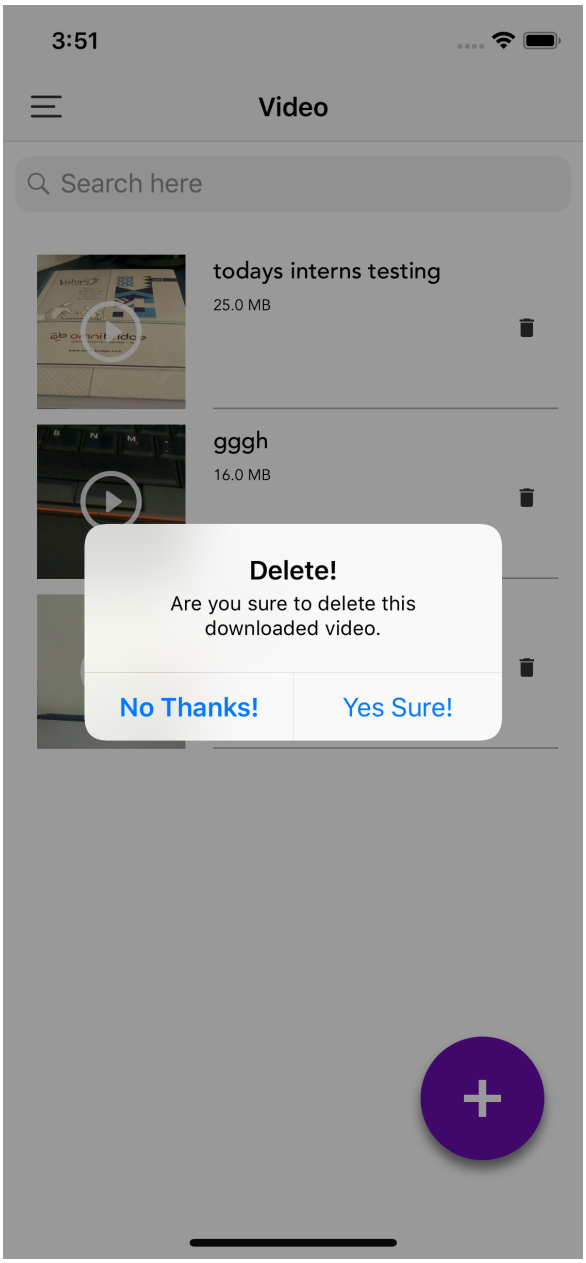
**Description:** Showing status and state of paused video.



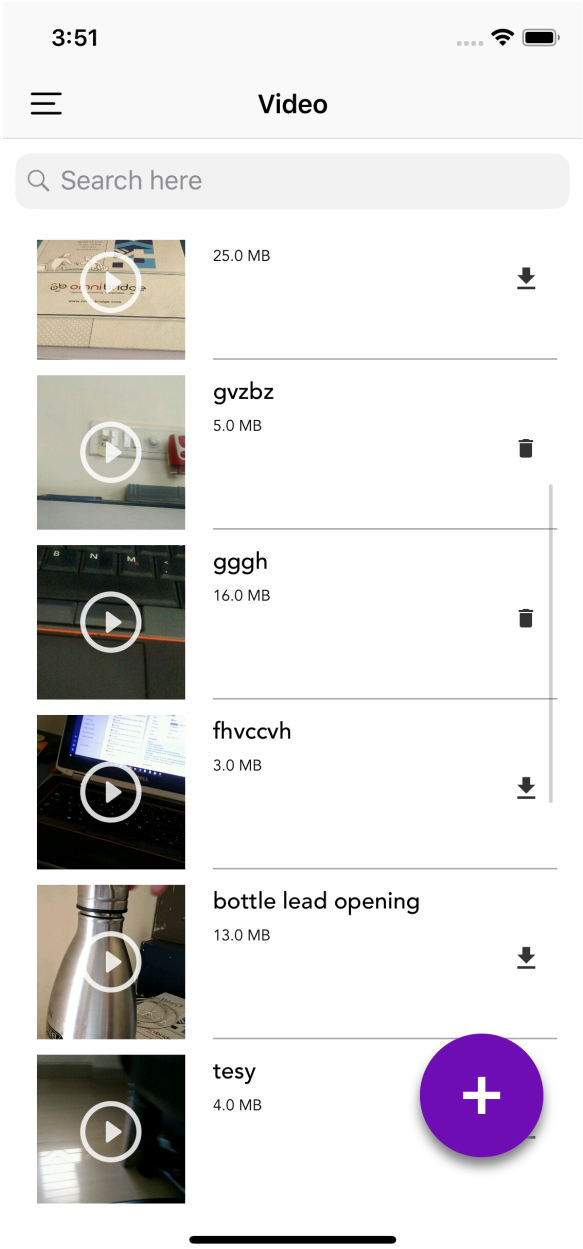
**Description:** Showing status and state of resume  pause video with persistence progress.



**Description:** Showing delete video alert.



**Description:** Showing delete again available for download.



## 5. Steps to integrate / use / installation

1. Function used get uploaded video list from server

```
    /// Used get list of uploaded video
    ///
    /// - Parameter comletion: comletion closure
    func getUploadedVideoDataFromServer(skip : Int,
comletion : @escaping (NSDictionary)->Void){
        Alamofire.request("\
(UserDefaults.standard.value(forKey:
"StartURLFromServer") ?? "")api/video/getvideolist?
skip=\(skip)&take=\
(PER_PAGE_TAKE_COUNT)").validate().responseJSON
{ (json) in
            if let jsonDict = json.result.value as?
NSDictionary {
                comletion(jsonDict)
            }else{
                comletion(NSDictionary())
            }
        }
    }
```

2. Function used to uploaded video with thumbnail, title and category to server

```
/// Used to upload video
    ///
    /// - Parameter fileName: fileName
    func uploadVideo(fileName : String , category :
String) {
        let url : URL! = URL(string: strURL)
        let thumbnailImage : UIImage! =
self.generateThumbnailFromVideoAt(path: url)
        let thumbnailImageData =
thumbnailImage.jpegData(compressionQuality: 1)
        Alamofire.upload(multipartFormData:
{ multipartFormData in
            multipartFormData.append(url, withName:
"file",fileName: "\(fileName).mp4", mimeType: "mov/
mp4")
```

```
multipartFormData.append(thumbnailImageData!, withName:
"thumbnail", fileName: "\(fileName.utf8).jpeg",
mimeType: "image/jpeg")
        },to:"\(UserDefaults.standard.value(forKey:
"StartURLFromServer") ?? "")api/video/upload?title=\
(fileName)&category=\
(category)".addingPercentEncoding(withAllowedCharacters
: CharacterSet.urlQueryAllowed)!)
        { (result) in
            switch result {
            case .success(let upload, _, _):
                upload.uploadProgress(closure:
{ (progress) in
                    //print("Upload Progress: \
(progress.fractionCompleted)")
                    self.progressView.progress =
Float(progress.fractionCompleted)
                })

                upload.responseJSON { response in
                    if response.result.value == nil{

self.progressView.removeFromSuperview()

self.view.isUserInteractionEnabled = true
                        Generals.presentAlert(title:
WARNING_TITLE, msg: SOMETHING_WENT_WRONG_MSG, VC: self)
                        return
                    }
                    //print(response.result.value!)
                    guard let resopnse =
response.result.value as? NSDictionary ,let
resopnseDict = resopnse.value(forKey: "data") as?
NSDictionary else{ return}
                    guard let message =
resopnseDict.value(forKey: "message") as? String else {
                        Generals.presentAlert(title:
WARNING_TITLE, msg: SOMETHING_WENT_WRONG_MSG, VC: self)
                        return
```

```
                        }

self.progressView.removeFromSuperview()
                    if let bgView =
self.view.viewWithTag(101){
                        bgView.removeFromSuperview()
                    }
                    self.view.isUserInteractionEnabled
= true
                    if message == "Video saved
succesfully."{

self.navigationController?.popToRootViewController(anim
ated: true)
                    }else{
                        Generals.presentAlert(title:
WARNING_TITLE, msg: SOMETHING_WENT_WRONG_MSG, VC: self)
                    }
                }
            case .failure(let encodingError):
                print(encodingError)
            }
        }
    }
}
```

3. Function used to generate thumbnail image from selected/capture video url

```
    /// Used to generate thumbnail image
    ///
    /// - Parameter path: video path url
    /// - Returns: thumbnail image
    func generateThumbnailFromVideoAt(path: URL) ->
UIImage? {
        do {
            let asset = AVURLAsset(url: path, options:
nil)
            let imgGenerator =
AVAssetImageGenerator(asset: asset)
```

```
            imgGenerator.appliesPreferredTrackTransform
= true
            let cgImage = try
imgGenerator.copyCGImage(at: CMTimeMake(value: 0,
timescale: 1), actualTime: nil)
            let thumbnail = UIImage(cgImage: cgImage)
            return thumbnail
        } catch let error {
            print("*** Error generating thumbnail: \
(error.localizedDescription)")
            return nil
        }
    }
```

4. Function used to start downloading video

```
    /**
     This method is used to download the video.
     - Parameters:
     - videoModel: VideoDetails

     ### Usage Example: ###
     ````
     DownloadManager.shared.beginDownloadVideo(with
videoModel : VideoDetails) {
     }
     ````
     */
    func beginDownloadVideo(with videoModel :
VideoDetails, index : Int) {

        let destination =
self.getDistinationPath(videoName:
videoModel.strVideoName)
        var downloadProgress = 0.0
        let videoInfo:[String: Int] = ["index": index]
        DownloadManager.request =
Alamofire.download( URL(string:
videoModel.strVideoUrl)!, method: .get, parameters:
```

```
nil, encoding: JSONEncoding.default, headers: nil, to:
destination).downloadProgress(closure: { (progress) in
            downloadProgress =
progress.fractionCompleted
            CoreDataOperations.updateDownloadStatus(id:
videoModel.id, progress: downloadProgress, status:
"start")
            //if (Int(downloadProgress * 100) % 2) ==
0{
                //NotificationCenter.default.post(name:
NSNotification.Name(DOWNLOAD_PAUSE_NOTIF_NAME), object:
nil)
            NotificationCenter.default.post(name:
NSNotification.Name(DOWNLOAD_PAUSE_NOTIF_NAME), object:
nil , userInfo : videoInfo)
            //}
        }).response(completionHandler:
{ (defaultDownloadResponse) in
            if defaultDownloadResponse.error == nil{

CoreDataOperations.updateDownloadStatus(id:
videoModel.id, progress: 1, status: "finish")
            }else if let downloadedData =
defaultDownloadResponse.resumeData{
                self.userDefault.set(downloadedData,
forKey: videoModel.strVideoUrl)

CoreDataOperations.updateDownloadStatus(id:
videoModel.id, progress: downloadProgress, status:
"pause")
            }
            //NotificationCenter.default.post(name:
NSNotification.Name(DOWNLOAD_PAUSE_NOTIF_NAME), object:
nil)
            NotificationCenter.default.post(name:
NSNotification.Name(DOWNLOAD_PAUSE_NOTIF_NAME), object:
nil , userInfo : videoInfo)
        })
    }
```

5. Function used to restart downloading video

```
/**
     This method is used to resume downloading video.
     - Parameters:
     - videoModel: VideoDetails

     ### Usage Example: ###
     ````
     DownloadManager.shared.resumeDownloadingVideo(with
videoModel : VideoDetails) {
     }
     ````
     */
    func resumeDownloadingVideo(with videoModel :
VideoDetails , index : Int) {

        guard  let data =
self.userDefault.value(forKey: videoModel.strVideoUrl)
as? Data else {
            return
        }
        var downloadProgress = 0.0
        let destination =
self.getDistinationPath(videoName:
videoModel.strVideoName)
        let videoInfo:[String: Int] = ["index": index]
        DownloadManager.request =
Alamofire.download(resumingWith: data, to:
destination).downloadProgress(closure: { (progress) in
            downloadProgress =
progress.fractionCompleted
            CoreDataOperations.updateDownloadStatus(id:
videoModel.id, progress: downloadProgress, status:
"start")
            //NotificationCenter.default.post(name:
NSNotification.Name(DOWNLOAD_PAUSE_NOTIF_NAME), object:
nil)
```

```
            NotificationCenter.default.post(name:
NSNotification.Name(DOWNLOAD_PAUSE_NOTIF_NAME), object:
nil , userInfo : videoInfo)
        }).response(completionHandler:
{ (defaultDownloadResponse) in
            if defaultDownloadResponse.error == nil{
                self.userDefault.removeObject(forKey:
videoModel.strVideoUrl)

CoreDataOperations.updateDownloadStatus(id:
videoModel.id, progress: 1, status: "finish")
            }else if let downloadedData =
defaultDownloadResponse.resumeData{
                self.userDefault.set(downloadedData,
forKey: videoModel.strVideoUrl)

CoreDataOperations.updateDownloadStatus(id:
videoModel.id, progress: downloadProgress, status:
"pause")
            }
            //NotificationCenter.default.post(name:
NSNotification.Name(DOWNLOAD_PAUSE_NOTIF_NAME), object:
nil)
            NotificationCenter.default.post(name:
NSNotification.Name(DOWNLOAD_PAUSE_NOTIF_NAME), object:
nil , userInfo : videoInfo)
        })
    }
```

6. Function used to update UI and capture notification

```
/**
     used to capture download notification
     - Parameter :
     - notification : Notification
     */
    @objc func
captureDownloadingNotification(notification :
Notification){
```

```
        if let cellIndex = notification.userInfo?
["index"] as? Int {
            if cellIndex >= uploadedVideoArray.count{
              return
            }
            if let offlineModel =
CoreDataOperations.fetchOfflineVideoData(id:
self.uploadedVideoArray[cellIndex].id){
                self.uploadedVideoArray[cellIndex] =
offlineModel
                for indexPath in
(tableView?.indexPathsForVisibleRows)!{
                    if indexPath.row == cellIndex &&
searchBar.text == ""{
                        let cell =
tableView.cellForRow(at: IndexPath(row: cellIndex,
section: 0)) as! TableViewCellForVideoDetails

self.progressButtonArray[offlineModel.id] =
cell.btnForVideoProgress

self.progressLabelArray[offlineModel.id] =
cell.lblForVideoProgress
                        DispatchQueue.main.async {
                            if offlineModel.status ==
"start"{

self.progressButtonArray[offlineModel.id]?.drawCircle()

self.progressButtonArray[offlineModel.id]?.status
= .downloading

self.progressButtonArray[offlineModel.id]?.progress =
offlineModel.progress

self.progressLabelArray[offlineModel.id]?.isHidden =
false

self.progressLabelArray[offlineModel.id]?.text = "\
```

```
(String(format: "%.2f", (offlineModel.progress * 100)))
% Downloading"
                                }else if
offlineModel.status == "finish"{

self.progressButtonArray[offlineModel.id]?.status
= .downloaded

self.progressLabelArray[offlineModel.id]?.isHidden =
true
                                }else if
offlineModel.status == "pause"{

self.progressLabelArray[offlineModel.id]?.isHidden =
false

self.progressButtonArray[offlineModel.id]?.status
= .paused

self.progressButtonArray[offlineModel.id]?.progress =
offlineModel.progress

self.progressLabelArray[offlineModel.id]?.text =
"Downloading Paused"
                                    }
                                }
                            }
                        }
                    }
                }
            }
        }
```

## 6. Pods/Supporting files

```
  pod 'IQKeyboardManagerSwift'
  pod 'Alamofire'
  pod 'AlamofireImage'
  pod 'ReachabilitySwift'
```

## 7. Contact us / Support

Support@plug-able.com

Reference link:

https://developer.apple.com/documentation/uikit/uiactivityindicatorview?changes=_5

https://stackoverflow.com/questions/3439853/replace-occurrences-of-space-in-url

_https://stackoverflow.com/questions/31779150/creating-thumbnail-from-local-video-in-swift

_https://github.com/Alamofire/Alamofire

https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/CoreData/