PetsGallery App-> A SwiftUI video-based application with AVPlayer and the Pexels API that is full of cute pets videos. The App contains different components and views, with JSON data, JSON data conversion into a custom SwiftUI model, API is called asynchronously, and also uses AVPlayer (from AVKit) to integrate the media player.

Developing Languages, Tools, and Techniques Needed:

Xcode Version 14.1 https://developer.apple.com/xcode/
Swift 5.7 https://docs.swift.org/swift-book/
SwiftUI https://developer.apple.com/xcode/swiftui/
Pexels API https://www.pexels.com/api
JSON API https://designcode.io/swiftui-advanced-handbook-data-from-json
AVKit https://developer.apple.com/documentation/avkit

Synchronous Developing Notes:

Create Query Tag:

Code query tag in `QueryTag.swift`:

```
    var isSelected: Bool
    var body: some View {
        Text(query)
            .font(.caption)
            .bold()
            .foregroundColor(isSelected ? .black : .gray)
            .padding(10)
            .background(.thinMaterial)
            .cornerRadius(10)
```

Make categories visible in `ContentView.swift`:

```
struct ContentView: View {
    var body: some View {
        VStack {
            HStack {
                ForEach(Query.allCases, id: \.self){
                    searchQuery in
                    QueryTag(query: searchQuery, isSelected: false)
                }
```

all categories showed.PNG

Customize Video cards:

Create a new SwiftUI file `VideoCard.swift`:

```
      ZStack(alignment: .bottomLeading){
            AsyncImage(url: URL(string: "")) { image in
                image.resizable()
                    .aspectRatio(contentMode: .fill)
                    .frame(width: 160, height: 250)
            } placeholder: {
                Rectangle()
                    .foregroundColor(.gray.opacity(0.3))
                    .frame(width: 160, height: 250)
```
video card initial frame.PNG

Import video play button:
```
  Image(systemName: "play.fill")
                .foregroundColor(.white)
                .font(.title)
                .padding()
                .background(.ultraThinMaterial)
                .cornerRadius(50)
```
video play button.PNG

<u>The Pexels API and ResponseBody model:</u>

Based on pexel page, attach the needed attributes into `VideoManager.swift`:
```
struct ResponseBody: Decodable {
    var page: Int
    var perPage: Int
    var totalResults: Int
    var url: String
    var videos: [Video]
```
<u>Add JSON data:</u>

Import `videoData.json` as preview content, and in `VideoCard.swift`:
```
AsyncImage(url: URL(string: video.image))
```
video cover image displayed.PNG

In `VideoView.swift,` add play button:
```
    var video: Video
    @State private var player = AVPlayer(
    var body: some View {
        VideoPlayer(player: player)
    }
  }
struct VideoView_Previews: PreviewProvider {
    static var previews: some View {
        VideoView(video: previewVideo)
    }
```
video view play button.PNG

Add constraints and video link for it to play:
```
 .edgesIgnoringSafeArea(.all)
            .onAppear{
                if let link = video.videoFiles.first?.link {
                    player = AVPlayer(url: URL(string: link)!)
                    player.play()
```
video preview played.PNG

<u>Generate API key for Pexels API:</u>

Go to https://www.pexels.com/api/new and obtain a private API.
In `ContentView.swift,` fetch videos in various categories with the imported API:

```
 NavigationView{
            VStack {
                HStack {
                    ForEach(Query.allCases, id: \.self){
                        searchQuery in
                        QueryTag(query: searchQuery, isSelected: 
false)
                    }
                }
                ScrollView {
                    ForEach(videoManager.videos, id: \.id) {
                        video in
                        NavigationLink {
                            VideoView(video: video)
                        } label: {
                            VideoCard(video: video)
                        }
                    }
                }
                .frame(maxWidth: .infinity)
            }
            .background(Color("AccentColor"))
```

videos in categories fetched.PNG

Use `LazyVGrid(columns: columns, spacing: 20)` method to make videos align better:

videos aligned better.PNG

Fetch all categories in dispatch queue:

```
  DispatchQueue.main.async {
          // Reset the videos (for when we're calling the API again)
                self.videos = []
           // Assigning the videos we fetched from the API
                self.videos = decodedData.videos
            }
```