6COSC021W
W1633664
EvanBabaker

Compositional View of Maps and Own Implementation

Code implementation

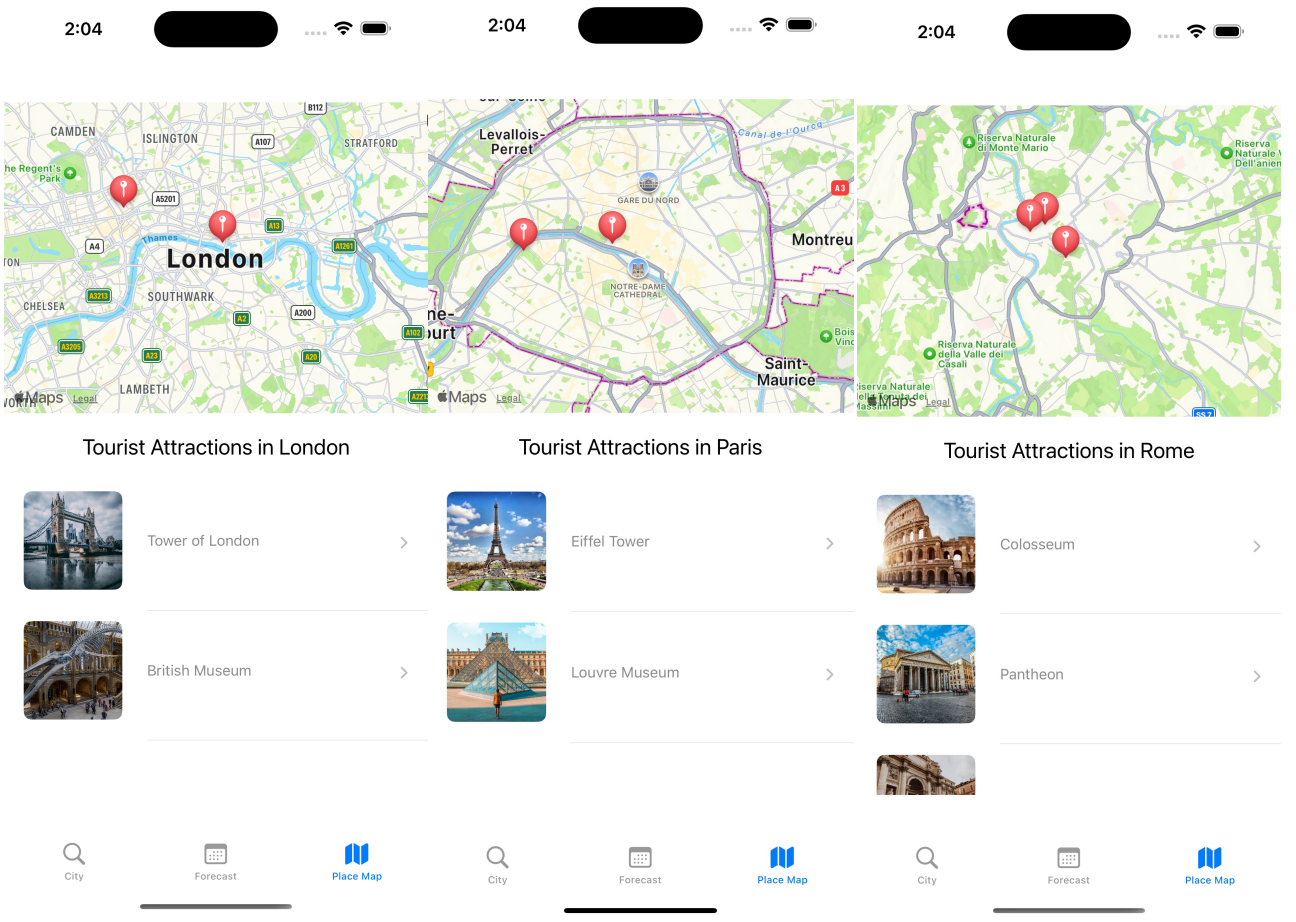
```
List{
                    if !locations.isEmpty{
                        ForEach(locations) { location in
                            NavigationLink(destination: LocationDetailView(location: location)) {
                                HStack(spacing: 25) {
                                    VStack(alignment: .leading) {
                                        Image(location.imageNames[0])
                                            .resizable()
                                            .aspectRatio(contentMode: .fill)
                                            .frame(width: 100, height: 100)
                                            .cornerRadius(8)
                                    }
                                            VStack {
                                                Text(location.name)
                                                    .font(.subheadline)
                                                    .foregroundColor(.secondary)
                                            }
                                        }
                                        .padding(.bottom, 5)
                                    }
                                    .padding(.bottom, 5)
                                }
                            } else{
                                Text("There are no attractions for
\(weatherMapViewModel.city)")
                                }
                            }.ignoresSafeArea()
                                .listStyle(.plain)
```

This was the code I wrapped around a `NavigationView {` inside my "touristplacesMapViewmodel", the destination of the `NavigationLink` navigates to the LocationDetailView, which provides the user with further information about the tourist locations.

The navigation Link wraps the location's image names and the location name. each location in the "Places.json" provided will show in the list with the location image and the name of the location side by side.

The images below provide a compositional view of my code description and maps with map annotation linking to the locations of tourist attractions.

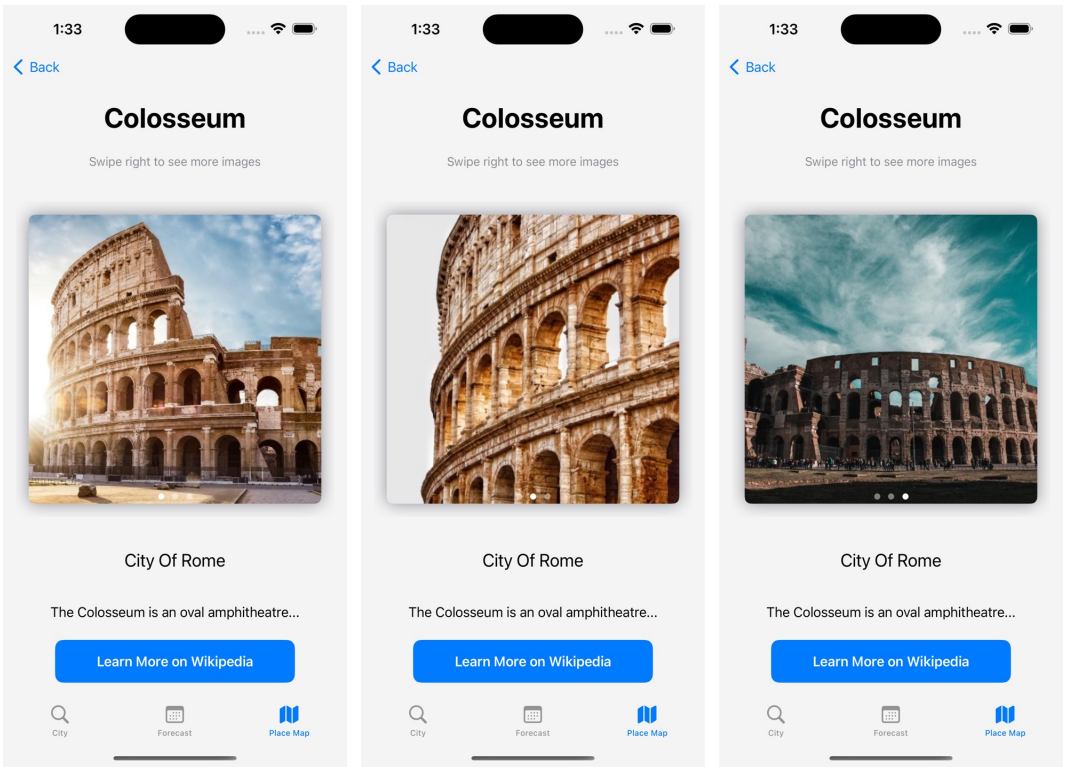
In the iterative development of the list view within the TouristPlacesMapView, several enhancements were strategically incorporated to optimise the display of tourist locations, providing users with a visually appealing and information-rich experience. The conditional check ensures a seamless transition between two states, dynamically adapting the content based on the presence or absence of available tourist attractions. A NavigationLink has been employed for each location to seamlessly navigate users to a dedicated LocationDetailView, offering a deeper insight into the selected attraction.

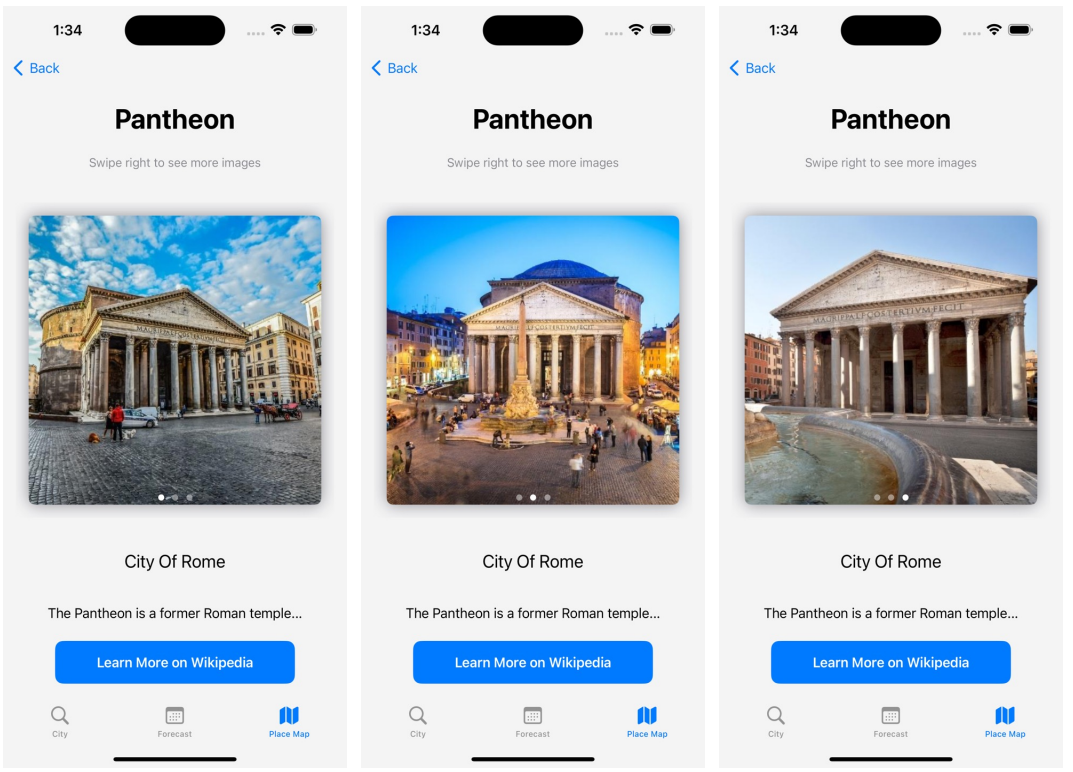


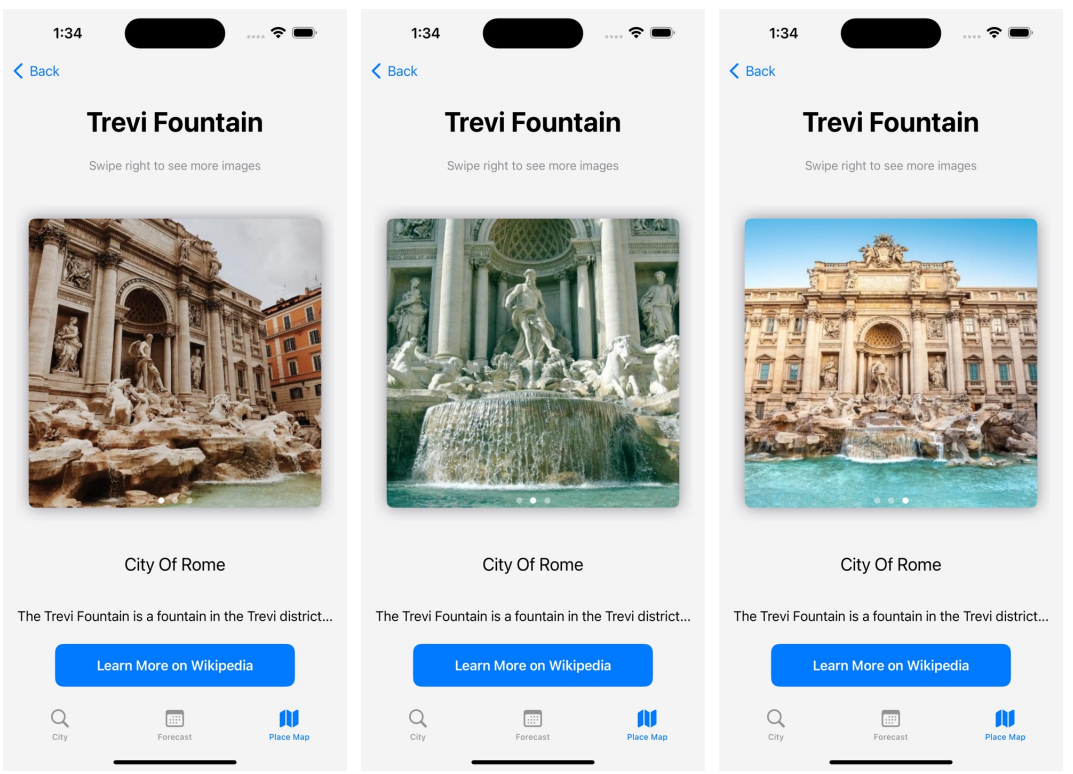
The list view is designed to be responsive, user-friendly, and aesthetically coherent. These refinements create an immersive experience, allowing users to seamlessly explore and engage with the application's diverse tourist attractions.

The layout of each item within the list has been meticulously crafted, featuring a horizontal stack (HStack) with a well-spaced vertical stack (VStack). The first vertical stack encompasses an image showcasing the location's allure, dynamically sourced from the imageNames array. The image is thoughtfully resized and adorned with rounded corners, fostering a visually pleasing and consistent presentation across different destinations.
The second vertical stack encapsulates the location's name, presented in a subheadline font with a secondary colour, contributing to a cohesive and harmonious aesthetic. The strategic use of padding ensures an organised and visually balanced layout, enhancing readability and user engagement. The absence of attractions is gracefully communicated through a text element, maintaining transparency and guiding users through the application's functionality.

Rome







In my mobile application, I was advised to add my implementation to improve the usability and functionality of the application.

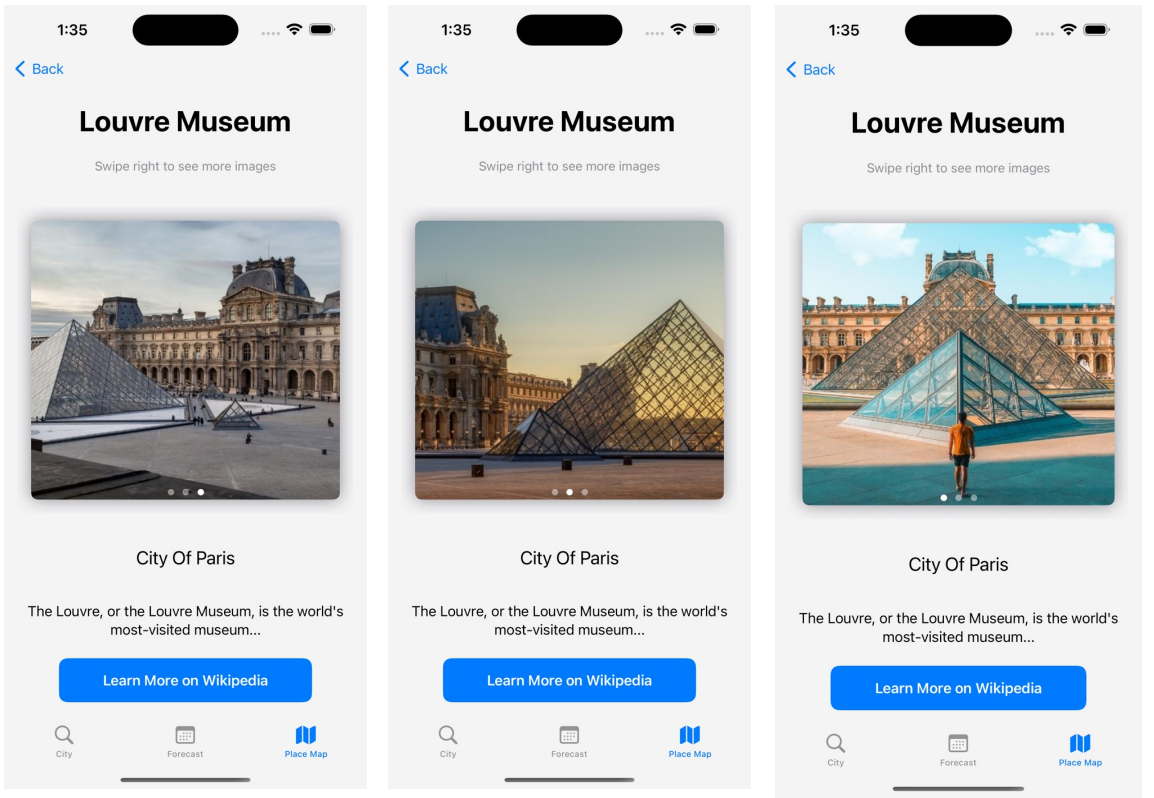
I noticed from the JSON file Provided named “Places.json” that there was additional information that I could project and display into a new existing view.

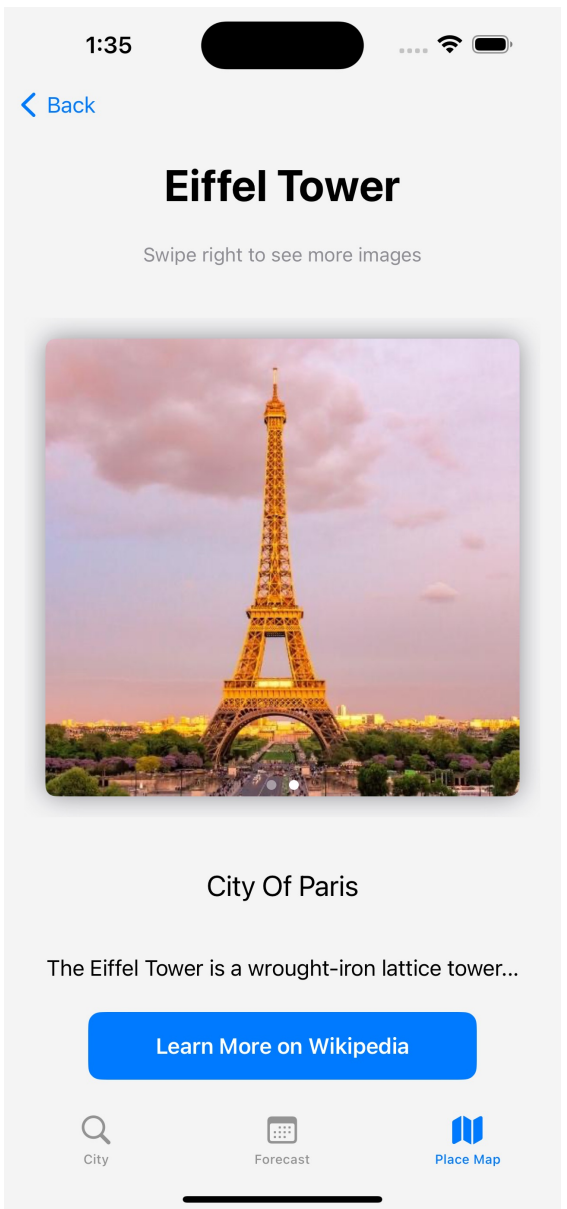

```
    {
        "name": "Colosseum",
        "cityName": "Rome",
        "latitude": 41.8902,
        "longitude": 12.4922,
        "description": "The Colosseum
is an oval amphitheatre...",
        "imageNames": [
            "rome-colosseum-1",
            "rome-colosseum-2",
            "rome-colosseum-3"
        ],
        "link":
"https://en.wikipedia.org/wiki/Coloss
eum"
    },
```

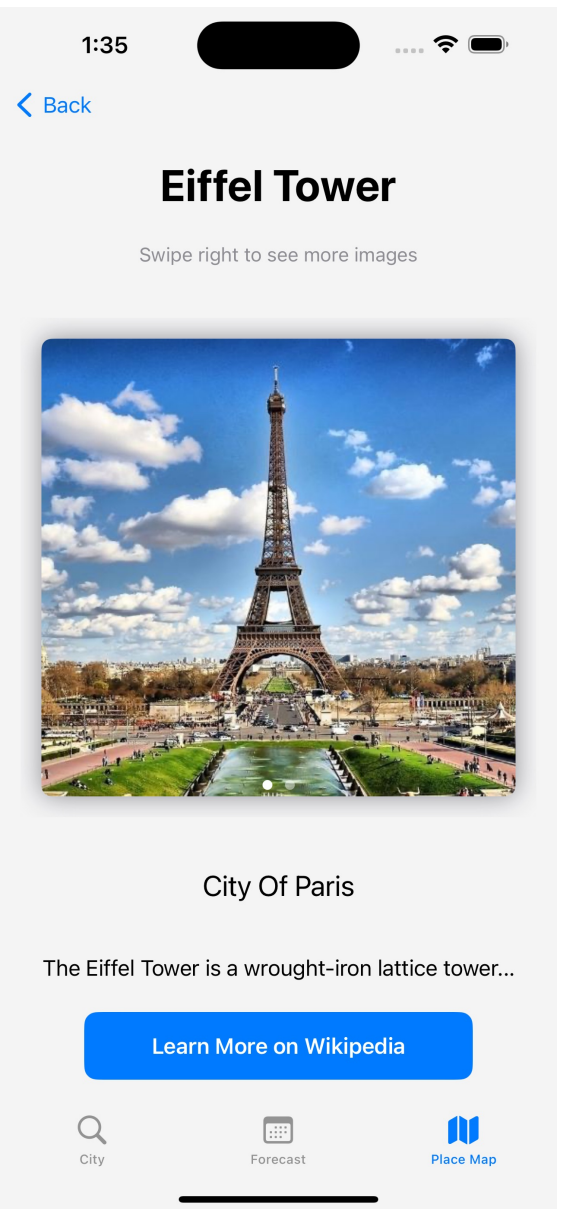

Given that the “imageNames” was an array of images. I produced a Tabview that showed the images in a slideshow format. Giving the user access to these images by swiping right provided them with more information about the tourist attractions In the city.

The page is accessed through a navigation view from the “ToursitPlacesMapView”. I found that using the navigation link allowed the user to navigate from the tourist page to the location page with access to a back button.

Paris







In my mobile application, I was advised to add my implementation to improve the usability and functionality of the application.

I noticed from the JSON file Provided named “Places.json” that there was additional information that I could project and display into a new existing view.


```
    {
        "name": "Eiffel Tower",
        "cityName": "Paris",
        "latitude": 48.8584,
        "longitude": 2.2945,
        "description": "The
Eiffel Tower is a wrought-iron
lattice tower...",
        "imageNames": [
            "paris-eiffeltower-
1",
            "paris-eiffeltower-2"
        ],
        "link":
"https://en.wikipedia.org/wiki/Ei
ffel_Tower"
    },
```


Given that the “imageNames” was an array of images. I produced a Tabview that showed the images in a slideshow format. Giving the user access to these images by swiping right provided them with more information about the tourist attractions In the city.

The page is accessed through a navigation view from the “ToursitPlacesMapView”. I found that using the navigation link allowed the user to navigate from the tourist page to the location page with access to a back button.

LocationDetailView

```swift
import SwiftUI


struct LocationDetailView: View {
    var location: Location


    var body: some View {
        VStack {
            Text(location.name)
                .font(.largeTitle)
                .fontWeight(.bold)
                .padding(.top, 20)
            Spacer()
            Text("Swipe right to see more images")
                            .font(.subheadline)
                            .foregroundColor(.gray)
                            .padding()
            TabView {
                ForEach(location.imageNames, id: \.self) { imageName in
                    Image(imageName)
                        .resizable()
                                    .frame(maxWidth: 600, maxHeight: 600)
                                    .cornerRadius(8)
                                    .overlay(RoundedRectangle(cornerRadius: 20)
                                                .stroke(Color.clear, lineWidth: 10))
                                    .padding()
                                    .shadow(color: .gray, radius: 10)
                                    .tag(imageName)

            }
                    .tabViewStyle(PageTabViewStyle(indexDisplayMode: .automatic))
                    .padding()


                Text("City Of \(location.cityName)")
                    .font(.title2)
                    .padding()


                Text("\(location.description)")
                    .font(.body)
                    .multilineTextAlignment(.center)
                    .padding()


                Link(destination: URL(string: location.link)!) {
                    Text("Learn More on Wikipedia")
                        .foregroundColor(.white)
                        .font(.headline)
                        .padding()
                        .frame(maxWidth: 300)
                        .background(Color.blue)
                        .cornerRadius(10)
                }
                .padding(.bottom, 20)
            }
            .background(Color.gray.opacity(0.1))

        }
}
```

The LocationDetailView SwiftUI code has undergone significant improvements to enhance the application's visual appeal and user interaction. The name of the location is now presented in a large and bold font, giving it more prominence. Additionally, a subtle prompt has been added to guide users through the image slideshow, which can be accessed by swiping right.

A TabView has been integrated to provide a seamless and visually pleasing image browsing experience. Each image is displayed optimally with rounded corners and a subtle shadow effect for added depth. The integration of a dynamic link to Wikipedia allows users to access further details about the tourist attraction.

To create an organised and aesthetically cohesive layout, stylistic enhancements such as consistent padding and background colour with a transparent touch have been added. Moreover, the text sizes and alignment are dynamically adjusted to ensure a harmonious presentation of information, regardless of the content length.

The meticulous attention to detail has resulted in an enriched LocationDetailView, showcasing the beauty of tourist locations and providing a seamless and engaging user experience.