Setting Up UPI Intent Launch and Opening UPI Apps from iOS

medium.com/@arindamxd/setting-up-upi-intent-launch-and-opening-upi-apps-from-ios-6229dced52e6

Arindam Karmakar 30 August 2024

Arindam Karmakar

Photo by on

Understanding UPI

UPI (Unified Payments Interface) is a real-time payment system in India that allows users to transfer money instantly using their smartphones. To facilitate UPI payments, many apps have integrated UPI functionality.

Setting Up UPI Intent Launch in Your iOS App

- Open your project's Info.plist file and add a new key-value pair.
- Set the key to "URL types" and create a new dictionary.
- Inside the dictionary, add a new dictionary and set the key to "URL schemes."
- Add your app's unique URL scheme as the value. For example, if your app's bundle identifier is "com.yourcompany.yourApp," you could use "upi" as the URL scheme.

- In your app delegate, implement the `application(_:open:sourceApplication:annotation:)` method.
- Check if the URL scheme matches your app's UPI scheme.
- If it does, parse the URL to extract the necessary UPI payment details.
- Use these details to initiate the payment process within your app.

```
funcapplication(_application: UIApplication, openurl: URL, sourceApplication:
String?, annotation: Any?) -> Bool {
if url.scheme =="upi" {
// Parse the URL to extract UPI payment details
let components =URLComponents(url: url, resolvingAgainstBaseURL: false)!
let queryItems = components.queryItems!


// Extract payment details (e.g., payee name, amount, transaction ID)
let payeeName = queryItems.first(where: { $0.name =="payeename" })?.value
let amount = queryItems.first(where: { $0.name =="amount" })?.value
let transactionId = queryItems.first(where: { $0.name =="transactionid" })?.value


// Initiate payment process using the extracted details
// ...

                     }     }
```

Opening UPI Apps from Your iOS App

- Use the `UIApplication.shared.canOpenURL(_:)` method to check if a specific UPI app is installed on the device.
- Iterate over a list of popular UPI apps (e.g., Google Pay, PhonePe, Paytm) to find installed ones.

  If a compatible UPI app is found, use the `UIApplication.shared.open(_:options:completionHandler:)` method to open the app with the appropriate URL scheme and query parameters.

```
let upiApps = ["googlepay://", "phonepe://", "paytm://"]


for upiApp in upiApps {
ifUIApplication.shared.canOpenURL(URL(string: upiApp)!) {
// Construct the URL with payment details
let paymentURL =URL(string: "?payeename=JohnDoe&amount=100&transactionid=12345")!

                              .shared.open(paymentURL, options: [:],
completionHandler: )            }}
```

Additional Considerations

- Implement robust error handling to gracefully manage potential issues during the payment process. This includes handling cases where the UPI app is not installed, the payment fails, or there are network errors. Consider providing informative feedback to the user in case of errors.

- Prioritize the security of sensitive payment data. Use secure communication channels and encryption to protect user information. Ensure that your app adheres to relevant security best practices and complies with industry standards.
- Design a user-friendly interface that guides users through the payment process seamlessly. Provide clear instructions, helpful tooltips, and visual cues to enhance the overall experience. Consider offering customization options to allow users to personalize the payment flow.
- Thoroughly test your app on various devices and with different UPI apps to ensure compatibility and functionality. Simulate different scenarios, such as successful and failed payments, to identify and address potential issues. Consider involving beta testers to get feedback from real users.
- Allow users to customize certain aspects of the payment process, such as choosing their preferred UPI app or setting default payment details. This can enhance user satisfaction and make the experience more personalized.
- Ensure that your app is accessible to users with disabilities. Adhere to accessibility guidelines and provide features like voiceover, high-contrast mode, and large text options.
- If you plan to target a global audience, consider localizing your app to support different languages and regions. This will make your app more inclusive and accessible to users from various backgrounds.
- Optimize your app's performance to ensure smooth and responsive payment transactions. Minimize network latency, optimize code, and use efficient algorithms to avoid delays or slowdowns.
- Stay updated with the latest regulations and guidelines related to UPI payments. Ensure that your app complies with all relevant laws and regulations to avoid legal issues.

By carefully considering these factors and implementing best practices, you can create a robust and user-friendly UPI payment experience within your iOS app.

Happy Coding!