

Power of Solana Mobile Stack: A Technical Deep Dive

Welcome to the world of Solana Mobile Stack (SMS), where the future of decentralized mobile applications meets the efficiency of the Solana network. This is your road map to start building with Flutter/React Native and integrate SMS to make your next dApp idea a reality.

Contents

- 👉 Key SDKs of Solana Mobile Stack
- 👉 Getting Started
 - 👉 Prerequisites and Setup
 - 👉 Integrating SMS with Flutter/React Native
- 👉 Advantages of building with Saga Phone
- 👉 Additional Tools
- 👉 My experience and final thoughts
- 👉 Resources

Key SDK's of Solana Mobile Stack

Want to integrate and support different Solana wallets in your app?

Mobile Wallet Adapter(MWA) is just the SDK you want.

MWA is like a common language that allows mobile apps (aka dApps) to easily talk to different mobile wallets for Solana.

This means developers only have to integrate MWA once in their dApp, and then it can work with any wallet that follows the MWA rules, like Phantom, backpack, Solfare, etc.

It saves developers from having to create specific code for each wallet, making things simpler and more efficient.

The image represents how MWA bridges the gap between dApps and Wallets.



Few Dapps (at the top) + Wallets (at the bottom)

Resource: <https://github.com/solana-mobile/mobile-wallet-adapter>

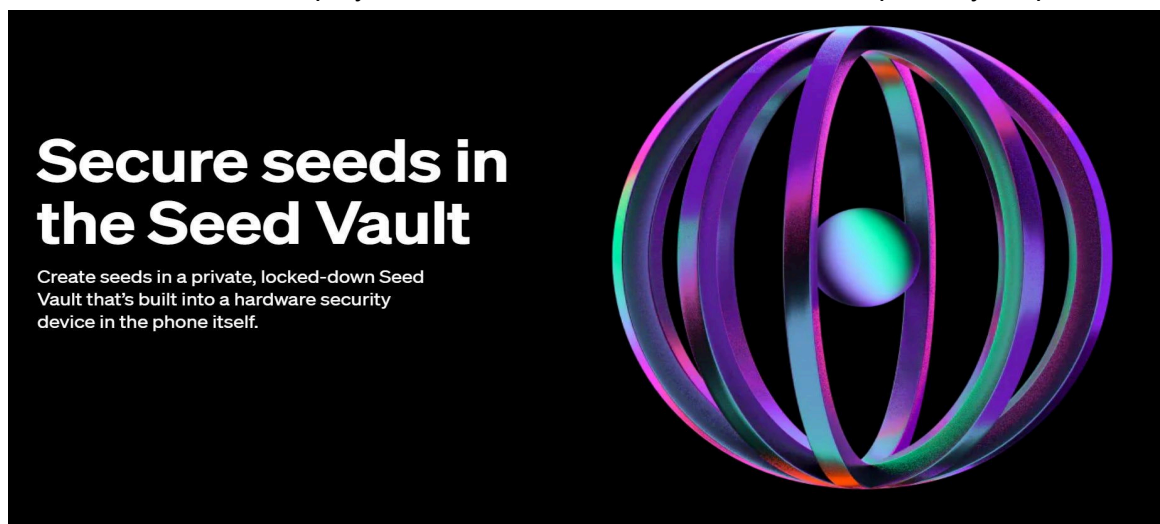
🙄 **Want to enhance security to protect keys and secrets for wallets?**

Seed Vault is a protective service for wallet apps on your phone.

It uses the most secure parts of your device to safeguard keys and secrets.

Your important information stays in this secure space, and the Android interface ensures secure transactions without your secrets leaving this protected environment.

In short, Seed Vault keeps your wallet data safe in the most secure part of your phone.



Resource: <https://github.com/solana-mobile/seed-vault-sdk>

🙄 **Looking for a special store just for dApps, to distribute your own?**

The Solana dApp Store is a different way of distributing apps, especially those created within the Solana ecosystem.

It offers a channel for apps to connect directly with users, without being bound by the rules or revenue demands of other app stores.

The aim is to encourage the Solana community to actively contribute to managing the store's content.

Resource: <https://github.com/solana-mobile/dapp-publishing>

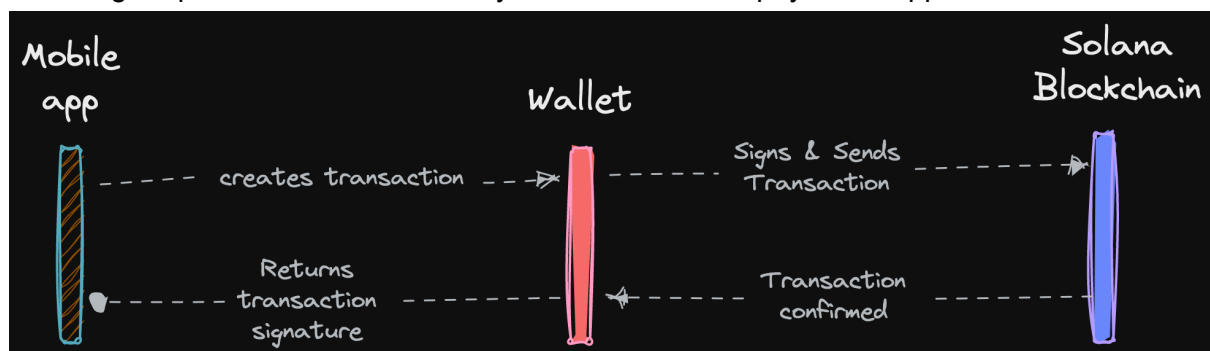
🤔 **What if there was a web3 version of Razorpay or Stripe for mobile payments?**

Say no more, Solana Pay has got you covered!

Solana Pay is a protocol designed separately from the Solana Mobile Stack, but it's a perfect match for mobile payments.

For developers wanting to add Solana Pay, they have created an example that shows how Wallet apps can use Android features to handle Solana Pay URLs from QR codes, NFC taps, messages, and web browsers, making it easy to launch Solana Pay requests.

The image represents how Solana Pay makes it easier to pay with dApps.



Resource: <https://docs.solanapay.com/>

🌟 Getting Started

In this section, I will be covering how to set up and develop a basic dApp with React Native and Flutter. You might think why not cover Android as well? The simple answer is that all the SDKs are natively made in Kotlin(Android) and you will find overwhelming examples of how to use them by just visiting their documentation and GitHub repos. Whereas in the cases of Flutter and React Native, choosing the right packages is more important than the code itself.

📦 Prerequisites and Setup

- ✓ A fundamental understanding of blockchain and Solana.
- ✓ Basic knowledge of either Flutter or React Native.

⚙️ Installation guide to set up Flutter on your machine:

<https://docs.flutter.dev/get-started/install>

⚙️ React Native project setup guide: <https://reactnative.dev/docs/environment-setup>

✓ Create or connect to a Solana wallet. A wallet is essential for interacting with the Solana blockchain. You can leverage existing wallets or create a custom one using the available SDKs.

🔧 Integrating SMS with Flutter

Flutter integration with SMS mainly works with the following packages

⚡ solana_wallet_adapter (https://pub.dev/packages/solana_wallet_adapter)

It is a common package used to integrate SMS Mobile Wallet Adapter.

⚡ solana_mobile_client (https://pub.dev/packages/solana_mobile_client)

This package is commonly used to communicate with Solana client.

⚡ solana (<https://pub.dev/packages/solana>)

This package is Solana web3js package equivalent built for dart.

Additionally,

⚡ solana_pay (https://pub.dev/packages/solana_pay)

This package can be used to integrate Solana Pay Protocol with your app.

As I mentioned earlier, choosing the right packages is important and as devs, it is easy to figure out the rest from there so instead of me giving some random code examples, here is well well-written comprehensive tutorial that covers building a basic dApp in Flutter.

<https://medium.com/@ronak01.raj/a-comphrensive-tutotial-building-dapps-with-flutter-and-solana-mobile-stack-sms-e452356a0adb>

🔧 Integrating SMS with React Native

React Native integration with SMS mainly works with the following packages

⚡ wallet-adapter-mobile

(<https://www.npmjs.com/package/@solana-mobile/wallet-adapter-mobile>)

It is a common package used to integrate SMS Mobile Wallet Adapter for react.

⚡ wallet-adapter-react-native

(<https://www.npmjs.com/package/@solana-mobile/mobile-wallet-adapter-walletlib>)

This is a package that provides React Native bridge for the native

mobile-wallet-adapter-walletlib library and it is designed for Wallet apps built in React Native.

It provides an API to implement the wallet endpoint of the mobile wallet adapter protocol.

⚡ solana (<https://www.npmjs.com/package/@solana/web3.js>)

Package used to communicate with the Solana web rpc

Additionally,

⚡ solana-pay (<https://www.npmjs.com/package/@solana/pay>)

This package is used to integrate Solana Pay in React native.

Just like for Flutter, we have a well-written tutorial that covers the basics

<https://medium.com/@laura.estupinand/solana-mobile-stack-developer-tutorial-building-a-dapp-in-react-native-125998ba3681>

📱 Advantages of building with Saga Phone

Do you own a Saga phone and want to unlock its full potential for development? Say no further here it is!

Saga has an in-built hardware wallet called Seed Vault, which not even the Android OS can touch.

It makes working with Web3 much easier and smoother than regular web apps.

With apps like Phantom or Solflare, doing transactions on the Solana network is as simple as a screen touch, with occasional PIN inputs for security.

It's all about a straightforward and user-friendly experience!



Get yours here <https://two.solanamobile.com/>

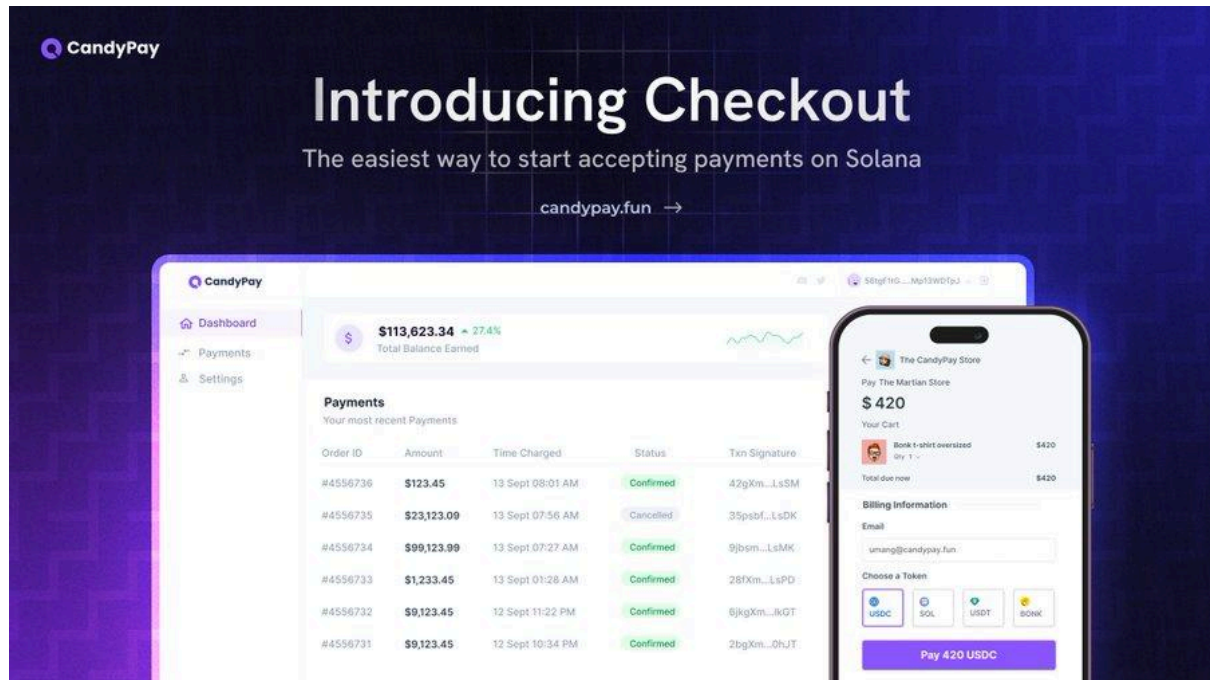
🔧 Additional Tools

Many emerging tools want to make the process of integrating Solana in dApps a seamless task. I have listed and briefed about 2 such tools that you might find useful and want to use in your next dApp.

🍬 CandyPay (<https://candypay.fun/>)

CandyPay simplifies Solana payments for businesses by providing an easy-to-use solution.

It overcomes technical challenges, offers a user-friendly experience, and ensures mobile support, making it hassle-free for merchants to accept Solana transactions.

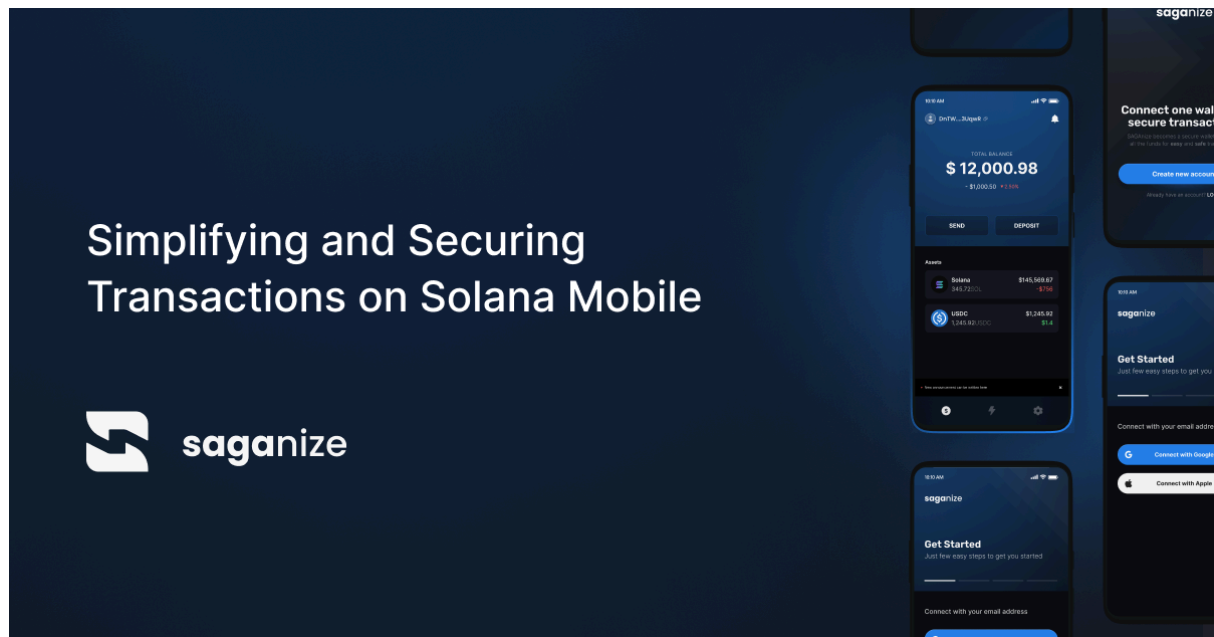


 **Saganize** (<https://saganize.com/>)

Saganize simplifies and secures Solana Mobile transactions with Solwave, a Kotlin SDK.

Developers can effortlessly add in-app transactions without building a separate wallet, ensuring swift approvals, seamless user experiences, and top-notch security through advanced encryption.

Users can onboard easily, log in once for hassle-free transactions, and approve with a quick PIN entry within the Saganize ecosystem.



My experience and final thoughts

SMS is a good entry point for developers to work on Solana based dApps.

I recommend starting with one of the blogs shared earlier to build your first dApp and slowly customize it to your needs. Why so?





1. You will be able to debug and solve any issues pretty quickly, which will help you get a grip on things. This will be beneficial when you start customizing the app for your specifications and come across new/general errors.
2. You will learn the use cases covered by the packages you will integrate.

During development, only some of your use cases might be covered and you have to work with some unstable packages but SMS is actively improving day by day.

All the packages/SDKs are open source and checking on all the issues and active PRs are some good starting points to debug and fix errors in your app. And don't forget to join the super-active discord community of SMS, which is always there to guide and help you.

If you come across a solution for existing issues, contribute to the SDK's and help out the community to improve the SDK's for everyone.

Resources

-  Official Website: <https://solanamobile.com/>
-  Join the Discord: <https://discord.com/invite/solanamobile>
-  Official Docs: <https://docs.solanamobile.com/getting-started/overview>
-  And all the other links provided throughout the blog

References

<https://docs.solanamobile.com/getting-started/overview>

<https://heybeluga.com/articles/solana-saga-phone-review/>

<https://docs.candypay.fun/checkout/introduction.html>

<https://github.com/saganize>

Image credits

<https://hornbull.substack.com/p/the-1st-crypto-phone>

<https://twitter.com/candypayfun/status/1621227243006734337>