Solana Web Standard

Decentralized Web Hosting with Solana

@guillaumetch

Table of Contents

Table of Contents

Executive Summary

The **Solana Web Standard (SWS)** introduces a forward-thinking framework for **decentralized web hosting** that leverages Solana's blockchain infrastructure alongside decentralized storage solutions like **Arweave** and **IPFS**. This standard enables a shift from traditional, centralized web hosting models—dominated by servers and DNS systems controlled by governments and corporations—to a **user-controlled, censorship-resistant, and permanent web architecture**.

Key components of SWS include:

- **.sol Domain Names**: Managed as **NFTs** on the Solana blockchain, these domains represent user-owned, transferable identifiers that replace centralized DNS with on-chain authenticity.

- **Decentralized Storage**: Static site files (HTML, CSS, JavaScript) are stored on permanent, tamper-resistant networks like **Arweave** or peer-to-peer systems such as **IPFS**. This ensures content integrity, durability, and censorship resistance.

- **Content Hash Linking**: Instead of traditional URLs, site content is linked via immutable content hashes stored in domain records, ensuring that even if storage locations change, the content remains accessible and verifiable.

- **Edge Computing and Decentralized Backends**: While static content is efficiently handled by decentralized storage, **dynamic content delivery** and **real-time interactivity** are achieved through a combination of **Edge Computing** (e.g., Cloudflare Workers, Akash Network) and emerging **decentralized databases** (e.g., Tableland, Ceramic, GunDB).

Major Takeaways

- **Censorship Resistance and Ownership**: SWS empowers users to control their web presence without dependence on centralized hosting providers or DNS authorities.

- **Permanent and Tamper-Proof Content**: Leveraging Arweave ensures that websites and data are preserved permanently, creating a verifiable historical record.

- **Hybrid Approach as a Practical Solution**: While fully decentralized architectures are still evolving, a **hybrid model** combining decentralized storage, edge computing, and decentralized backends offers a feasible path forward today.

- **Web3 Integration and Future Scalability**: SWS aligns with the broader Web3 ecosystem by supporting wallet-based identity, smart contracts, and decentralized APIs, enabling future-proof applications.

Background and Motivation

The internet, as we know it today, is predominantly hosted and controlled by **centralized entities**—corporations, cloud providers, and DNS authorities. While this model has enabled massive global connectivity, it comes with significant drawbacks:

Centralization Risks

- **Censorship and Control:** Websites and applications can be taken down or restricted by governments, service providers, or domain registrars.
- **Single Points of Failure:** Outages at major providers (e.g., AWS, Cloudflare) can bring down vast portions of the web.
- **Data Vulnerabilities:** Centralized databases and servers are prime targets for hacks, leaks, and unauthorized access.

Erosion of Digital Sovereignty

- **Users and creators have limited control** over their digital identities, domains, and content. DNS domains can be revoked, content can be deleted, and platforms can restrict access.
- **Censorship has become a growing threat**, particularly in regions with limited freedom of expression or authoritarian control.

The Motivation for Solana Web Standard (SWS)

The motivation behind the **Solana Web Standard** is to provide a clear, interoperable framework for:

- Empowering creators and developers with **control over their web presence**.
- Offering **resilience and permanence** for critical web content.
- Bridging the gap between **static decentralized storage** and **dynamic content delivery** using decentralized backends and edge computing.
- Aligning with the **Web3 ethos of openness, ownership, and censorship resistance**.

By shifting to a decentralized hosting model, the web can become a more **inclusive, user-owned, and resilient ecosystem**, paving the way for a truly **decentralized internet**.

Core components

The **Solana Web Standard (SWS)** is built upon a robust combination of **decentralized technologies** and **blockchain-based protocols** that work seamlessly together to enable a **resilient, censorship-resistant, and user-controlled web ecosystem**.

Here's an overview of the **major components** that make this possible :

Identity and Domain System: .sol Names

- **.sol domains** are NFTs registered on the **Solana blockchain**, representing ownership of web addresses in a decentralized manner.

- They replace centralized DNS with **on-chain identity and ownership**, ensuring that domains can't be seized or censored.

- Domain owners manage metadata, including **content hashes (Arweave, IPFS)**, directly on-chain.

Decentralized Storage Solutions

- **Arweave**: A blockchain-based, permanent data storage solution. Once uploaded, content is preserved indefinitely, ensuring immutability and tamper-resistance.

- **IPFS (InterPlanetary File System)**: A peer-to-peer storage network where files are addressed by their unique **content hashes (CIDs)**. Ideal for hosting large files with optional pinning.

- **Skynet & Hypercore (Optional)**: Advanced decentralized file storage systems offering P2P data sharing, versioning, and redundancy.

Content Hashing and Linking

- Websites and content stored on decentralized networks are identified using **content hashes** (e.g., Arweave transaction ID or IPFS CID).

- These hashes are stored in the `.sol` domain's metadata, creating a verifiable and immutable link between a domain name and its content.

Resolvers and Gateways

- Users and applications resolve `.sol` domains via:
    - On-chain lookups of domain metadata (content hashes).
    - Decentralized gateways like **arweave.net** or **ipfs.io**, which fetch the linked content and serve it to browsers.

- This system bypasses traditional DNS and centralized hosting, ensuring availability and censorship resistance.

Dynamic Content and Decentralized Backends

- While static content is handled by Arweave/IPFS, **dynamic functionality** (user-specific data, real-time interactions) is supported through:
    - **Tableland**: Web3-native, SQL-like database with on-chain control.
    - **Ceramic/ComposeDB**: Document-based decentralized data streams for identities and user-generated content.
    - **GunDB**: Peer-to-peer, CRDT-based NoSQL database for real-time, offline-first apps.

Edge Computing for Performance

- **Edge computing platforms** (e.g., Cloudflare Workers, Akash, Fastly) complement the decentralized architecture by:
    - Handling dynamic logic at the network edge.
    - Reducing latency for personalized content.
    - Ensuring global scalability and resilience.

The **Solana Web Standard's core components** create a **layered, modular architecture** that:

- **Empowers users** with domain ownership via .sol names.
- **Ensures content permanence and integrity** through decentralized storage.
- **Bridges static and dynamic content** with edge computing and decentralized backends.
- **Enables a fully decentralized web experience** with interoperability across protocols.

Hosting Workflow

The **Solana Web Standard (SWS)** defines a **clear, step-by-step process** for deploying websites in a decentralized manner. This workflow integrates decentralized storage, on-chain identity, and Web3 protocols, enabling developers to **launch websites that are censorship-resistant, permanent, and user-owned**.

Here's a streamlined overview of the **workflow**:

Step 1: Build Your Website

- Create your website using modern frameworks like **React, Vue, or Svelte**.
- Run a production build to generate **static files**: HTML, CSS, JavaScript, images.
- These static files are ready for decentralized deployment.

Step 2: Upload to Decentralized Storage

- **Arweave**: Use **Bundlr** to upload your build directory to Arweave's permanent storage using : `bundlr upload-dir ./build --index-file index.html`

  This generates a permanent **content hash** (Arweave TX ID).

- **IPFS**: Add files to IPFS using: `ipfs add -r ./build`

  This outputs an IPFS **Content Identifier (CID)**.

  Choose your preferred storage layer (Arweave for permanence, IPFS for flexibility) or use both for redundancy.

Step 3: Link Content Hash to .sol Domain

- Using **Solana Name Service (SNS)**, update your `.sol` domain's metadata to include the **content hash** (e.g., `ar://<hash>` or `ipfs://<CID>`).
- This links your domain to your website's content, ensuring users can resolve the domain to the correct decentralized files.

Step 4: Access Your Site

- Users can now access your site through:
  - Web3-enabled browsers (e.g., Phantom, Solflare, Brave).
  - Custom resolvers that translate `.sol` domains to Arweave/IPFS URLs.
  - Gateways like **arweave.net** or **ipfs.io** which bridge the decentralized storage to HTTP access.

Optional: Integrate Dynamic Content

- For user-specific data or real-time updates:
    - Deploy **Edge Functions** (Cloudflare Workers, Akash, Fastly) to handle requests dynamically at the network edge.
    - Connect to **decentralized backends** like Tableland, Ceramic, or GunDB to manage data in a decentralized manner.

The **SWS Hosting Workflow** combines the simplicity of **static site generation** with the resilience and ownership of **decentralized storage** and **on-chain identity**. With optional edge computing and decentralized backends, it supports both **static** and **dynamic** web experiences that align with the principles of **Web3**.

Architecture Diagram

The **Solana Web Standard (SWS)** brings together decentralized technologies in a **layered, interoperable architecture**. This design ensures that websites are **resilient, censorship-resistant, and user-controlled**, with both static and dynamic components delivered securely and efficiently.

Here's a detailed look at the **core architectural layers** and how they work together:

1. User Interface Layer (Frontend)

- **React/Vue/Svelte static files** (HTML, CSS, JS) are built and prepared for deployment.
- Deployed to **Arweave** or **IPFS** for **permanent, tamper-proof storage**.
- Accessed by users through Web3-enabled browsers or gateways.

2. Domain and Identity Layer

- **.sol domains** are registered on the Solana blockchain as NFTs.
- These domains store **metadata and content hashes** (Arweave TX IDs or IPFS CIDs) on-chain.
- When users request a domain, the system retrieves the associated **content hash** for the correct website content.

3. Decentralized Storage Layer

- **Arweave**: Ensures permanent, immutable storage of website files.
- **IPFS**: Offers distributed, content-addressed storage with optional pinning and redundancy.
- Content hashes (CIDs or TX IDs) serve as unique, verifiable references to the stored data.

4. Resolver and Gateway Layer

- Acts as the bridge between **on-chain domain records** and **decentralized storage**.
- When a user enters a `.sol` domain:
    1. The **on-chain resolver** looks up the domain's metadata.
    2. The resolver extracts the **content hash**.

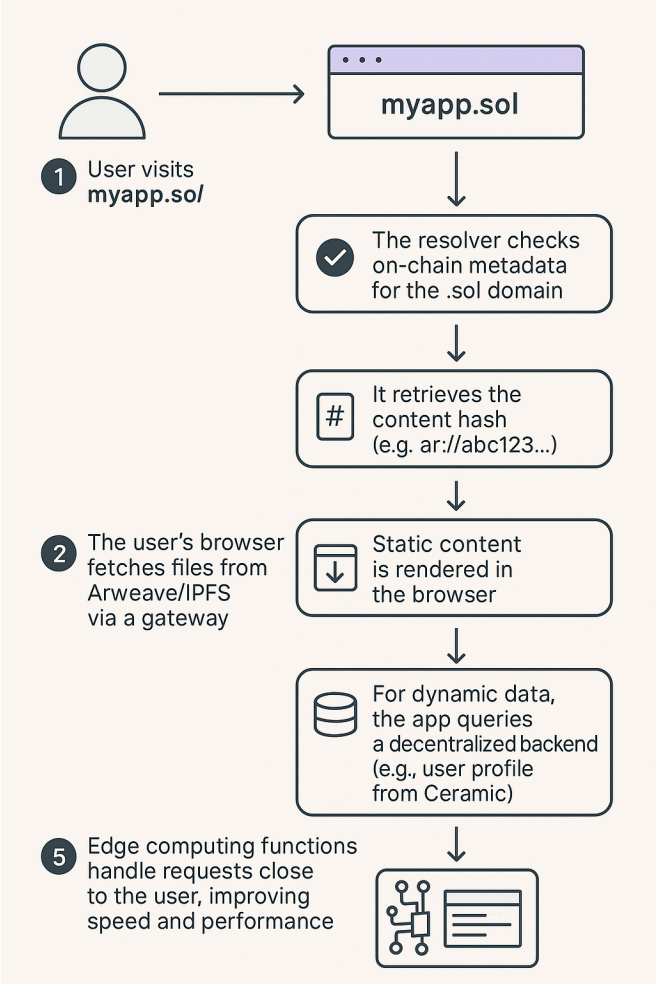
3. A **gateway** (e.g., arweave.net, ipfs.io) fetches the corresponding content from the storage network.

- Content is then served to the user's browser.

5. Optional Dynamic Content and Backend Layer

- **Dynamic user data** or **real-time interactions** can be integrated using:
    - **Edge computing platforms** (e.g., Cloudflare Workers, Akash) for low-latency processing.
    - **Decentralized backends** (Tableland, Ceramic, GunDB) for decentralized data management and querying.
- These systems ensure a hybrid model, combining the security of static decentralized content with dynamic features.

Architecture Flow Example



The **SWS architecture** seamlessly combines:

- **User-owned domains (.sol)** for identity and control.
- **Decentralized storage networks** (Arweave, IPFS) for permanent content.
- **Resolvers and gateways** for dynamic content retrieval.
- **Edge computing and decentralized backends** for scalability and flexibility.

This layered approach ensures **resilience, censorship-resistance, and a true Web3 user experience**.

Key Benefits

The **Solana Web Standard (SWS)** presents a **paradigm shift** in web hosting by combining **blockchain-based identity**, **decentralized storage**, and **Web3 technologies**. This approach brings significant benefits to developers, businesses, and end-users alike.

Here are the **major advantages** of adopting the SWS framework:

1. Censorship Resistance

- Websites hosted via SWS are stored on **decentralized networks** like Arweave and IPFS, making them immune to takedown by governments, corporations, or ISPs.
- Domains (.sol) are **owned as NFTs on Solana**, providing a tamper-proof system for managing digital identities and web presence.

2. True Ownership and User Control

- Users and developers control both the **domain names** (via .sol) and the **content** itself.
- Eliminates reliance on centralized registrars, DNS providers, or hosting platforms.

3. Permanent and Immutable Content

- Using **Arweave** for storage ensures that once content is uploaded, it is preserved **forever**—unchanged, verifiable, and tamper-resistant.
- Ideal for historical records, creative works, and critical data.

4. Low-Cost and Scalable Deployment

- Hosting static sites on Arweave/IPFS is **cost-effective**—a one-time payment stores files permanently (Arweave) or minimal pinning costs (IPFS).
- No ongoing hosting fees or bandwidth charges.

5. Web3 Interoperability and Identity

- Integrates seamlessly with the broader Web3 ecosystem—wallet authentication, smart contracts, decentralized data sources, and more.
- Enables dApps to function with **on-chain data** and **decentralized authentication systems**.

6. Enhanced Security and Reliability

- Content is distributed across a **global decentralized network**, eliminating single points of failure and reducing vulnerability to attacks.

- Resilience is built into the architecture, ensuring high availability even in the face of network disruptions.

7. Dynamic Content and Edge Performance

- Supports **hybrid architectures** with **Edge Computing** for dynamic data handling and personalization.

- Leverages decentralized backends (Tableland, Ceramic, GunDB) to manage user-generated content, profiles, and real-time data.

The **Solana Web Standard** delivers a **secure, cost-effective, and future-ready framework** for decentralized web hosting. It empowers users with true ownership, ensures data permanence, resists censorship, and aligns perfectly with the evolving Web3 landscape.

Limitations & Solutions

While the **Solana Web Standard (SWS)** offers a robust framework for decentralized web hosting, it's essential to acknowledge and address the **current limitations** that may impact adoption or functionality. Here's a balanced look at these challenges—and how to overcome them.

1. IPFS Requires Pinning for Persistence

- **Limitation:** By default, IPFS doesn't guarantee that content stays available. Files are removed unless actively pinned or stored.

- **Solution:**
  - Use **Filecoin**, **Pinata**, or **web3.storage** to pin and back up IPFS content.
  - Prefer **Arweave** for critical content needing permanent, immutable storage.

2. Smart Contract Costs and Complexity

- **Limitation:** Writing and managing smart contracts (e.g., for domain management or dynamic content) can be costly and complex, especially for new developers.

- **Solution:**
  - Use tools like **Bundlr** to optimize Arweave uploads and reduce on-chain interactions.
  - Leverage **pre-built contracts** and APIs provided by SNS for managing .sol domains.

3. No Native Support for Dynamic Content

- **Limitation:** Arweave and IPFS excel at static content, but dynamic, personalized data (e.g., user profiles, comments) isn't natively supported.

- **Solution:**
  - Integrate **decentralized backends** (e.g., Tableland for structured data, Ceramic for documents, GunDB for real-time sync).
  - Use **Edge Computing** (e.g., Cloudflare Workers, Akash) for dynamic data processing close to users.

4. Limited Browser Support for .sol Domains

- **Limitation:** Mainstream browsers don't natively resolve `.sol` domains or decentralized storage links.

- **Solution:**
  - Build **custom resolvers**, browser extensions, or redirect services to bridge `.sol` domains to Arweave/IPFS content.
  - Educate users to use **Web3-enabled browsers** (Brave, Phantom)

5. Security and Key Management Risks

- **Limitation:** Decentralized systems shift the responsibility of **key management** to users, increasing risks of lost access or compromised keys.

- **Solution:**
    - Encourage best practices like **hardware wallets**, **multi-sig setups**, and **key recovery plans**.
    - Leverage emerging decentralized identity solutions for improved key security.

6. Adoption and Ecosystem Maturity

- **Limitation:** While promising, the decentralized web ecosystem (including SWS) is still **maturing**, with evolving tooling and standards.

- **Solution:**
    - Start with a **hybrid approach**: combine decentralized frontends with traditional APIs and storage where needed.
    - Contribute to open-source projects and collaborate with the **Solana and Web3 communities** to accelerate ecosystem development.

The **Solana Web Standard** provides a **strong foundation** for decentralized web hosting, but real-world deployment requires addressing **current limitations**. By implementing the recommended solutions—pinning services, edge computing, decentralized backends, user education, and hybrid architectures—developers can build **resilient, scalable, and user-friendly decentralized websites** today while preparing for a fully Web3-native future.

Future Developments

The **Solana Web Standard (SWS)** lays the foundation for a resilient, user-owned, decentralized web. As we look ahead, several key developments will enhance its functionality, usability, and adoption—bringing decentralized web hosting into the mainstream.

Here's a **vision for the next phase of SWS**, aligned with practical goals for implementation and ecosystem support:

1. Deploy a Proof of Concept Website

- **Objective:** Showcase the SWS architecture in action with a fully functional decentralized website.

- **Actions:**
  - Build a React/Vue app with static assets deployed on **Arweave/IPFS**.
  - Link the site to a **.sol domain** (managed via **SNS**).
  - Demonstrate seamless **content resolution** via decentralized storage.
  - Optionally, integrate dynamic content using **Tableland** or **Ceramic** for profiles, comments, or activity feeds.

2. Build a Custom Resolver as a Chrome Extension

- **Objective:** Enable standard browsers (Chrome) to resolve `.sol` domains without additional infrastructure.

- **Actions:**
  - Develop a Chrome extension that intercepts `.sol` domain requests.
  - Query the **Solana blockchain** to retrieve associated content hashes (e.g., Arweave TX IDs, IPFS CIDs).
  - Directly load content from decentralized gateways (arweave.net, ipfs.io) or nodes.
  - Ensure user-friendly UX with settings, domain history, and fallback mechanisms.

3. Develop an API for Wallet and Browser Integration

- **Objective:** Allow wallets and browsers to integrate SWS seamlessly, resolving `.sol` domains and loading decentralized content.

- **Actions:**
  - Design an **open API standard** for resolving domains via on-chain lookups and fetching content from decentralized storage.
  - Include functions for wallet-based authentication, domain verification, and secure signing of updates.

    - Provide SDKs/libraries for easy integration into **Phantom, Solflare, Brave, and custom browsers**.
    - Support **on-chain metadata caching** and **gateway selection** for performance optimization.

4. Empower the Developer Community

- **Objective:** Support and scale adoption of the Solana Web Standard across the Web3 ecosystem.

- **Actions:**
    - Publish **developer guides, tutorials, and code samples** for integrating SWS into projects.
    - Contribute to **open-source tooling** (e.g., Bundlr, Ceramic SDKs) to improve ecosystem readiness.
    - Host **workshops, webinars, and hackathons** to onboard developers and projects.
    - Collaborate with **wallets, dApps, and storage providers** to align on standards and interoperability.

5. Enhance Performance and Flexibility

- **Objective:** Make SWS scalable and production-ready for mainstream applications.

- **Actions:**
    - Integrate **Edge Computing platforms** (Cloudflare Workers, Akash) to handle dynamic content close to users.
    - Explore additional decentralized backends (Tableland, GunDB) for scalable data management.
    - Optimize **browser-side caching**, **resolver logic**, and **content delivery paths** to improve performance.

The **future of SWS** is both exciting and actionable. By deploying a **proof of concept website**, building a **custom Chrome resolver**, and designing an **API for wallet and browser integration**, we will not only validate the framework but also empower the broader community to embrace decentralized web hosting. Through collaboration, open-source contribution, and ecosystem partnerships, **SWS** can become a cornerstone of a **truly decentralized, user-owned internet**.

Conclusion

The **Solana Web Standard (SWS)** presents a compelling vision for the future of web hosting—one that is **decentralized, censorship-resistant, and user-controlled**. By combining Solana's **high-speed blockchain** and **domain system** with **decentralized storage protocols** like **Arweave and IPFS**, SWS offers an innovative framework for building and serving web content without reliance on centralized servers.

Our exploration has demonstrated:

- The role of **.sol domains** as NFT-based web identifiers, enabling user-controlled ownership and decentralized DNS.
- The strengths of decentralized storage systems (**Arweave, IPFS, Hypercore**) for hosting **static content** immutably and efficiently.
- The current **limitations** of decentralized web hosting (e.g., lack of backend processing, dynamic content challenges), and emerging solutions such as **Tableland, Ceramic, and GunDB** for decentralized data handling.
- The importance of **Edge Computing** in bridging the gap between static, decentralized files and dynamic, user-specific web experiences.

These findings reveal a **hybrid model** as the most viable approach today:

- Host static frontends on **Arweave/IPFS** for permanence and resilience.
- Use **Edge Computing** (e.g., Cloudflare Workers) for dynamic logic, personalization, and integration with Web3.
- Store dynamic and structured data in **decentralized backends** (Tableland, Ceramic, GunDB) to maintain decentralization while achieving flexibility

The **Solana Web Standard** sets the stage for a new era of web hosting where control resides with users, not intermediaries. By blending blockchain technology, decentralized storage, dynamic computation, and edge delivery, it empowers developers to build **secure, censorship-resistant, and performant websites**. As the ecosystem matures, and tools like decentralized backends and edge computing advance, the vision of a truly decentralized web is not just possible—it's inevitable.

Recommendations

For developers and organizations seeking to adopt decentralized web hosting on Solana, we recommend:

Embrace Solana's .sol Domains: Establish your Web3 identity and site ownership on-chain, linking domains to decentralized content hashes.

Leverage Arweave or Bundlr for Static Content: Store immutable website files efficiently and cost-effectively.

Integrate Edge Computing for Dynamic Content: Utilize platforms like Cloudflare Workers or decentralized edge solutions to handle dynamic requests and personalize user experiences.

Explore Decentralized Backends: For applications requiring user data, structured content, or real-time updates, adopt solutions like **Tableland** for SQL-like storage, **Ceramic** for document-based data, or **GunDB** for P2P sync.

Design Hybrid Architectures: Recognize that a fully decentralized stack is evolving. Start with a hybrid approach to combine decentralization with performance and user experience.

Build with Composability in Mind: Ensure your decentralized web components integrate with wallets, identities, and protocols to enable future-proof, scalable dApps.

References

Solana Ecosystem

- **Solana Website**: https://solana.com
- **Solana Name Service (SNS)**: https://sns.id

Decentralized Storage

- **Arweave**: https://www.arweave.org/
- **Bundlr**: https://bundlr.network/
- **IPFS**: https://ipfs.io/
- **Filecoin**: https://filecoin.io/
- **Pinata (IPFS pinning)**: https://pinata.cloud/
- **Web3.storage**: https://web3.storage/

Decentralized Backends

- **Tableland**: https://tableland.xyz/
- **Ceramic Network**: https://ceramic.network/
- **ComposeDB (Ceramic)**: https://composedb.js.org/
- **GunDB**: https://gun.eco/

Edge Computing and Web3 Tools

- **Cloudflare Workers**: https://workers.cloudflare.com/
- **Akash Network**: https://akash.network/
- **Fluence**: https://fluence.network

- **Deno Deploy**: https://deno.com/deploy

Developer Tools and Standards

- **Solana Web3.js SDK**: https://solana-labs.github.io/solana-web3.js/
- **IPFS Documentation**: https://docs.ipfs.tech/
- **Arweave Bundlr Docs**: https://docs.bundlr.network/
- **Ceramic ComposeDB Docs**: https://composedb.js.org/docs

Additional References

- **Fleek (Decentralized Frontend Hosting)**: https://fleek.co/
- **Skynet (Sia-based decentralized storage)**: https://siasky.net/
- **Filebase (IPFS and S3 compatible)**: https://filebase.com/
- **Backpack Wallet**: https://backpack.app/
- **Phantom Wallet**: https://phantom.app/
- **Solflare Wallet**: https://solflare.com/