

BoB v.0.1.2
Bitcoin-based organization Builder
tetakta
2024

Bitcoin is revolutionizing the landscape. The allure of this adventure lies in its educational voyage: no matter which path you venture down, one constant remains. Bitcoin represents the ultimate frontier and the Holy Grail that leads us toward uncovering life as it was always meant to be. Bitcoin breaks free from constraints; it resonates at the highest frequency. It's the melody of innovation. Bitcoin resides in the space between circumstances and reality, pulsating with all human emotions and actions.

It is the truth that many seek.*

BoB, short for **Bitcoin Organization Builder**, is envisioned as a premier tool for facilitating organizational finance through the Bitcoin blockchain. It aims to capitalize on the inherent strengths of Bitcoin to offer a platform that is not only secure due to advanced cryptographic techniques but also maintains transparency in financial transactions. Moreover, BoB is designed to optimize the efficiency of financial operations using Bitcoin, by streamlining processes such as fund allocation, transaction tracking, and multi-signature authorizations. The tool will serve as a bridge between the traditional organizational financial management needs and the modern decentralized financial systems powered by Bitcoin.

Transparency: By utilizing the Bitcoin mainnet, BoB will offer an open and verifiable record of transactions, ensuring all organizational financial activities are transparent to authorized stakeholders.

Security: BoB will emphasize strong security practices, incorporating features like multi-signature authorization and time-locked contracts to safeguard assets. The Masterkey system will add a layer of security by providing a secure method for organizational users to manage signatures and authorizations, potentially through a convenient mobile application.

Efficiency: The application will streamline financial processes with automated scripts for transactions and address management. It will allow for the creation of a complex system of Bitcoin wallets, enabling treasuries to manage funds with agility and precision.

1. Description

- 1.1. [Define the functionalities: wallet management, transaction scripting, and multi-signature processes.](#)

2. User Experience

- 2.1. [User roles and permissions.](#)
- 2.2. [User flow diagrams and wireframes for interactions.](#)

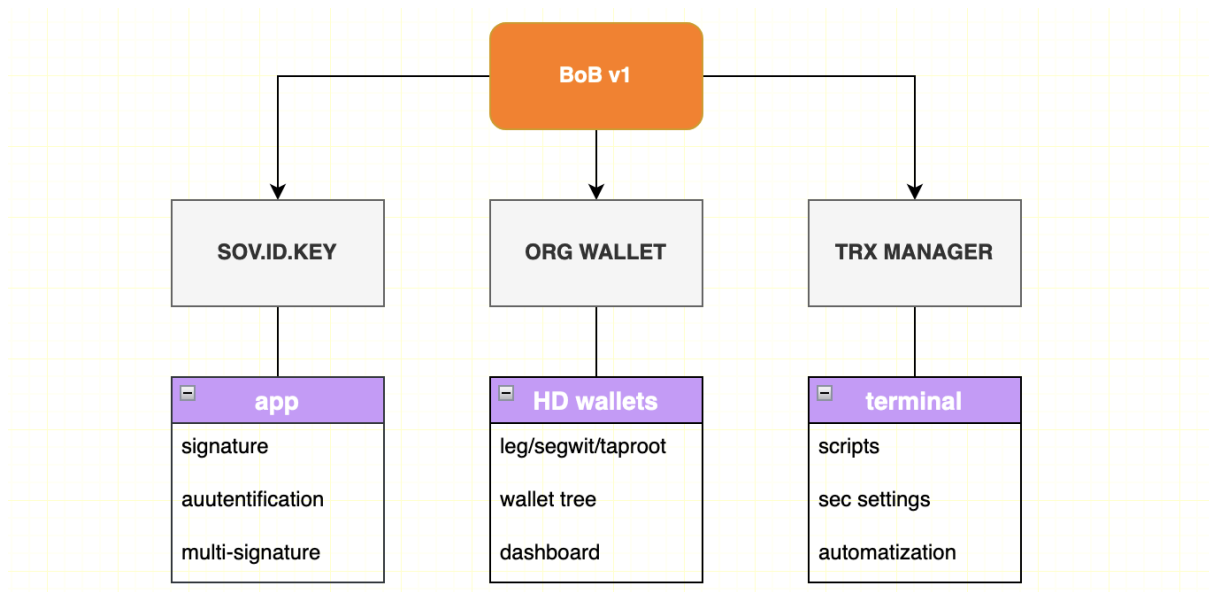
3. Technical Architecture

- 3.1. [Utilization of Bitcoin mainnet as the primary layer.](#)
- 3.2. [Front-end development with React.](#)

- 3.3. [Back-end services using Node.js and bitcoinjs-lib.](#)
- 4. **Masterkey System**
 - 4.1. [Description of the masterkey infrastructure.](#)
 - 4.2. [Integration with multi-signature wallet operations.](#)
- 5. **Scripting Mechanism**
 - 5.1. [Specification for scripts handling address programming and transaction automation.](#)
- 6. **Security Plan**
 - 6.1. [Encryption methods, key management, and other security best practices.](#)
- 7. **Acceptance Criteria**
 - 7.1. [Metrics and benchmarks for successful delivery.](#)
- 8. **UI UX Description**
 - 9.1 [Creating an organization](#)
 - 9.2 [Organization management](#)
- 9. [Project links](#) and [References](#)

Define the functionalities:

wallet management, transaction scripting, and multi-signature processes



BoB is to provide a comprehensive suite of Bitcoin-based tools designed for organizational use. It will include the following functionalities:

Wallet Management: Users can create and manage a hierarchical structure of Bitcoin wallets, facilitating organized fund storage and access. This may include the ability to generate new addresses, monitor balance and transaction history, and ensure wallets are compatible with HD (Hierarchical Deterministic) standards.

Transaction Scripting: BoB will allow users to automate Bitcoin transactions through scripting. This feature is intended to simplify complex transaction types, such as

batched transactions, automatic redistribution of funds according to predefined rules, and other advanced Bitcoin script-based operations.

Multi-Signature Processes: The platform will incorporate multi-signature authorization for transactions, enhancing security by requiring multiple parties to approve a transaction before it is broadcasted to the Bitcoin network. This process will also support flexible multi-signature setups, catering to various organizational governance models.

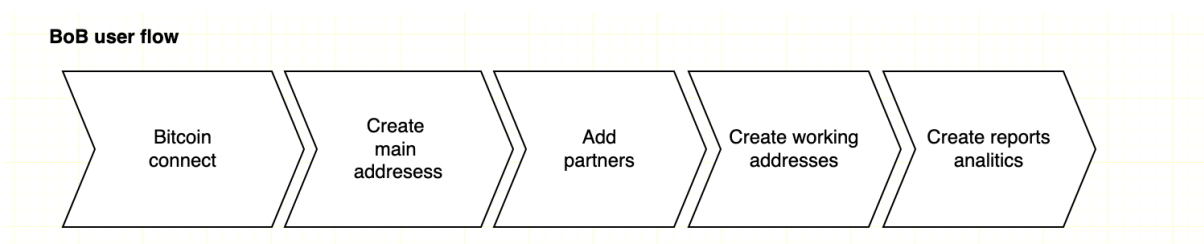
Each of these functionalities will be developed with a focus on usability for non-technical users while maintaining the rigorous security standards required for blockchain-based operations.

For the BoB application, the user experience is meticulously designed to accommodate various roles within an organization, each with distinct permissions and capabilities.

User Roles and Permissions:

- Administrators have the highest level of access, capable of setting up the master wallet, creating or revoking user permissions, and defining the rules for transaction scripts.
- Financial Officers manage the treasury functions, oversee transaction scripting logic, and initiate transactions subject to multi-signature approvals.
- Auditors have read-only access to ensure transparency and compliance, allowing them to verify transactions and wallet integrity without initiating or altering transactions.
- Standard Users can create individual wallets tied to the master wallet, propose transactions, and participate in multi-signature authorizations as defined by their role.

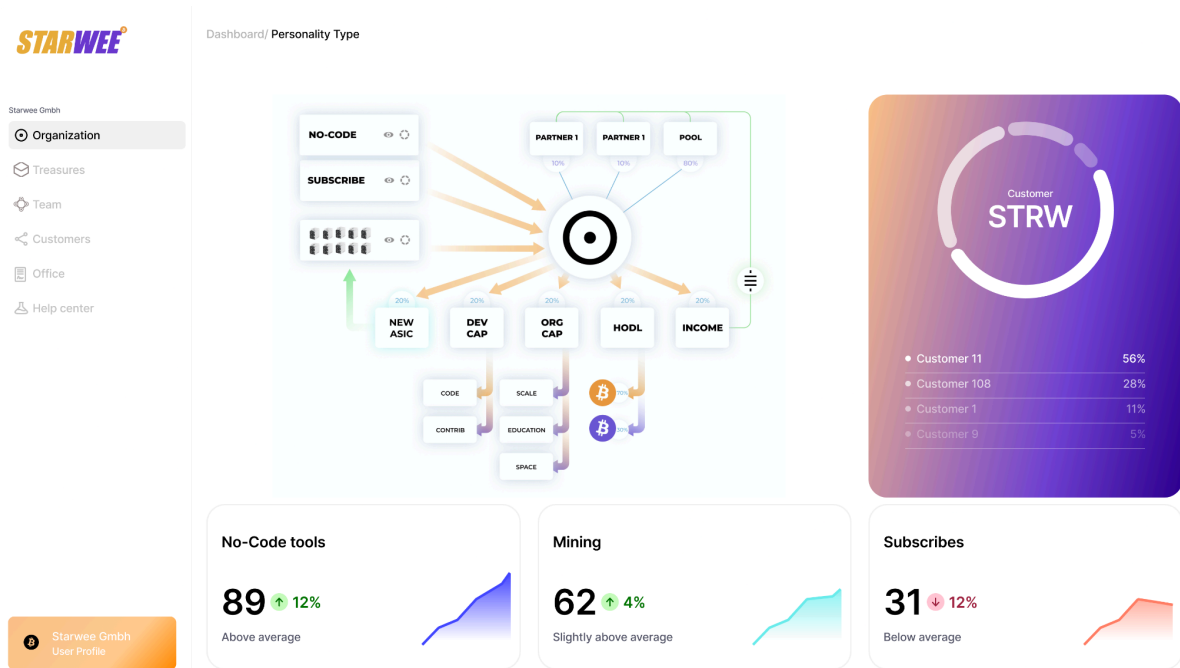
User Flow Diagrams and Wireframes:



1. Login and Authentication: Users are greeted by a secure login screen, leading to a dashboard customized to their role.
2. Dashboard: The main interface provides a summary of wallet statuses, pending transactions requiring signatures, and recent activity logs.
3. Wallet Management: From the dashboard, users navigate to wallet management where they can generate new addresses, view balances, and transaction history.
4. Transaction Management: Users with the necessary permissions can access a transaction creation interface, which includes scripting capabilities and multi-signature setup.

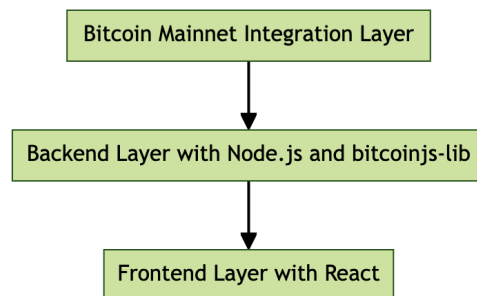
- Audit Trails: Auditors can access detailed logs and records of all transactions, including timestamps and involved parties, from their dashboard.

The wireframes will reflect a clean and intuitive design, ensuring that complex Bitcoin operations are simplified for the end-user. The emphasis on clarity and ease of navigation aims to facilitate a smooth and efficient user journey through the application.

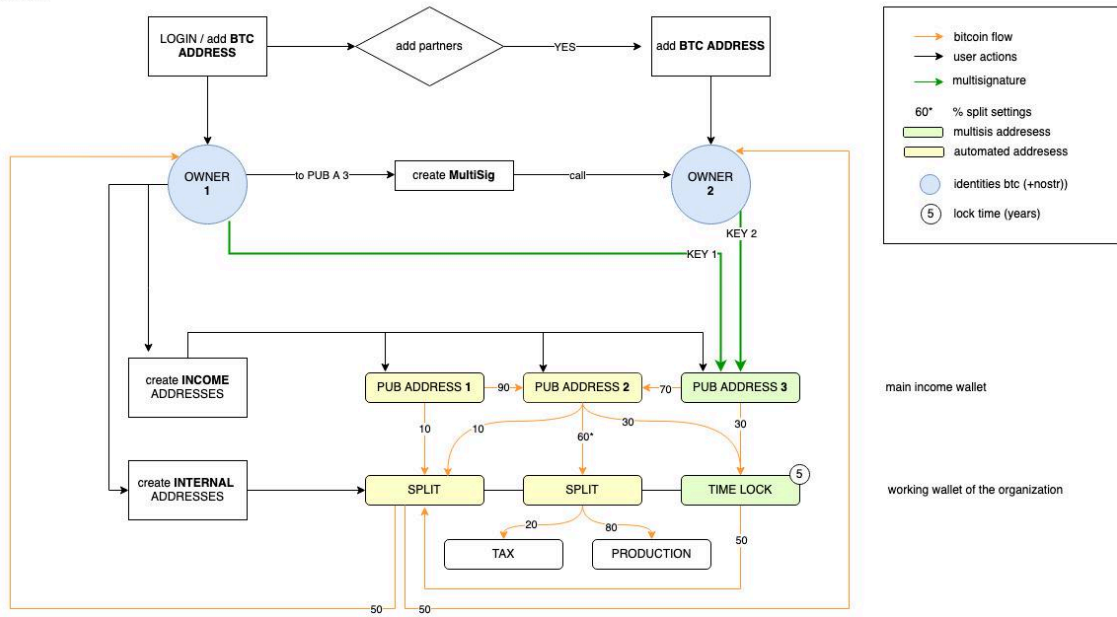


Technical Architecture

The BoB application's technical architecture is structured to ensure robustness, scalability, and security, aligning with the foundational principles of the Bitcoin network:



BoB v0.0.1



1. Bitcoin Mainnet Integration: BoB will interact with the Bitcoin mainnet to perform transactions, ensuring real-time synchronization with the blockchain. This is the foundational layer of the BoB application. Direct integration with the Bitcoin mainnet ensures that all transactions conducted through BoB are recorded on Bitcoin's decentralized and immutable ledger. This layer is responsible for interacting with the Bitcoin blockchain, facilitating secure and verifiable Bitcoin transactions.

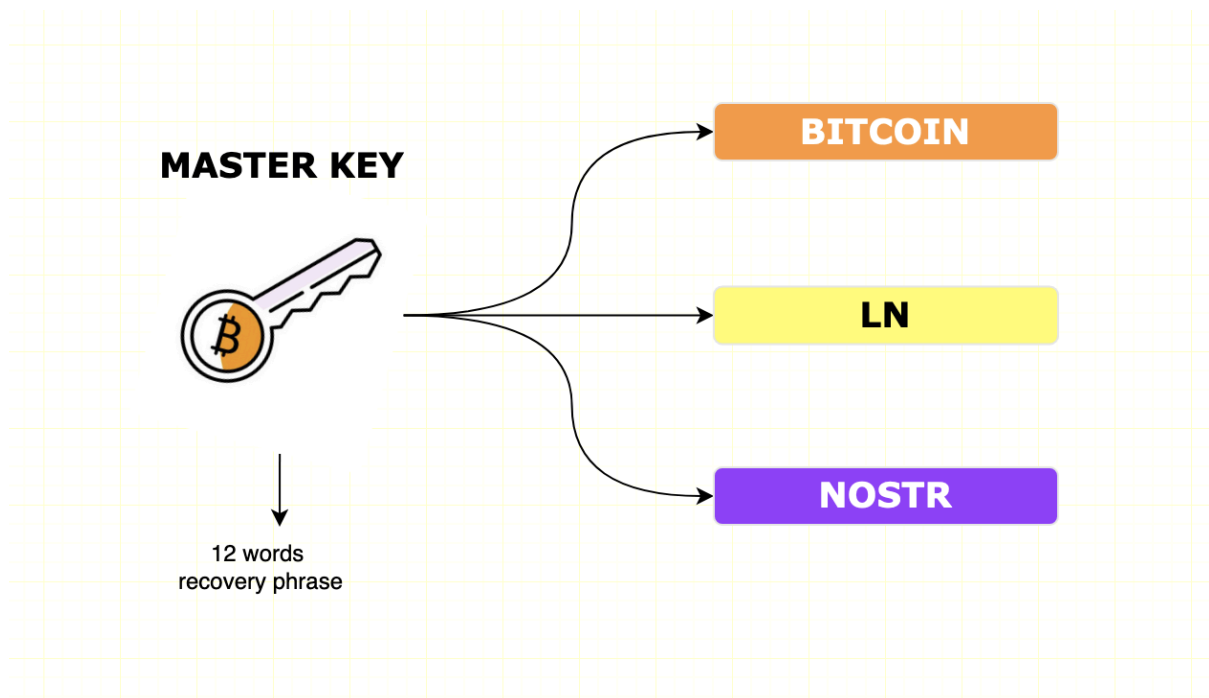
2. Front-end Development with React: The client-side of the application will be built with React, a modern JavaScript library for building user interfaces. The topmost layer of the BoB architecture is the frontend, developed using React. React's component-based architecture allows for the creation of isolated, state-managed components. This contributes to a secure and dynamic user interface, minimizing risks associated with cross-component vulnerabilities. The frontend layer is responsible for presenting information to the users in an accessible and interactive format. It fetches processed data from the backend and displays it, facilitating user interactions like transaction initiation, wallet management, and viewing transaction histories.

3. Back-end Services using Node.js and `bitcoinjs-lib`: Above the Bitcoin integration layer sits the backend, powered by Node.js. This layer serves as the application's engine room, handling business logic, processing data, and managing interactions with the Bitcoin network. The use of bitcoinjs-lib within this layer is crucial. It provides the necessary tools for implementing Bitcoin-specific operations like address generation, transaction signing, and ensuring these processes adhere to Bitcoin's security standards. The backend layer acts as a bridge between the blockchain and the frontend, processing and relaying information to and from the Bitcoin mainnet.

Back-end Services using Node.js and `bitcoinjs-lib`: The server-side logic will be handled by Node.js, a JavaScript runtime built on Chrome's V8 JavaScript engine. It will facilitate the creation of RESTful APIs that the React front-end will consume. `bitcoinjs-lib`, a well-known library for Bitcoin applications, will be used to handle wallet creation, transaction signing, and other Bitcoin-specific operations.

Masterkey System

The Master Key concept in Bitcoin and related technologies like Nostr and the Lightning Network (LNURL-Auth) is indeed a foundational element for secure interactions across different protocols. Each of these use cases employs the Master Key differently:



Bitcoin Base Layer

In the base layer of Bitcoin, the Master Key is used to derive all the addresses and private keys for transactions. With HD wallets, you can generate a tree of keys from the Master Key without the need to back up each individual key—only the Master Key is crucial for recovery.

Nostr

Nostr is a decentralized network that can use cryptographic keys for identity verification. A Master Key in this context could be used to derive subkeys that sign messages or transactions, effectively proving identity without exposing the Master Key itself.

LNURL-Auth

LNURL-Auth is a method for authenticating with services using keys from your Lightning wallet. Instead of traditional username and password combinations, LNURL-Auth uses a private key to sign a challenge from the service, which the service can verify using the corresponding public key. This process is often facilitated by a Master Key, from which the signing key is derived.

The Master Key System's integration with multi-signature wallet operations in BoB is a pivotal aspect:

Operational Control: The master key is central to the multi-signature setup, allowing for distributed control over transactions. It enables the definition of transaction authorization policies, requiring multiple signatures for execution.

Enhanced Security: This setup ensures no single individual can unilaterally move funds, aligning with robust security protocols.

Flexible Governance Models: The system accommodates various governance structures, fitting different organizational needs.

User Authorization Levels: Different levels of transaction authority can be set, based on the organization's hierarchy and risk management protocols.

This integration ensures secure, efficient, and flexible management of Bitcoin transactions within the organizational framework.

Scripting Mechanism

The BoB application's scripting mechanism is a crucial component of its security architecture, focusing on automating Bitcoin transaction processes and address management. This mechanism is designed to provide both operational efficiency and enhanced security:

Automated Bitcoin Transactions: The scripting mechanism automates various aspects of Bitcoin transactions. This includes the automatic generation of transactions based on predefined conditions, handling of batch transactions, and the execution of complex, multi-step transaction processes. By automating these tasks, the system minimizes human error and increases the efficiency and reliability of transaction handling.

Dynamic Address Management: The scripting also extends to the dynamic management of Bitcoin addresses. It can programmatically generate new addresses as needed, ensuring that address reuse is minimized. This is crucial for maintaining privacy and security in the Bitcoin ecosystem, as address reuse can make it easier to trace transaction histories and wallet balances.

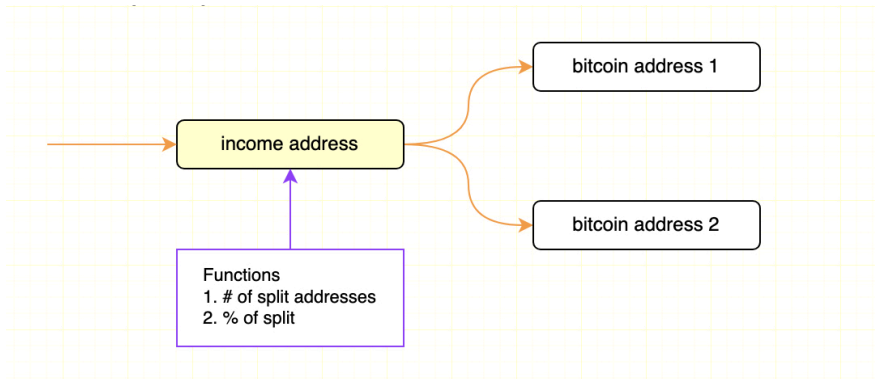
Security Through Automation: By automating these processes, the scripting mechanism in BoB significantly enhances the security of the system. Automated scripts reduce the risk of manual errors in transaction creation and address management. They also allow for the implementation of sophisticated transaction rules that can enhance the overall security of the funds managed by the application.

The following scripts are integrated in the first version of the system:

E224 263C 11A5 83D1 96C4 F33E EADF 9B89 0C5E F5B9

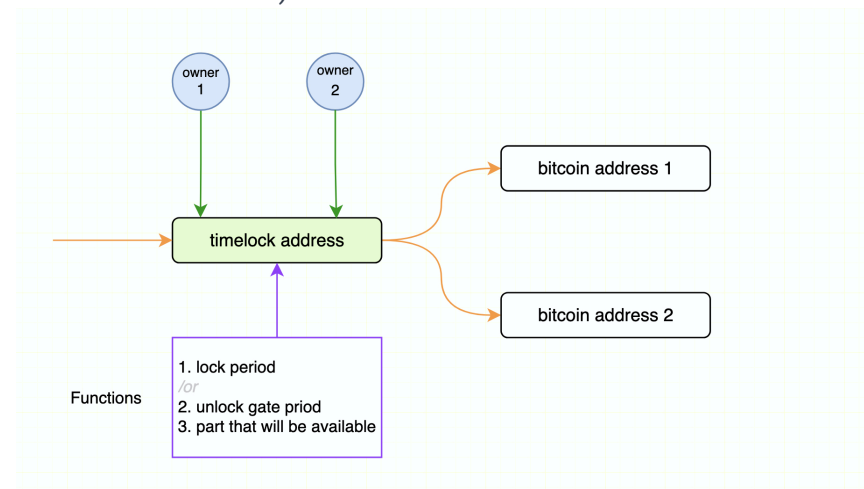
SC1 - Split

This is a script to automatically distribute bitcoin from one (income) address to two others.



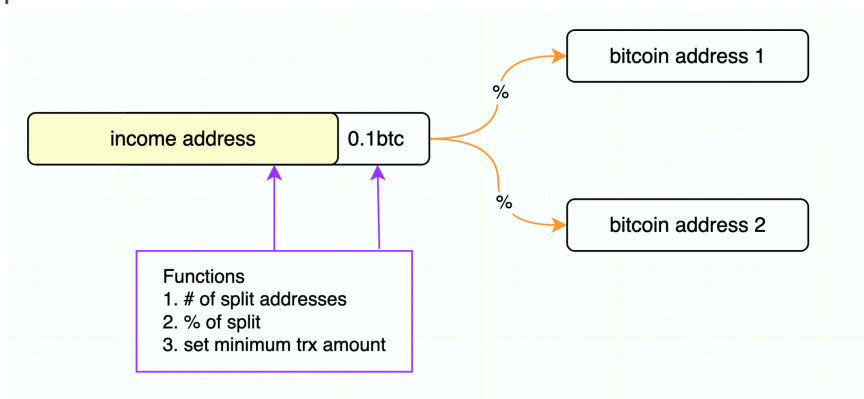
SC2 - Timelock

This script blocks the address in time. there are two formats:- blocking until a certain time (the end date of the freeze is set)- permanent blocking with defrosting periods (a period in time with dates from and to is set)



SC3 - Dam trx balance

This is a script to automatically distribute bitcoin from one (income) address to two others, when the required balance is reached



Security plan

Robust Encryption Protocols: At the heart of BoB's security is the implementation of advanced encryption methods. Data at rest and in transit is protected through industry-standard encryption techniques. This includes the use of TLS (Transport Layer Security) protocols to secure all communication between the client and server. Sensitive data, particularly private keys, are encrypted using state-of-the-art algorithms, ensuring they remain impervious to unauthorized access.

Comprehensive Key Management Strategy: Key management is a critical aspect of BoB's security framework. The system is designed to ensure private keys are never exposed over the network. Key storage is handled with utmost care, employing secure, encrypted storage solutions. For heightened security, the integration with Hardware Security Modules (HSMs) is considered. HSMs provide an added layer of security, safeguarding the cryptographic keys and managing key generation, encryption/decryption, authentication, and digital signing processes in a tamper-resistant hardware device.

Adherence to Security Best Practices: BoB's security architecture is underpinned by a commitment to best practices in the industry. This includes regular security audits conducted to identify and rectify vulnerabilities. A robust access control system is in place to ensure only authorized personnel have access to sensitive operations and data. The software components are regularly updated to address new threats and vulnerabilities as they emerge. Additionally, a significant emphasis is placed on user education, promoting a culture of security awareness and hygiene among all users.

Acceptance Criteria

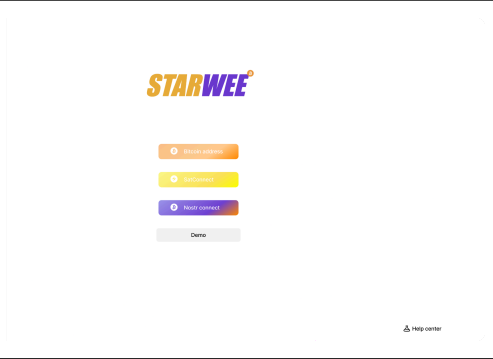
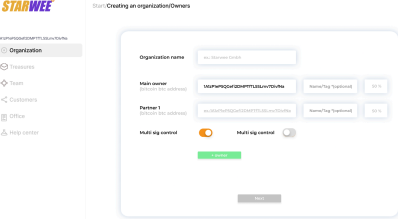
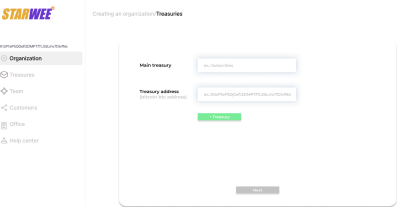
| Process | Status | Notes |
|---------------|---------------|---|
| Functionality | in. process ▾ | All features, such as wallet management, transaction scripting, and multi-signature processes, must work as intended without errors |
| Performance | in. process ▾ | The application must meet predefined benchmarks for speed and responsiveness |

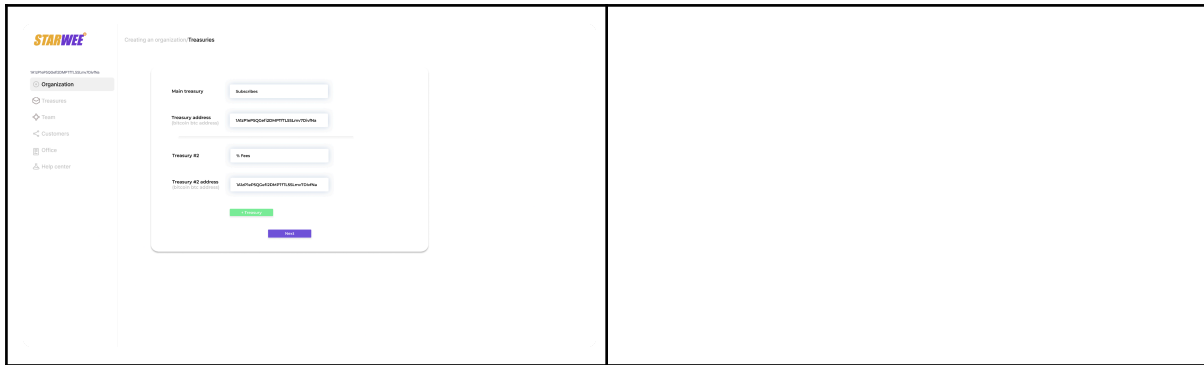
| Process | Status | Notes |
|-----------------|---------------|--|
| Security | in. process ▾ | The application must pass security audits, demonstrating robust encryption, secure key management, and adherence to best practices |
| User Experience | in. process ▾ | The application should align with the designed user flow and wireframes, offering an intuitive and responsive interface |
| Testing | waiting ▾ | All testing phases (unit, integration, acceptance) must be successfully completed with no critical bugs |
| Deployment | waiting ▾ | Successful deployment in the production environment with essential functionalities operational |
| Documentation | waiting ▾ | Comprehensive and clear documentation for users and administrators must be provided |

UI UX Description / System Interface

Creating an organisation

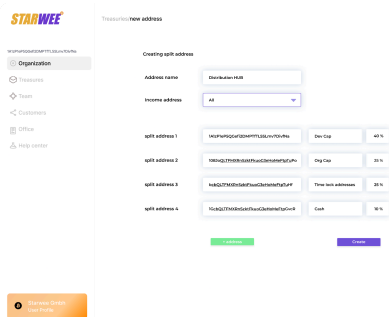
| | |
|------------------------|--|
| 1. Connect page | <p>There are several ways to join the system:</p> <ol style="list-style-type: none"> 1. the field for entering the btc address 2. the "wallet/sat connect" button [when pressed, satconnect appears]. |
|------------------------|--|

| | |
|--|---|
|  | <p>3. the "nostr connect" button [when pressed, nip connect appears].</p> <p>footer:</p> <ul style="list-style-type: none"> "help" button [when pressed, a submenu appears (dropdown)] <ul style="list-style-type: none"> - FAQ - Guides / Instruction - Online chat support |
| <p>2. Creating and configuring the system. Server</p> | <p>select one of the server options:</p> <ul style="list-style-type: none"> - own node (connect) [fields for entering your node data appear]. - Starwee node centre/server - demo server - Starwee Node coming soon |
| <p>3. Create and configure. Owners</p>  | <p>add an owner or signatory (якщо треба):</p> <ul style="list-style-type: none"> - add ourselves - a field for entering the bitcoin address (the btc address is already entered if it was specified during the connection) - "add owner" button [after clicking it, a new field for entering the bitcoin address of another owner appears] <p><i>*The system does not limit the number of owners</i></p> <ul style="list-style-type: none"> - Button "add global M2S (multi signature) which divides votes into ALL non-automatic transactions, including the transaction of creating an organization) - Button "help center" |
| <p>4. Create and configure. Main treasuries</p>  | <p>first, you already have the first department,</p> <ul style="list-style-type: none"> - you need to set the main btc address of the department and - its name/tag <p>it is possible to add a new department</p> <ul style="list-style-type: none"> - added by clicking on the "new" button <p>fill in the data in all input (main) treasuries to complete the creation of the organization.</p> <p>after clicking on the FINISH/NEXT button, you will be taken to the main page. Now you can use the wallet to receive payments in bitcoin</p> |



Organization management

5. Script development. Split



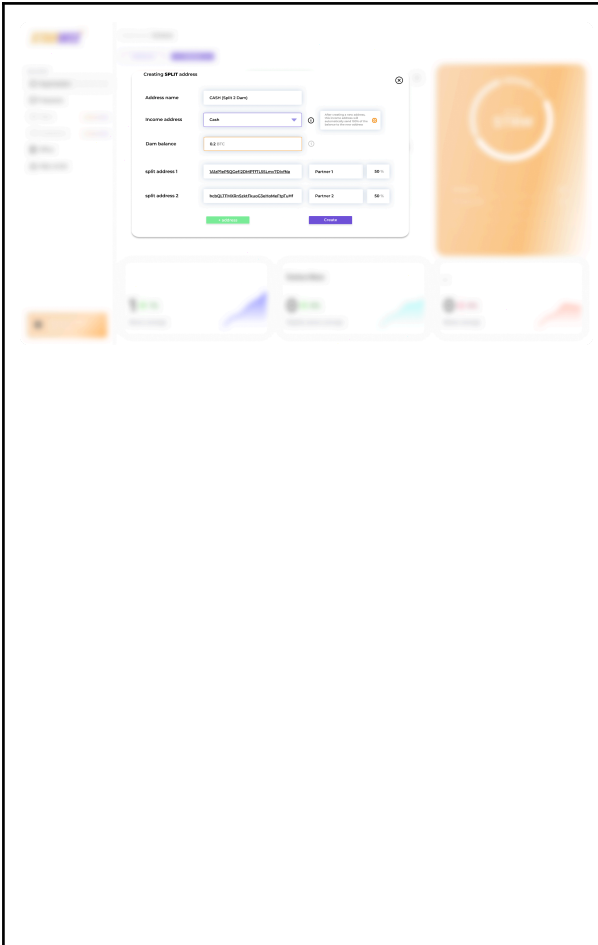
Create a script for distributing funds from the main address:

1. Address name
2. Dropdown menu - Selecting the address from which to send funds to the newly created address (you can choose one of the available ones, and when it is the first internal address of the organization, there is also an option to select all front addresses at once)
3. New address:
 - BTC address
 - name or tag
 - % of distribution
4. Button "create new address" after clicking it, another new address for distribution appears
5. Button "create" after pressing the system creates a new address on the binds all the settings, income addresses are now programmed and automated.
6. The Multisignature option allows you to increase the security of the path address by adding a multisignature with multiple keys. You can select specific keys to sign the actions of this address

6. Script development. Split Dam

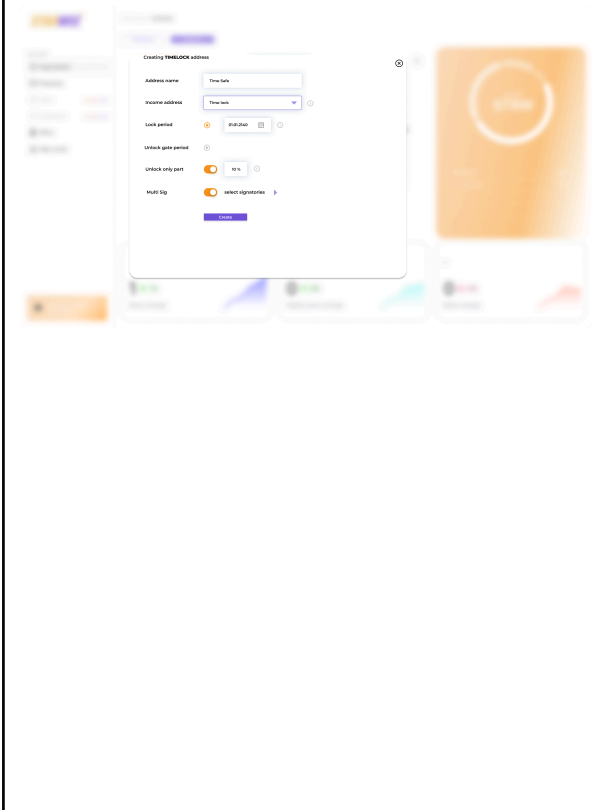
Creating a distribution script with a fixed balance. This script distributes funds from its address to other programmed addresses, but only after reaching a certain balance

1. New address name



2. Dropdown menu - Selecting the address from which to send funds to the newly created address (the selected input address will change its format from "standard" to "automated" and will send 100% of its balance to this newly created address)
3. Amount (dam) as a limit for sending that must be reached for the script to work
4. New address:
 - BTC address
 - name or tag
 - % of distribution
5. Button "create new address" after clicking it, another new address for distribution appears
6. Button "create" after pressing the system creates a new address
7. The Multisignature option allows you to increase the security of the path address by adding a multisignature with multiple keys. You can select specific keys to sign the actions of this address.

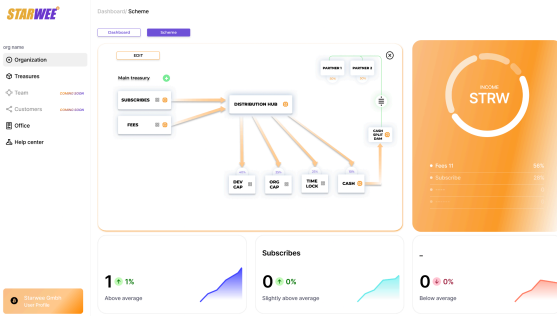
7. Script development. **Timelock**




Create a script to block an address in time.
 This script blocks the address and all transactions from it until the programmed time.

1. Name of the new address
 2. Dropdown menu - Selecting the address from which to send funds to the newly created address (the selected input address will change its format from "standard" to "automated" and will send 100% of its balance to this newly created address)
- Selection:
- Blocking period (select the end date and time of the blocking)
 - Unblock at a specified time (here you need to select the time range in which the address will be unblocked)
3. The "Unblock only part" option allows you to select the % of the address balance that will be

| | |
|--|--|
| | <p>unblocked and available for withdrawal/transaction</p> <p>4. The Multisignature option allows you to increase the security of the path address by adding a multisignature with multiple keys. You can select specific keys to sign the actions of this address</p> <p>Button "create" after pressing the system creates a new address</p> |
|--|--|

| | |
|---|---|
| <p>8. Organization scheme</p>  | <p>Main screen. Organization chart. Here you can see the structure of addresses and their connections that you have created.</p> <ol style="list-style-type: none"> 1. Button switch "Dashboard/Scheme" - when pressed, you can switch screens between the dashboard and the scheme 2. The "Edit" button switches 3. Address block: <ul style="list-style-type: none"> - address name - Structures of the scheme of interaction with other addresses - Button icon - view the address as a wallet - Itatus icon - appears when the address becomes automated |
|---|---|

| | |
|--|---|
| <p>8. Organization scheme editing</p>  | <p>Edit the organization's address scheme</p> <p>Screen to the editing state, where you can:</p> <ul style="list-style-type: none"> - add main incoming address - add internal address - add a partner - delete an address (may change the structure if it is linked to other addresses) |
|--|---|

| | |
|---------------------------------|--|
| <p>8. Main dashboard</p> | <p>Home screen. organization's dashboard.</p> <p>are displayed here:</p> <ol style="list-style-type: none"> 1. all wallets (final, non-automated |
|---------------------------------|--|

The screenshot shows a dashboard for 'STARWEE' with a user ID of 834777. The interface includes a sidebar with navigation options: Organization, Treasurers, Team, Customers, Office, and Help center. The main content area is divided into several sections:

- WALLETS:** A table listing wallets with columns for name, balance, and percentage.

| Wallet Name | Balance | Percentage |
|-------------|-------------|------------|
| REV CAP | 0.18483 BTC | 12% |
| ORG CAP | 0.29493 BTC | 12% |
| TIME LOCK | 0.18483 BTC | 12% |
| CASH | 0.18483 BTC | 12% |
- TEAM:** A list of team members with roles and counts.

| Role | Count |
|----------|-----------|
| INITIATA | 5 members |
| INITIATA | 3 members |
| INITIATA | 2 members |
| INITIATA | 1 member |
- INCOME STRW:** A circular progress indicator for income.
- Subscribers:** Three small charts showing subscriber trends with labels like '1 @ 1%', '0 @ 0%', and '0 @ 0%'.

On the right side of the dashboard, there is a list of items:

- name
- BTC balance
- percentage of balance growth
- button to open the wallet

Below the dashboard, there is a list of items:

2. Dashboard of personal income statements
3. *Team*
4. *Customer*

These dashboards are still under development

Project links:

Github: <https://github.com/Bitcoin-Based/bitcoin-organization-builder>

Nostr: <https://primal.net/p/npub1ejn6vu9rkqyqldq4ufn6sdhwyyua97uuglkk8ekpa0xx4wdkvqksdvdst0>

Telegram: @tetakta

E224 263C 11A5 83D1 96C4 F33E EADF 9B89 0C5E F5B9

References

[1] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System., 2008.

[2] Tetakta. Bitcoin-based Organizations.