WalleTx

Software Research Specification

Prestige Worldwide

Group Members:
Arron Solano
Brian Howell
Daniel Carroll
Michael Danko

Contents

1 Introduction

1.1 Purpose

Bitcoin WalleTx (WalleTx) is a bitcoin wallet tracker tool for Android that assists users in monitoring their bitcoin balance, transaction history, and spending trends across multiple wallets. It is common for bitcoin users to possess numerous wallets, yet there do not exist many services that are capable of aggregating wallet information in order to provide the user with an overall picture of their bitcoin funds. Bitcoin WalleTx solves this problem by allowing users to group or categorize their bitcoin wallets, as well as tag their transactions with real-world information related to how their bitcoins are being spent. Charts and graphs help to identify trends in both single wallet spending and across wallet groups, a useful feature that is essential for enabling users to integrate their bitcoin finances with a more traditional budget.

The purpose of this document is to define the requirements for Bitcoin WalleTx. The intended audience of this document includes the CEN4020 instructional staff at FSU and the end users of the WalleTx. It is also intended that this document serve as the principle point of reference for the Prestige Worldwide team throughout the development of WalleTx.

1.2 Scope

1.2.1 Product

Bitcoin WalleTx (referred to as WalleTx)

1.2.2 Scope of Work

WalleTx is an android application that enables users to import their bitcoin wallets by public key and tag their bitcoin transactions in order to obtain an aggregated overview of their bitcoin finances. WalleTx shall allow users to manage their public keys; manage wallet groups; manage transaction tags; tag individual transactions; and receive feedback regarding trends via charts and graphs. A full listing of functional requirements for WalleTx is located in the Functional Requirements section of this document.

1.2.3 Out of Scope

WalleTx shall not provide any functionality allowing users to spend or receive bitcoins since only public keys shall be stored on the device of the user.

1.2.4 Application

Bitcoin is a new technology and the ecosystem is currently undergoing major infrastructure development. There currently do not exist any tools that allow users to label their bitcoin transactions in order to identify trends in their bitcoin spending behavior. WalleTx aims to bring this functionality to the Bitcoin space.

1.3 Definitions

- **WalleTx** - Shorthand for Bitcoin WalleTx
- **Bitcoin** - a tradeable virtual asset existing on the ledgers of a distributed and synchronized network of ledgers
- **bitcoin** - a tradeable virtual asset unit, divisible into 100 million units (also known as a satoshi)
- **Blockchain** - A block chain is made of a blocks, linked to the block before and block after, that contain information pertinent to the unerlying system. In the case of Bitcoin these blocks contain transaction information.
- **Public key** - A hash of a wallet's public key that allows other users on the Bitcoin network to send bitcoins to the Public Key.
- **Tx** - A transaction that moves bitcoins from one address to another, usually in exchange for goods or services.
- tag?
- Master Public Key

1.4 Acronyms

BTC Common unit of Bitcoin currency

1.5 References

- Bitcoin: A Peer-to-Peer Electronic Cash System
  `https://bitcoin.org/bitcoin.pdf`
- Blockchain Data API
  `https://blockchain.info/api/blockchain_api`

1.6 Overview

2 General Description

2.1 Product Perspective

2.2 Product Functions

2.3 User Characteristics

2.4 General Constraints

2.5 Assumptions and Dependencies

3 Functional Requirements

3.1 The system shall allow users to manage their imported wallets or public keys

3.1.1. Users shall be able to add new single address wallets into the system

3.1.2. Users shall be able to edit previously existing wallets

3.1.3. Users shall be able to delete existing wallets from the system

3.1.4. Users shall be able to name individual wallets and assign them to a group

3.1.5. The system should be designed to allow for future integration with other wallet types such as deterministic seed wallets or service based wallets such as Coinbase and Circle

3.2 The system shall allow users to customize wallet groups

3.2.1. System shall designate a default wallet group

3.2.2. Users shall be able to add new wallet groups by name

3.2.3. Users shall be able to edit previously existing group names

3.2.4. Users shall be able to delete existing wallet groups, an action that should reassign all wallets associated with the group to the default group

3.3 The system shall allow users to manage transaction tag categories

3.3.1. User shall be able to define new tag categories by name

3.3.2. User shall be able to edit existing tag categories

3.3.3. User shall be able to delete existing tag categories, an action that should remove the tag from any transactions it has been assigned to

3.4 The system shall allow users to manage tags for individual transactions

3.4.1. User shall be able to add a tag to an individual transaction

3.4.2. User shall be able to remove a tag from an individual transaction

3.5 System shall provide an aggregate view of all imported wallets that contains the total balance and number of transactions

3.6 System shall provide a summary view for each wallet group that provides an overview of the group balance, transactions, and charts

3.7 System shall provide a summary view for individual wallets that provides an overall summary of its balance, transactions, and charts

3.8 System shall provide a transactions detail view for each wallet group that lists the transactions associated with the group

3.9 System shall provide a transactions detail view for each individual wallet that lists its transactions

3.10 System shall provide graphical charts for identifying transaction trends

3.10.1. System shall provide pie charts views breaking down transaction percentage by tag for all wallet groups and for individual wallets

3.10.2. System should provide additional trend identifying charts for wallet groups and individual wallets

3.11 The system should provide a view containing individual transaction information

3.12 The system should allow users to backup their data to an external file and import data from a previously created backup file

3.13 The system should allow users to passcode protect the application

4 Non-Functional Requirements

4.1 Query and reporting times - this will be dependent on the end users RAM and processor speed. Also database placement and architecture. Must account for moderate processor speeds and run on little as RAM as possible. Android Framework provides tools to test and profile CPU and RAM usage.i Querying the blockchain will be done via the Blockchain API and speed to return a query will be dependent on outside services and network connection speed.

4.2 Storage - this will be dependent on the end users storage memory on the phone. Must have small footprint on phone's storage. Need to compare application size for similar available applications to scope size. Records will be stored locally to prevent repeated connections ot the blockchain network for duplicatin information. Records and their added tagging information will probably be measured in hunders of bytes meaning that the storage footprint will be very reasonable.

4.3 Response time (between activities) - this will be dependent on the end users RAM and design of UI. Must account for low RAM and optimize UI for quick activity. This evaluation will fall under query and reporting time.

4.4 Screen Resolution - dependent on end users phone. Must find resolution to work on most phones. (font, layout tweaks, image changes)

4.5 Versions - Need to specify and target revision to reach optimal user targets. Evaluated by most popular revision available or with best compared resources.

4.6 No system downtime - All updates to the app will be done through Android updates pushed through Google Play Store. Standard update process for life of application on Android platform.

4.7 Battery Usage - this will be dependent on the end users battery size, installed applications, and screen size. Must try to minimize battery drain. Need to evaluate against controls to test battery usage.

5 System Architecture

6 System Model

7 Appendices

7.1 Data Dictionary

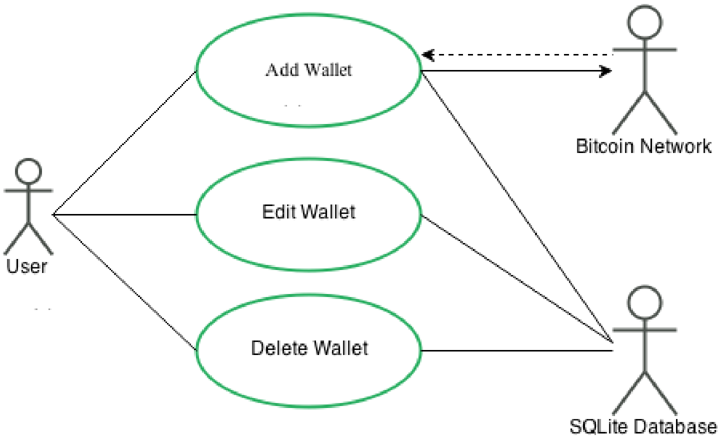
7.1.1 Actor Descriptions

Identifying actors:

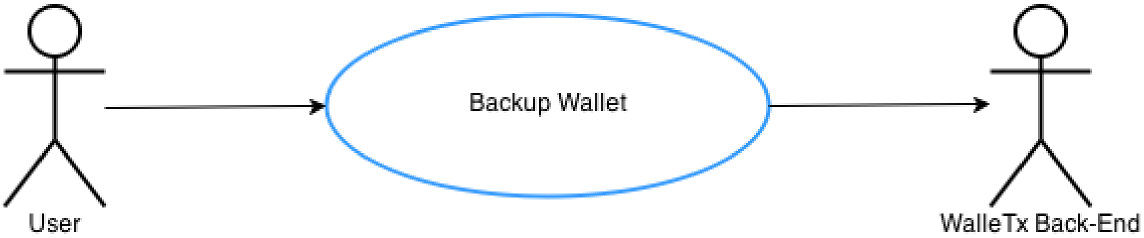
- Who will supply, use, or remove information from the system?
- Who will use the system?
- Who is interested in a certain feature or service provided by the system?
- Who will support and maintain the system?
- What are the system's external resources?
- What other systems will need to interact with the system under development?

Actors will be the customer/user of the application. Most interested in our application would be those interested in Bitcoin. Service and maintenance of the system will be performed by the developers(Prestige Worldwide). Systems external resources will be blockchain, SQLite, and other external wallets providing information. (Michael and Brian please confirm external resources of Bitcoin)

7.1.2 Use Case Descriptions



WalleTx: Managing User Wallets	
Actors	User
Description	A user will be able to backup all wallet information, transaction hashes, and tags associate with those transactions to an XML or JSON file.
Data	public wallet identification, transaction hashes, tags associate with transactions
Stimulus	User initiates a data backup
Response	Dialog box will confirm backup
Comments	Is there any additional data that needs to be backed up? Where will the database backup be stored?



WalleTx: Backup Data	
Actors	User
Description	A user will be able to backup all wallet information, transaction hashes, and tags associate with those transactions to an XML or JSON file.
Data	public wallet identification, transaction hashes, tags associate with transactions
Stimulus	User initiates a data backup
Response	Dialog box will confirm backup
Comments	Is there any additional data that needs to be backed up? Where will the database backup be stored?

Identifying use cases:

- What are the goals that the actor will attempt to accomplish with the system?
- What are the primary tasks that the actor wants the system to perform?
- Will the actor create, store, change, remove, or read data in the system?
- Will the actor need to inform the system about sudden external changes?
- Does the actor need to be informed about certain occurrences, such as unavailability of a network resource, in the system?
- Will the actor perform a system startup or shutdown?

  Use cases will be outlined in diagrams provided.

The primary task of the customer/user will be to interact with the application. Their main purpose will be to monitor their bitcoin currency from multiple wallets and tag transactions to better organize all transactions. The actor will have the ability to add and remove wallets/add and remove tags pertaining to transactions. The actor can observe aggregate information in the form of graphs, charts, etc. The actor will be alerted to certain trigger events. (updated wallet info, specific transactions, etc). The actor has the ability to close and open application in running OS and also has the ability to completely remove the application from existing OS.

7.1.3 Class Descriptions

7.1.4 Attribute Descriptions

7.2 Raw Use Case Point Analysis

7.2.1 Actor Summary Table

Actors	Summary
Customer/End User	This actor will be the end user. They will interact with WalleTx, providing it with wallet information, tagging transactions, and viewing aggrated wallet information
Development Team/Maintenance	This actor will be interacting via the support and maintance of the application. They will maintain DB orginzation, revision handling, and overall maintaince of the application
External Bitcoin Wallets	External Bitcoin Wallets will be providing our application with vital information. All external wallets' totals and transactions will aggreated in WalleTx.
SQLite Database	This actor is a database stored on each end user's phone. This will contain data from wallets, transactions, and display information. It will interact soley with the application via code.
Bitcoin Blockchain	The bitcoin blocktrain will provide transaction specific data for all included wallets. This actor will interact with the application via database backend.

7.2.2 Use Case Summary Table

7.2.3 Screens and Reports with Navigation Matrix

7.3 Other Appendices