Decentralised Application Project

Team: Yihan Xiao

Hao Chen

Chenwei Qian

Li Yu

Abstract: a decentralised shopping application cart supports online business operations by using a blockchain to manage data. All trades happen directly between buyers and sellers with no middleman to take a cut from each sale.

Background/Problem Domain:

Background

The concept of decentralized digital currency, as well as alternative applications like property registries, has been around for decades. The anonymous e-cash protocols of the 1980s and the 1990s, mostly reliant on a cryptographic primitive known as Chaumian blinding, provided a currency with a high degree of privacy, but the protocols largely failed to gain traction because of their reliance on a centralized intermediary. In 1998, Wei Dai's bmoney became the first proposal to introduce the idea of creating money through solving computational puzzles as well as decentralized consensus, but the proposal was scant on details as to how decentralized consensus could actually be implemented.

In 2005, Hal Finney introduced a concept of "reusable proofs of work", a system which uses ideas from b-money together with Adam Back's computationally difficult Hashcash puzzles to create a concept for a cryptocurrency, but once again fell short of the ideal by relying on trusted computing as a backend. In 2009, a decentralized currency was for the first time implemented in practice by Satoshi Nakamoto, combining established primitives for managing ownership through public key cryptography with a consensus algorithm for keeping track of who owns coins, known as "proof of work".

The mechanism behind proof of work was a breakthrough in the space because it simultaneously solve two problems：

First, it provided a simple and moderately effective consensus algorithm, allowing nodes in the network to collectively agree on a set of canonical updates to the state of the Bitcoin ledger.

Second, it provided a mechanism for allowing free entry into the consensus process, solving the political problem of deciding who gets to influence the consensus, while simultaneously preventing sybil attacks. So here draws our idea of making a decentralised shopping application cart to support online business operations by using a blockchain to manage data.

Existing system and drawbacks

Excessive commission fees, privacy violations, frauds and censorship are common issues encountered during daily interactions with centralized online shopping service providers. Given these well-recognized problems with centralization, a wide range of blockchain technologies attempting to resolve these issues have been developed since the launch of Bitcoin in 2008. Specialized projects like SteemIt, Bitshares, and Syscoin, as well as more versatile projects like Ethereum and EOS, are some of many examples. However, most of those attempts are either too specialized in certain applications, or burdened by low transaction throughput. Due to these limitations in flexibility and transaction throughput, it is impossible for developers and enterprises to bring heavy services like Facebook or Amazon onto the blockchain- not to mention something more complicated like digital asset exchanges.

These problems are also drawbacks of the existing system for online shopping and trading. However, we could use blockchain as the fundamental technologies to prevent these problems, and we choose Ethereum to build this shopping cart. The design and concept of Ethereum protocol provides for a platform with unique potential; rather than being a closed-ended, single-purpose protocol intended for a specific array of applications in data storage, gambling or finance, Ethereum is open-ended by design, and we believe that it is extremely well-suited to serving as a foundational layer for our shopping cart development.

Aims:

The aim of this project is to build a functional distributed shopping website using blockchain technology. Instead of building our own blockchain from nothing, we will build the website based on ethereum, a widely-used blockchain-based platform, which can handle most of foundational problem. We are going to focus on implementing our own smart contract, data storage and user interface.

The deliverables of this project include:

Smart contract
The smart contract is kernel of our system. It handles the operation between the blockchain and user operation. Also it defines business logic and algorithm used to make the shopping website work at the backend. Clients authority control is also handle by this part to keep user's privacy and data safety.

Data storage
Data storage is made from two separate part. Public information like trading history and product information is stored on blockchain so that once data is stored in it will be unchangeable. Another part is private data like user profile and shopping cart, those data need to be modified frequently, those will be stored in a database on AWS.

Shopping system
Shopping system is the interface that user interact with the website. It includes user registration, profile management, purchase system and so on. Products catalogue and filter can help customer easily find the item they needed. All features above aims to make the website more user-friendly and enhance clients' shopping experience.

Epics:

The project is consisted of three main epics, the smart contract module, web interface module with middle-layer for connecting smart contract, and the local database module.

Smart Contract Module(Medium)

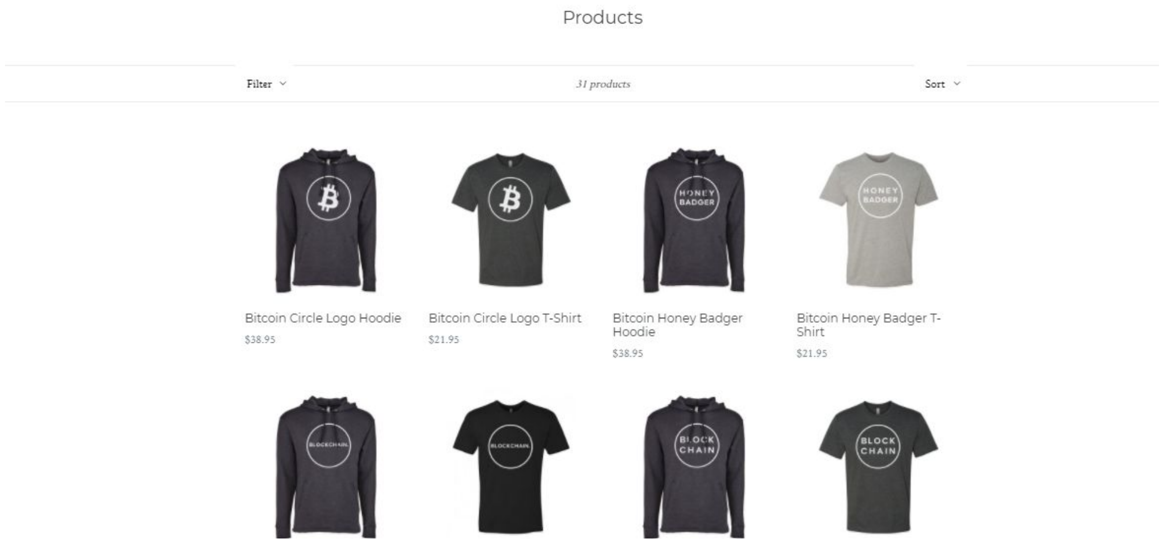
This module is for data transmitting and storing, which is implemented with the interface provided by Ethereum. Through sending necessary data to the public blockchain of

Ethereum, information will be permanently solidated in the blockchain which assures the safety of data. The module will implement the basic objects that needs to be stored in Ethereum, including all essential functions for manipulating the object such as initialize(), get(), set(), update(), etc.

The language for implementation is Solidity. For connecting Ethereum, we use MetaMask, which is a plug-in for browsers, as a gateway between smart contract module and Ethereum.

Middle-layer Module(Small)
This module is for transmitting information between smart contract built in Ethereum and web interface module which is for displaying. The object taken out from blockchain is not ready-to-use, which requires an extra module to extract and unpack from blockchain as well as pack and send to blockchain.

Web-interface Module(Large)
Web module is just as what we usually see from a normal shopping website. It has basic displaying outlayer for placing merchandises of different categories, showing their names and prices respectively and a hyperlink which leads to a new page of details for each of them. An example can be like the snapshot below.

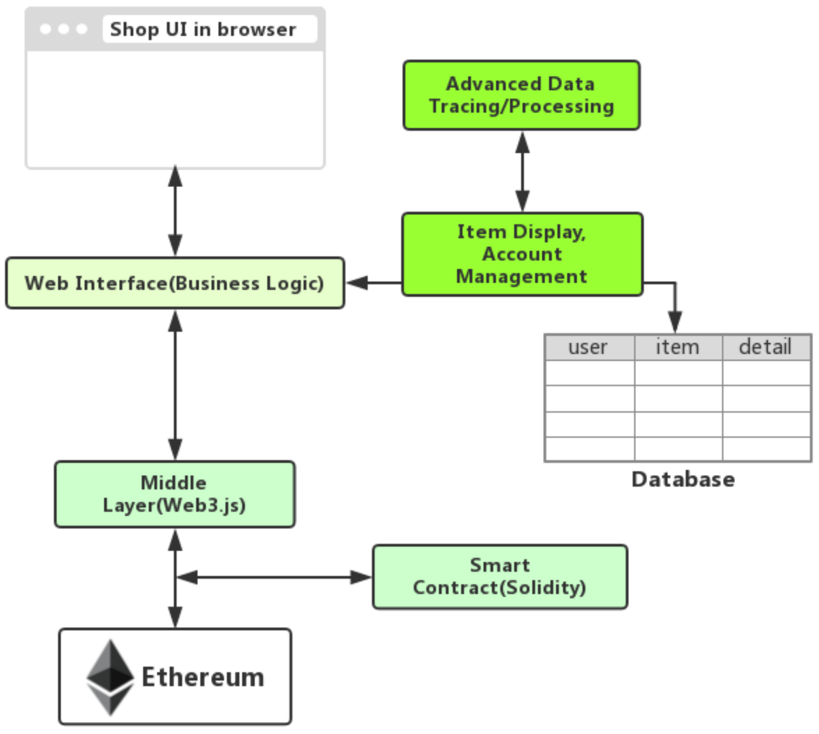

Products
Filter
31 products
Sort
Bitcoin Circle Logo Hoodie
$38.95
Bitcoin Circle Logo T-Shirt
$21.95
Bitcoin Honey Badger Hoodie
$38.95
Bitcoin Honey Badger T-Shirt
$21.95

Database Module(Small)
This module is for storing all necessary data in local database. Since interacting with blockchain is somehow expensive(every operation cost a certain amount of ‘gas’ which is a form of cryptocurrency of Ethereum) and time-consuming(every operation cost more than 10 seconds in Ethereum). We only store data that is essential on Ethereum, such as the record of purchase and transaction. Other data like pictures of items and account details will be stored in local database.

Account Management(Large)
Either buyer or seller can have access to management of their own accounts, which includes creating, updating, deleting(deleting is rather sophisticated because a user shall not be able to delete account if there are pending transactions). A seller will be able to upload new items or unload old items. A buyer will be able to check the shopping cargo and purchase history.

Advanced Data Tracing/Processing(Large Optional)
Buyer can search for the commodity with several useful options like sorting items with prices and filtering items with certain conditions. It is also possible for buyers to look up the selling history of a specific seller, and the change of prices. All other data processing functions will be implemented in this module if applicable.

The conceptual flow chart:


Shop UI in browser
Advanced Data Tracing/Processing
Item Display, Account Management
Web Interface(Business Logic)
user
item
detail
Database
Middle Layer(Web3.js)
Smart Contract(Solidity)
Ethereum

Project Methodology:

The project team is a group of four, with Product Owner and Scrum Master roles within the group members throughout the project duration.

IDE: Sublime 3
Code management: BitBucket
Programming language: Solidity, web3.js, HTML, CSS, JavaScript, SQL
SCRUM: Trello

Project Scrum
Scrum team:

Yihan Xiao	z5099956@ad.unsw.edu.au
Hao Chen	z5102446@ad.unsw.edu.au
Chenwei Qian	z5107753@ad.unsw.edu.au
Li Yu	z3492782@ad.unsw.edu.au

Scrum master: Li Yu
Developer: All scrum team
Sprint Review Meeting: 9:00 - 11:00 Tuesday
Sprint Retrospective Meeting 10:00 - 12:00 Friday

Project deliverables includes fortnightly project progression demos, with a final project demonstration and report on the 22nd of May. More details on the schedule are shown below.

Project Time-frame/Schedule:

Friday 16th March:	Project Proposal Release	(Week 3)
Tuesday 20th March:	Progress Report	(Week 4)
Tuesday 10th April:	Progress Report	(Week 6)
Tuesday 24th April:	Progress Report	(Week 8)
Tuesday 8th May:	Progress Report	(Week 10)
Tuesday 15th May:	Project Improvement	(Week 11)
Tuesday 22nd May:	Project Demo & Report	(Week 12)
Tuesday 22nd May:	Software Quality	
Tuesday 22nd May:	Peer Assessment	