VeChain “CrowdSale” DApp

About CrowdSale:

CrowdSale is a decentralized marketplace for users to make online purchases of non-physical products. Being decentralized, CrowdSale records all transaction history on the **VeChain** blockchain network. Cryptocurrency is used for transfer of funds, without the hassle of credit cards.

In a recent paper by MIT technology review, (https://www.technologyreview.com/s/613135/blockchain-powered-marketplaces-will-be-like-steroids-for-crowd-investing/?fbclid=IwAR2AVwY0c9M1o9gQuvYzlK_VB8fBGm1QATh8YBmMhe81d0vW1dEsCeEat0w) the strong potential of marketplaces and crowd concepts with blockchain applications were highlighted.

CrowdSale’s vision is to be the go-to marketplace for online assets sales. We plan to onboard products from many categories, including media, software keys, e-books, news, freelance services, and many more.

Features of CrowdSale: CrowdSale is a beautifully designed marketplace that provides the option to sell and also to auction products. On CrowdSale, users can easily view all buy and bid history for each product recorded securely on blockchain and make trusted decisions.

In the near future, CrowdSale will take advantage of VeChain's multi-party-payment system to allow either buyer/seller to pay the transaction fees.

CrowdSale’s Seller Traceability You can see a Seller’s history easily with the assurance that it has not been doctored. Furthermore, all sellers must first be approved by the CrowdSale Admin before they can begin selling their products. This adds an additional layer of confidence for the user.

CrowdSale’s Auction Feature The auction seller puts up "x" number of his/her product for sale. Bidders bid for the particular product, and each bid is recorded on the blockchain. Take note that when submitting a bid, the bidder will need to pay the respective number of tokens. When the number of bids equals or exceeds the total number of products available, the seller has the option to finish the Auction. Upon the seller's finish auction confirmation, CrowdSale will send the blockchain tokens of the successful bidders over to the seller, and return the blockchain tokens of the failed bidders back to them.

How CrowdSale Can Endorse the VeChain Ecosystem:

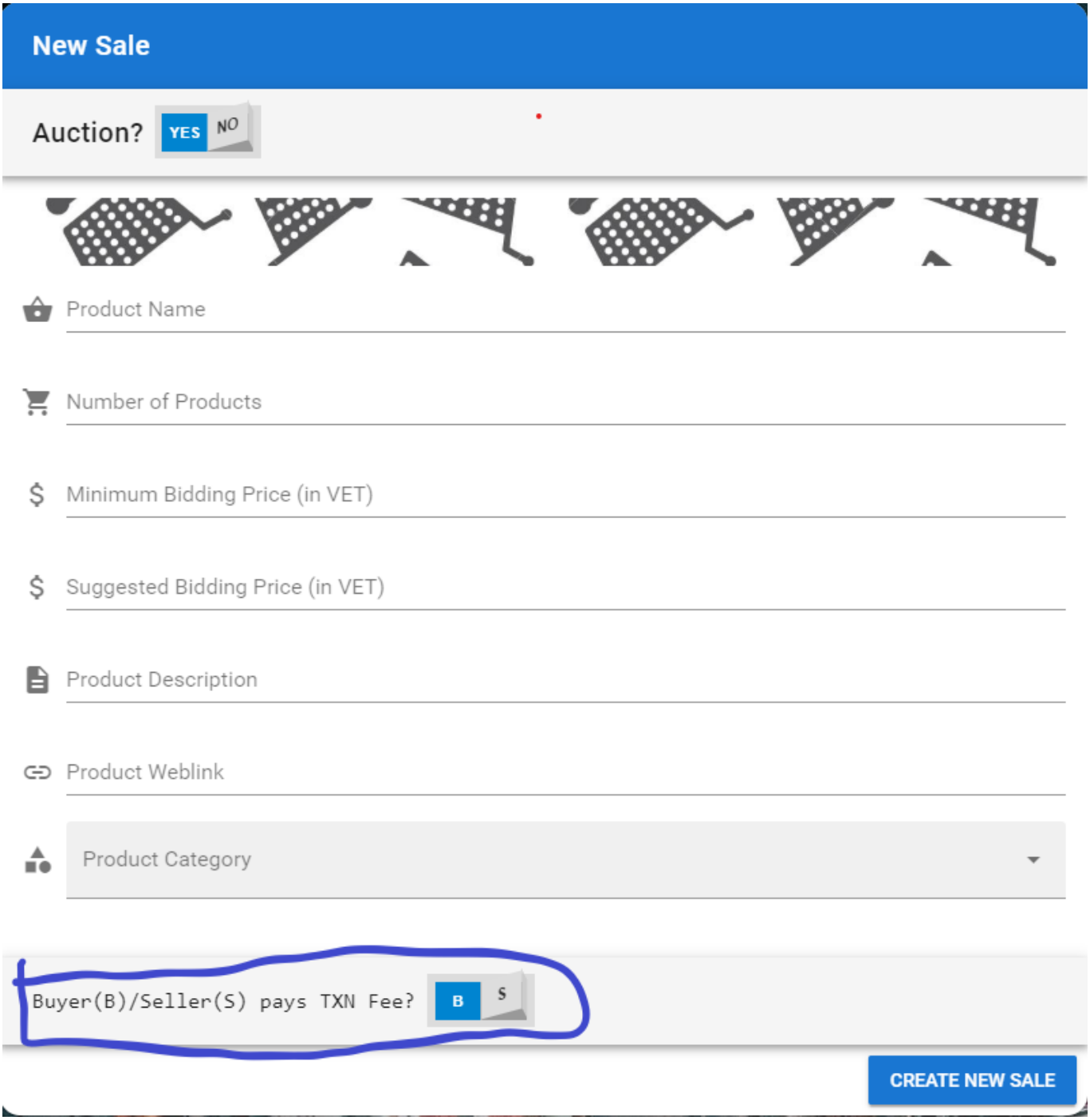
CrowdSale is a clear endorsement of the VeChain ecosystem in developing non-trivial decentralized applications. CrowdSale DApp has selected VeChain blockchain because of the many unique features the blockchain offers that is central to its required features.

CrowdSale is a traiblazer for VeChain in that it has adopted **Connex** as the middle layer, rather than the common web3.js front-end standard. This would allow CrowdSale to be able to use VeChain's unique features.

The Front-End of CrowdSale is developed using *Vue.js* framework, the rising star of front-end frameworks. The highly dynamic Vuex store is used, allowing for asynchronous viewing of products upon making new sale

and other data. While most implementation you find online regarding DApps are done with React.js, CrowdSale validates the compatibility of Vue.js framework with VeChain backend smart contracts. This will highly draw the ever-increasing pool of Vue.js frontend developers' interest in the Vechain environment and larger blockchain ecosystem.

The **Multi-party-payment feature** provided by VeChain shall be a key feature of CrowdSale.



When making a new sale, the buyer/seller can determine who to pay the transaction fee. In the future this will be further expanded to allow the seller to determine specific percentages of the transaction fee to be paid by the buyer/seller. This helps buyers to make better decisions when making purchases of products.

CrowdSale has specifically selected the VeChain blockchain also because of the unique **transaction dependency/multi-task transactions** features that is available on VeChain.

The multi-task transaction feature is important to CrowdSale because in the auction contract logic. We have an intermediate wallet which is for the product. A bidder submitting a bid (where he has to make a transaction on VeChain) should be independent from the seller finishing the auction. Furthermore, in the logic of finish auction, it is important that the multiple tasks of closing the auction, sending the money to the seller, and returning the money to failed bidders are done in the correct order. The VeChain model helps us to achieve this.

Technology Summary of CrowdSale:

This app is for crowd sale with both sale and auction functions leveraging VeChain blockchain.

The app consists of a Web App, APIs and Smart contracts. The Web App is implemented using Vue.JS framework, APIs are RESTful APIs using Node.js which integrates with smart contracts using VeChain APIs. The smart contracts are written using Solidity.

The Web App and APIs will be hosted on a trusted platform or a hosting platform, such as Morpheuslabs BPaaS, GoDaddy, and the smart contracts will be running on VeChain MainNet.

All amount, price and bids referred in this App is based on the blockchain native token. (VET in this case)

Core Technology Stacks

Vue.JS frontend

- Vue, Vuex, Vue Router
- Vuetify Material Component Framework
- Firebase web application development platform (User authentication and authorization)

Middle Client API

- Connex

Backend Smart Contracts and Blockchain Technology

- Solidity
- Truffle Framework v4.1.15
- VeChain web3-gear
- VeChain Thor
- VeChain Sync Browser

Technical Features of CrowdSale

CrowdSale smart contracts are written in Solidity and deployed on VeChain as backend infarstructure. And front applications can use Connex Client to interact with the smart contracts.

CrowdSale dApp supports two kinds of sales for now: normal sale and auction sale. In near future, more functions will be added.

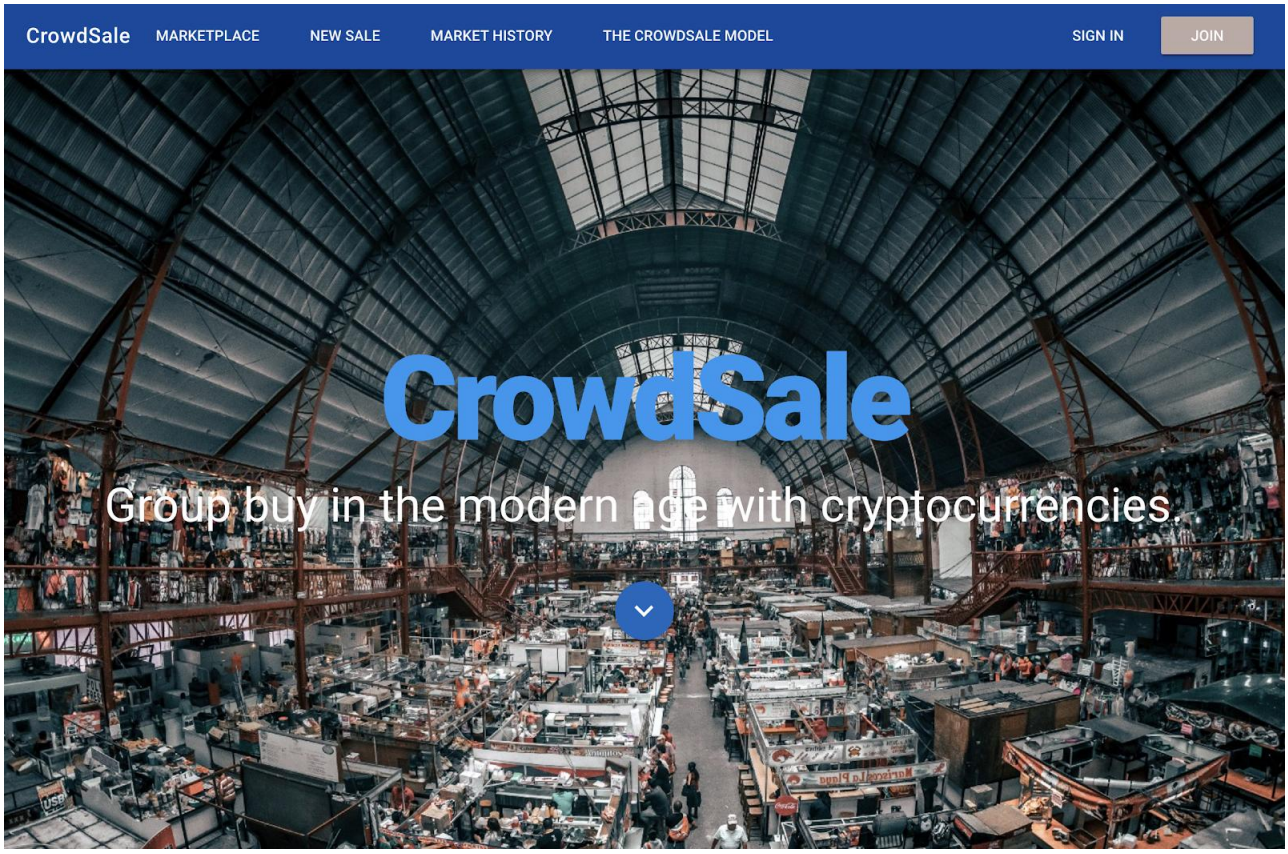
The followings are the main functions provide by the CrowdSale smart contracts (refer to ABI for detailed spec of smart contract functions).

- registerSeller: register a seller to be able to sell products on dApp
- getSellers: return list of sellers
- getAllProductList: get a list of all products
- getProductListOfSeller: get list of products of one seller
- createNewProduct: create a new product for sale (it requires seller permission)
- getBuyRequests: get all the buying requests of a product
- finishSale: finish sale of a product, then users are not able to buy the project any more, and tokens are released to the seller.
- createAuctionProduct: create an auction product
- getListBiders: get the list of all bidders
- finishAuction: settle auction sale, tokens will be refunded to people who did not win the auction.

UI/UX Features and Walkthrough

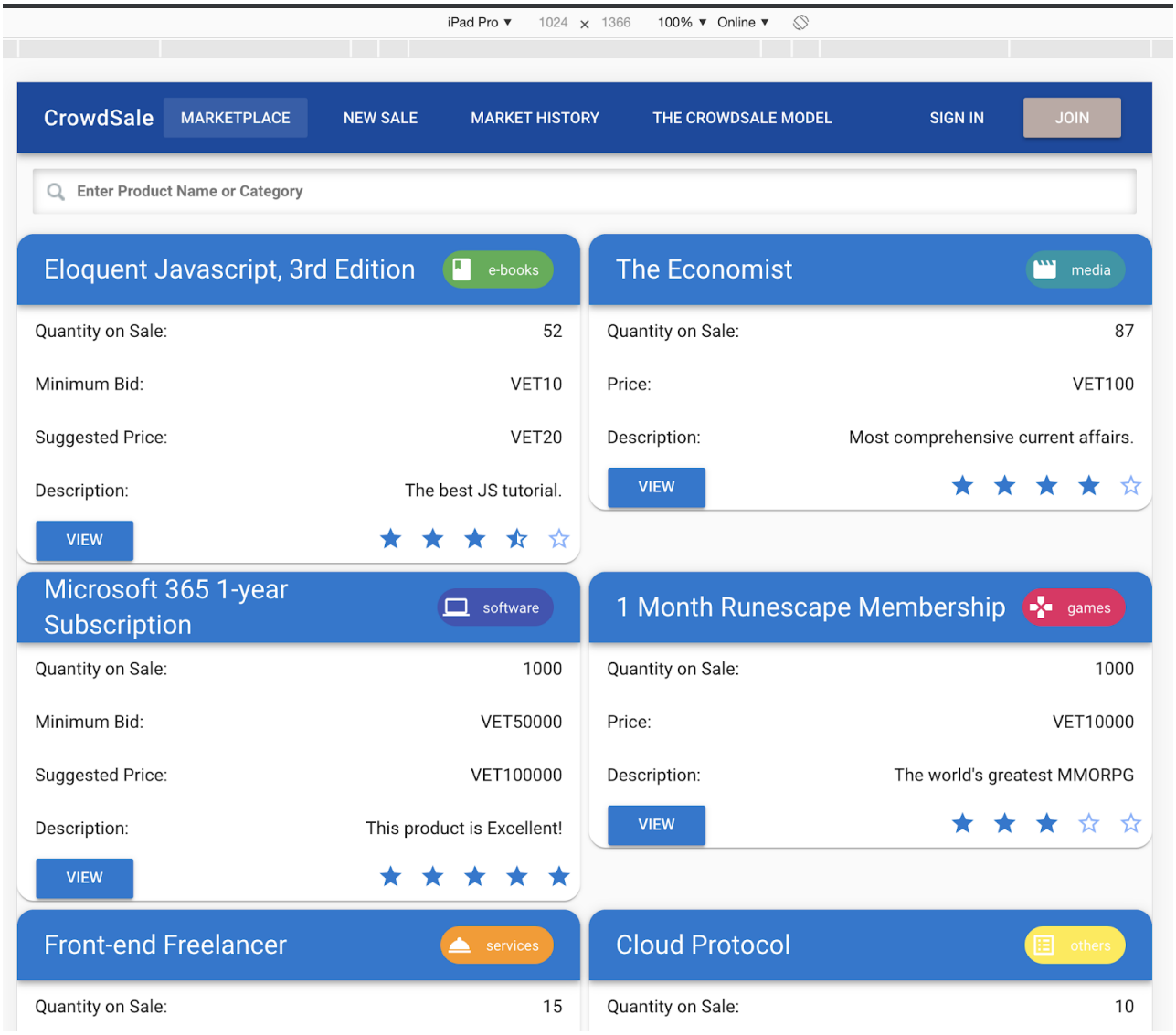
CrowdSale web application UI is configured to work well in mobile, tablet and computer displays. Below are some examples showing different pages on different displays.

- Home Page, on 15” computer display.



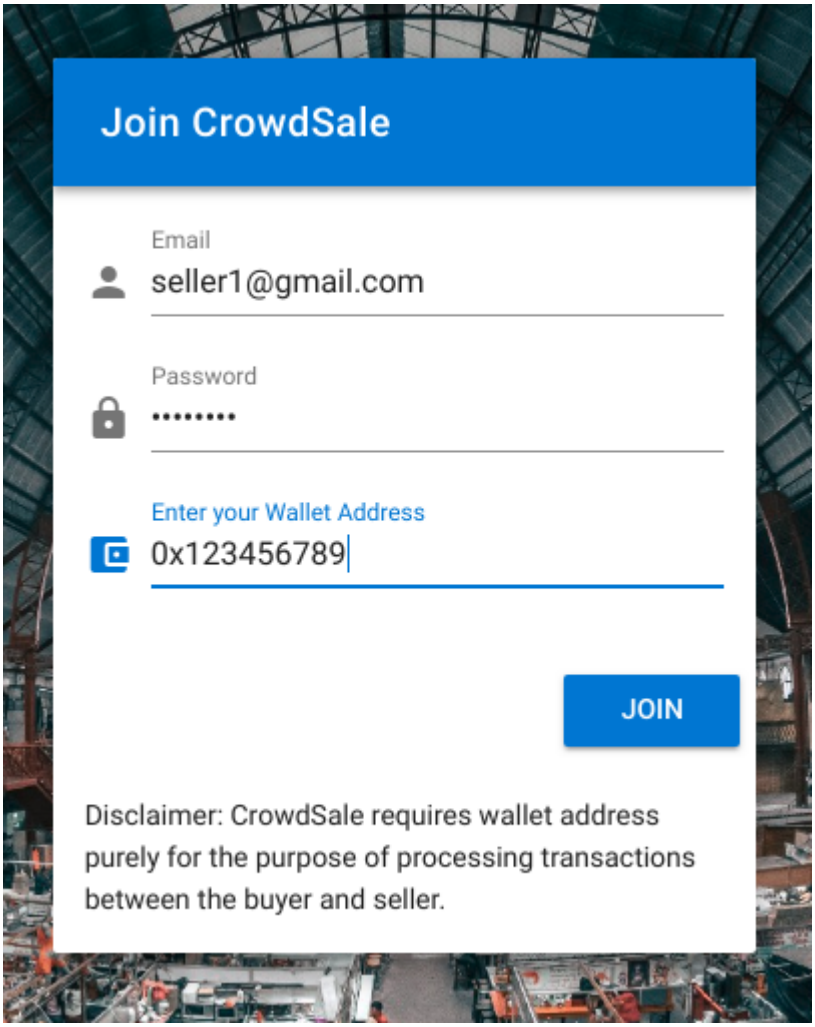
- Home Page, on iPhone X screen.

- Marketplace Page, on iPad Pro screen.

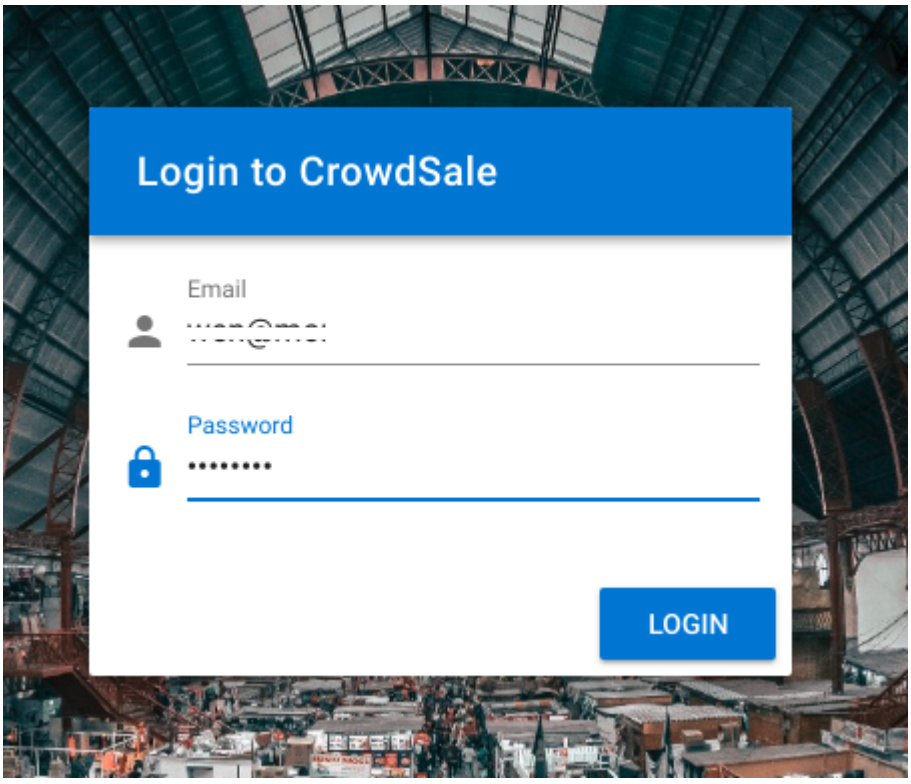


Join and Login

Users should begin exploring CrowdSale by joining with a simple sign-up.

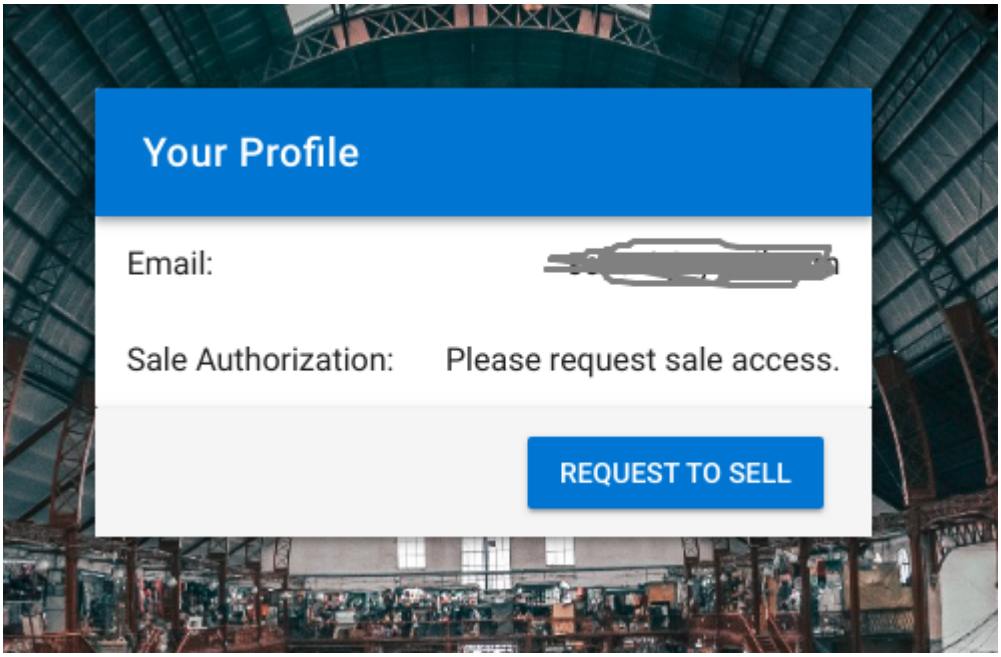


The login data is saved on Google's Firebase web application development platform.

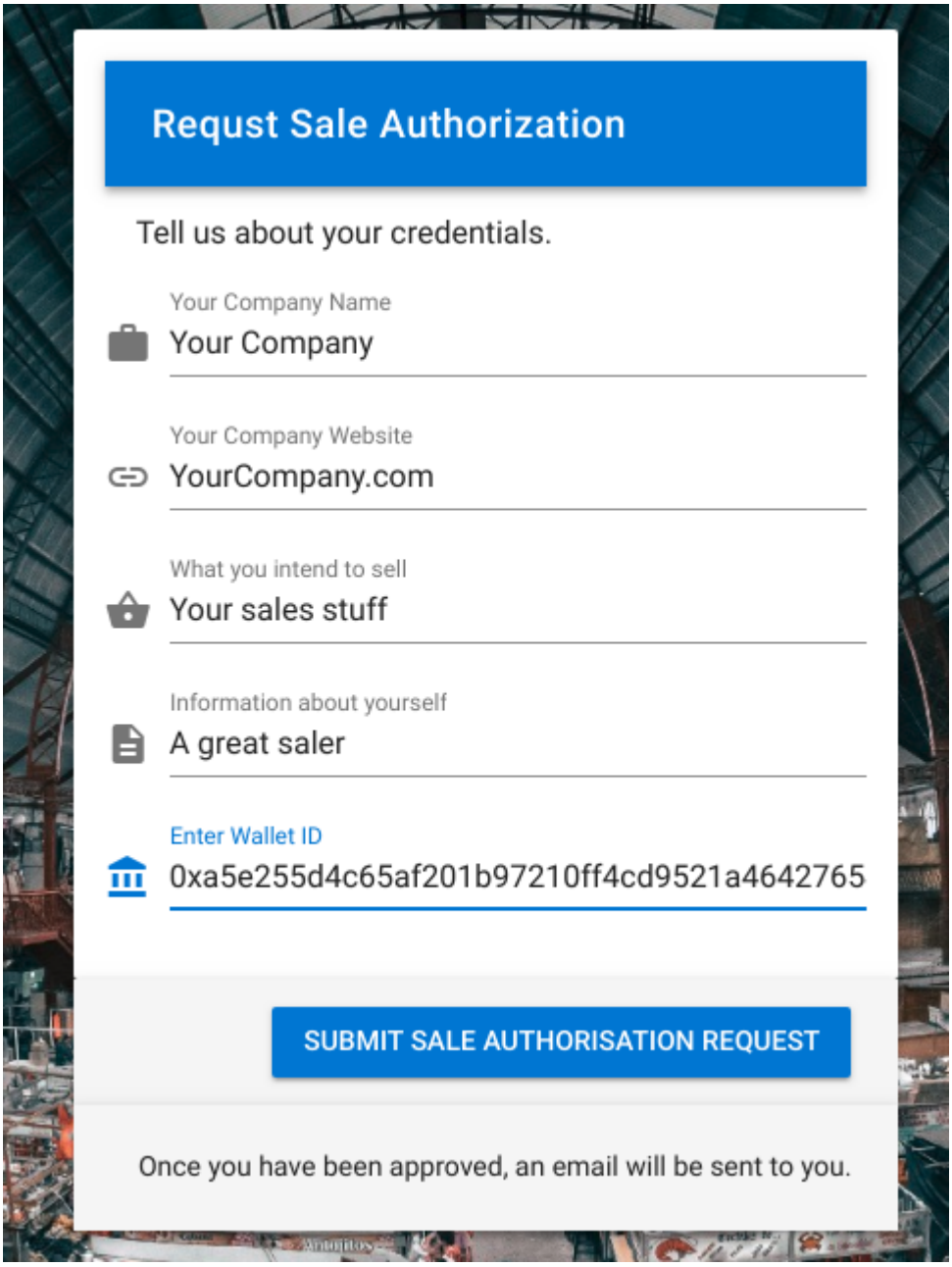


Request for Sale Authorization

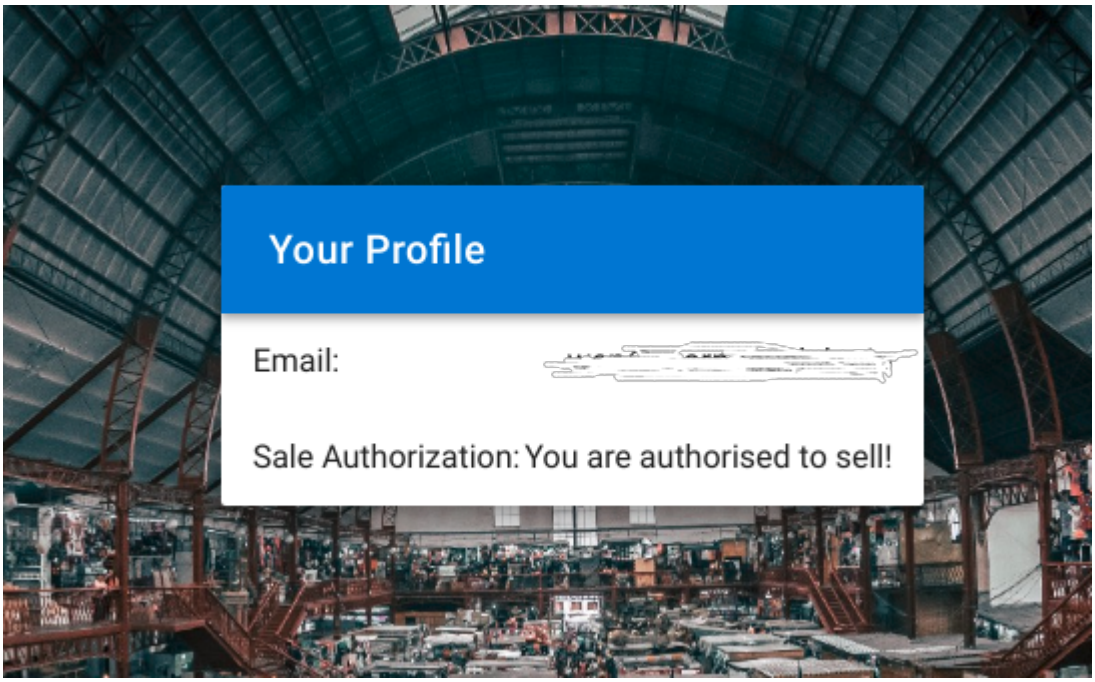
Upon the time of first login, you may need to request for sale authorization so that you can create new sales items in CrowdSale.



Click the REQUEST TO SELL button, fill in the pop-up,

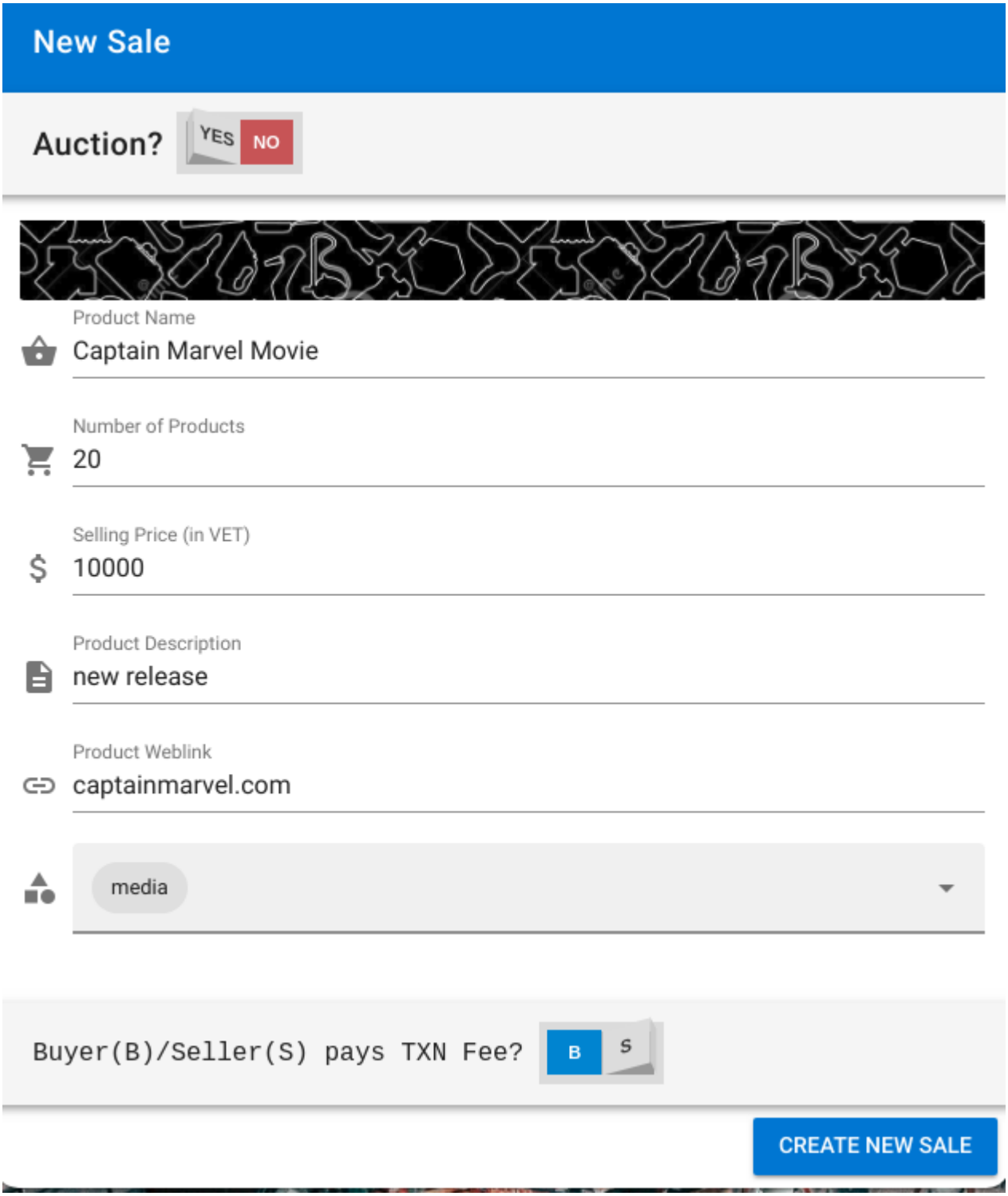


The Wallet ID is your Wallet address. Submit the request. The Administrator will sign and approve your request. You are then authorized to sell. Validate it by checking your profile.



Create a New Sale Item

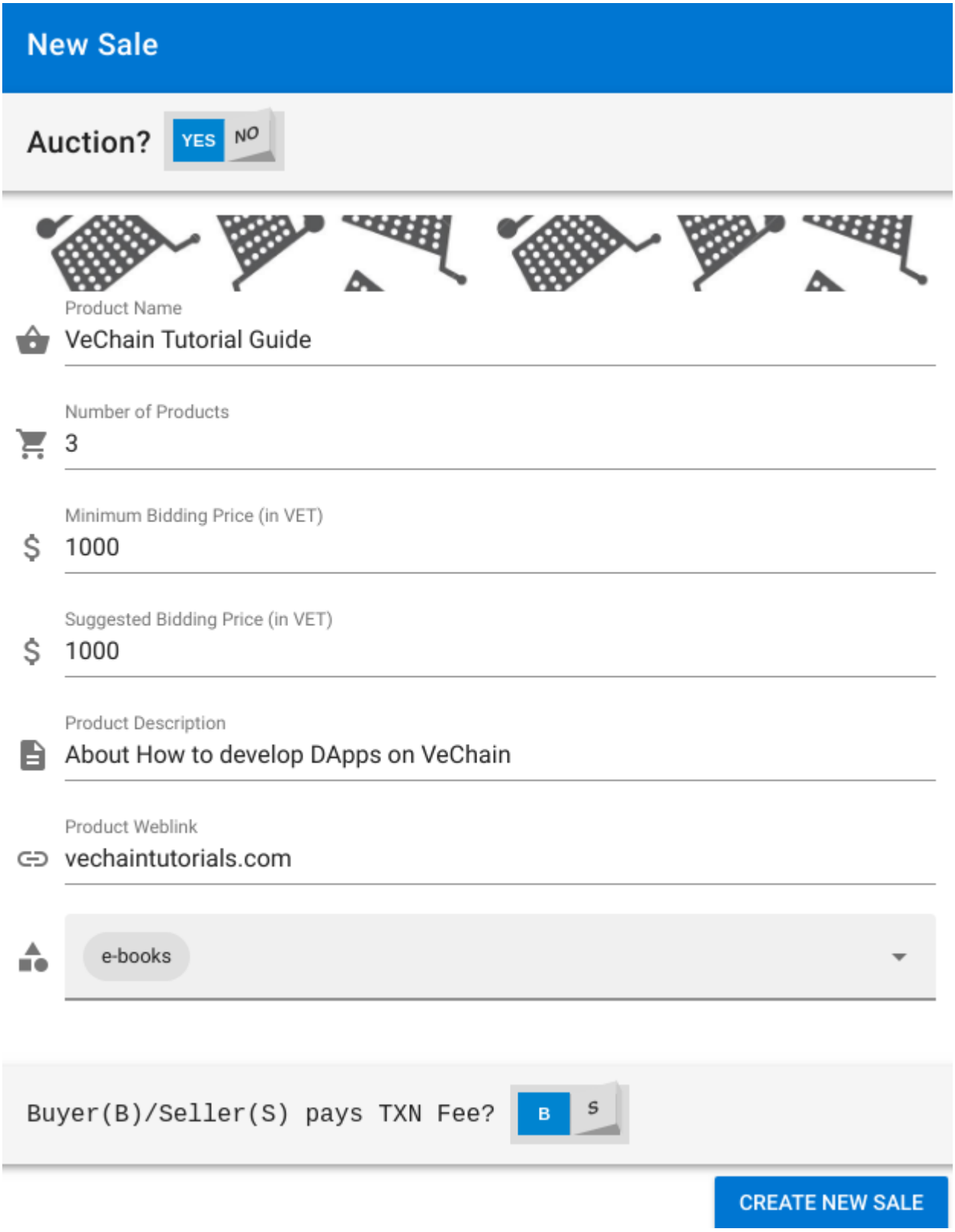
Click the New Sale on the toobar. Toggle the switch to non-auction mode, and fill in the details.



Click on "Create New Sale" to submit the sale to the marketplace.

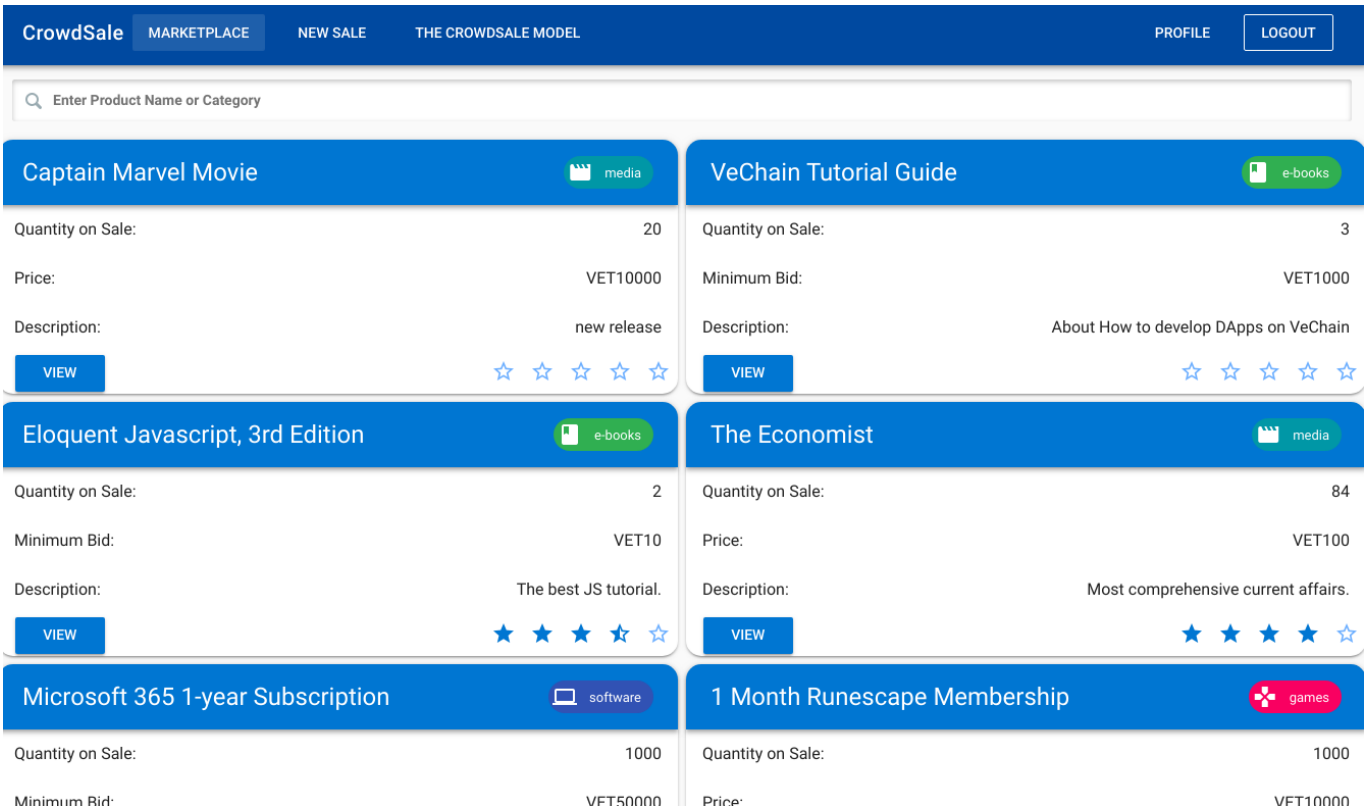
Create a New Auction Item

Click the New Sale on the toobar. Toggle the switch to auction mode, and fill in the details.



Click on "Create New Sale" to submit the sale to the marketplace.

Viewing Your Sell Items on Marketplace

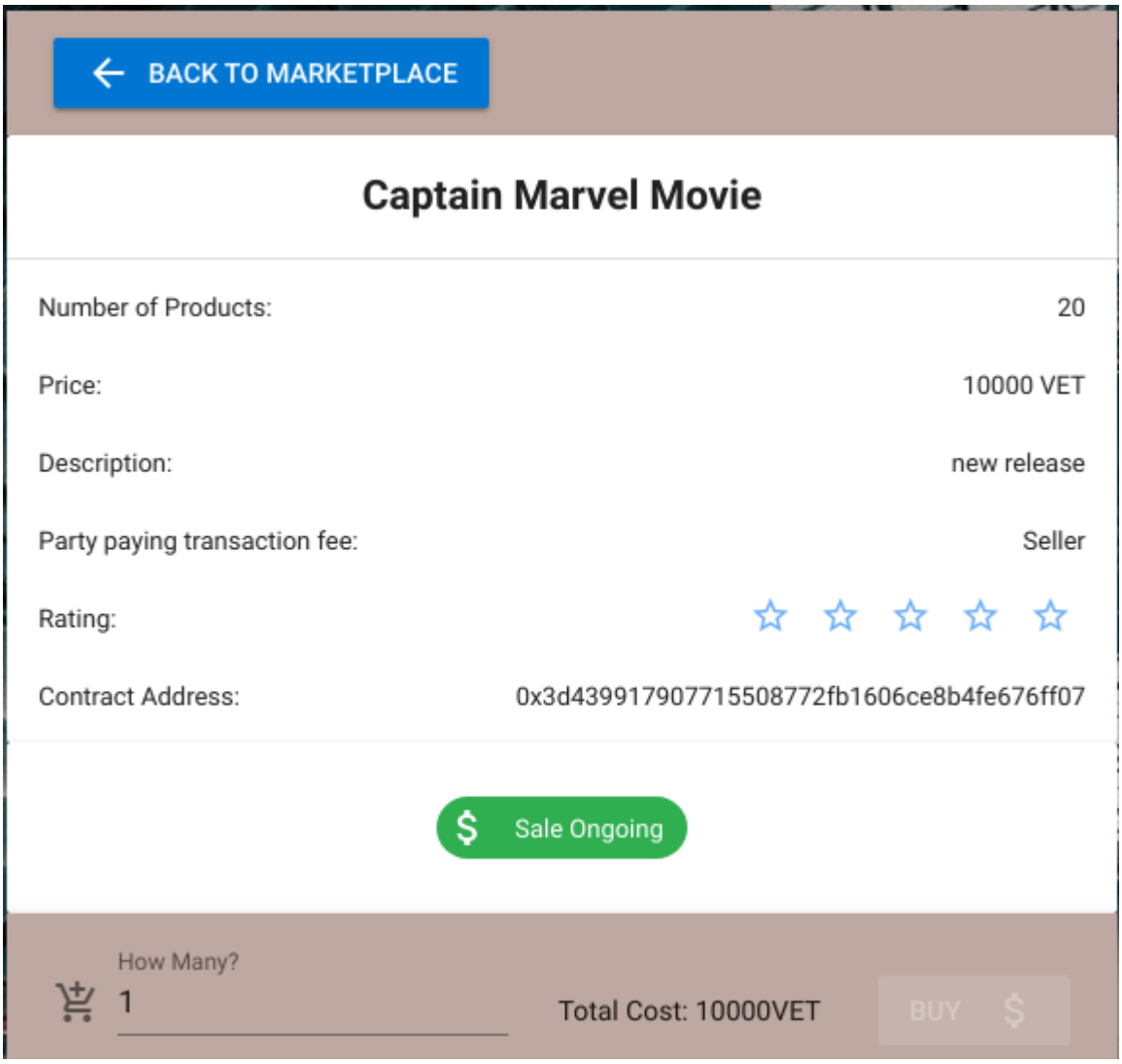


Your items have now been asynchronously added to the marketplace once you add a new sale item or a new auction item.

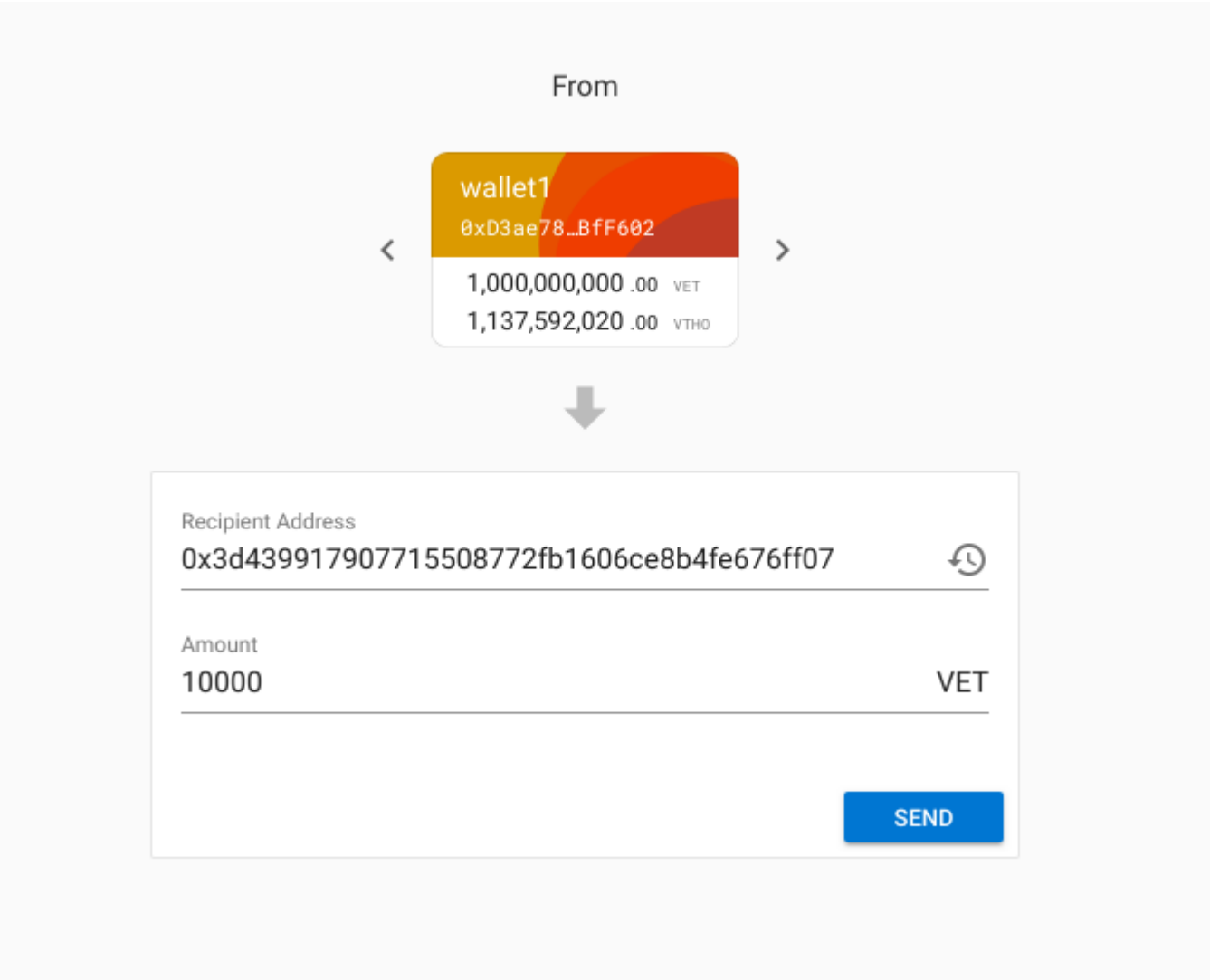
A search bar is also implemented to help you to filter for your specific product.

Purchasing a Non-Auction Product

On the market place page, for the specific item, click on the "View" button and go into the product details.

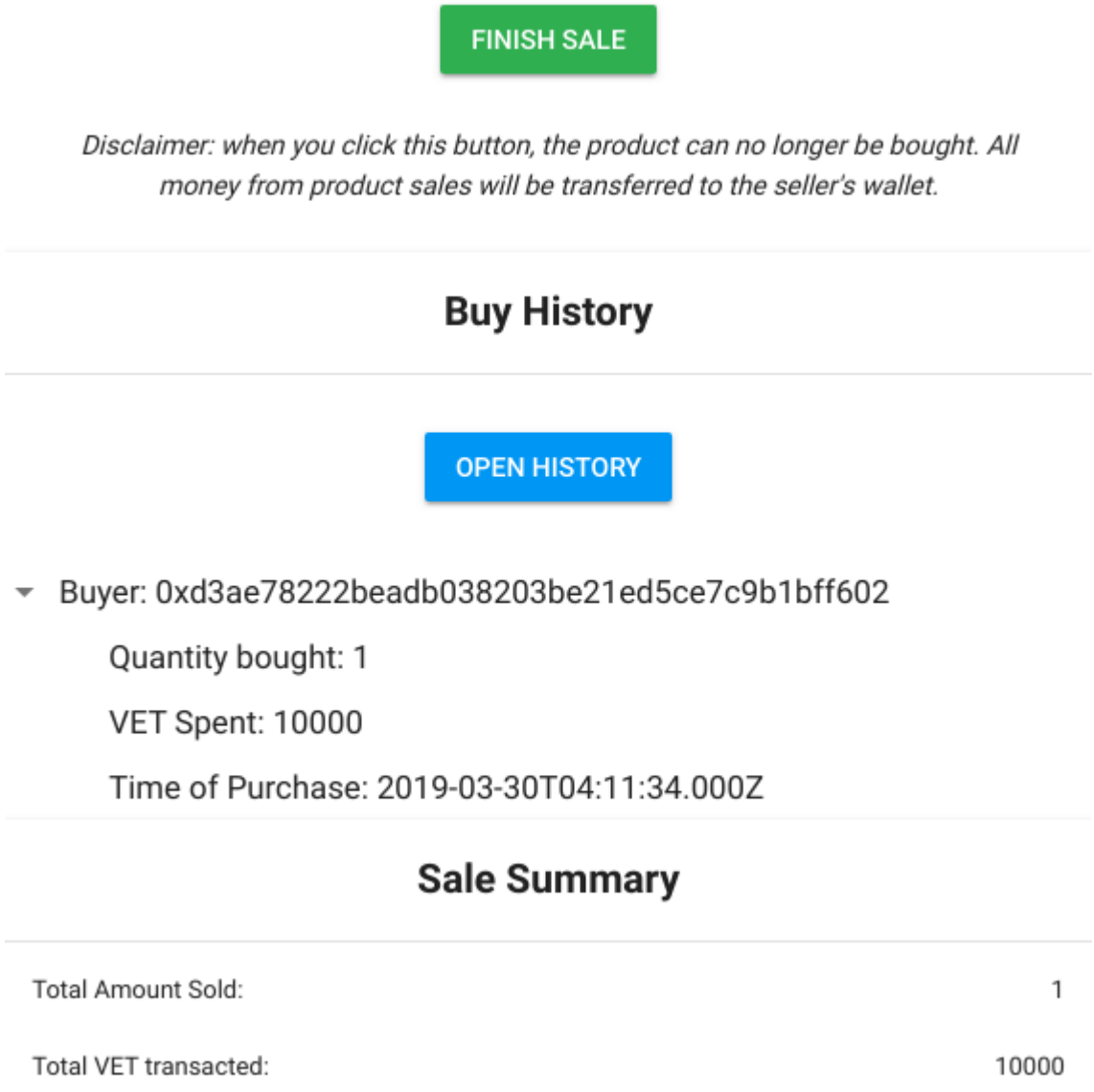


You can key in the number of products you want to buy, and the "total cost" will be shown to you. Copy this total cost, and make a transfer from your wallet to the contract address of the product.



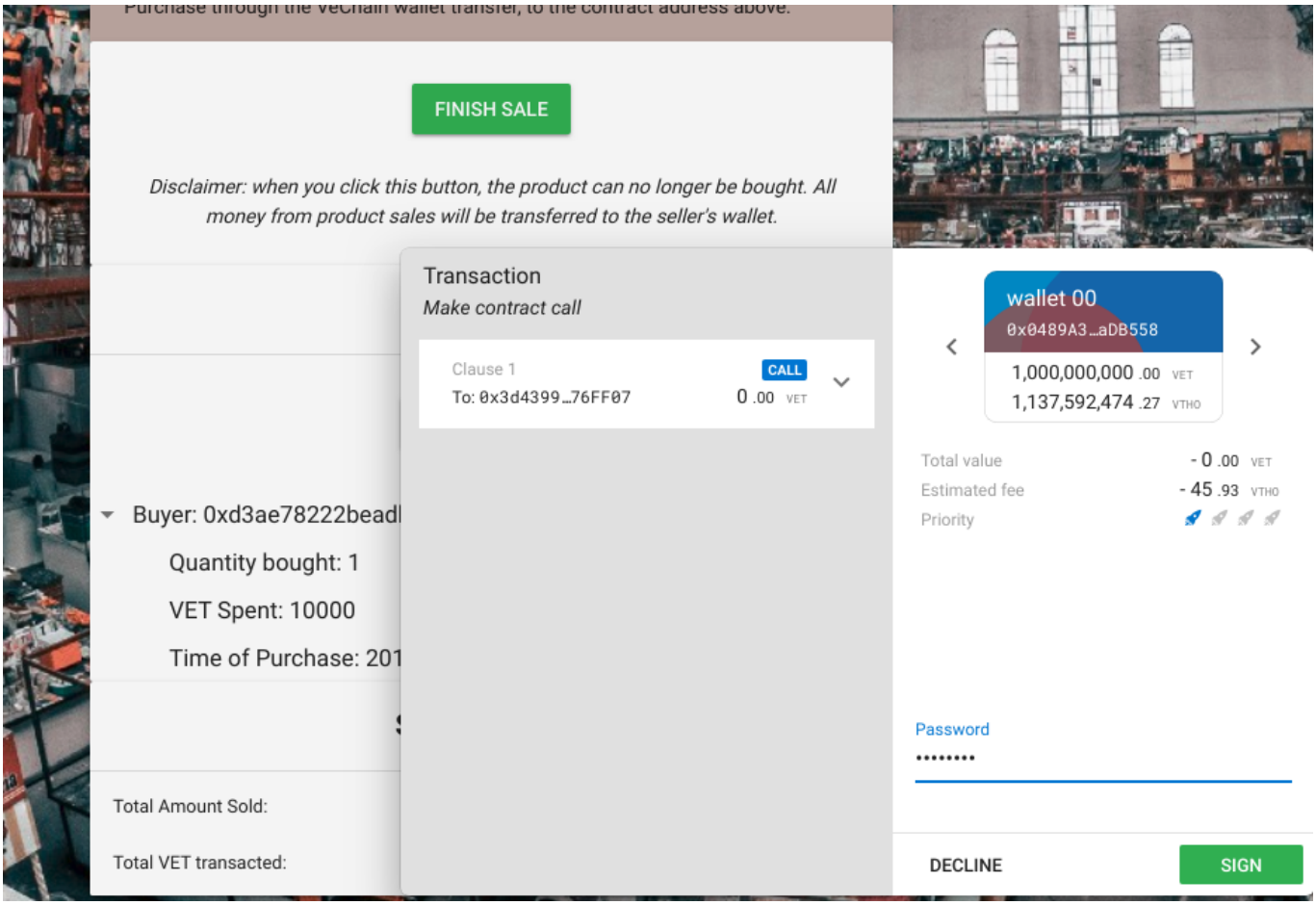
Finishing a Sale

The sale history can be viewed from the product page as below, in a tree-view format.

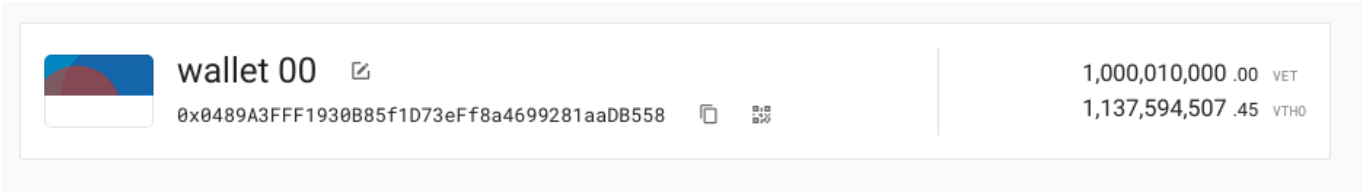


In this case, only buyer *0xd3ae78222beadb038203be21ed5ce7c9b1bff602* has bought the product. The Sale Summary summarises all the transactions.

The seller and only the seller can click on *Finish Sale* to end the sale, preventing further purchases (transferring VET to the wallet) and also sending all the money to his wallet.

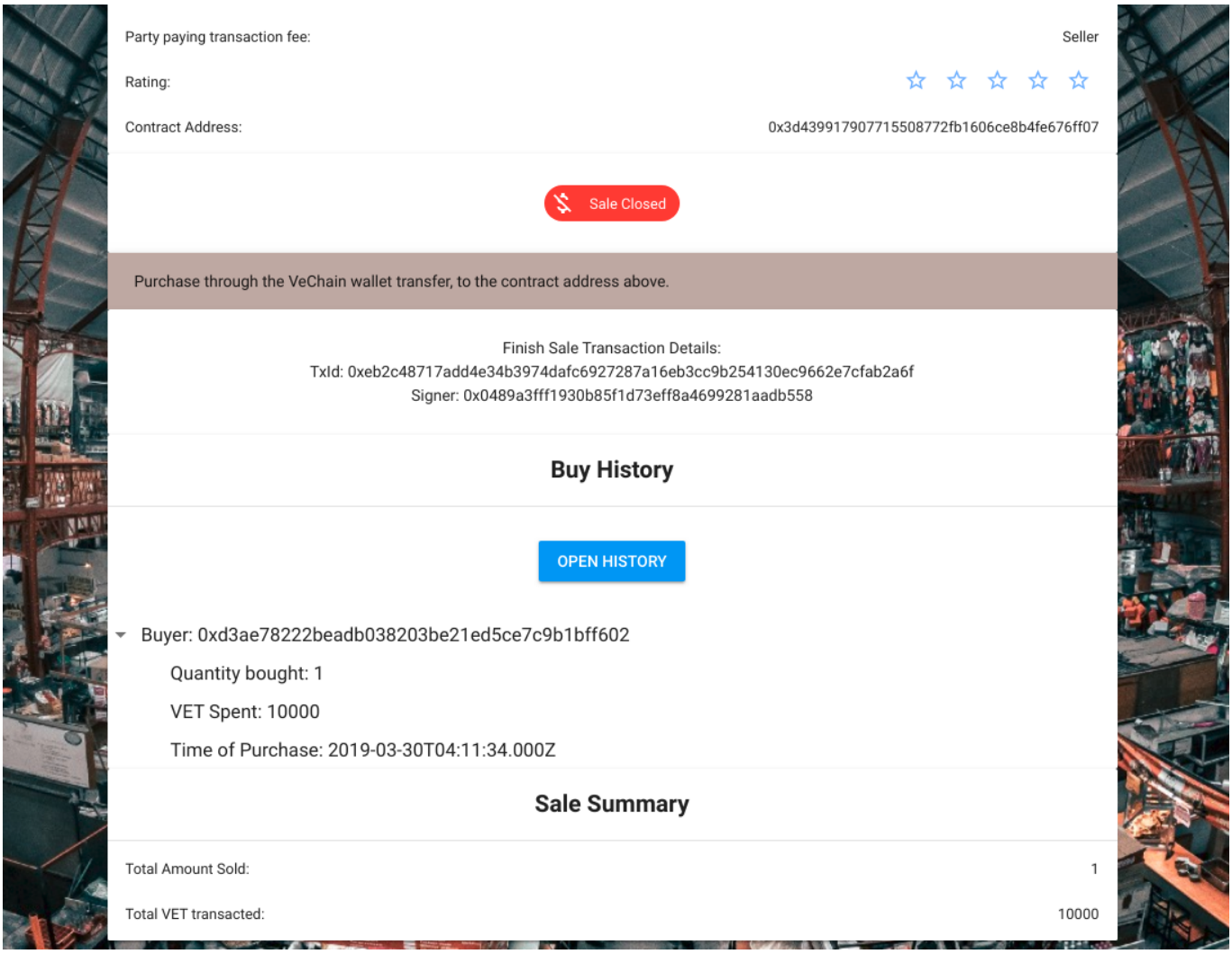


Sign the authorization... (transfer to seller, which is *wallet 00* here)



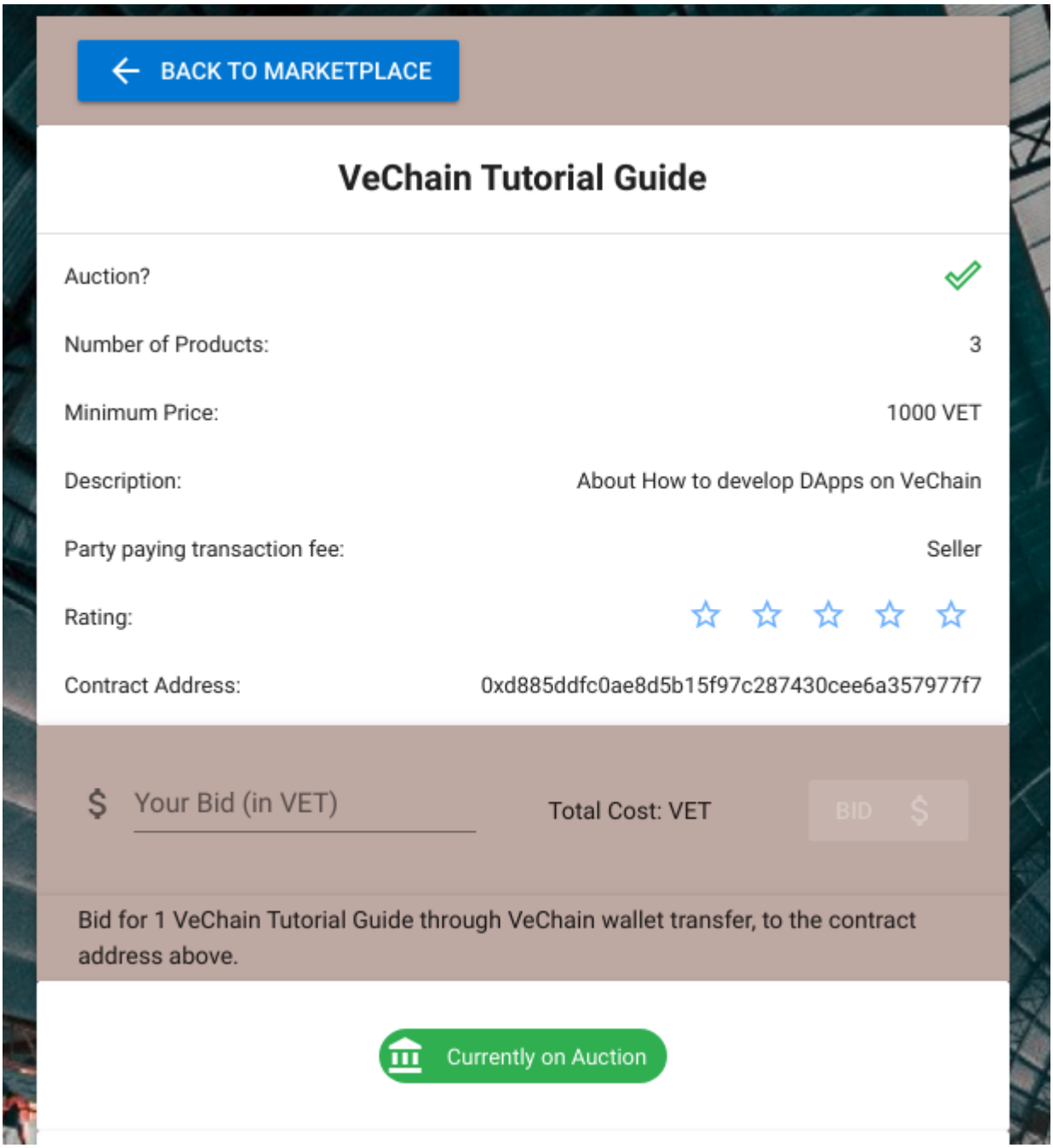
...And now *wallet 00* is 10,000 VET richer!

The sale is now closed, and the TxId as well as the Signer are displayed for verification purposes.

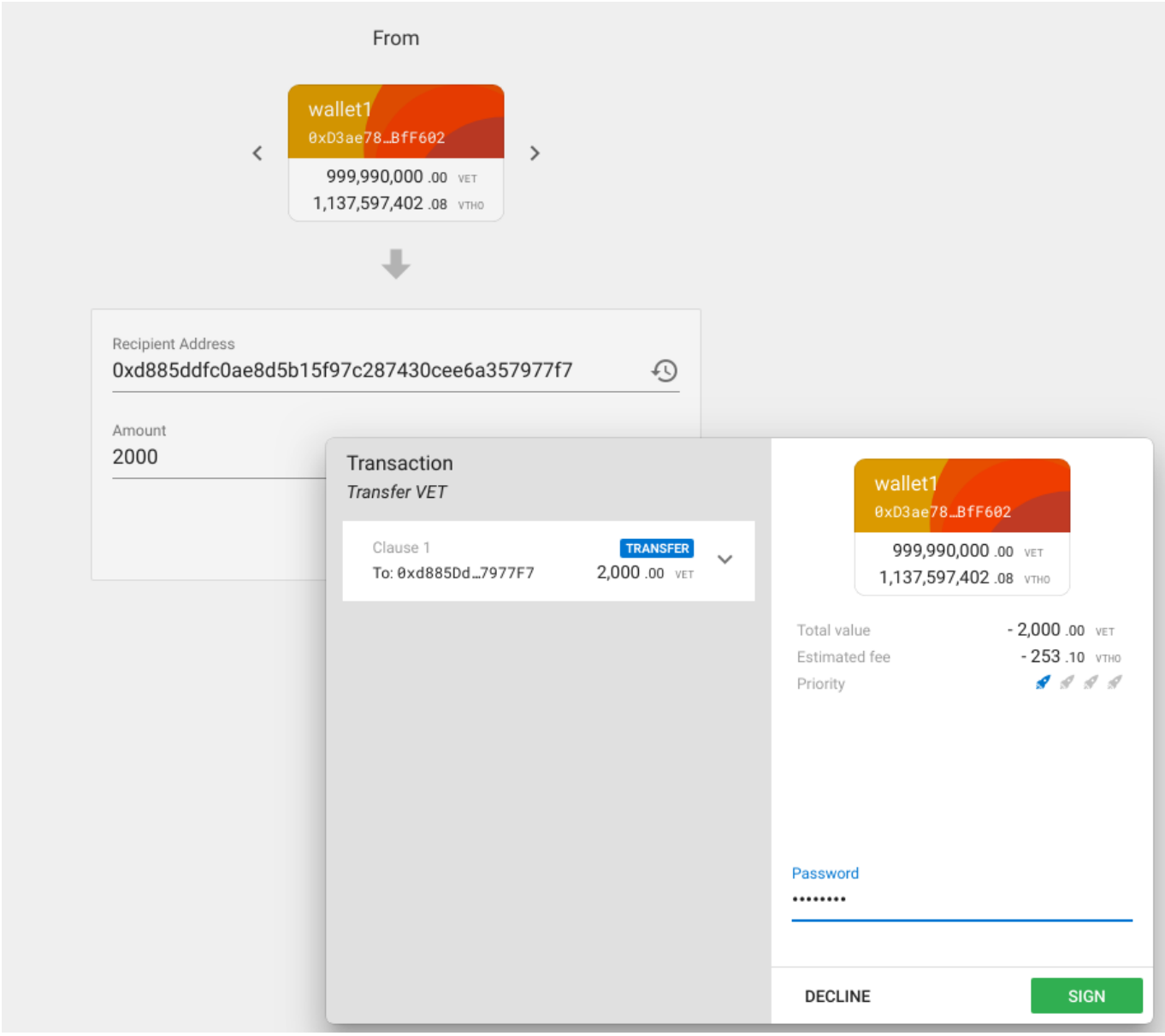


Bidding for an Auction Product

Click on the "View" button and go into the product view page.

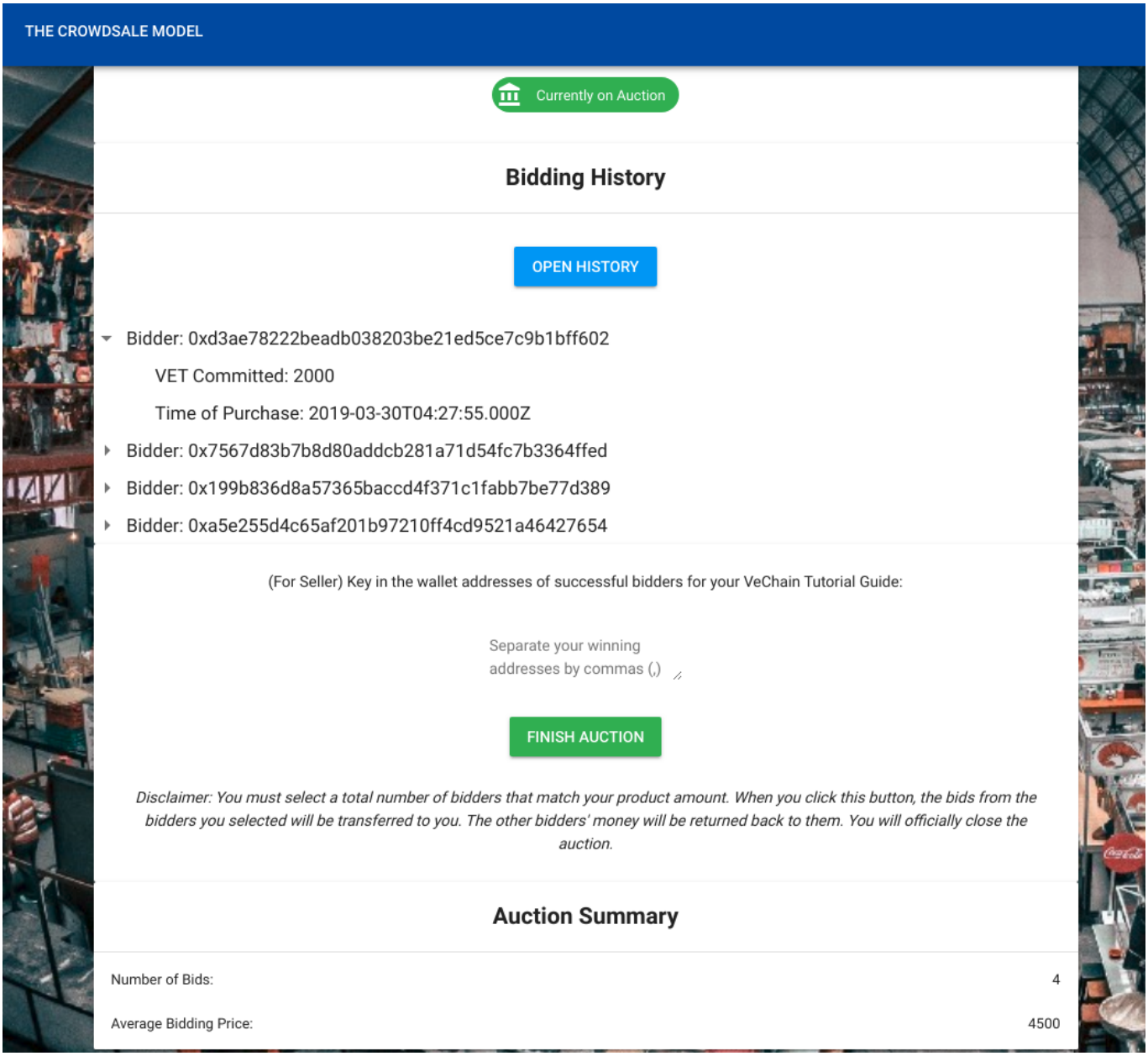


For auction products, you can only bid for 1 product at a time. Submit your bid to the contract address of the product. A bid will only be submitted if it is higher than the minimum price stated.



Finishing an Auction

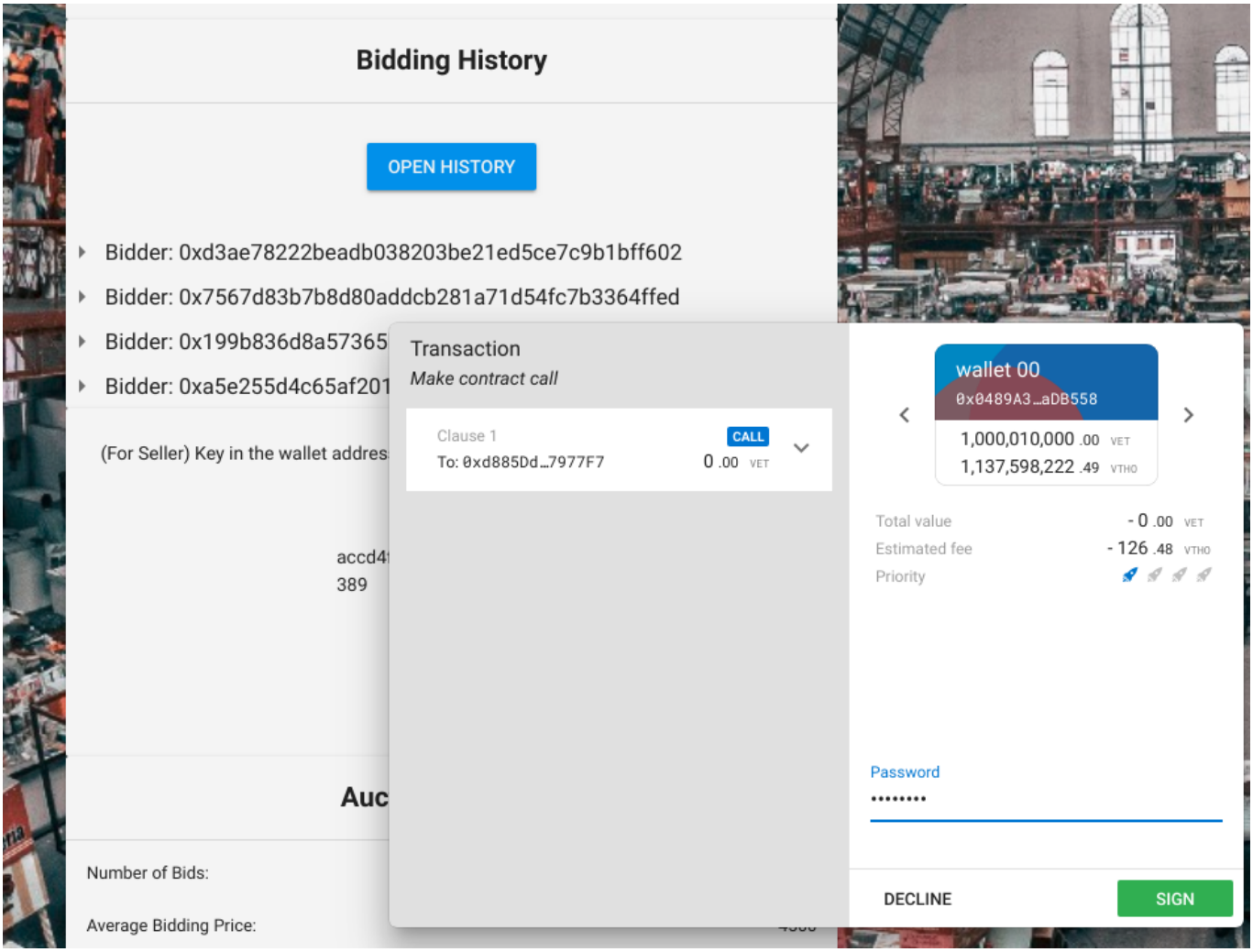
The auction history can be viewed from the product page as below, in a tree-view format.



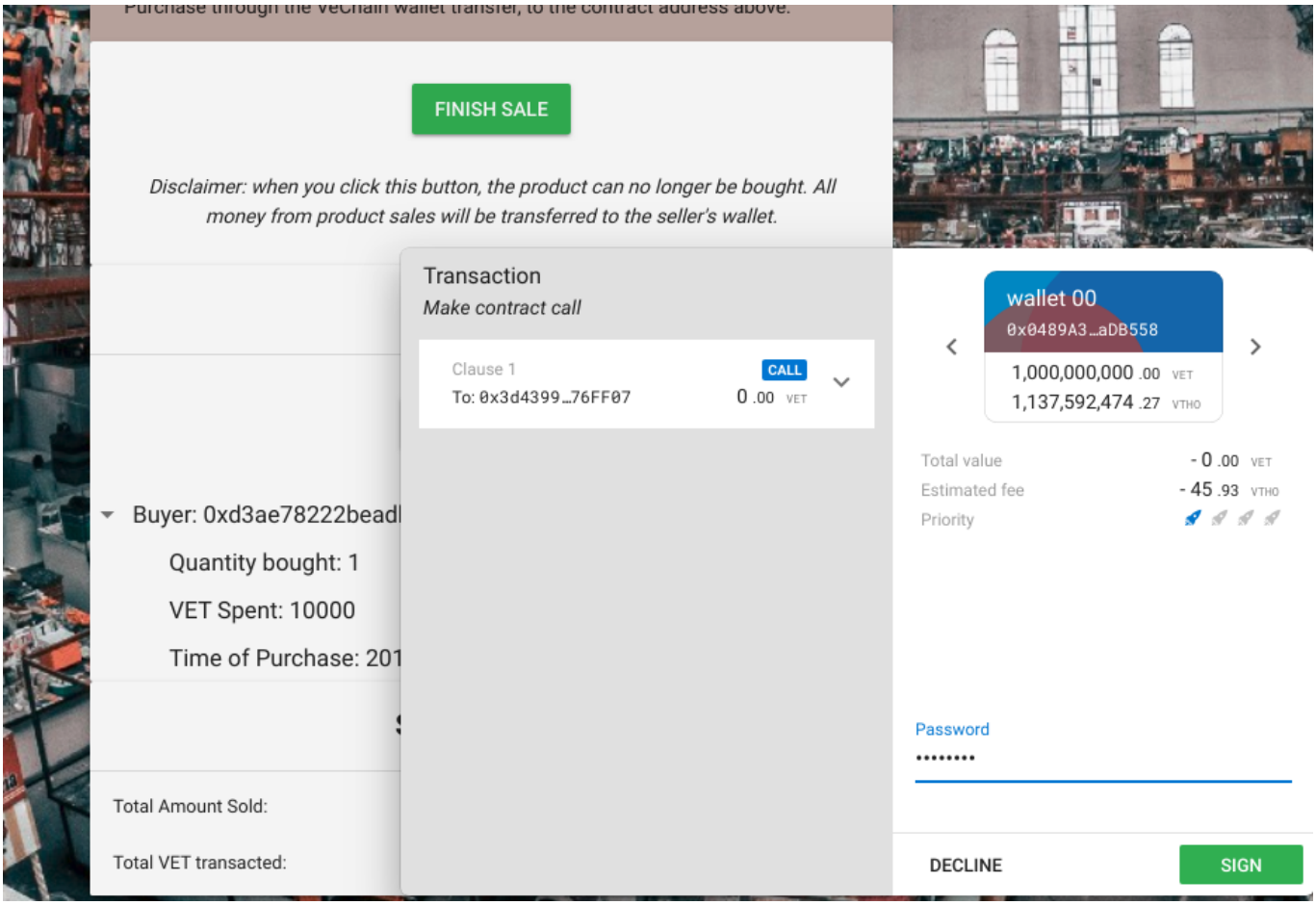
In this case, 4 bidders have submitted their bids. An Auction Summary summarises all the transactions.

Now, the seller can determine the bidders that win the auction. He is to select a number of bidders equalling to the number of his products. In this demo case, he selects the first 3 bidders because they have the highest

bids.

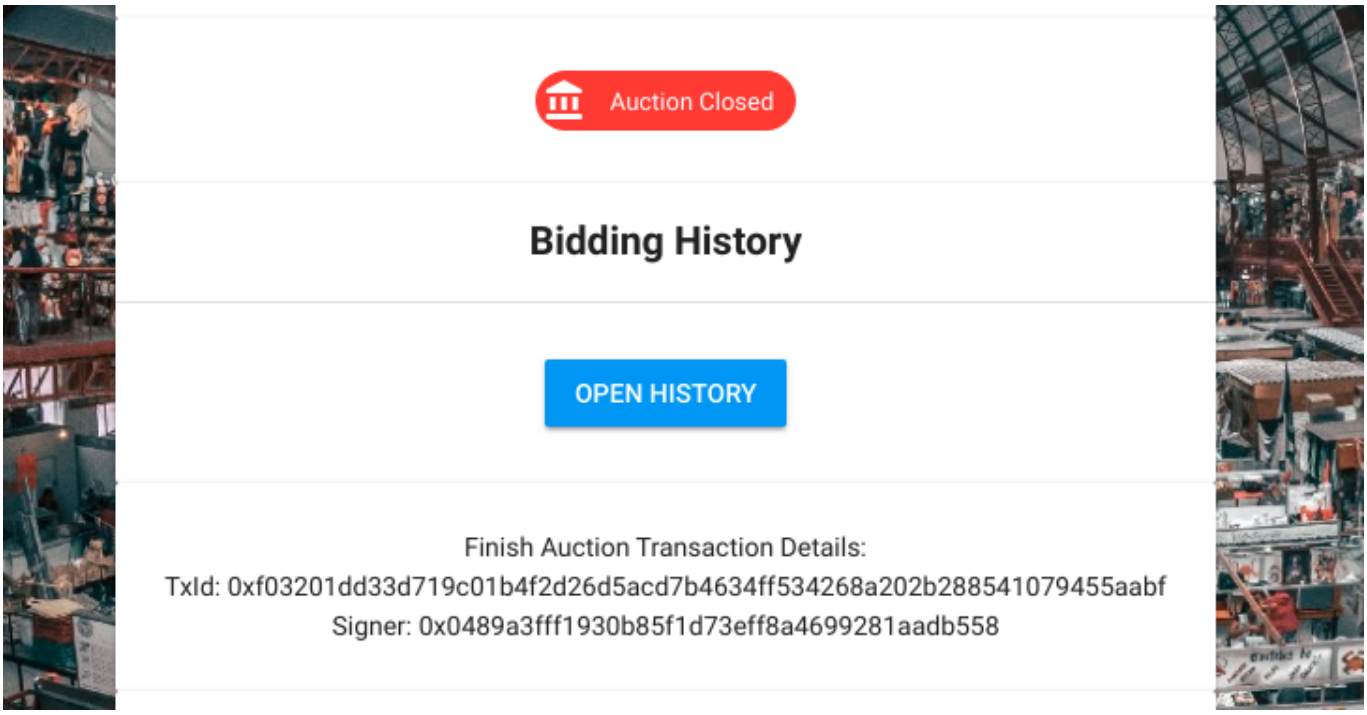


The seller (and only the seller) can click on *Finish Auction* to end the auction, preventing further bids (transferring VET to the wallet) and also sending all the money to his wallet.



Sign the authorization... (transfer to seller, which is *wallet 00* here)

The auction is now closed, and the TxId as well as the Signer are displayed for verification purposes.



Deployment Procedure

Deploying CrowdSale dApp on Local Test Node

Linux or Mac OS is recommened for the development setup.

Prepare development environment

Install the following enablement technology

- Node.js
- Vue-cli (`npm install -g @vue/cli`)
- python3 and pip3
- Golang
- Truffle v4.1.15 (not v5+ which is in beta)
- Web3-gear
- VeChain Thor (https://github.com/vechain/thor#installation)
- VeChain Sync Browser

Running Blockchain Node Locally

Run each of the following in a separate shell window.

1. Run thor solo node `bin/thor solo --on-demand`
2. Run web3-gear `web3-gear`
3. Deploy smart contract on VeChain (Compile is already done) `truffle migrate`
4. Get CrowdSaleFactory contract address and update it in the CrowdSaleFactory.js file `public/client/CrowdSaleFactory.js`

Launch the CrowSale dApp Locally

```
cd src
npm run serve
```

Deploying CrowdSale dApp on VeChain Testnet and Mainnet

The smart contracts have been deployed to the VeChain Testnet and Mainnet. Please refer to the information below for the address and ABIT of the smart contract on Testnet and Mainnet respectively.

Testnet deployoment information

```
    Contract address: 0x740aDeDf4f42Da12d4C177f329D11995bb24000E
    ABI: [{"constant":true,"inputs":[],"name":"getSellers","outputs":
[{"name":"","type":"address[]"}],"payable":false,"stateMutability":"view","
type":"function"},{"constant":true,"inputs":[],"name":"owner","outputs":
[{"name":"","type":"address"}],"payable":false,"stateMutability":"view","ty
pe":"function"},{"constant":false,"inputs":
[{"name":"_newOwner","type":"address"}],"name":"transferOwnership","outputs
":[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":true,"inputs":[],"name":"pageNumber","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability":"view","ty
pe":"function"},{"anonymous":false,"inputs":
```

```
[{"indexed":false,"name":"_person","type":"address"},
{"indexed":false,"name":"_isAdd","type":"bool"}],"name":"RoleAdded","type":
"event"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"previousOwner","type":"address"}],"name":"Ownershi
pRenounced","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"previousOwner","type":"address"},
{"indexed":true,"name":"newOwner","type":"address"}],"name":"OwnershipTrans
ferred","type":"event"},{"constant":false,"inputs":
[{"name":"newPage","type":"uint256"}],"name":"changePageRecords","outputs":
[{"name":"","type":"bool"}],"payable":false,"stateMutability":"nonpayable",
"type":"function"},{"constant":false,"inputs":
[{"name":"_sellerName","type":"string"},
{"name":"_sellerId","type":"string"},{"name":"_wallet","type":"address"},
{"name":"_seller","type":"address"}],"name":"registerSeller","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":true,"inputs":
[{"name":"_sellerAddress","type":"address"}],"name":"getSellerInfo","output
s":[{"name":"","type":"string"},{"name":"","type":"string"},
{"name":"","type":"address"},
{"name":"","type":"address"}],"payable":false,"stateMutability":"view","typ
e":"function"},{"constant":false,"inputs":
[{"name":"_wallet","type":"address"},
{"name":"_seller","type":"address"}],"name":"updateSellerInfo","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":false,"inputs":
[{"name":"_seller","type":"address"}],"name":"disableSeller","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":false,"inputs":[{"name":"_productId","type":"string"},
{"name":"_sellerId","type":"string"},
{"name":"_productName","type":"string"},
{"name":"_description","type":"string"},
{"name":"_weblink","type":"string"},
{"name":"_productCategory","type":"string"},
{"name":"_productAmount","type":"uint256"},
{"name":"_price","type":"uint256"}],"name":"createNewProduct","outputs":
[{"name":"","type":"address"}],"payable":false,"stateMutability":"nonpayabl
e","type":"function"},{"constant":false,"inputs":
[{"name":"_productId","type":"string"},
{"name":"_sellerId","type":"string"},
{"name":"_productName","type":"string"},
{"name":"_description","type":"string"},
{"name":"_weblink","type":"string"},
{"name":"_productCategory","type":"string"},
{"name":"_productAmount","type":"uint256"},
{"name":"_minPrice","type":"uint256"}],"name":"createAuctionProduct","outpu
ts":
[{"name":"","type":"address"}],"payable":false,"stateMutability":"nonpayabl
e","type":"function"},{"constant":true,"inputs":
[{"name":"_page","type":"uint256"}],"name":"getAllProductList","outputs":
[{"name":"","type":"address[]"}],"payable":false,"stateMutability":"view","
type":"function"},{"constant":true,"inputs":
[{"name":"_page","type":"uint256"}],"name":"getProductListOfSeller","output
s":
```

```
[{"name":"","type":"address[]"}],"payable":false,"stateMutability":"view","
type":"function"}]
```


Mainet deployoment information


```
Contract Address: 0x00c307a02f68c1A5dd9232C1150983F482a5d173
ABI: [{"constant":true,"inputs":[],"name":"getSellers","outputs":
[{"name":"","type":"address[]"}],"payable":false,"stateMutability":"view","
type":"function"},{"constant":true,"inputs":[],"name":"owner","outputs":
[{"name":"","type":"address"}],"payable":false,"stateMutability":"view","ty
pe":"function"},{"constant":false,"inputs":
[{"name":"_newOwner","type":"address"}],"name":"transferOwnership","outputs
":[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":true,"inputs":[],"name":"pageNumber","outputs":
[{"name":"","type":"uint256"}],"payable":false,"stateMutability":"view","ty
pe":"function"},{"anonymous":false,"inputs":
[{"indexed":false,"name":"_person","type":"address"},
{"indexed":false,"name":"_isAdd","type":"bool"}],"name":"RoleAdded","type":
"event"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"previousOwner","type":"address"}],"name":"Ownershi
pRenounced","type":"event"},{"anonymous":false,"inputs":
[{"indexed":true,"name":"previousOwner","type":"address"},
{"indexed":true,"name":"newOwner","type":"address"}],"name":"OwnershipTrans
ferred","type":"event"},{"constant":false,"inputs":
[{"name":"newPage","type":"uint256"}],"name":"changePageRecords","outputs":
[{"name":"","type":"bool"}],"payable":false,"stateMutability":"nonpayable",
"type":"function"},{"constant":false,"inputs":
[{"name":"_sellerName","type":"string"},
{"name":"_sellerId","type":"string"},{"name":"_wallet","type":"address"},
{"name":"_seller","type":"address"}],"name":"registerSeller","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":true,"inputs":
[{"name":"_sellerAddress","type":"address"}],"name":"getSellerInfo","output
s":[{"name":"","type":"string"},{"name":"","type":"string"},
{"name":"","type":"address"},
{"name":"","type":"address"}],"payable":false,"stateMutability":"view","typ
e":"function"},{"constant":false,"inputs":
[{"name":"_wallet","type":"address"},
{"name":"_seller","type":"address"}],"name":"updateSellerInfo","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":false,"inputs":
[{"name":"_seller","type":"address"}],"name":"disableSeller","outputs":
[],"payable":false,"stateMutability":"nonpayable","type":"function"},
{"constant":false,"inputs":[{"name":"_productId","type":"string"},
{"name":"_sellerId","type":"string"},
{"name":"_productName","type":"string"},
{"name":"_description","type":"string"},
{"name":"_weblink","type":"string"},
{"name":"_productCategory","type":"string"},
{"name":"_productAmount","type":"uint256"},
{"name":"_price","type":"uint256"}],"name":"createNewProduct","outputs":
```

```
[{"name":"","type":"address"}],"payable":false,"stateMutability":"nonpayabl
e","type":"function"},{"constant":false,"inputs":
[{"name":"_productId","type":"string"},
{"name":"_sellerId","type":"string"},
{"name":"_productName","type":"string"},
{"name":"_description","type":"string"},
{"name":"_weblink","type":"string"},
{"name":"_productCategory","type":"string"},
{"name":"_productAmount","type":"uint256"},
{"name":"_minPrice","type":"uint256"}],"name":"createAuctionProduct","outpu
ts":
[{"name":"","type":"address"}],"payable":false,"stateMutability":"nonpayabl
e","type":"function"},{"constant":true,"inputs":
[{"name":"_page","type":"uint256"}],"name":"getAllProductList","outputs":
[{"name":"","type":"address[]"}],"payable":false,"stateMutability":"view","
type":"function"},{"constant":true,"inputs":
[{"name":"_page","type":"uint256"}],"name":"getProductListOfSeller","output
s":
[{"name":"","type":"address[]"}],"payable":false,"stateMutability":"view","
type":"function"}]
```

Onboarding CrowdSale dApp on Morpheus Labs Application Library

This dApp aims to provide a reference implementation of VeChain dApp that would help promote VeChain adoption. Morpheus Labs Blockchain Platform as a Service (ML BPaaS) recognizes VeChain as a primary public blockchain business solution and now the end to end development and test environments are available on ML BPaaS. So the application will be published into the Application Library on ML BPaaS in near future. Then developers can use the application as a reference and leverage the platform to quickly build and test VeChain dApps.

Future Add-Ons:

This version of the dApp is just the starting point, more functional features for crowd sale and technical features leveraging VeChain unique features will be added in the future add-ons. Stay with us for future dApp developments using VeChain.