Woocommerce Ethereum Payment Method

Abstract

This payment plugin makes it possible to make a payment on a woocommerce site by sending ether. Main features include QR codes for payment address, payment address rotation, custom order attributes to include relevant payment details to the order on the back end. Attached are screenshot of how the plugin should look. Along with linked code examples of payment plugins. The plugin should be as simple as possible and avoid extra code and complexity when possible.

Requirements

1. Database needs to hold the following: The plugin's various settings, api urls, receiving eth addresses. The plugin should install and uninstall with little or no effect to the database and little to no left over files. Use wordpress' best practices for plugin development. Use wordpresses default menu layout, no need to create extra menu or setting paths, just whats needed.

2. Should follow the standard url paths for woocommerce orders and payments. Use standard open source methods or tools to generate the QR code image.

3. The Plugin only needs to work for ether but should have a switch to work on an ether test net and some way to let users know its on the test net. Both on the front end and back end.

4. The price information should be pulled from the etherscan.io api. The database should be able to store this account information securely. Users of the plugin will need a free account with etherscan. Will need a cron job or some way to time api calls as not to get limit blocked. Should have a way to store price information in the database. Will also need method to convert from the store's base currency to eth and display the appropriate number when this payment method is displayed. When the api is unavailable the store will use the average of the last 15 price updates. A box or message should show the time last updated and if the last price update was successful or not. Keep it simple.

5. The plugin should automatically rotate the payment addresses used and add which address the user was presented with as an attribute attached to the order. The webstore owner can add up to 10 eth account addresses to be rotated through. Each order only has 1 address associated with it.

6. All orders and the payment status is checked by a person. If the payment was received the person will update the order's status from "awaiting payment" to "processing", "completed", or any other appropriate status. The user can/should also update the orders notes with payment information. This payment method should work just like any other in regards to the order steps and flow.

7. The plugin should work like any other woocommerce payment method and work with any standard wordpress theme. All CSS code should be generic so that it could work with whatever theme is currently installed on the WordPress site. Should use existing menu layouts and styling.

8. Should work for all woocommerce product types (digital and physical)

9. All payments should have an attached order number. The order number is attached by appending 'OrderNumber' to the end of an 18 digit eth payment.

For example: an order of .01223 eth would turn into .012230000000000865 for order 865. the final payments amount displayed to the user must be 18 digits. The user can still pay .01223 eth and most wallets do some sort of rounding but in the future the order number will be used to automatically link. Orders and payments.

Expected Deliverables

Custom Code for a wordpress/woocommerce compatible plugin that can be zipped and installed on a standard system.

Note and README documentation

Time Frame for Delivery

30 days for completed project.
Weekly Emails updates, at least 4 total.
Weekly git uploads with notes and code comments.

Included Examples, Files, and Links

CryptoWoo: A general example of how the plugin should work but built for the bitcoin blockchain.
https://www.cryptowoo.com/

Please review the standard Woocommerce payment methods to understand the standard structure and layout.
https://woocommerce.com/

Please Review the Etherscan.io api
https://etherscan.io/apis