Chillieman's Mobile Application

The purpose of this document is to explain the functionality of Chillieman's mobile application, and its associated utility token.

I cannot provide solid delivery dates when things will be complete. I am a professional Software Engineer, specializing in Android development, and I'm taking this Project on completely by myself. I work full time, have family obligations, and I spend too much time on Twitter. XD

This is an application that will give any crypto trader the ability to easily use Limit Orders on any Decentralized Exchange that is based on Ethereum's network. This App will be scalable to use all EVM block chains such as BNB, ETH, MATIC, AVAX, etc. Limit Orders will have the ability to buy a token once it dips below a certain price, and then automatically sell when the token has risen in price. Stop Loss functionality will also allow the user to sell tokens in a crashing market. I intend to provide this functionality for an extremely low fee, if not completely free – And only collect taxes when the app has successfully made the user a profit. – This is the motivation behind the app – To empower an average human to have a fighting chance against the rich and powerful who already have trading algorithms at their disposal!

I want to make this explicit: **This token and application are designed to be as fair as I can possibly make it.** Trust is one of the hardest things to gain, especially in the crypto space, as the number of scams out here is just ridiculous. Even though I have gone to great lengths to make the Smart Contract code as fair and safe as possible, I understand that it may not be easy, for those who aren't a geek like I am, to *see* that the Contracts and App will fair and safe. I will embrace feedback from everyone, even my biggest critics, and address any issues that arise. I am genuinely interested in helping other people make money.

What are the features of Chillieman's mobile application?

Privacy is number one, ChillieWallet will never capture any personal information about you such as Email, Real Name, etc.

The idea of App is to turn your phone into your own, full-time, day trader. The focus area is Tokens that are listed on a Decentralized Exchange such as UniSwap or PancakeSwap. One of the most annoying things about these exchanges, is there is no meaningful method of creating Limit Orders! It also can be annoying using the website interface of the DEX, especially when using a phone. With the app, you can be faster than the competition because the App is communicating directly with the Block Chain (Via RPC Node) and its Smart Contracts – Best of all, it will use the preprogrammed trading logic that you teach it to use. Come up with a good strategy? Well you can just sell it as a (SHH – Not yet, Chillieman!!)

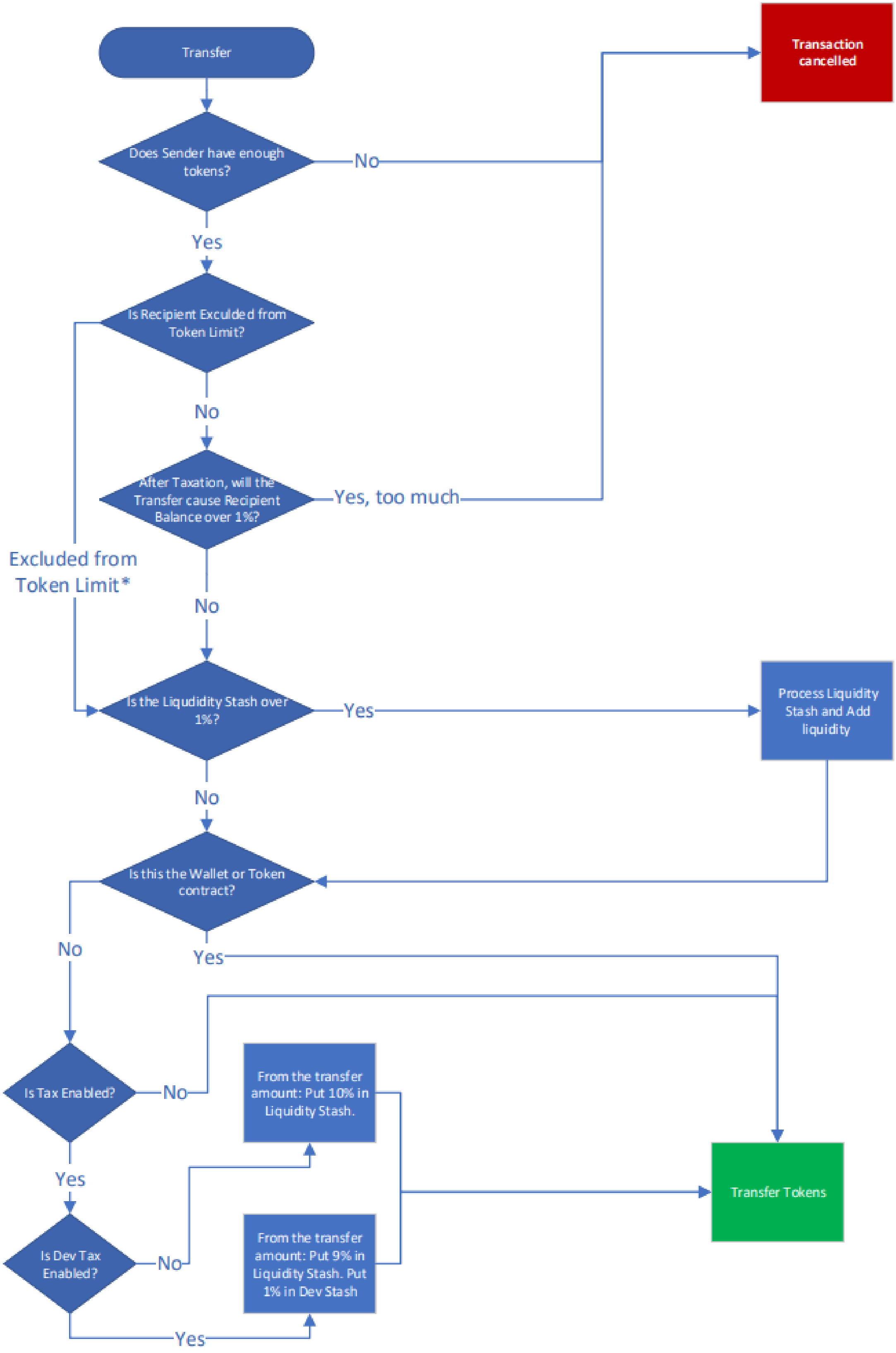
Wouldn't it be nice if you could tell your phone to Buy a certain Token once the price dips to your target? Wouldn't it be nice to have a Stop Loss on your tokens, that way your exit strategy can automatically be executed when the market is crashing? Wouldn't it be nice to give your phone precise instructions on how it should Flip a certain token, and it just does it for you? And wouldn't it be nice to use this functionality on any DEX on any EVM blockchain, even ones that you can manually add yourself? This application will be the solution to these questions!

Another plan is to have robust vetting of any token that is being purchased. V2 of the app will include detection of common scams and warn the user before they buy something that they will most likely not be able to sell back. I have been scammed, rug pulled, honey potted, and lied to – I'm sick of it! I will provide a tool that to mitigate the damage.

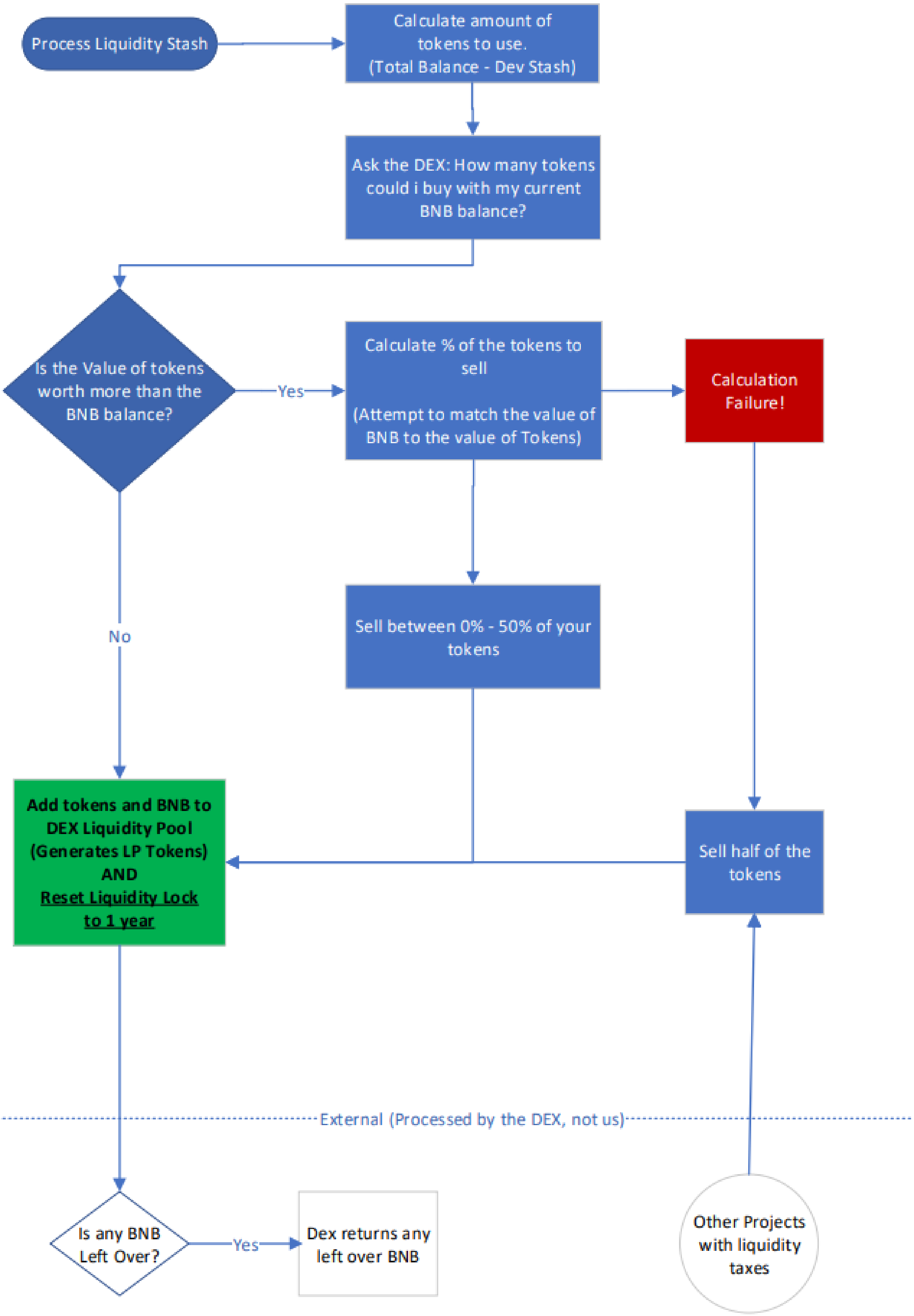
Much more info to come on the App! – This is just the first step. I plan to make apps that will help the average joe in EVERY industry! I'm not going to let the AI strip 100% of wealth from 99.99% of humans.

And if no one ends up using it – I'll still have an automated trader in my pocket – that does what no other app currently does. I'll still be happy =] – **So let's do this!**

Token Transfers:

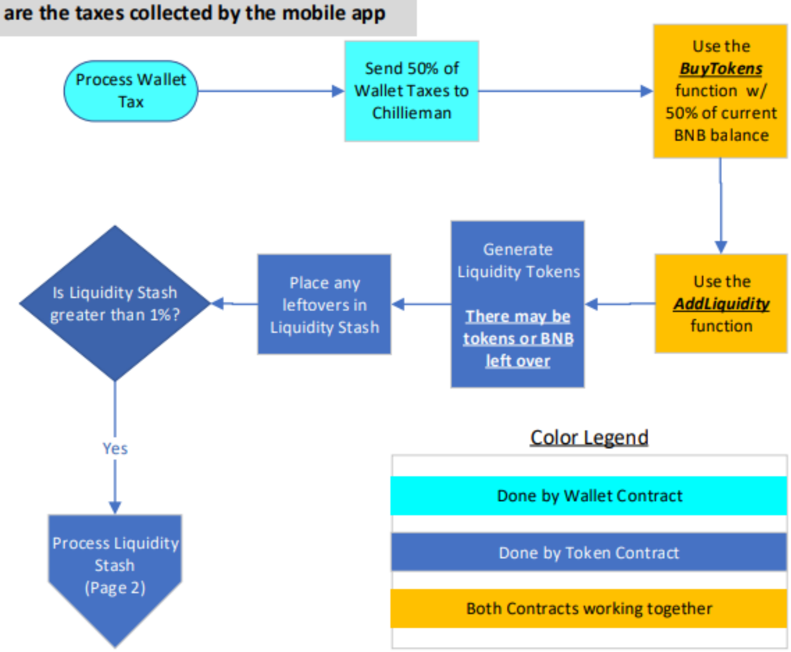

Transfer
Does Sender have enough tokens?
No
Transaction cancelled
Yes
Is Recipient Exculded from Token Limit?
No
After Taxation, will the Transfer cause Recipient Balance over 1%?
Yes, too much
Excluded from Token Limit*
No
Is the Liqudidity Stash over 1%?
Yes
Process Liquidity Stash and Add liquidity
No
Is this the Wallet or Token contract?
No
Yes
Is Tax Enabled?
No
From the transfer amount: Put 10% in Liquidity Stash.
Yes
Transfer Tokens
Is Dev Tax Enabled?
No
From the transfer amount: Put 9% in Liquidity Stash. Put 1% in Dev Stash
Yes

Process Liquidity Stash:

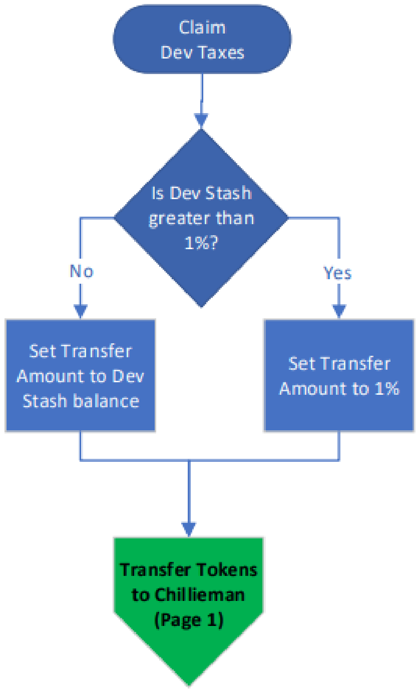

Process Liquidity Stash
Calculate amount of tokens to use. (Total Balance - Dev Stash)
Ask the DEX: How many tokens could i buy with my current BNB balance?
Is the Value of tokens worth more than the BNB balance?
Yes
Calculate % of the tokens to sell
(Attempt to match the value of BNB to the value of Tokens)
Calculation Failure!
No
Sell between 0% - 50% of your tokens
Add tokens and BNB to DEX Liquidity Pool (Generates LP Tokens) AND Reset Liquidity Lock to 1 year
Sell half of the tokens
External (Processed by the DEX, not us)
Is any BNB Left Over?
Yes
Dex returns any left over BNB
Other Projects with liquidity taxes

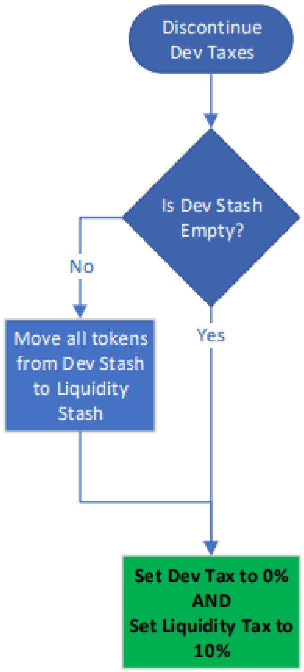
Process Application Taxes:

These are the taxes collected by the mobile app


Process Wallet Tax
Send 50% of Wallet Taxes to Chillieman
Use the ***BuyTokens*** function w/ 50% of current BNB balance
Use the ***AddLiquidity*** function
Generate Liquidity Tokens
There may be tokens or BNB left over
Place any leftovers in Liquidity Stash
Is Liquidity Stash greater than 1%?
Yes
Process Liquidity Stash (Page 2)
Color Legend
Done by Wallet Contract
Done by Token Contract
Both Contracts working together

Development Stash:


Claim Dev Taxes
Is Dev Stash greater than 1%?
No
Yes
Set Transfer Amount to Dev Stash balance
Set Transfer Amount to 1%
Transfer Tokens to Chillieman (Page 1)


Discontinue Dev Taxes
Is Dev Stash Empty?
No
Yes
Move all tokens from Dev Stash to Liquidity Stash
Set Dev Tax to 0% AND Set Liquidity Tax to 10%

Chillie's Utility Token:

Chillie's token is an ERC-20 token that is inspired by DeFi projects that automatically add liquidity by taxing the transfers of the token, so I did my own spin on this. My starting point was based on Open-Source code found in https://github.com/OpenZeppelin/openzeppelin-contracts.

I started by making a simple function for handling token transfers and implemented a straightforward function for collecting taxes. I made sure an account can never go above 1% of the total supply, or the transfer will fail. I believe this functionality will be a deterrent for whales and stop people from causing massive volatility if they chose to sell a massive portion of the supply. The only exception to this rule is reserved for the Smart Contract, Chillieman, and any Exchanges. Chillieman will have the ability to add decentralized exchanges to the list that is excluded from the max token limit. Any time an Exchange is added to the list, the Token will emit a **ExchangeAdded** event to retain full transparency. **ExchangeRemoved** event will be emitted if an exchange was removed from this special status.

I then started the code that will be responsible for adding liquidity. I took a look around other existing token with this Liquidity functionality and found a huge problem. The liquidity tokens were being purchased with the Owner listed as the recipient, giving the Owner instant access to all that liquidity. I analyzed the web and found another huge problem. UniCrypt wants 15,000 USDC to create a dynamic vault for the Liquidity Tokens… Since I'm a programmer, it just wouldn't feel right paying all that money when I can just implement a tiny bit of extra logic to get the Smart Contract to auto-lock its liquidity.

To resolve this Liquidity issue, the token assigns itself as the receiver of the liquidity tokens, not the Owner. With each new batch of liquidity, a timer is set (or reset) to 1 year until any liquidity can be claimed. The only way to get the liquidity out is to not have meaningful token activity for a full year – Which I hope never happens, so that Liquidity should never see the light of day! If the liquidity is ever claimed, the Contact will emit a **LiquidityUnlocked** event to retain full transparency. Liquidity is added after the Liquidity stash (collected from transaction taxes) has grown larger than 1% of total supply (10 billion). Liquidity is also added when the Mobile Application Fee stash is processed.

I added logic that allows Chillieman to turn the fees on and off globally (for everyone) which may come in handy in the future for events, holidays, and possible migrations. I added a lot of functionality that is designed to be used by the Wallet contract. A function for paying Mobile App Fees, and a function for processing those Fees, further fueling the liquidity of Chillie's token. The two projects will constantly fuel the success of the other, as long as people are using the Application!

--- Flash Forward --

I enhanced the function that is called when the App is processing Taxes. In rare cases where the App's Smart Contract would Inject much more ETH than exists in the liquidity pool, resulting in a ton of tokens left in the Token Address – If this occurs, a second processing of the Liquidity occurs – Just like what happens when the Liquidity Stash goes above 1% (See Page 4 for diagram). Any excess ETH will stay in the Token Contract and be used the next time liquidity is generated. This allows the case where the Token Contract will not need to sell any tokens to add liquidity, avoiding the price from falling from a future processing of liquidity.

Furthermore, the calculation of how many tokens to sell when generating liquidity is very robust. Each time liquidity is generated, the ETH balance of the token will creep to 0, and in some case even reaching 0!! This is extremely exciting because there will never be the case where ETH get lost forever because they were accidently sent to the Token address. If you look out there, there are Millions if not Billions of dollars' worth of ETH and BNB just sitting in a Tokens Balance, never to be used. - This contract solves that issue.

I used a fancy try / catch 🤓 in the Liquidity Generation function, which attempts to calculate a prime ratio of tokens to sell, but if the fancy logic does not work for whatever reason, the execution reverts to being simple and just selling Half of your tokens and provide liquidity with that. Apart of that calculation is to see if the $ value of ETH in the balance is greater than the $ value of the tokens that have been taxed to the liquidity stash. If the value of ETH is greater, then you don’t need to sell any tokens, and simply add everything you have!

Oh, and I attempted to make the code as readable as possible to aspiring Token developers can have a solid example to go by! =D