The only file that will be necessary to edit is ‘wallet.py’ inside the src/core/ directory. Inside of this are ten functions that need to be completed. These functions are already hooked up through the bot commands and thus won’t require any other work outside the functions to reflect in the bot’s behavior. This also means these functions must NOT be renamed and need to return exactly what is specified, in the format specified.

The WalletDebug class exists solely for testing purposes, allowing commands to be tested using fake user balances existing only inside the program’s run. It also implements its own dummy versions of most of the ten functions, which they currently hook through to keep things simple. Once these functions are implemented the WalletDebug class will be completely unneeded and can be either deleted or commented out (I would recommend the latter in case it ever comes in use for debugging in the future).

The functions -

get_staking_weight()
- Returns, as a string, the staking weight to display in the !blockchain command.

get_staking_reward()
- Returns, as a string, the staking reward to display in the !blockchain command

get_blockchain_size()
- Returns, as a string, the staking reward to display in the !blockchain command

get_balance(user)
- Argument ‘user’ is a discord User object
- Returns, as a float/decimal, a user’s current KekBot balance
- The commented out ‘address’ variable gets the user’s deposit address from the database table, so feel free to comment that back in and use that variable. It will already be verified elsewhere that the provided user has a registered deposit address.

create_deposit_address(user)
- Argument ‘user’ is a discord User object
- Returns, as a string, the deposit address
- The commented out line “database.set_deposit_address(user, address)” MUST be commented back in and used before the end of the function, with ‘address’ being the new deposit address

get_withdrawals(address)
- Argument ‘address’ is a deposit address of a user’s tipping account as a string
- Returns a LIST of transaction IDs as strings
- Gets recent withdrawal transaction IDs for the given address. The list does not need to have any particular limit as the !withdrawals command will trim the list itself

get_deposits(address)
- Argument ‘address’ is a deposit address of a user’s tipping account as a string
- Returns a LIST of transaction IDs as strings

- Gets recent deposit transaction IDs for the given address. The list does not need to have any particular limit as the !deposits command will trim the list itself

make_transaction(sender_address, receiver_address, amount)
- Argument ‘sender_address’ is a deposit address of the sender’s tipping account as a string
- Argument ‘receiver_address’ is a deposit address of the receiver’s tipping account as a string
- Argument ‘amount’ is the desired amount of KekCoin to send as a float
- Doesn’t return anything
- Already wrapped in a try/except block elsewhere so can be allowed to just throw an exception if it failed

is_valid_address(address)
- Argument ‘address’ is a supposed deposit address as a string
- Used for arg-checking !withdraw commands, to make sure the deposit address they listed actually makes sense - for example, if they listed ‘abcdefg’ as a deposit address.
- Returns True if the address is valid, otherwise returns False

can_withdraw_amount(amount, address)
- Argument ‘amount’ is a proposed withdrawal amount as a float
- Argument ‘address’ is a deposit address as a string
- Returns True if the given address is able to/has enough to withdraw the proposed amount, otherwise returns False