BTC is based on the concept of unspent outputs of UTXO
Initially the miners get coins for mining blocks.
The chain holds record for the coins they have. If they have not spent them they are termed as unspent and can be used as inputs.
In order to use coins as input, you need to get the transaction ID for the transaction which gave you ownership of the coin.
The coins are sent to and received from addresses which are derived using Elliptic curve cryptography.

There are three different networks.
Mainnet - The real BTC network
Testnet - For testing and trying out BTC
Regtest - Local network for developers

In order to interact with the network you need to have a node running on the network. Syncing up a full node can take days since it very computationally expensive and requires 100s of GBS of data to download.

The Testnet chain is smaller and requires less time to download but you can expect to spend a day or more waiting for it to sync up.

The regtest chain syncs up immediately and does not require huge amounts of space and bandwidth.

NOTE: If you're chain is not in sync, your node will not be able to transact on the chain and the code would throw errors since it will not find the transaction reference.

How the code works:
The code requires you to have a relevant address for the chain you wish to transact on.

NOTE: The utilities available online do not generate correct key formats and the code throws errors. If you need to generate keys use "./bitcoin-cli generatenewaddress" command
And to get the private key in wif format use the command

```
./bicoin-cli dumprivkey YOURGENERATEDADDRESS
```

Now you need to send coins to this address.
For Mainnet - Ask your friend
For TestNet - use a facuet
For Regtest - I have included a script that sends satoshis to the address specified in the wallet.js file and stores the transaction reference in a file for use, so you don't have to copy paste stuff.

The code then takes the TXID i.e the transaction reference and the key and makes a new transaction. It used the TXID as input since that coin has not been spent and then generates an OP_RETURN transaction with the data as you requested and broadcasts it to the network. It also fetches the raw transaction from the network to verify that the data was indeed added.

In order to interact with the full node you have manually issue laborious steps and have to copy paste a lot.

I have written a special server that uses RPC to interact directly with the BTC node. The file btc.js connects to and sends data to the server.

BTC.js => server.js => BTC node => Rest of the network.

It was a lot of work completing this project. Enjoy sending OP_RETURN transactions.