Day 15 blockchain bootcamp

Event Trigger Implementation

* Customer will interact with item manager.
* Each item in the item manager would be a smart contract to sends funds to.
* In item manager they would be three functions,
* Create item that take a identifier which would be a string, and itemPrice.
* Trigger payment it would take a index of some sort
* Trigger delivery it would also take a index.
* Mapping containing all of the items.
* Each item would contain a identifier, price and state.
* To use a enum, reference the smart contract.<name of enum>
* In the createitem, insert a new item. Pass a identifier and item price, then increment index indicating an item has been added.
* In the triggerpayment, it would set the status to paid.
* If it’s not a created state revert a exception saying that the items are further up the chain.
* If the payment is not the full price of item then revert gas then throw a exception saying you must pay full price.
* In the triggerdelivery, it would set the status to delivered.
* Create a event or data that can be searched on. Pass the index of the item and state of that item at that time. Emit event whenever a item is created or status changes.
* How to improve implementation.
* Each customer needs a specific function he needs to call
* Can make it easier by making new smart contract, one index of mapping. Reflected as extra address to send money there

Testing the blockchain

* In truffle whenever a project is unboxed. It has a test folder
* Javascript tests are more important than blockchain based tests
* In unit tests, use the contracts keyword. Give it a description, and a callback. In the callback it would provide 10 accounts. Results from web3.eth.getAccounts()
* Then for each test use the it keyword.
* It(description of specific test)
* In tests it uses truffle-contract not web3. Whenever deploying a contract in a unit test via a json file provided by truffle. It would deploy it via truffle-contract not web3
* Clean room tests, it will always start from scratch. It will always redeploys smart contracts, it doesn’t store the smart contract.
* **Truffle test** to run unit tests from test folder.
* Truffle test will go into truffle.config.js file find port and node configuration to find truffle development console on given port.