Database

Receipt class

Variables:

- Int order_id
    - This ID connects each receipt to its addons in the database and to the live orders
- Float subtotal
    - total based on purely store prices
- Float tax
    - taxes associated with each price
- Float total
    - subtotal + tax
- Item_entry[] order
    - array of each item in the order

Item_entry class

Functions:

- jsonify() - takes information from an object and translates it into a json object
    - This can just be imported since it exists already
- unpack() - takes a json object and makes an Item_entry object from it

Variables:

- String name
- Float cost
- String category_path (ie grill/grill_entrees)
- Bool[] sale_time [morning, lunch, dinner]
- Bool taxable
- String image
- Option[] addons

Option class

Variables:

- String name
- Float cost

Database Methods (implemented as microservices)

- Req 1.6

add_entry(Item_entry data) - adds a json object of an item to the database

- Returns the key for this new entry

- Store keys in a dictionary for lookup from item name?
- Req 1.1

remove_entry(string(?) key) - removes a json object of the corresponding key from the database
- Requires an item with this key in the database
- Req 1.2

update_entry(Item_entry data) - updates a json object in the database to a new state
- Could make this so it takes in (key, traits) and specifically updates those
- Req 1.3

query_database(string(?) key) - returns a json object from the database
- Can return item entries OR invoices
- Req 1.4, Req 1.5

Customer UI

Order class

Functions:
- get_total(Item_entry[] purchases)
    - returns double total
    - 5.4

Variables:
- Int ID
- Double cost
- Item_entry[] purchases

Customer UI Methods

load_menu() - queries database for the highest level categories
- Req 2.1, 2.4

load_category(string category) - queries database to navigate into a category
- Req 2.2

load_item(string item)
- 2.3, 2.5

add_to_order(Item_entry item) - adds an item with its options to an order in process
- 2.6

compile_order(Order order) - categorizes the order and prepares it as an object to be sent to the cook UI
- 4.3

send_order(Order order) - sends an order to the server to be filled
- ID association and sending are both network based and are the last step of the Customer UI
- Since information determined at the network stage is implanted into the order, encryption also needs to be done here

- Also sends a request to the database to add an invoice entry based on the order information
    - Req 1.5
- Reqs 4.1, 4.2, 4.4

Cook UI

get_orders() - receive any incoming, relevant orders
- 3.1, 4.1
- Verify orders are genuine by checking their verification
    - 4.5

list_orders(Order[] orders)
- 3.1
- gives an ordered list (by ID) of all the current orders

display_order(Order order)
- 3.1
- gives specific details of the selected order

complete_order(int idx)
- 3.2
- removes the order from list of orders
- ask for confirmation to prevent accidental order removal

Payment Processing

process_payment(total) - interacts with the physical parts set up to process credit/debit cards and ursinus id cards.
- 5.1, 5.2, 5.3
- returns verification that the order has been payed for