CZ3005

Artificial Intelligence

Assignment 4: Talking Box

Subway Sandwich Interactor

Kannan Shivani

U1822998H

TSP6

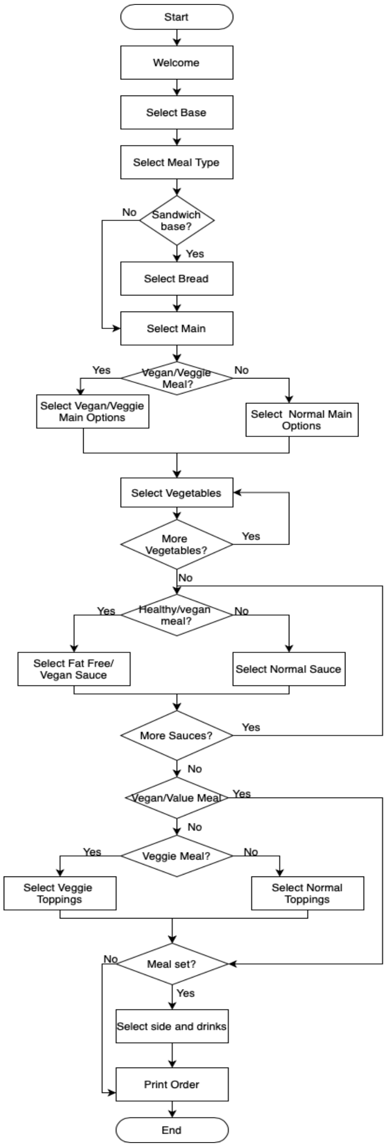
Solution Design

The customer places their order by specifying the type of base that they're ordering through an interactive prolog script. The script will then proceed to ask the customer to specify the meal type, bread, main, vegetables, sauces, toppings, side and drinks. A list of options, based on the customer's previous choices, would be shown for each category. At the end of the order, the customer will receive a receipt which contains the order summary of the choice they've made.

The following features have been added to the script:

- The customer can select multiple vegetables and sauces
- Any invalid selection by the customer will be handled
- The order will be reset once the current one is over/ individual order choices would be reset if error occurs in the particular choice

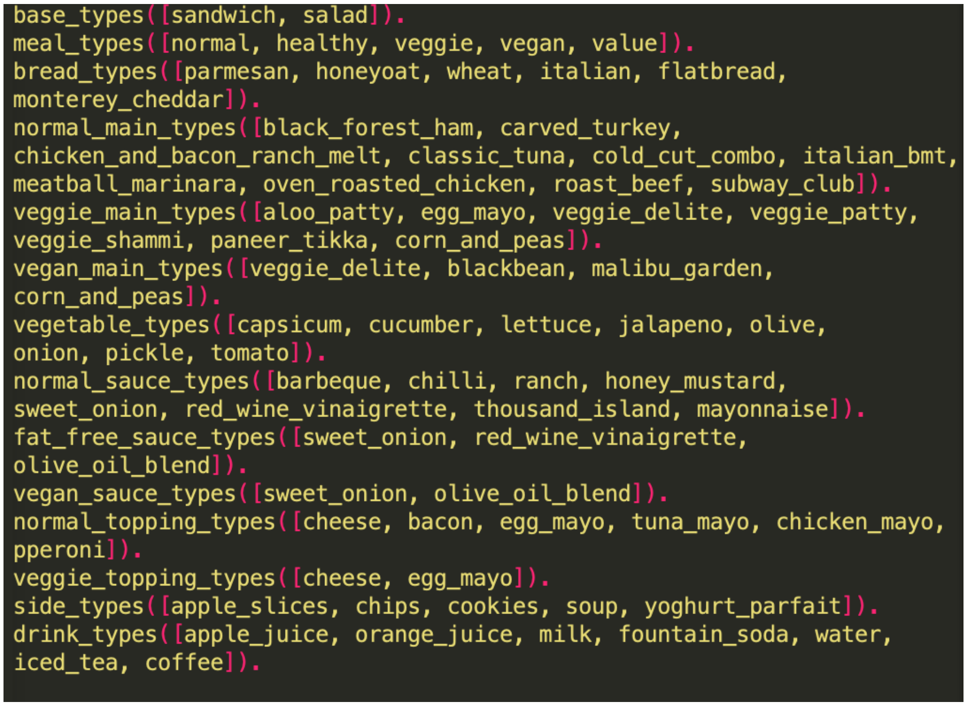
Flowchart



Implementation

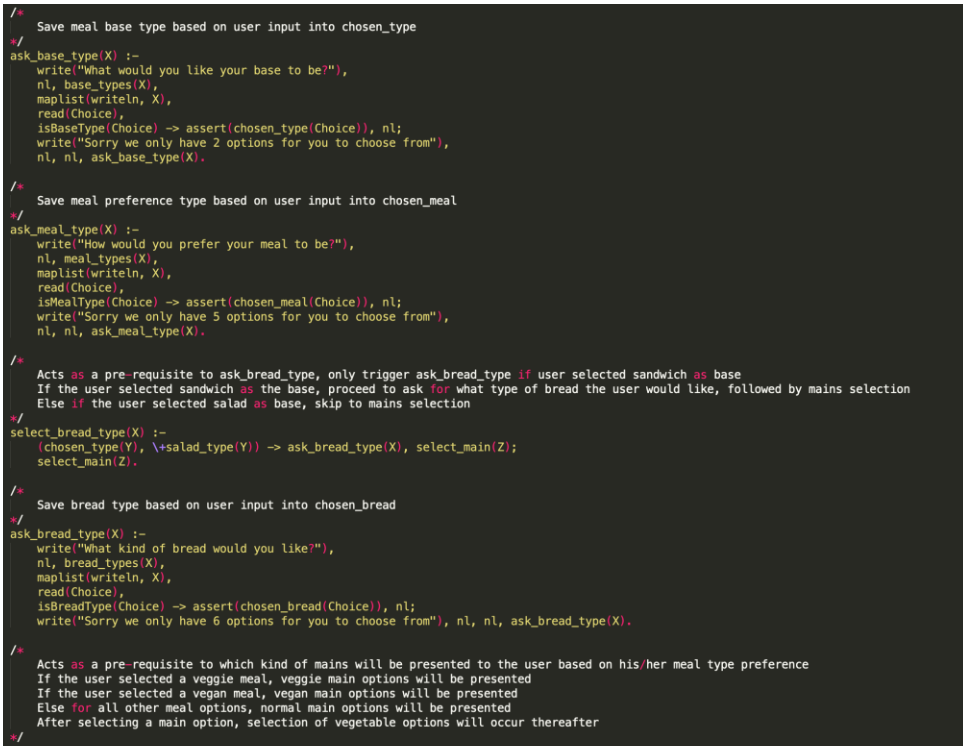
Prolog was used as the Knowledge Base as well as the interface for the assignment, “Subway Sandwich Interactor”. The prolog script, ‘subway-sandwich.pl’ contains the implementation code. The program will be started after calling the predicate ‘ask(0)’.

The Subway Singapore Website was referred for the list of breads, meats, vegetables and other options. Below is the code snippet that declares the facts.

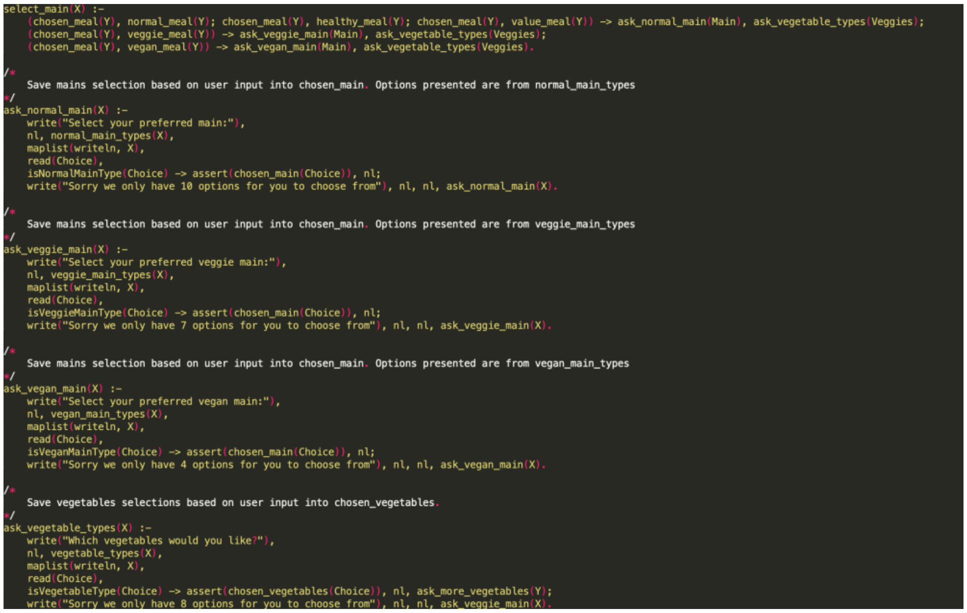


```
base_types([sandwich, salad]).
meal_types([normal, healthy, veggie, vegan, value]).
bread_types([parmesan, honeyoat, wheat, italian, flatbread,
monterey_cheddar]).
normal_main_types([black_forest_ham, carved_turkey,
chicken_and_bacon_ranch_melt, classic_tuna, cold_cut_combo, italian_bmt,
meatball_marinara, oven_roasted_chicken, roast_beef, subway_club]).
veggie_main_types([aloo_patty, egg_mayo, veggie_delite, veggie_patty,
veggie_shammi, paneer_tikka, corn_and_peas]).
vegan_main_types([veggie_delite, blackbean, malibu_garden,
corn_and_peas]).
vegetable_types([capsicum, cucumber, lettuce, jalapeno, olive,
onion, pickle, tomato]).
normal_sauce_types([barbeque, chilli, ranch, honey_mustard,
sweet_onion, red_wine_vinaigrette, thousand_island, mayonnaise]).
fat_free_sauce_types([sweet_onion, red_wine_vinaigrette,
olive_oil_blend]).
vegan_sauce_types([sweet_onion, olive_oil_blend]).
normal_topping_types([cheese, bacon, egg_mayo, tuna_mayo, chicken_mayo,
pperoni]).
veggie_topping_types([cheese, egg_mayo]).
side_types([apple_slices, chips, cookies, soup, yoghurt_parfait]).
drink_types([apple_juice, orange_juice, milk, fountain_soda, water,
iced_tea, coffee]).
```

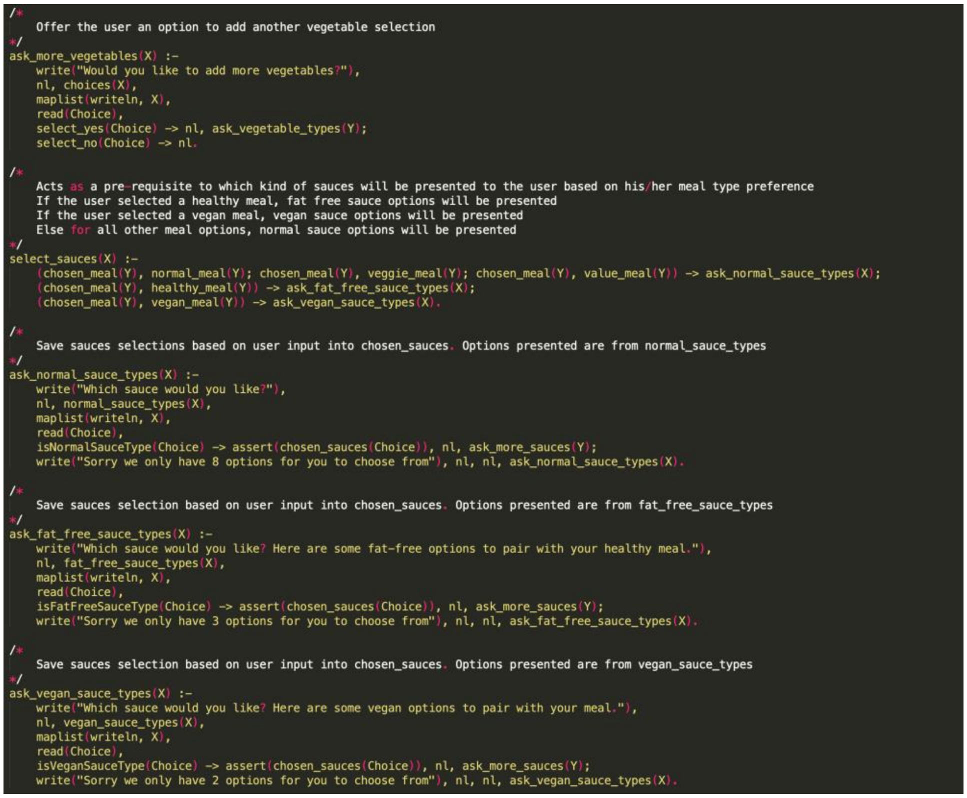
To show the list of options, I created a predicate, ask_type that will query the user which option they would like to choose, as well as read and write the user's choices into the list that contains the user's order. Then, to assert the fact, I created a rule, select_type, that is used to assert the fact dynamically based on the given item and the fact list. The code snippets below show the implementation of ask_type and select_type for all the lists (base type, meal type, etc.) in the program.



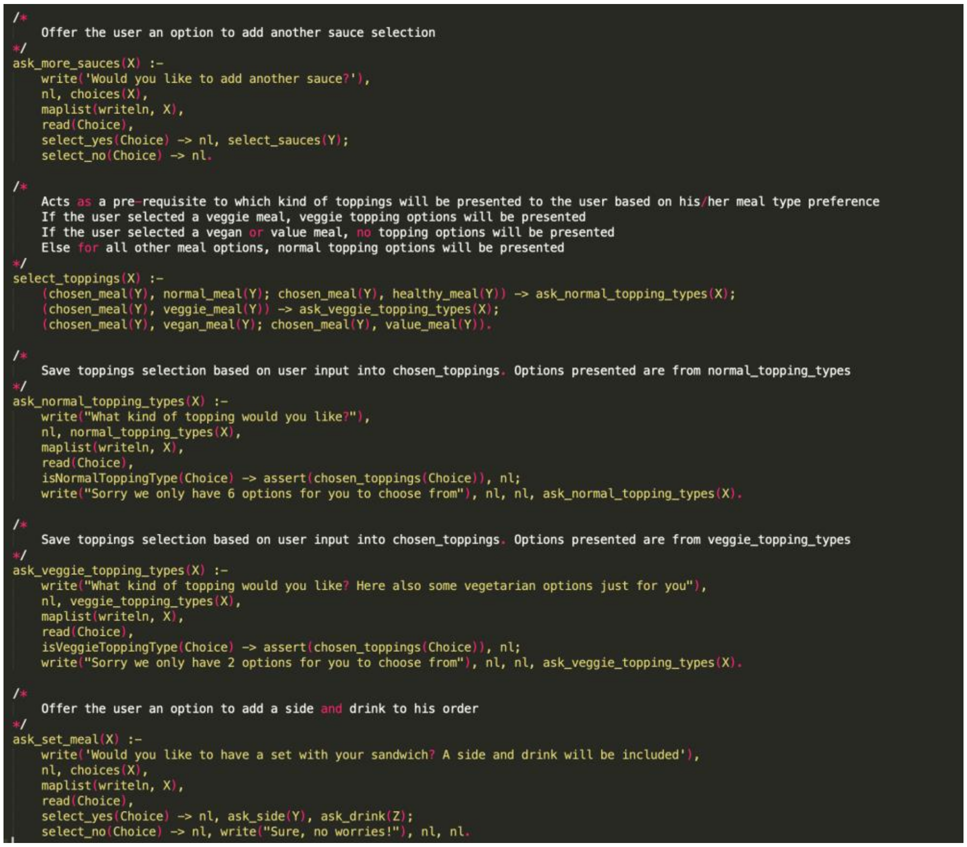
```
/*
    Save meal base type based on user input into chosen_type
*/
ask_base_type(X) :-
    write("What would you like your base to be?"),
    nl, base_types(X),
    maplist(writeln, X),
    read(Choice),
    isBaseType(Choice) -> assert(chosen_type(Choice)), nl;
    write("Sorry we only have 2 options for you to choose from"),
    nl, nl, ask_base_type(X).

/*
    Save meal preference type based on user input into chosen_meal
*/
ask_meal_type(X) :-
    write("How would you prefer your meal to be?"),
    nl, meal_types(X),
    maplist(writeln, X),
    read(Choice),
    isMealType(Choice) -> assert(chosen_meal(Choice)), nl;
    write("Sorry we only have 5 options for you to choose from"),
    nl, nl, ask_meal_type(X).

/*
    Acts as a pre-requisite to ask_bread_type, only trigger ask_bread_type if user selected sandwich as base
    If the user selected sandwich as the base, proceed to ask for what type of bread the user would like, followed by mains selection
    Else if the user selected salad as base, skip to mains selection
*/
select_bread_type(X) :-
    (chosen_type(Y), \+salad_type(Y)) -> ask_bread_type(X), select_main(Z);
    select_main(Z).

/*
    Save bread type based on user input into chosen_bread
*/
ask_bread_type(X) :-
    write("What kind of bread would you like?"),
    nl, bread_types(X),
    maplist(writeln, X),
    read(Choice),
    isBreadType(Choice) -> assert(chosen_bread(Choice)), nl;
    write("Sorry we only have 6 options for you to choose from"), nl, nl, ask_bread_type(X).

/*
    Acts as a pre-requisite to which kind of mains will be presented to the user based on his/her meal type preference
    If the user selected a veggie meal, veggie main options will be presented
    If the user selected a vegan meal, vegan main options will be presented
    Else for all other meal options, normal main options will be presented
    After selecting a main option, selection of vegetable options will occur thereafter
*/
```



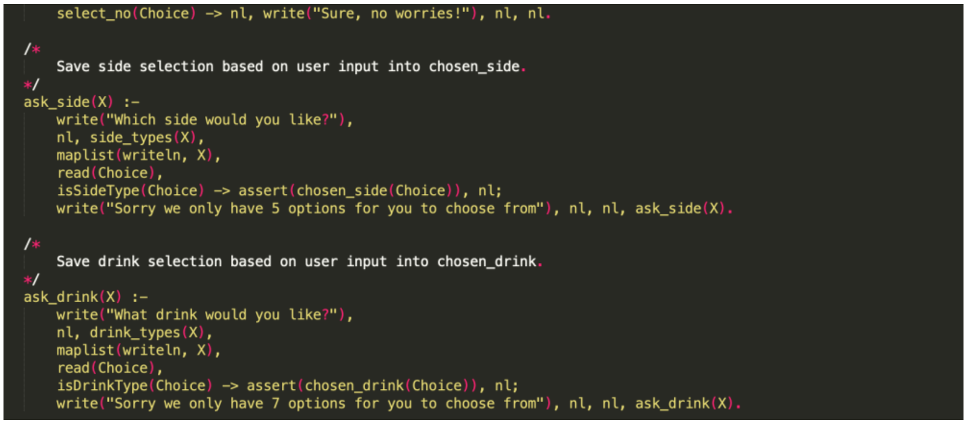
```
select_main(X) :-
    (chosen_meal(Y), normal_meal(Y); chosen_meal(Y), healthy_meal(Y); chosen_meal(Y), value_meal(Y)) -> ask_normal_main(Main), ask_vegetable_types(Veggies);
    (chosen_meal(Y), veggie_meal(Y)) -> ask_veggie_main(Main), ask_vegetable_types(Veggies);
    (chosen_meal(Y), vegan_meal(Y)) -> ask_vegan_main(Main), ask_vegetable_types(Veggies).

/*
    Save mains selection based on user input into chosen_main. Options presented are from normal_main_types
*/
ask_normal_main(X) :-
    write("Select your preferred main:"),
    nl, normal_main_types(X),
    maplist(writeln, X),
    read(Choice),
    isNormalMainType(Choice) -> assert(chosen_main(Choice)), nl;
    write("Sorry we only have 10 options for you to choose from"), nl, nl, ask_normal_main(X).

/*
    Save mains selection based on user input into chosen_main. Options presented are from veggie_main_types
*/
ask_veggie_main(X) :-
    write("Select your preferred veggie main:"),
    nl, veggie_main_types(X),
    maplist(writeln, X),
    read(Choice),
    isVeggieMainType(Choice) -> assert(chosen_main(Choice)), nl;
    write("Sorry we only have 7 options for you to choose from"), nl, nl, ask_veggie_main(X).

/*
    Save mains selection based on user input into chosen_main. Options presented are from vegan_main_types
*/
ask_vegan_main(X) :-
    write("Select your preferred vegan main:"),
    nl, vegan_main_types(X),
    maplist(writeln, X),
    read(Choice),
    isVeganMainType(Choice) -> assert(chosen_main(Choice)), nl;
    write("Sorry we only have 4 options for you to choose from"), nl, nl, ask_vegan_main(X).

/*
    Save vegetables selections based on user input into chosen_vegetables.
*/
ask_vegetable_types(X) :-
    write("Which vegetables would you like?"),
    nl, vegetable_types(X),
    maplist(writeln, X),
    read(Choice),
    isVegetableType(Choice) -> assert(chosen_vegetables(Choice)), nl, ask_more_vegetables(Y);
    write("Sorry we only have 8 options for you to choose from"), nl, nl, ask_veggie_main(X).
```



```
/*
    Offer the user an option to add another vegetable selection
*/
ask_more_vegetables(X) :-
    write("Would you like to add more vegetables?"),
    nl, choices(X),
    maplist(writeln, X),
    read(Choice),
    select_yes(Choice) -> nl, ask_vegetable_types(Y);
    select_no(Choice) -> nl.

/*
    Acts as a pre-requisite to which kind of sauces will be presented to the user based on his/her meal type preference
    If the user selected a healthy meal, fat free sauce options will be presented
    If the user selected a vegan meal, vegan sauce options will be presented
    Else for all other meal options, normal sauce options will be presented
*/
select_sauces(X) :-
    (chosen_meal(Y), normal_meal(Y); chosen_meal(Y), veggie_meal(Y); chosen_meal(Y), value_meal(Y)) -> ask_normal_sauce_types(X);
    (chosen_meal(Y), healthy_meal(Y)) -> ask_fat_free_sauce_types(X);
    (chosen_meal(Y), vegan_meal(Y)) -> ask_vegan_sauce_types(X).

/*
    Save sauces selections based on user input into chosen_sauces. Options presented are from normal_sauce_types
*/
ask_normal_sauce_types(X) :-
    write("Which sauce would you like?"),
    nl, normal_sauce_types(X),
    maplist(writeln, X),
    read(Choice),
    isNormalSauceType(Choice) -> assert(chosen_sauces(Choice)), nl, ask_more_sauces(Y);
    write("Sorry we only have 8 options for you to choose from"), nl, nl, ask_normal_sauce_types(X).

/*
    Save sauces selection based on user input into chosen_sauces. Options presented are from fat_free_sauce_types
*/
ask_fat_free_sauce_types(X) :-
    write("Which sauce would you like? Here are some fat-free options to pair with your healthy meal."),
    nl, fat_free_sauce_types(X),
    maplist(writeln, X),
    read(Choice),
    isFatFreeSauceType(Choice) -> assert(chosen_sauces(Choice)), nl, ask_more_sauces(Y);
    write("Sorry we only have 3 options for you to choose from"), nl, nl, ask_fat_free_sauce_types(X).

/*
    Save sauces selection based on user input into chosen_sauces. Options presented are from vegan_sauce_types
*/
ask_vegan_sauce_types(X) :-
    write("Which sauce would you like? Here are some vegan options to pair with your meal."),
    nl, vegan_sauce_types(X),
    maplist(writeln, X),
    read(Choice),
    isVeganSauceType(Choice) -> assert(chosen_sauces(Choice)), nl, ask_more_sauces(Y);
    write("Sorry we only have 2 options for you to choose from"), nl, nl, ask_vegan_sauce_types(X).
```

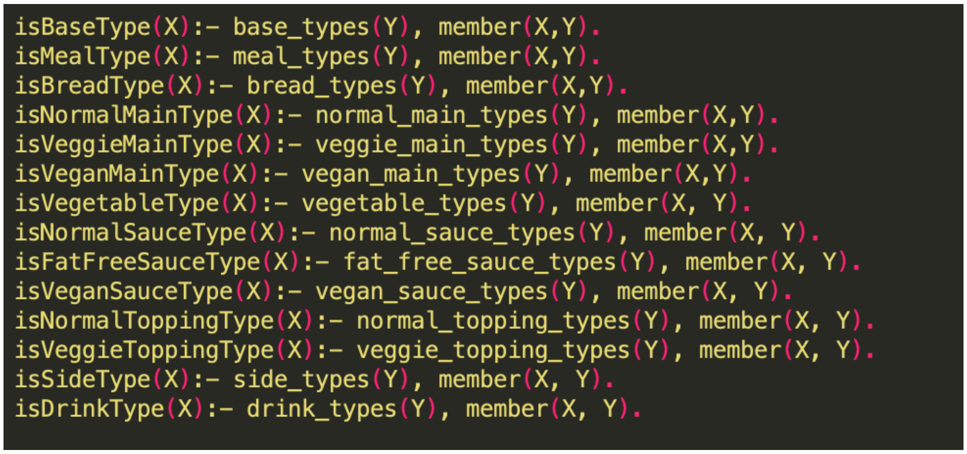


```
/*
    Offer the user an option to add another sauce selection
*/
ask_more_sauces(X) :-
    write('Would you like to add another sauce?'),
    nl, choices(X),
    maplist(writeln, X),
    read(Choice),
    select_yes(Choice) -> nl, select_sauces(Y);
    select_no(Choice) -> nl.

/*
    Acts as a pre-requisite to which kind of toppings will be presented to the user based on his/her meal type preference
    If the user selected a veggie meal, veggie topping options will be presented
    If the user selected a vegan or value meal, no topping options will be presented
    Else for all other meal options, normal topping options will be presented
*/
select_toppings(X) :-
    (chosen_meal(Y), normal_meal(Y); chosen_meal(Y), healthy_meal(Y)) -> ask_normal_topping_types(X);
    (chosen_meal(Y), veggie_meal(Y)) -> ask_veggie_topping_types(X);
    (chosen_meal(Y), vegan_meal(Y); chosen_meal(Y), value_meal(Y)).

/*
    Save toppings selection based on user input into chosen_toppings. Options presented are from normal_topping_types
*/
ask_normal_topping_types(X) :-
    write("What kind of topping would you like?"),
    nl, normal_topping_types(X),
    maplist(writeln, X),
    read(Choice),
    isNormalToppingType(Choice) -> assert(chosen_toppings(Choice)), nl;
    write("Sorry we only have 6 options for you to choose from"), nl, nl, ask_normal_topping_types(X).

/*
    Save toppings selection based on user input into chosen_toppings. Options presented are from veggie_topping_types
*/
ask_veggie_topping_types(X) :-
    write("What kind of topping would you like? Here also some vegetarian options just for you"),
    nl, veggie_topping_types(X),
    maplist(writeln, X),
    read(Choice),
    isVeggieToppingType(Choice) -> assert(chosen_toppings(Choice)), nl;
    write("Sorry we only have 2 options for you to choose from"), nl, nl, ask_veggie_topping_types(X).

/*
    Offer the user an option to add a side and drink to his order
*/
ask_set_meal(X) :-
    write('Would you like to have a set with your sandwich? A side and drink will be included'),
    nl, choices(X),
    maplist(writeln, X),
    read(Choice),
    select_yes(Choice) -> nl, ask_side(Y), ask_drink(Z);
    select_no(Choice) -> nl, write("Sure, no worries!"), nl, nl.
```



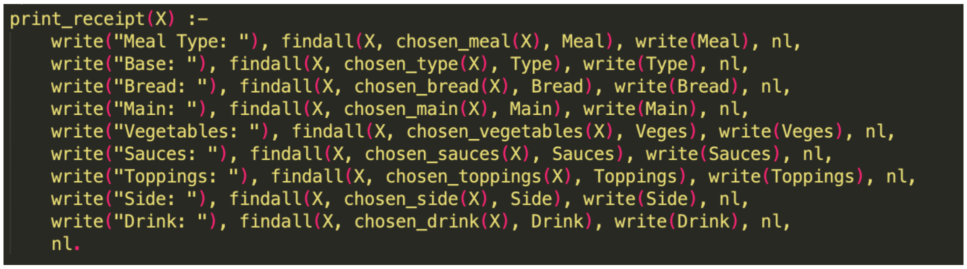
```
    select_no(Choice) -> nl, write("Sure, no worries!"), nl, nl.

/*
    Save side selection based on user input into chosen_side.
*/
ask_side(X) :-
    write("Which side would you like?"),
    nl, side_types(X),
    maplist(writeln, X),
    read(Choice),
    isSideType(Choice) -> assert(chosen_side(Choice)), nl;
    write("Sorry we only have 5 options for you to choose from"), nl, nl, ask_side(X).

/*
    Save drink selection based on user input into chosen_drink.
*/
ask_drink(X) :-
    write("What drink would you like?"),
    nl, drink_types(X),
    maplist(writeln, X),
    read(Choice),
    isDrinkType(Choice) -> assert(chosen_drink(Choice)), nl;
    write("Sorry we only have 7 options for you to choose from"), nl, nl, ask_drink(X).
```

A check is made to verify that the user's choice exists in the options list, and if not, an error message is shown (in the ask_type predicate). The code snippet below shows the checks which are to ensure that selected options are within the list of possible options for its corresponding type.



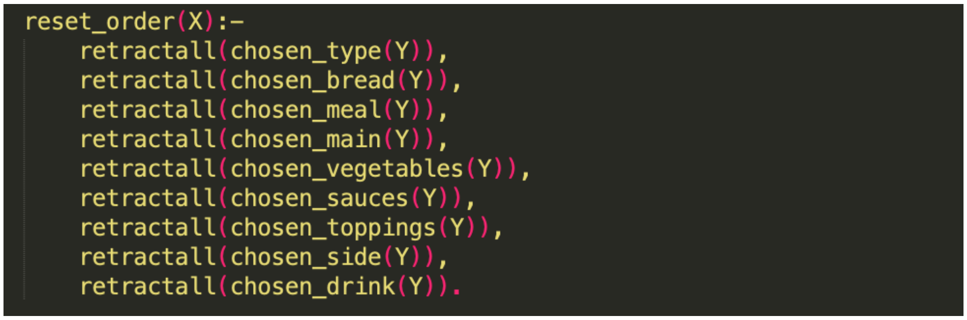
```
isBaseType(X):- base_types(Y), member(X,Y).
isMealType(X):- meal_types(Y), member(X,Y).
isBreadType(X):- bread_types(Y), member(X,Y).
isNormalMainType(X):- normal_main_types(Y), member(X,Y).
isVeggieMainType(X):- veggie_main_types(Y), member(X,Y).
isVeganMainType(X):- vegan_main_types(Y), member(X,Y).
isVegetableType(X):- vegetable_types(Y), member(X, Y).
isNormalSauceType(X):- normal_sauce_types(Y), member(X, Y).
isFatFreeSauceType(X):- fat_free_sauce_types(Y), member(X, Y).
isVeganSauceType(X):- vegan_sauce_types(Y), member(X, Y).
isNormalToppingType(X):- normal_topping_types(Y), member(X, Y).
isVeggieToppingType(X):- veggie_topping_types(Y), member(X, Y).
isSideType(X):- side_types(Y), member(X, Y).
isDrinkType(X):- drink_types(Y), member(X, Y).
```

Finally, I created the print_receipt predicates which writes into the order list, the user's choices. This is used to show the complete list of items that the user wishes to order, at the end of the program.



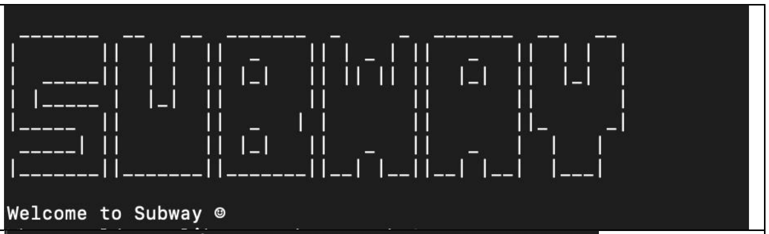
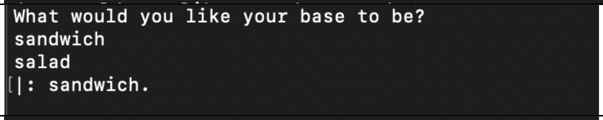
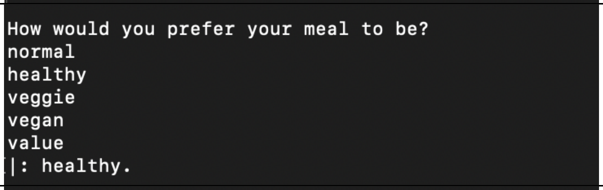
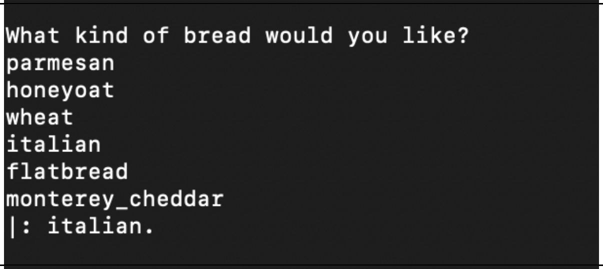
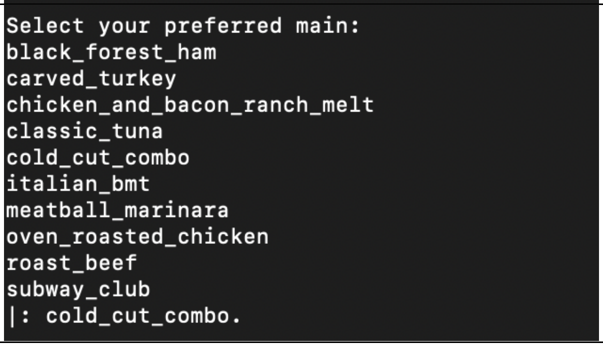
```
print_receipt(X) :-
    write("Meal Type: "), findall(X, chosen_meal(X), Meal), write(Meal), nl,
    write("Base: "), findall(X, chosen_type(X), Type), write(Type), nl,
    write("Bread: "), findall(X, chosen_bread(X), Bread), write(Bread), nl,
    write("Main: "), findall(X, chosen_main(X), Main), write(Main), nl,
    write("Vegetables: "), findall(X, chosen_vegetables(X), Veges), write(Veges), nl,
    write("Sauces: "), findall(X, chosen_sauces(X), Sauces), write(Sauces), nl,
    write("Toppings: "), findall(X, chosen_toppings(X), Toppings), write(Toppings), nl,
    write("Side: "), findall(X, chosen_side(X), Side), write(Side), nl,
    write("Drink: "), findall(X, chosen_drink(X), Drink), write(Drink), nl,
    nl.
```

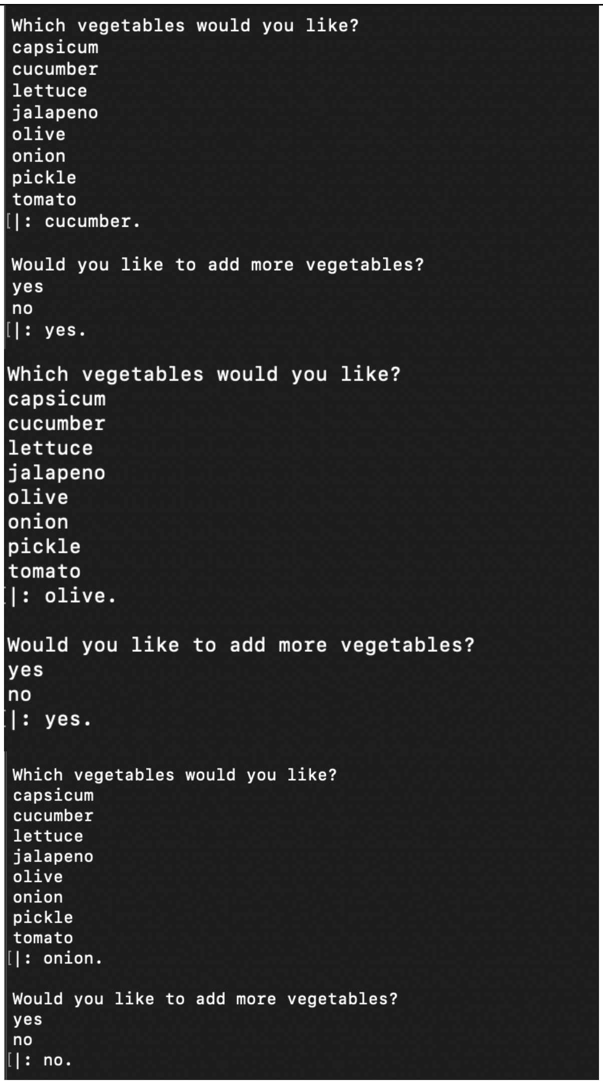
I used a reset predicate, reset_order, which will be used to reset the entire order once the current one is over or also after resetting a certain option.



```
reset_order(X):-
    retractall(chosen_type(Y)),
    retractall(chosen_bread(Y)),
    retractall(chosen_meal(Y)),
    retractall(chosen_main(Y)),
    retractall(chosen_vegetables(Y)),
    retractall(chosen_sauces(Y)),
    retractall(chosen_toppings(Y)),
    retractall(chosen_side(Y)),
    retractall(chosen_drink(Y)).
```

Program Execution

<table>
<tr><td>Welcome</td><td>Welcome to Subway ☻</td></tr>
<tr><td>Select Base</td><td>What would you like your base to be?
sandwich
salad
|: sandwich.</td></tr>
<tr><td>Select Meal Type</td><td>How would you prefer your meal to be?
normal
healthy
veggie
vegan
value
|: healthy.</td></tr>
<tr><td>Select Bread</td><td>What kind of bread would you like?
parmesan
honeyoat
wheat
italian
flatbread
monterey_cheddar
|: italian.</td></tr>
<tr><td>Select Main</td><td>Select your preferred main:
black_forest_ham
carved_turkey
chicken_and_bacon_ranch_melt
classic_tuna
cold_cut_combo
italian_bmt
meatball_marinara
oven_roasted_chicken
roast_beef
subway_club
|: cold_cut_combo.</td></tr>
</table>

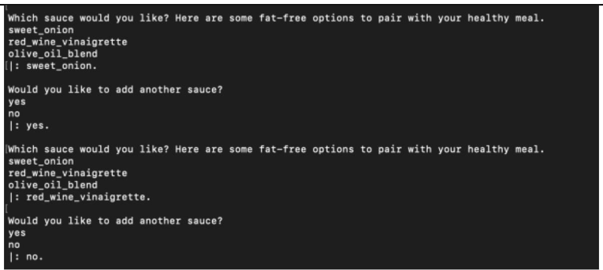
<table>
<tr><td>Select Vegetables</td><td>
Which vegetables would you like?
capsicum
cucumber
lettuce
jalapeno
olive
onion
pickle
tomato
|: cucumber.

Would you like to add more vegetables?
yes
no
|: yes.

Which vegetables would you like?
capsicum
cucumber
lettuce
jalapeno
olive
onion
pickle
tomato
|: olive.

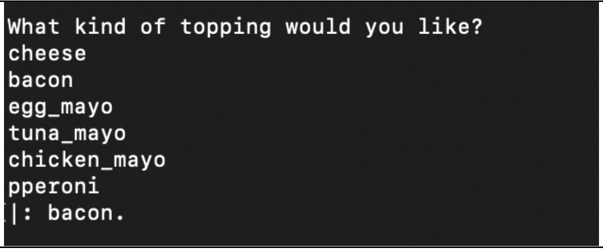
Would you like to add more vegetables?
yes
no
|: yes.

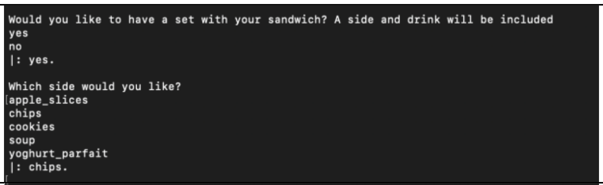
Which vegetables would you like?
capsicum
cucumber
lettuce
jalapeno
olive
onion
pickle
tomato
|: onion.

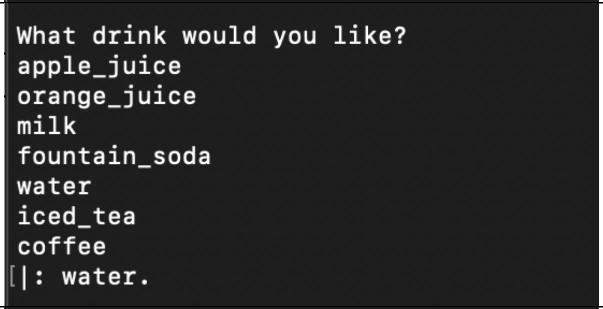
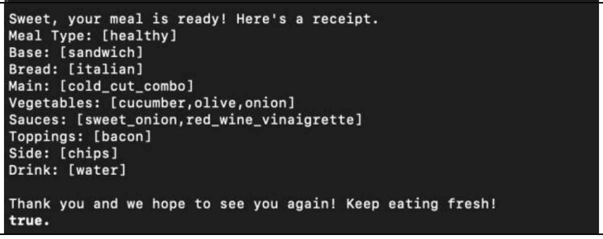
Would you like to add more vegetables?
yes
no
|: no.</td></tr>
<tr><td>Select Sauce</td><td>
Which sauce would you like? Here are some fat-free options to pair with your healthy meal.
sweet_onion
red_wine_vinaigrette
olive_oil_blend
|: sweet_onion.

Would you like to add another sauce?
yes
no
|: yes.

Which sauce would you like? Here are some fat-free options to pair with your healthy meal.
sweet_onion
red_wine_vinaigrette
olive_oil_blend
|: red_wine_vinaigrette.

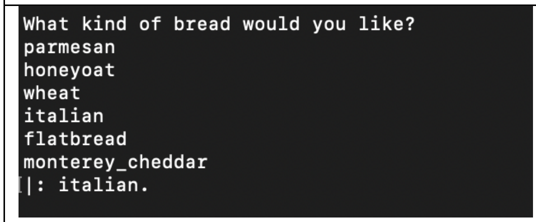
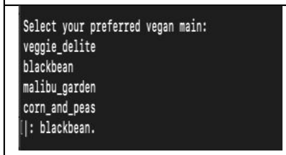
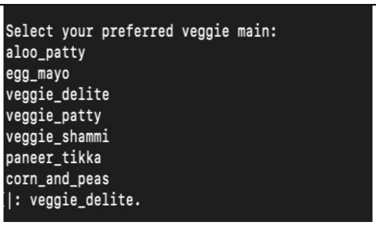
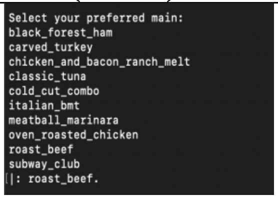
Would you like to add another sauce?
yes
no
|: no.</td></tr>
<tr><td>Select Toppings</td><td>
What kind of topping would you like?
cheese
bacon
egg_mayo
tuna_mayo
chicken_mayo
pperoni
|: bacon.</td></tr>
</table>

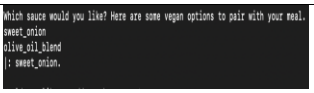
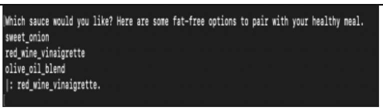
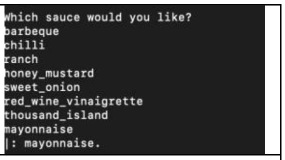
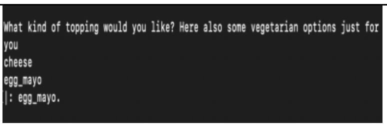
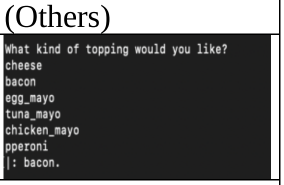
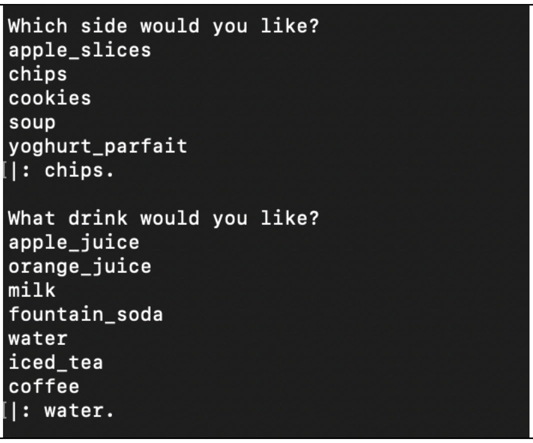
<table>
<tr><td>Select Side</td><td>
Would you like to have a set with your sandwich? A side and drink will be included
yes
no
|: yes.

Which side would you like?
apple_slices
chips
cookies
soup
yoghurt_parfait
|: chips.</td></tr>
<tr><td>Select Drink</td><td>
What drink would you like?
apple_juice
orange_juice
milk
fountain_soda
water
iced_tea
coffee
|: water.</td></tr>
<tr><td>Receipt</td><td>
Sweet, your meal is ready! Here's a receipt.
Meal Type: [healthy]
Base: [sandwich]
Bread: [italian]
Main: [cold_cut_combo]
Vegetables: [cucumber,olive,onion]
Sauces: [sweet_onion,red_wine_vinaigrette]
Toppings: [bacon]
Side: [chips]
Drink: [water]

Thank you and we hope to see you again! Keep eating fresh!
true.</td></tr>
</table>

Variations

<table>
<tr><td>Select Bread</td><td colspan="2">Sandwich Base</td><td>Salad Base</td></tr>
<tr><td></td><td colspan="2">
What kind of bread would you like?
parmesan
honeyoat
wheat
italian
flatbread
monterey_cheddar
|: italian.</td><td>NA</td></tr>
<tr><td>Select Main</td><td>Vegan Meal</td><td>Veggie Meal</td><td>Normal Meal(Others)</td></tr>
<tr><td></td><td>
Select your preferred vegan main:
veggie_delite
blackbean
malibu_garden
corn_and_peas
|: blackbean.</td><td>
Select your preferred veggie main:
aloo_patty
egg_mayo
veggie_delite
veggie_patty
veggie_shammi
paneer_tikka
corn_and_peas
|: veggie_delite.</td><td>
Select your preferred main:
black_forest_ham
carved_turkey
chicken_and_bacon_ranch_melt
classic_tuna
cold_cut_combo
italian_bmt
meatball_marinara
oven_roasted_chicken
roast_beef
subway_club
|: roast_beef.</td></tr>
<tr><td>Select Sauce</td><td>Vegan Meal</td><td>Healthy Meal</td><td>Normal Meal(Others)</td></tr>
</table>

<table>
<tr><td></td><td>Which sauce would you like? Here are some vegan options to pair with your meal.
sweet_onion
olive_oil_blend
|: sweet_onion.</td><td>Which sauce would you like? Here are some fat-free options to pair with your healthy meal.
sweet_onion
red_wine_vinaigrette
olive_oil_blend
|: red_wine_vinaigrette.</td><td>Which sauce would you like?
barbeque
chilli
ranch
honey_mustard
sweet_onion
red_wine_vinaigrette
thousand_island
mayonnaise
|: mayonnaise.</td></tr>
<tr><td rowspan="2">Select Toppings</td><td>Vegan/Value Meal</td><td>Veggie</td><td>Normal Meal (Others)</td></tr>
<tr><td>NA</td><td>What kind of topping would you like? Here also some vegetarian options just for you
cheese
egg_mayo
|: egg_mayo.</td><td>What kind of topping would you like?
cheese
bacon
egg_mayo
tuna_mayo
chicken_mayo
pperoni
|: bacon.</td></tr>
<tr><td rowspan="2">Select Side and Drinks</td><td colspan="2">Set Meal</td><td>No Set Meal</td></tr>
<tr><td colspan="2">Which side would you like?
apple_slices
chips
cookies
soup
yoghurt_parfait
|: chips.

What drink would you like?
apple_juice
orange_juice
milk
fountain_soda
water
iced_tea
coffee
|: water.</td><td>NA</td></tr>
</table>