NANYANG TECHNOLOGICAL UNIVERSITY

Lab 3: Subway Sandwich Interactor

CZ3005: Artificial Intelligence

Poh Ying Xuan
U1821489B
TS4

Exercise: Subway sandwich interactor

Assumptions

This subway sandwich interactor follows the flow of a subway branch in Singapore.

Normal meal would be able to get everything.
Healthy meal would be able to get everything except fatty sauces.
Value meal would be able to get everything except top-ups.
Vegan meal would be able to get everything except cheese and meat. In summary, only vegetables are offered.
Veggie meal would be able to get everything except meat. Cheese would be offered.

Prolog codes with explanation

This are meal checkers to verify if the following meals are selected so that the appropriate response could be given.

```
is_healthy_meal(healthy).
is_value_meal(value).
is_vegan_meal(vegan).
is_veggie_meal(veggie).
```

These are all the list of available meals.

```
meals([healthy, normal, value, vegan, veggie]).
```

These are the list of breads

```
breads([italian, hearty_italian, mulitgrain, honey_oat, parmesan,
flatbread, parmesan_oregano]).
```

These are the list of Veggies

```
veggies([lettuce, tomatoes, cucumbers, green_peppers, onions, olives,
pickles, jalapenos]).
```

These are the list of sauces, fatty sauces and non-fatty sauces.

```
fatty_sauces([spicy_mayo, mayo, bbq, ranch, chipotle_southwest, mustard]).

non_fatty_sauces([honey_mustard, sweet_onion, chilli, ketchup, salt,
pepper]).
```

These are the list of top-ups, meat top-ups, vegan top-ups and veggie top-ups.

```
meat_topups([tuna, bacon]).

vegan_topups([avocado]).

veggie_topups([american_cheese, shredded_cheddar, egg_mayo]).
```

These are the list of sides available.

```
sides([chips, cookies, hashbrowns]).
```

These are the list of drinks.

```
drinks([water, coke, ice_lemon_tea, hot_cappuccino,
hot_coffee_without_milk, hot_tea_without_milk, orange_juice]).
```

These are the list of mains, separated into veggies and mains.

```
mains([chicken_bacon_ranch, chicken_teritaki, cold_cut_trio, egg_mayo,
italian_bmt, meatball_marinara_melt, roast_beef, roasted_chicken_breast,
steak_cheese, subway_club, subway_melt, turkey]).

veggie_mains([veggie_delite, veggie_patty]).
```

The following code is used to show all the available meal in prolog.

```
ask_meal(X):-meals(X).
```

The following code is used to show all the available breads in prolog.

```
ask_bread(X):-breads(X).
```

The following code is used to generate the mains according to the meal chosen, which would be labelled under chosen_meal(Y).
If the chosen_meal(Y) is vegan or veggie, veggie mains would be returned. Otherwise, all mains would be shown, which is the combination of all meat and veggies meals.

```
ask_mains(X):-chosen_meal(Y)->((\+is_vegan_meal(Y), \+is_veggie_meal(Y))-
>(veggie_mains(M1), mains(M2), append(M1, M2, X)); veggie_mains(X)).
```

The following code is used to show all the available veggies in prolog.

```
ask_veggie(X):-veggies(X).
```

The following code is used to generate the top-ups according to the meal chosen, which would be labelled under chosen_meal(Y).
If the chosen_meal(Y) is value meal, nothing would be returned, as value meal has no top-ups available.
If the chosen_meal(Y) is vegan meal, vegan top-ups would be returned, It offers only vegan top-up available
If the chosen_meal(Y) is veggie meal, veggies top-ups would be returned, It offers the combination of both veggies and vegan top-up available. Which is the inclusion of cheese top-ups.
Otherwise, all top-ups would be offered. It would be the combination of all the vegan, veggies and meat top-ups available.

```
ask_topup(X):-chosen_meal(Y)->
(\+is_value_meal(Y)->
(is_vegan_meal(Y)->vegan_topups(X);
 is_veggie_meal(Y)->(vegan_topups(V1), veggie_topups(V2), append(V1, V2,
X));
 ((vegan_topups(V1), veggie_topups(V2), append(V1, V2, M1)),
meat_topups(M2), append(M1, M2, X)))).
```

If the chosen_meal(Y) is healthy meal, only non-fatty sauces would be returned. Otherwise, every sauces would be offered

```
ask_sauces(X):-chosen_meal(Y)->(is_healthy_meal(Y)->non_fatty_sauces(X);
(non_fatty_sauces(S1), fatty_sauces(S2), append(S1, S2, X))).
```

The following code is used to show all the available sides in prolog.

```
ask_sides(X):-sides(X).
```

The following code is used to show all the available drinks in prolog.

```
ask_drinks(X):-drinks(X).
```

The following code is used to show all the choices made in prolog.

```
show_meals(X) :- findall(Y, chosen_meal(Y), X).
show_breads(X) :- findall(Y, chosen_bread(Y), X).
show_mains(X) :- findall(Y, chosen_main(Y), X).
show_veggies(X) :- findall(Y, chosen_veggie(Y), X).
show_sauces(X) :- findall(Y, chosen_sauce(Y), X).
show_topups(X) :- findall(Y, chosen_topup(Y), X).
show_sides(X) :- findall(Y, chosen_side(Y), X).
show_drinks(X) :- findall(Y, chosen_drink(Y), X).
```

Interface and how it works

For this illustration, we will explain what happens when a user clicks on the value meal. The system would ask the user a series of questions that would ask the user for their choices through a flow of events:

Index → Meal → Bread →Main → veggies → Top-ups → Sauces → Sides → Drinks → Summary

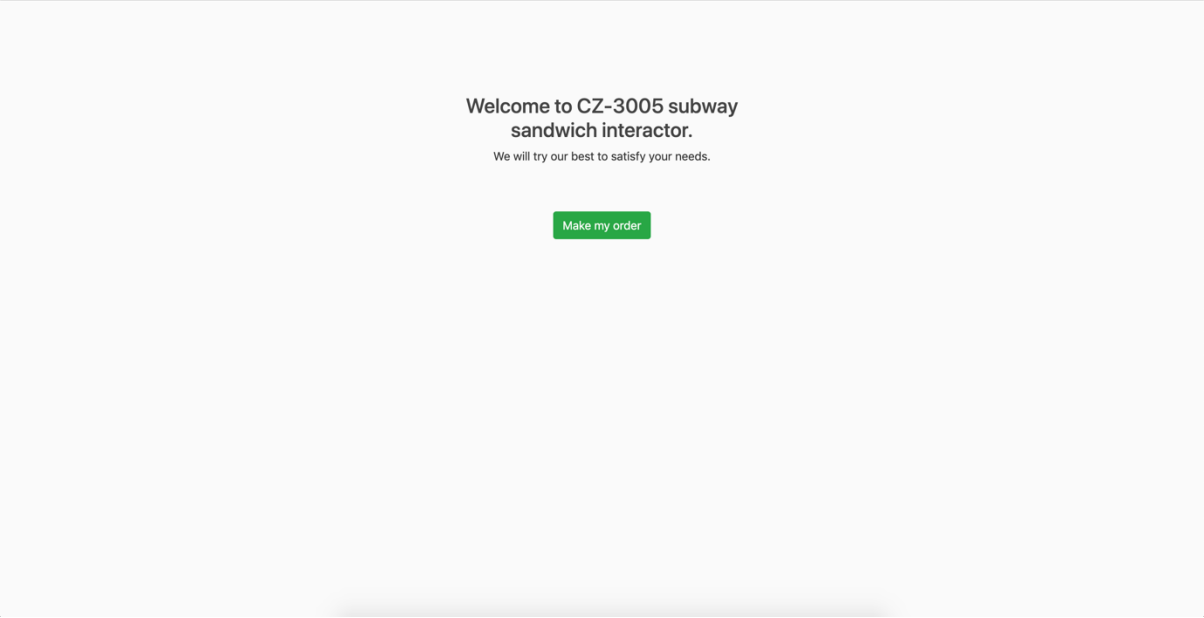

Figure 1: index page of the program

Over at figure 1, retractall commands would be implemented such that it starts off from a clean start.

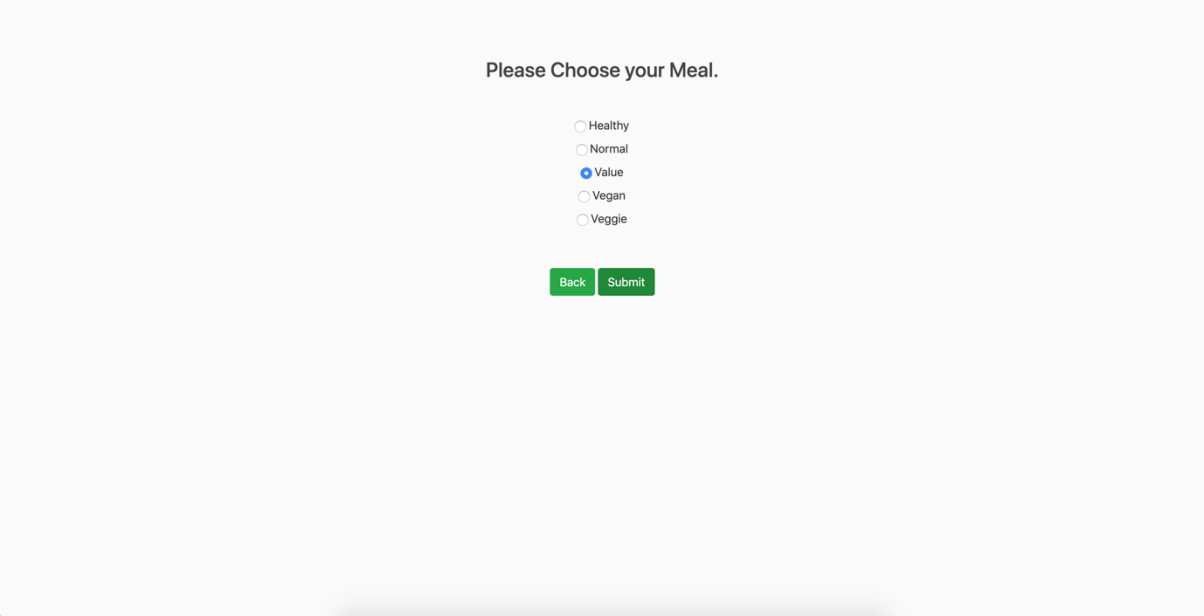

Figure 2: Asked to choose your meal

When this page is loaded, “ask_meal(X)” would be implemented into the prolog system to generate all the available meal.
Upon pressing submit button in figure 2, “assert(chosen_meal(value))”, would be implemented into the prolog.

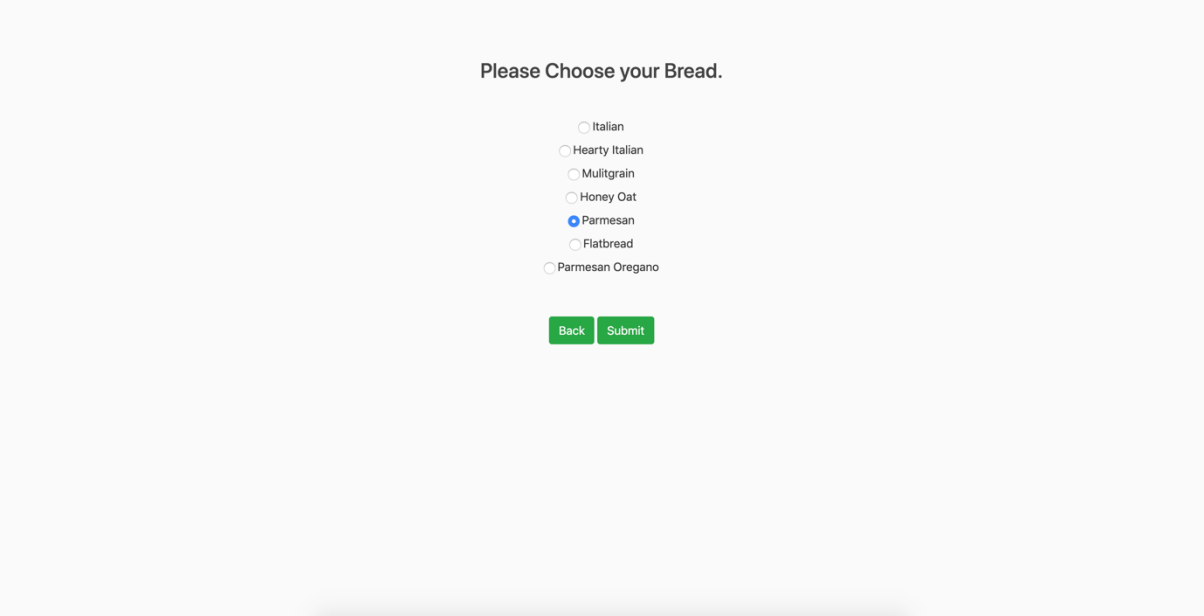

Figure 3: Ask to choose your bread

When this page is loaded, “ask_bread(X)” would be implemented into the prolog system to generate all the available breads.
Upon pressing submit button in figure 3, “assert(chosen_bread(parmesan))”, would be implemented into the prolog.

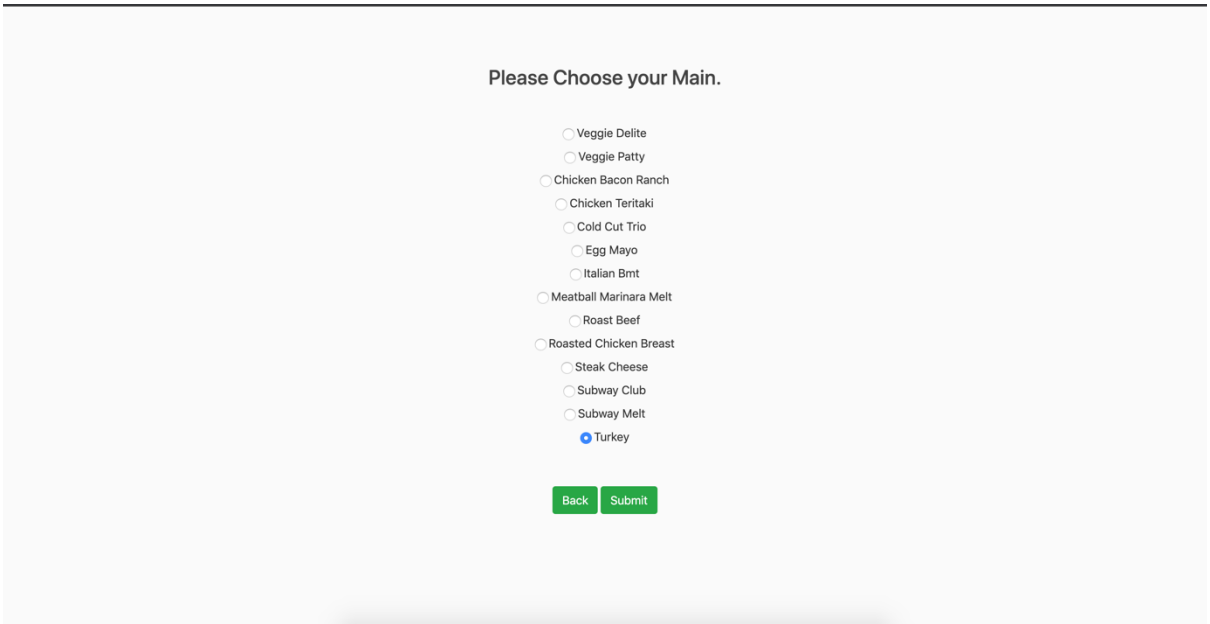

Figure 4: Asked to choose your mains

When this page is loaded, “ask_main(X)” would be implemented into the prolog system to generate all the available meal.
Upon pressing submit button in figure 4, “assert(chosen_main(turkey))”, would be implemented into the prolog.

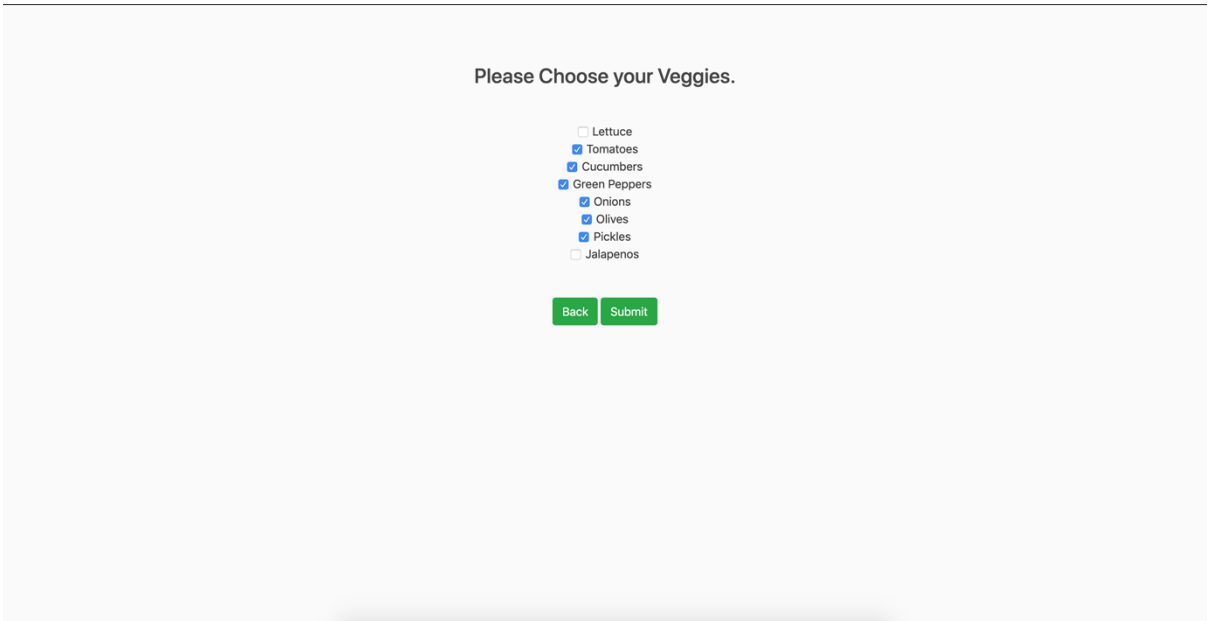

Figure 5: Ask to choose your veggies

When this page is loaded, “ask_veggie(X)” would be implemented into the prolog system to generate all the available veggies.
Upon pressing submit button in figure 5, it would incur a loop of assert commands into prolog. In this case, “assert(chosen_veggie(tomatoes))”, “assert(chosen_veggie(cucumbers))”, so on and so forth until “assert(chosen_veggie(pickles))”.

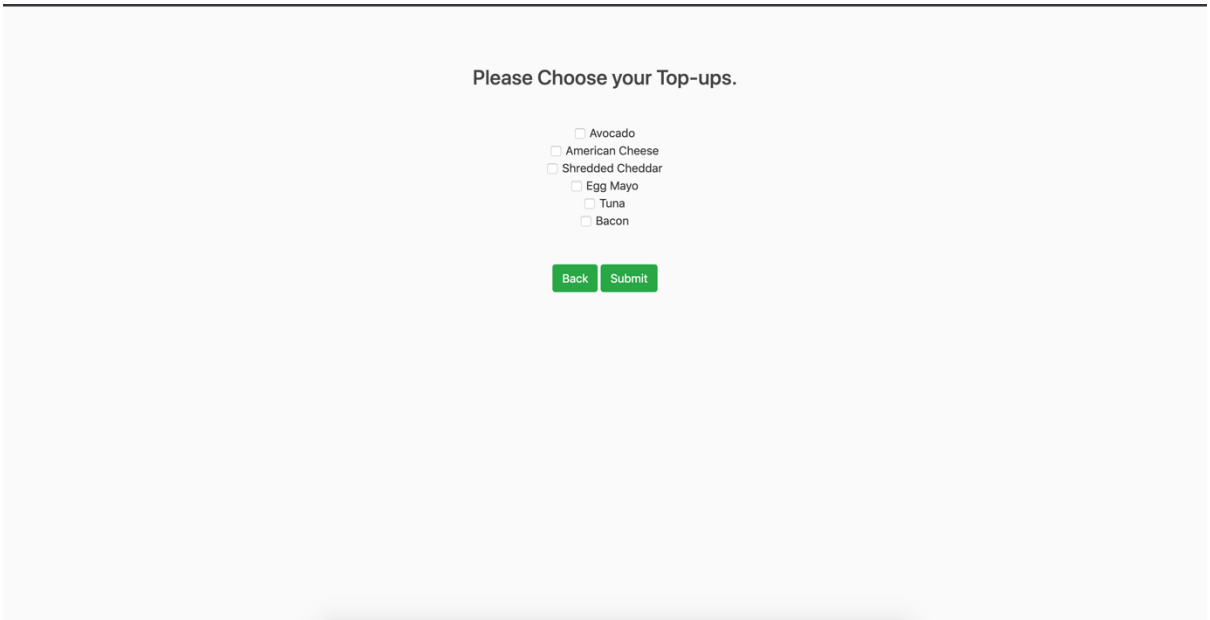

Figure 6: Ask to choose your top-ups.

When this page is loaded, “ask_topup(X)” would be implemented into the prolog system to generate all the available top-ups depending on your meal choices.
However, in this example for value meal, there would not be any top-ups. Hence this would not be shown, because “ask_top(X)” would return false. As a result, it will proceed straight over to the next page.

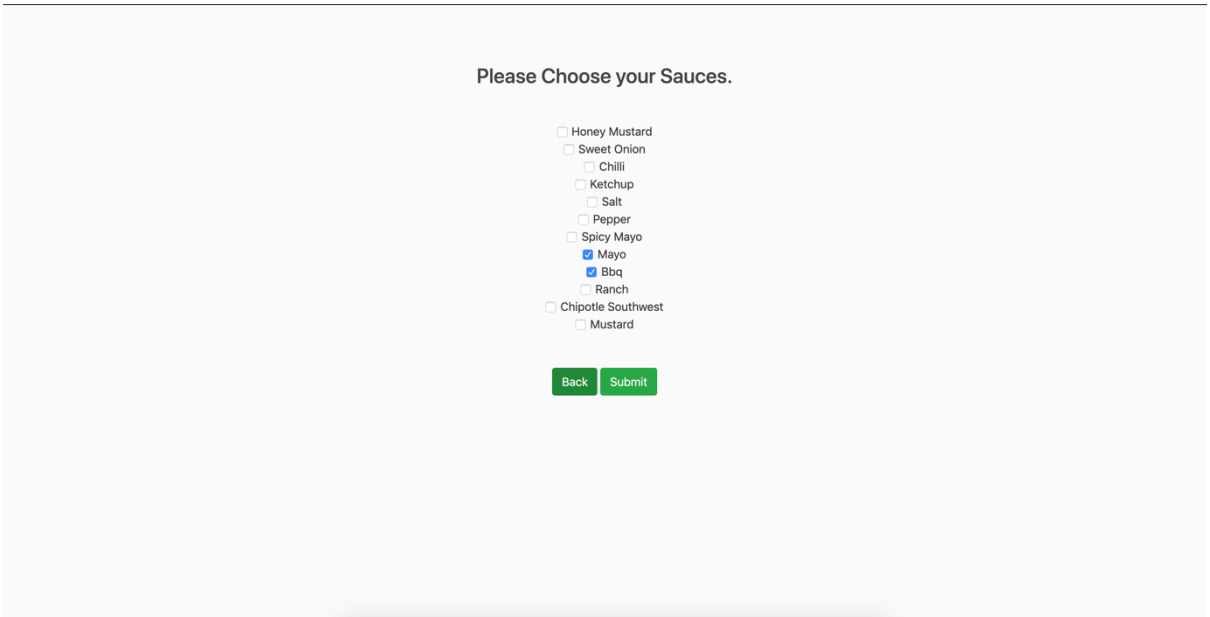

Figure 7: Asked to choose your sauces

When this page is loaded, “ask_sauce(X)” would be implemented into the prolog system to generate all the available sauces.
Upon pressing submit button in figure 7, it would incur a loop of assert commands into prolog. In this case, “assert(chosen_sauce(mayo))” and “assert(chosen_sauce(bbq))” commands would be asserted into prolog.

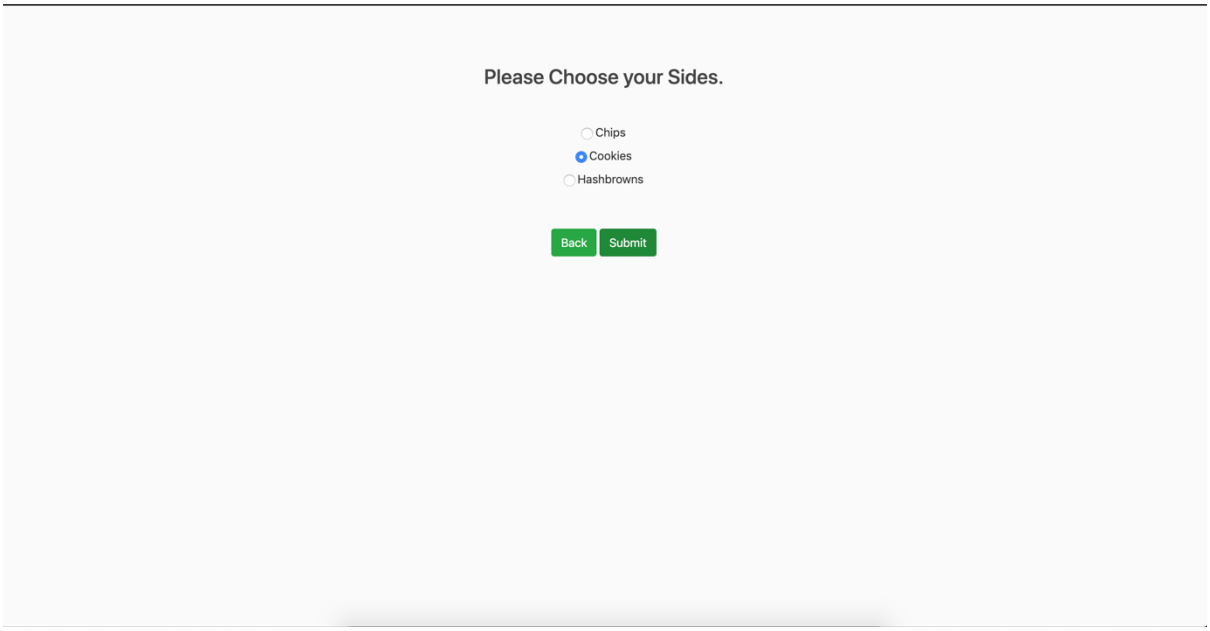

Figure 8: Asked to choose your sides

When this page is loaded, “ask_side(X)” would be implemented into the prolog system to generate all the available sides.
Upon pressing submit button in figure 8, “assert(chosen_side(cookies))”, would be implemented into the prolog.

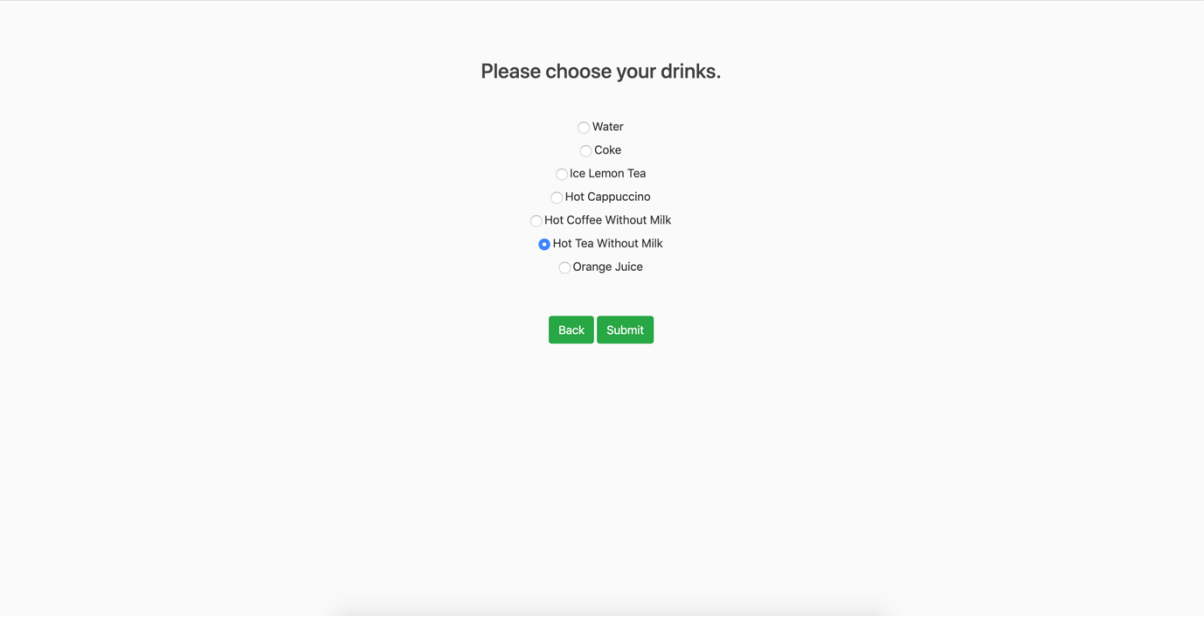

Figure 9: Asked to choose your drinks

When this page is loaded, “ask_drink(X)” would be implemented into the prolog system to generate all the available drinks.
Upon pressing submit button in figure 3, “assert(chosen_drink(hot_tea_without_milk))”, would be implemented into the prolog.

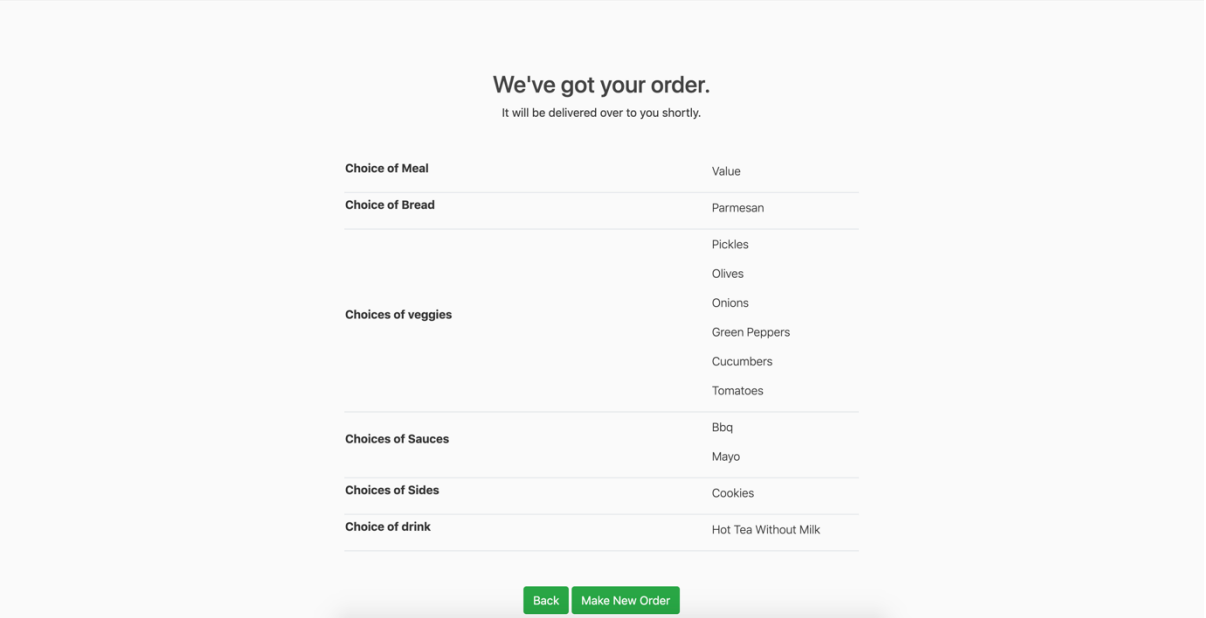

Figure 10: A summary of all your orders.

Lastly, when this page is loaded, a list of commands would be implemented into the prolog system to retrieve all the choices that the user made.
In this case, “show_meals(X)”, “show_bread(X), “show_veggies(X)”,”show_topup(X)”, “show_sauces(X)”, “show_sides(X)” and “show_drinks(X)” would be implemented and it would return the relevant information needed and shown on the page.
However, since no top-ups have been chosen because of value meal, show_topup(X) would return null and would not be reflected here.

Upon clicking “make new order” button, it will redirect the user to the index page which restarts the entire process.