Potato Pizza - Chatbot

Design a layout of program

create data as a dataframe / using while loop to check each input data that will be matched with dataset or not?

Create Data Variable

```
##Input data
# Pizza Type
TYPE <- c("Hawaiian", "Vegetarian", "Cheese", "Pepperoni", "H
PRICE <- c(6.99, 5.99, 6.99, 6.99, 7.99)
```

```
pizza_type <- data.frame(ID = 1:5, TYPE, PRICE)          #colu

# Pizza Size
SIZE <- c("small", "medium", "large", "extra large", "jumbo")
SLICE <- c(4, 6, 8, 10, 12)
CHARGE <- c(0, 2, 4, 6, 8)
pizza_charge <- data.frame(ID = 1:5, SIZE, SLICE, CHARGE)

# Pizza Topping
TOPPING <- c("mushroom", "pineapple", "pepperoni", "pizza sau
EXTRA <- c("1", "1", "1.5", "1.5", "2", "2", "3", "0")
pizza_topping <- data.frame(ID = 1:8, TOPPING, EXTRA)
```

Using a `data.frame` to create an ID for each data / show data in table form

ID	TYPE	PRICE
1	Hawaiian	6.99
2	Vegetarian	5.99
3	Cheese	6.99
4	Pepperoni	6.99
5	House Special	7.99

TYPE

ID	SIZE	SLICE	CHARGE
1	small	4	0
2	medium	6	2
3	large	8	4
4	extra large	10	6
5	jumbo	12	8

SIZE

ID	TOPPING	EXTRA
1	mushroom	1
2	pineapple	1
3	pepperoni	1.5
4	pizza sauce	1.5
5	cheese	2
6	bacon	2
7	seafood	3
8	no topping	0

TOPPING

Create Chatbot program

1. Welcome section

```
pizza_chatbot <- function() {

## Process
#Welcome
  cat("((_Welcome_)) > Type > Size > Topping > Extra Topping :
  cat("\n----------- Hello ! Welcome to Potato's Pizza House
```

```
  cat("What's your name ?  ")
  user_name <- readLines("stdin", n=1)
  cat("\n\t\t\t\t\tHello", toupper(user_name), ":) , Let's or
  cat("\n-------------------------------------------------
```

```
~/pizza$ Rscript Potato_Pizza.r
((_Welcome_)) > Type > Size > Topping > Extra Topping > Summarize

---------- Hello ! Welcome to Potato's Pizza House ! ------------
What's your name ?  m

                  Hello M :) , Let's order your pizza !
-----------------------------------------------------------------
```

2. Pizza Type Section

```
# Ordering type of pizza
while (TRUE) {
  cat("\nWelcome > ((_Type_)) > Size > Topping > Extra Toppin
  print(pizza_type, row.names = F)
    while (TRUE) {                      # use while to use th
      cat("\n\t\t\t\t\tChoose ID for select type of pizza :
      w <- readLines("stdin", n=1)                  # row sel
        if (w %in% pizza_type$ID) {                 # check i
          a <- pizza_type[w, 2]                     # w = id,
          b <- pizza_type[w, 3]                     # w = id,
          break                   # need to input correct ID
        } else {                                    # if ID i
          print("Incorrect pizza id!!")
        }                                           # return
    }
    cat("\t\t\t\t>>  You Choose: ", a, "//  $", b," <<")
    cat("\n-------------------------------------------------
```

```
Welcome > ((_Type_)) > Size > Topping > Extra Topping > Summarize

 ID          TYPE PRICE
  1      Hawaiian  6.99
  2    Vegetarian  5.99
  3       Cheese  6.99
  4     Pepperoni  6.99
  5 House Special  7.99

                  Choose ID for select type of pizza :  1
               >>  You Choose:  Hawaiian //  $ 6.99  <<
----------------------------------------------------------------------
```

if input ID (w) is on data

```
                    Choose ID for select type of pizza :  9
[1] "Incorrect pizza id!!"

                    Choose ID for select type of pizza :
```

if input ID (w) is not on data > return to while loop again

3. Pizza Size Section

```
# Ordering size of pizza
    cat("Welcome > Type > ((_Size_)) > Topping > Extra Toppin
    print(pizza_charge, row.names = F)
    while (TRUE) {
      cat("\n\t\t\t\t\tChoose ID for select size of pizza :
      x <- readLines("stdin", n=1)
        if (x %in% pizza_charge$ID) {
          c <- pizza_charge[x, 2]
          d <- pizza_charge[x, 3]
          e <- pizza_charge[x, 4]
          break
        } else {
          print("Incorrect size id!!")
        }
    }
```

```
    cat("\t\t\t\t>>  You Choose: ", c, "size // ", d, "slices
    cat("\n-------------------------------------------------
```

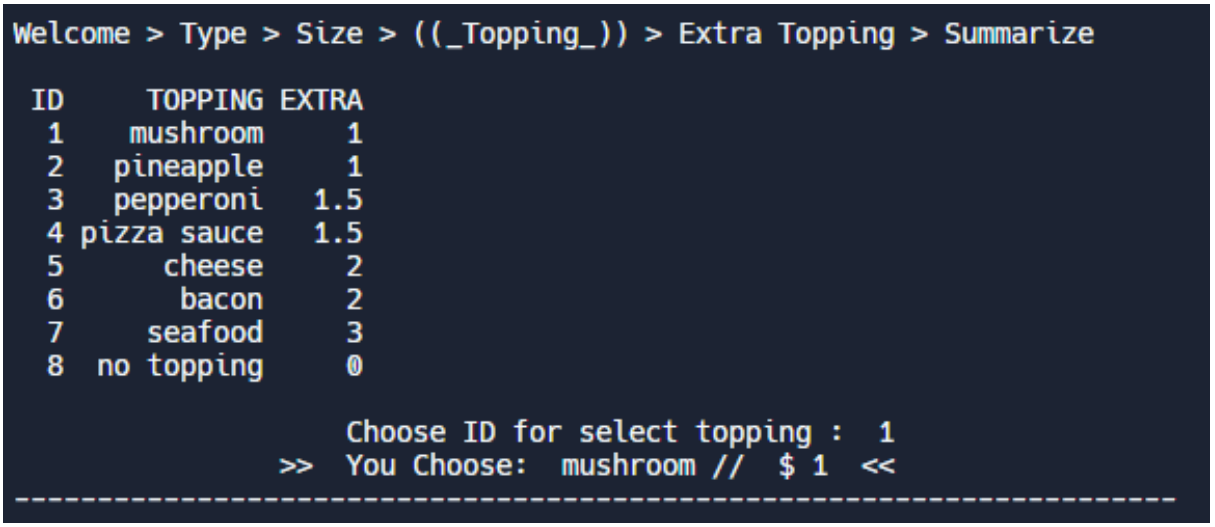
```
Welcome > Type > ((_Size_)) > Topping > Extra Topping > Summarize

 ID        SIZE SLICE CHARGE
  1       small     4      0
  2      medium     6      2
  3       large     8      4
  4 extra large    10      6
  5       jumbo    12      8

                  Choose ID for select size of pizza :  1
               >>  You Choose:  small size //  4 slices //  $ 0  <<
-----------------------------------------------------------------------
```

4. Pizza Topping Section

```
# Ordering topping of pizza
    cat("Welcome > Type > Size > ((_Topping_)) > Extra Toppin
    print(pizza_topping, row.names = F)
    while (TRUE) {
      cat("\n\t\t\t\t\tChoose ID for select topping :  ")
      y <- readLines("stdin", n=1)
        if (y %in% pizza_topping$ID) {
          f <- pizza_topping[y, 2]
          g <- as.numeric(pizza_topping[y, 3])
          break
        } else {
          print("Incorrect topping id!!")
        }
    }
    cat("\t\t\t\t>>  You Choose: ", f, "//  $", g, " <<")
    cat("\n-------------------------------------------------
```



5. Pizza Extra Topping Section

```
# Ordering extra topping of pizza
    cat("Welcome > Type > Size > Topping > ((_Extra Topping_)
    print(pizza_topping, row.names = F)
    while (TRUE) {
      cat("\n\t\t\t\t\tChoose ID for select topping :  ")
      z <- readLines("stdin", n=1)
        if (z %in% pizza_topping$ID) {
          h <- pizza_topping[z, 2]
          i <- as.numeric(pizza_topping[z, 3])
          break
        } else {
          print("Incorrect topping id!!")
        }
    }
    cat("\t\t\t\t>> You Choose: ", h, "//  $", i, "<<")
    cat("\n-------------------------------------------------
```

```
Welcome > Type > Size > Topping > ((_Extra Topping_)) > Summarize

 ID     TOPPING EXTRA
  1    mushroom     1
  2   pineapple     1
  3   pepperoni   1.5
  4 pizza sauce   1.5
  5      cheese     2
  6       bacon     2
  7     seafood     3
  8  no topping     0

                  Choose ID for select topping :  1
               >> You Choose:  mushroom //  $ 1 <<
-------------------------------------------------------------------
```

6. Summarize Section

```
# Summarize pizza order
count <- 0                # set count = 0 for counting an ord
sum <- 0                  # set sum = 0 for calculating a tot

    cat("Welcome > Type > Size > Topping > Extra Topping > ((
    pizza_list <- list(Type = pizza_type[w, c(2,3)],
                       Size = pizza_charge[x, c(2,3,4) ],
                       Topping =  pizza_topping[y, c(2,3)],
                       Extra_Topping = pizza_topping[z, c(2,3)
    print(pizza_list)
    price <- b + e + g + i
    cat(">>> Price : $", price, " <<<\n")
    count <- count + 1
    sum <- sum + price
```

```
Welcome > Type > Size > Topping > Extra Topping > ((_Summarize_))

$Type
      TYPE PRICE
1 Hawaiian  6.99

$Size
   SIZE SLICE CHARGE
1 small     4      0

$Topping
   TOPPING EXTRA
1 mushroom     1

$Extra_Topping
   TOPPING EXTRA
1 mushroom     1

>>> Price : $ 8.99  <<<
```

Reorder Section

▶️ Checking an User's answer >> Incorrect answer

```
while (TRUE) {
        cat("Would you like to order more pizza? (y/n) : ")
        reorder <- tolower(readLines("stdin", n=1))
        if (reorder != "y" & reorder != "n" ) {
          print(paste("Please select y or n"))
        } else {
          break
        }
      }
```

```
Would you like to order more pizza? (y/n) : o
[1] "Please select y or n"
Would you like to order more pizza? (y/n) : 2
[1] "Please select y or n"
Would you like to order more pizza? (y/n) : No
[1] "Please select y or n"
```

cat("Would you like to order more pizza? (y/n) : ")
reorder <- tolower(readLines("stdin", n=1))
user can answer in y, n, Y, N
if (reorder != "y" & reorder != "n") {
check an answer if it not y, n, Y, N >> return to a while loop

▶️ Checking an User's answer >> "No" answer >> Summarizing transaction

```
    if (reorder == "n") {
      cat("\n-------------------------------------------------
      cat("\nTotal of pizza : ", count)
      cat("\nTotal Price : $", sum, "\n")
      cat("\n\t\t\tThank you", toupper(user_name), "for order
      cat("\n\t\t\t\t\t\tHave a nice day :))\n\n")
      break
    } else {
      cat("\n-------------------------------------------------
    }
  }

}
```

```
Would you like to order more pizza? (y/n) : N

------------------------------------------------------------------------
Total of pizza :  1
Total Price : $ 8.99

            Thank you M for ordering pizza with Potato!
                        Have a nice day :))
```

if User's answer is n, N >> Summarize an order >> end program

▶️ Checking an User's answer >> "Yes" answer >> Return to order again

```
>>> Price : $ 8.99  <<<
Would you like to order more pizza? (y/n) : y

------------------------------------------------------------------------
Welcome > ((_Type_)) > Size > Topping > Extra Topping > Summarize

 ID          TYPE PRICE
  1       Hawaiian  6.99
  2    Vegetarian  5.99
  3        Cheese  6.99
  4     Pepperoni  6.99
  5 House Special  7.99

                   Choose ID for select type of pizza :
```

if User's answer is y, Y >> Return to while loop of pizza again >> Starting a program from Pizza Type

```
Welcome > Type > Size > Topping > Extra Topping > ((_Summarize_))

$Type
        TYPE PRICE
2 Vegetarian  5.99

$Size
    SIZE SLICE CHARGE
2 medium     6      2

$Topping
    TOPPING EXTRA
2 pineapple     1

$Extra_Topping
    TOPPING EXTRA
2 pineapple     1

>>> Price : $ 9.99  <<<
Would you like to order more pizza? (y/n) : n

------------------------------------------------------------------------
Total of pizza :  2
Total Price : $ 18.98

           Thank you M for ordering pizza with Potato!
                      Have a nice day :))
```

summarizing an transaction that include each orders together