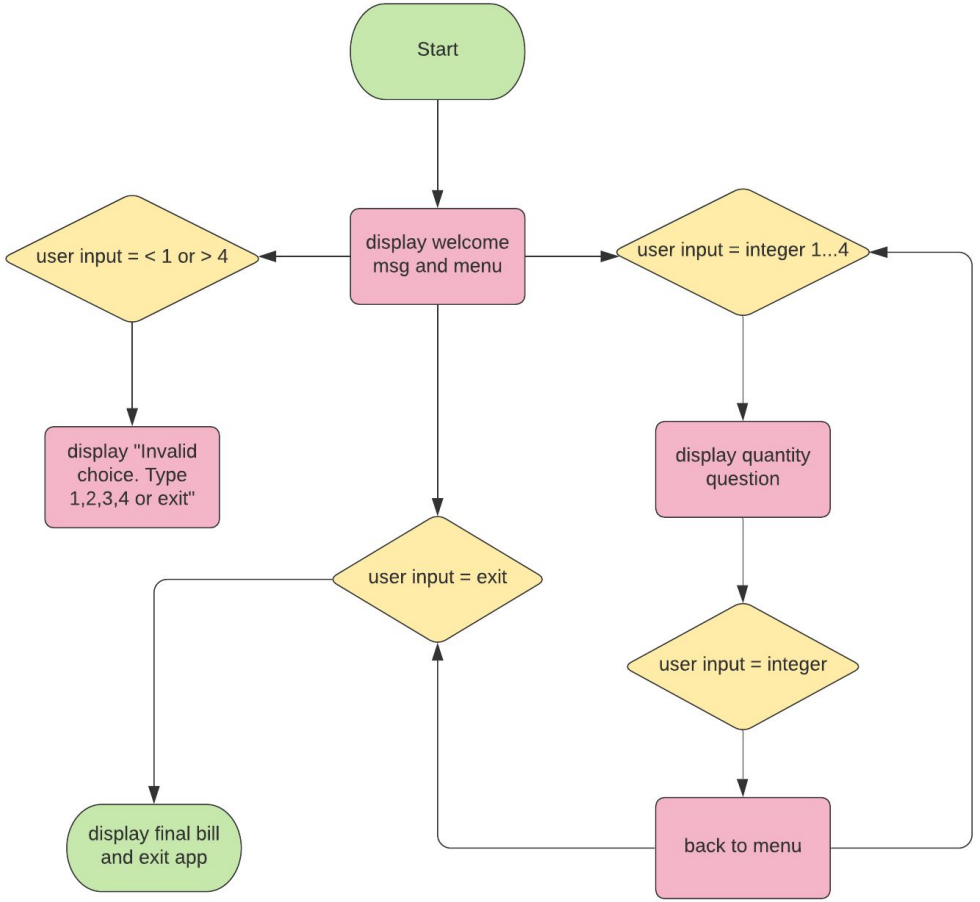
Alaw's Gin

Menu Application

Layout of Application

Purpose?

- Contactless waitressing
- Keep hospitality running
- Ease the work for staff
- Constant access to the menu



Overall Structure

- Application divided into two files, an index file and a menu file
- The index file contains the loop for the application
- Menu file contains the class GuestOrder where all the methods are grouped together.
- Menu file also holds the data for the menu in arrays and hashes

Main Features

- Menu Displayed (terminal table coming)
- Each ingredient displayed
- Automatically back to menu after choosing quantity (loop)
- Easy access to instructions throughout the whole experience (get_input)
- Receiving correctly calculated final bill when entering exit

Code Overview

While loop

- The while loop is checking for an input and only executes a set of instructions on a certain condition
- I used a while loop for picking up the user input and executing the next action based on the input

```
# create a flag for the loop to tell the program what to do when flag is true or false
ordering = true

# create a loop for ordering
while ordering
    guest.menu
# get user input, make it an integer
    guest_input = gets.chomp.to_i
    case guest_input
    when 1,2,3,4
        guest.place_order(guest_input)
    when 0
        ordering = false
        puts Rainbow("Thank you for visiting Alaw's Gin #{guest.name}, your bill is $#{guest.bill}. See ya next time!").indianred
    else
        puts Rainbow("Invalid choice. Type 1, 2, 3, 4 or exit").orange
    end
end
```

Calculated Final Bill

- Data in hashes
- @drink_list to make it easier to grab the price when user input choice

```
        @drink_menu = {"1. Gin Lane Martini - London Dry Gin, Lavender Bitters, Grapefruit Bitters & Tonic" => 19, "2. Italian Flip - Gin,
#update the menu list (containing item number and price)
        @drink_list = {1 => 19, 2 => 17, 3 => 21, 4 => 19}

        @drink_menu.each do |drink, price|
            puts "#{drink} $#{price}"
        end
    end

#show drink price based on the user choice
    def get_drink_price(user_input)
        @drink_price = @drink_list[user_input]
    end
```

- After getting quantity =>

```
#calculate the total bill when user enters exit
    def calculate_final_bill
        @bill += @drink_price * @quantity
    end
```

Terminal Table ??

- Trying to add data to the table
- Separator and titles coming

```ruby
    def menu
#terminal table for menu
    #    rows = [[1, "Gin Lane Martini - London Dry Gin, Lavender Bitters, Grapefruit Bitters & Tonic"], [2, "Italian Flip - G
    #    table = Terminal::Table.new :rows => rows
    #    puts table
```

```
alawmoradi@Alaws-MacBook-Air T1A3_Terminal_App % ruby index.rb
Hey there! Welcome to Alaw's Gin! Find our menu below and choose one of our sensational drinks by entering the drink number. When you've finished, just type exit to receive your final bill
+---+-------------------------------------------------------------------------------------------------------------------+
| 1 | Gin Lane Martini - London Dry Gin, Lavender Bitters, Grapefruit Bitters & Tonic                                    |
| 2 | Italian Flip - Gin, Campari, Antica Formula, Shaken Well and Ice Block                                            |
| 3 | Lightbulb Moment - Tanqueray London Dry Gin, Fresh Mint, Lemon and Peach, Blue Curacao, Rose Essence, Charged with Procecco |
| 4 | South Side - Gin, Fresh Pressed Lemon Juice, Fresh Pressed Mint                                                    |
+---+-------------------------------------------------------------------------------------------------------------------+
1. Gin Lane Martini - London Dry Gin, Lavender Bitters, Grapefruit Bitters & Tonic $19
2. Italian Flip - Gin, Campari, Antica Formula, Shaken Well and Ice Block $17
3. Lightbulb Moment - Tanqueray London Dry Gin, Fresh Mint, Lemon and Peach, Blue Curacao, Rose Essence, Charged with Procecco $21
4. South Side - Gin, Fresh Pressed Lemon Juice, Fresh Pressed Mint $19
```

Development Review

- Started off with all documentation

- Planned to use JSON, realised I took too long documenting and did not have time to learn JSON so kept the code as simple as possible

- Adding details regarding instructions etc on the way when trying out the app. Planning ahead and planning as I go, good combo?

- Be better with time management

Questions?