AJ's RECORDS

T1A3 Terminal App

WHAT IS IT?

AJ Records is a simple CLI app that imitates a real indie/pop culture store. It allows a user to enter the store, browse vinyl in stock, filtering their search by type (genre, artist, etc), add items to cart, and finalise a purchase with girl at the counter.

MAIN FEATURES

01

STORE NAVIGATION

Menu displaying options for user to explore (Vinyl, Cart, Counter, Exit)

02

FILTER PRODUCTS

Users can search by keywords to filter records in stock by Genre/Title/Album/Artist/Price

03

CART

Visitor can remove/add item to cart based on cat no.

04

CHECKOUT

Visitor can finalise their order at the Counter and have various interactions with the girl there.

STORE NAVIGATION

Feature 1

PROCESS/CODE

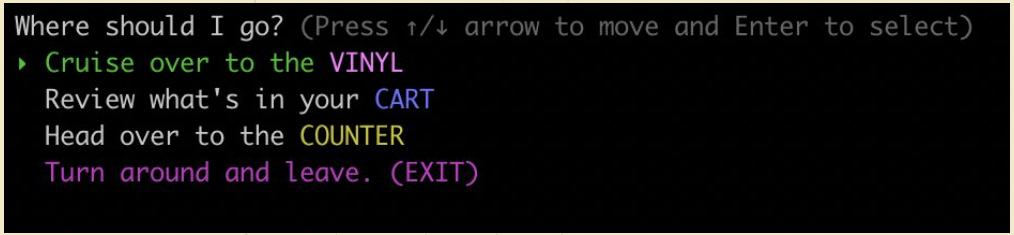
- Introduces the visitor to the store, displays a banner and asks if they want to enter
- Displays a nav menu in which visitor can decide where to go in store.
    - Their decision (input) is parsed from look_around into choose_store_section(choice)
- Serves as the main convergence point for the entire program, where users can come back at any time when a decision/interaction leads to look_around being called

```ruby
# welcome the visitor to the store, ask if they want to come in
puts @@store_name.yellow
puts "You arrive at a swanky looking retro record store.\n".yellow +
     "The aisles are lined with hundreds of records, CDs, games, comic books....\n".white +
     "The walls are covered in posters and memorabilia.\n".white +
     "There are literally 2 MILLION pop vinyl figurines staring directly at you.\n".white +
     "Oh my god is that Baby Yoda?".italic.green + " You're so overstimulated your eyeballs almost explode.\n".white

answer = PROMPT.yes?("The pop vinyl are super creepy. But I do need more records.. should I go in?".magenta)
if !answer
  system "clear"
  puts "You realise you have absolutely no understanding of how to talk to humans anymore since COVID. You turn around and leave.".red
  exit
else
  system "clear"
  look_around
end
end
```

```ruby
require_all "./lib"

FILE = File.read("./data/vinyl.json") # the store's stock on hand (498 records)
STOCK = JSON.parse(FILE)
PROMPT = TTY::Prompt.new

module StoreNavigation
  @@store_name = Artii::Base.new
  @@store_name = TTY::Box.frame @@store_name.asciify("AJ RECORDS *")

  # enact consequences of the visitors selection in look_around
  def self.choose_store_section(choice)
    system "clear"
    puts "\n· · ·~~~~~ᕕ( ᐛ )ᕗ" # imitate the visitor walking
    progressbar = ProgressBar.create
    10.times { progressbar.increment; sleep 0.01 }
    case choice
    when 1
      display_menu #vinyl section
    when 2
      display_cart #cart
    when 3
      go_to_counter #counter/checkout
    when 4
      exit
    when 5
      system "clear"
      puts "For me? Yay. So wonderful. So generous of you. I love my job.\nI love sorting. " +
      "What a gift you have given me.\nEVERYTHING. IS. GREAT.\n (╯°□°) ╯ ┻━┻\n".yellow
      puts "[...You back away slowly and exit the store]".italic
      exit
    end
  end

  # visitor decides where to explore in the store
  def self.look_around
    input = PROMPT.select("Where should I go?") do |menu|
      menu.choice "Cruise over to the " + "VINYL".light_magenta, 1
      menu.choice "Review what's in your " + "CART".light_blue, 2
      menu.choice "Head over to the " + "COUNTER".yellow, 3
      # visitor can't just run out with vinyl in their cart, they have to return it first
      if total_cost_of_cart == 0
        menu.choice "Turn around and leave. (EXIT)".magenta, 4
      else
        menu.choice "Drop your records to the girl at the counter and leave (EXIT)".magenta, 5
      end
    end
    choose_store_section(input)
  end
```

IN APP


You arrive at a swanky looking retro record store.
The aisles are lined with hundreds of records, CDs, games, comic books....
The walls are covered in posters and memorabilia.
There are literally 2 MILLION pop vinyl figurines staring directly at you.
Oh my god is that Baby Yoda? You're so overstimulated your eyeballs almost explode.
The pop vinyl are super creepy. But I do need more records.. should I go in? (Y/n)


Where should I go? (Press ↑/↓ arrow to move and Enter to select)
› Cruise over to the VINYL
Review what's in your CART
Head over to the COUNTER
Turn around and leave. (EXIT)

FILTER PRODUCTS

Feature 2

PROCESS/CODE

- Module VinylSection enacted when users select option 1 from store navigation (display_menu is called)
- Users can choose to search STOCK (a constant containing a json file of data) by genre, artist, album and price
- Artist/album/genre searches are defined in search_by_keyword(tag, tag_2 = nil)
- Price search is defined in search_by_price
- Both functions use while and for loops to iterate through the STOCK and see if the users input matches
- In the case of errors in the user's input (invalid, no results, empty string), relevant error & hint messages are displayed to the user to help them achieve valid input on the next iteration
- Prints results aesthetically according to display_filtered_records(record)

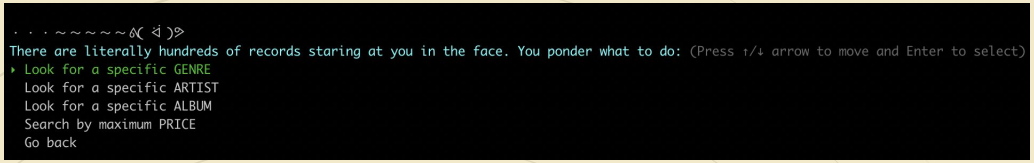
```ruby
require_all "./lib"

FILE = File.read("./data/vinyl.json") # the store's stock on hand (498 records)
STOCK = JSON.parse(FILE)
PROMPT = TTY::Prompt.new
```

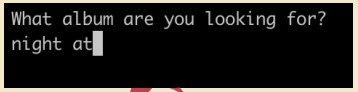
```ruby
module VinylSection
  include Cart

  def decide_whether_to_add_to_cart
    if PROMPT.yes?("Should I add anything?".green)
      add_items_to_cart
    else
      system "clear"
      display_menu
    end
  end

  def display_filtered_records(record)
    puts "CATALOGUE #: #{record["Catno"]}\n" +
         "ALBUM: #{record["Album"]}\n" +
         "ARTIST: #{record["Artist"]}\n" +
         "YEAR: #{record["Year"]}\n" +
         "GENRE: #{record["Genre"]}\n" +
         "SUBGENRE: #{record["Subgenre"]}\n" +
         "PRICE $#{record["Price"]}\n" +
         "======================="
  end

  def search_by_keyword(tag, tag_2 = nil) # Set tag_2 to nil for Artist and Album searches.
    found = false
    while !found # Loop the sequence until a record is found matching the user's criteria
      puts "What #{tag.downcase} are you looking for?"
      # Match the input to the same format of the json so it can be compared
      keyword = gets.split.map(&:capitalize).join(" ").chomp
      system "clear"
      # If user hits enter, display an error that tells them what is wrong
      # The program will skip to next iteration so they can try again.
      if keyword == ""
        puts "[No results. Woops, guess I should actually search for something..]".italic.red
        next
      end

      # Loop through each record (hash) in STOCK (array)
      for record in STOCK
        # When searching by Genre, BOTH Genre (tag) and Subgenre (tag_2) keys in the array will be searched for matches.
        # When searching by Artist or Album, tag_2 = nil and only their respective keys will be searched for matches.
        if (tag_2 == nil && (record[tag] == keyword || record[tag].include?(keyword))) || (tag_2 != nil && record[tag_2].include?(keyword))
          # Print each matching record out on screen according to the format of display_filtered_records(record)
          display_filtered_records(record)
          found = true  # Change found to true, so the loop ends.
        end
      end
      # If nothing has been found, the program helps the user by presenting an error and some hints about what to search.
      if !found
        puts "[No results found... maybe I should try and simplify my search?)]\n".italic.magenta + "HINT: 'beach' " +
        "will return The Beach Boys\n'rock' will return Psychedelic Rock, Hard Rock, Rockabilly etc,\n'bob' will return Bob Dylan, Bob Marley etc."
        display_menu # Allow them to search by something else.
      else
        decide_whether_to_add_to_cart
      end
    end
```

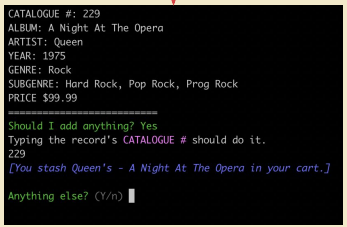
```ruby
def search_by_price
  found = false
  while !found # Loop the sequence until a record is found matching the user's criteria
    puts "Whats the max you're willing to pay for a record?"
    price = gets.to_i #
    system "clear"
    # There are no records < $40 in stock. So, if user input a value < 40, the program will
    # print an error & helper message to inform them of how to rectify it on the next loop iteration.
    if (..40).include?(price)
      puts "[What was I thinking, I haven't seen a single record here for less than ".italic + "40 bucks.\n".magenta.italic +
      "Guess I've gotta stretch the purse strings... :'( ]".italic
      # Go to next iteration, try input again.
    end
    # Loop through each record (hash) in STOCK (array)
    for record in STOCK
      if record["Price"] <= price
        # Print each record to the screen in the format of display_filtered_records(record)
        display_filtered_records(record)
        # Change to found so loop ends once all matching records have been printed.
        found = true
      end
    end
  end
  decide_whether_to_add_to_cart
end

# enact consequences of users choice in display_menu
def filter_records(filter_choice)
  system("clear")
  case filter_choice
  when 1
    search_by_keyword("Genre", "Subgenre")
  when 2
    search_by_keyword("Artist")
  when 3
    search_by_keyword("Album")
  when 4
    search_by_price
  when 5
    #if user changes their mind, go back to the main store navigation menu
    look_around
  end
end

# display a menu of filter options to the user
def display_menu
  input = PROMPT.select("There are literally hundreds of records staring at you in the face. You ponder what to do:".light_cyan) do |menu|
    menu.choice "Look for a specific GENRE", 1
    menu.choice "Look for a specific ARTIST", 2
    menu.choice "Look for a specific ALBUM", 3
    menu.choice "Search by maximum PRICE", 4
    menu.choice "Go back", 5
  end
  filter_records(input)
end
```

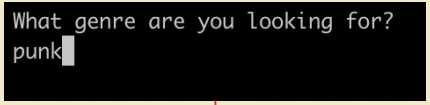
	GENRE	ARTIST	ALBUM	PRICE
USER INPUT?	“blues”	“beatles”	“night at”	40
DISPLAYS RECORDS W/:	“blues” in genre or subgenre keys. (e.g. Blues Rock, Rhythm & Blues)	“beatles” in the artist key. (e.g. The Beatles)	“night” in album key. (e.g. A Night at the Opera)	price key <= $40.00

IN APP

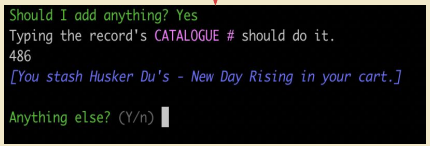

. . . ～～～～～◊(‹)>
There are literally hundreds of records staring at you in the face. You ponder what to do: (Press ↑/↓ arrow to move and Enter to select)
› Look for a specific GENRE
Look for a specific ARTIST
Look for a specific ALBUM
Search by maximum PRICE
Go back

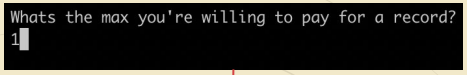

What album are you looking for?
night at

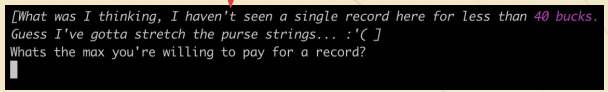

CATALOGUE #: 229
ALBUM: A Night At The Opera
ARTIST: Queen
YEAR: 1975
GENRE: Rock
SUBGENRE: Hard Rock, Pop Rock, Prog Rock
PRICE $99.99
==========
Should I add anything? Yes
Typing the record's CATALOGUE # should do it.
229
[You stash Queen's - A Night At The Opera in your cart.]
Anything else? (Y/n)

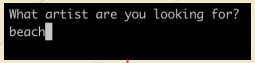

What genre are you looking for?
punk


CATALOGUE #: 459
ALBUM: Metal Box
ARTIST: Public Image Ltd.
YEAR: 1979
GENRE: Electronic, Rock
SUBGENRE: Post-Punk, Dub, Avantgarde, Experimental
PRICE $89.99
==========
CATALOGUE #: 481
ALBUM: Entertainment!
ARTIST: Gang Of Four
YEAR: 1979
GENRE: Rock
SUBGENRE: Post-Punk, New Wave
PRICE $89.99
==========
CATALOGUE #: 486
ALBUM: New Day Rising
ARTIST: Husker Du
YEAR: 1985
GENRE: Rock
SUBGENRE: Alternative Rock, Hardcore, Punk
PRICE $89.99
==========
Should I add anything? (Y/n)

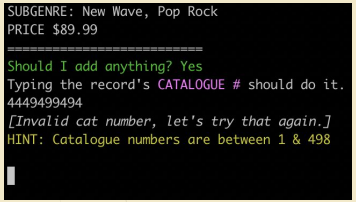

Should I add anything? Yes
Typing the record's CATALOGUE # should do it.
486
[You stash Husker Du's - New Day Rising in your cart.]
Anything else? (Y/n)

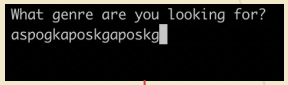

Whats the max you're willing to pay for a record?
1

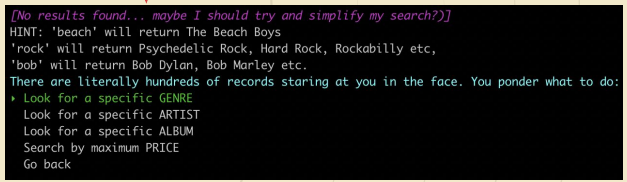

[What was I thinking, I haven't seen a single record here for less than 40 bucks.
Guess I've gotta stretch the purse strings... :'( ]
Whats the max you're willing to pay for a record?


What artist are you looking for?
beach


CATALOGUE #: 2
ALBUM: Pet Sounds
ARTIST: The Beach Boys
YEAR: 1966
GENRE: Rock
SUBGENRE: Pop Rock, Psychedelic Rock
PRICE $89.99
==========
CATALOGUE #: 269
ALBUM: The Beach Boys Today!
ARTIST: The Beach Boys
YEAR: 1965
GENRE: Rock
SUBGENRE: Pop Rock, Psychedelic Rock
PRICE $65.95
==========
CATALOGUE #: 379
ALBUM: The Smile Sessions
ARTIST: The Beach Boys
YEAR: 2011
GENRE: Rock
SUBGENRE: Pop Rock, Psychedelic Rock
PRICE $39.95
==========
Should I add anything? (Y/n)


SUBGENRE: New Wave, Pop Rock
PRICE $89.99
==========
Should I add anything? Yes
Typing the record's CATALOGUE # should do it.
4449499494
[Invalid cat number, let's try that again.]
HINT: Catalogue numbers are between 1 & 498


What genre are you looking for?
aspogkaposkgaposkg


[No results found... maybe I should try and simplify my search?)]
HINT: 'beach' will return The Beach Boys
'rock' will return Psychedelic Rock, Hard Rock, Rockabilly etc,
'bob' will return Bob Dylan, Bob Marley etc.
There are literally hundreds of records staring at you in the face. You ponder what to do:
› Look for a specific GENRE
Look for a specific ARTIST
Look for a specific ALBUM
Search by maximum PRICE
Go back

CART

Feature 3

PROCESS/CODE

- If user selects option 2 in look_around (store navigation menu), display_cart will run in module Cart
- If @@cart has anything in it, it will print each record to the screen surrounded by asterisks
- Users are prompted to choose whether to remove anything (and if they want to, remove_items_from_cart runs)
- When searching for products in the VinylSection, if results are returned they can choose to add_items_to_cart
- Both adding/removing items involves typing the respective catalogue number
- Total_cost_of_cart calculates the total amount owing and Counter uses this figure in order_summary

```ruby
module Cart
  @@cart = []
  @@cart_total = 0

  def display_cart
    # If the cart is empty, display a message and return the user to the menu
    if @@cart.empty?
      puts "What a sad little empty cart. :(".light_red
      look_around
    else # Display all items in cart according to this aesthetic
      puts "****************************************\n"
      for vinyl in @@cart
        puts "#{vinyl["Catno"]}: ".magenta + "#{vinyl["Artist"]} - #{vinyl["Album"]} (#{vinyl["Year"]}) " + "$#{vinyl["Price"]}".yellow
      end
      puts "****************************************\n"
    end
    remove = PROMPT.yes?("Should I put anything back?".green)
    if remove
      remove_items_from_cart
    else
      system "clear"
      look_around
    end
  end

  def add_items_to_cart
    puts "Typing the record's " + "CATALOGUE # ".light_magenta + "should do it."
    done = false
    while !done # Loop until user does not want to add any more items to cart
      product_selection = gets.chomp
      # Catalogue numbers are between 1 and 498. If the user inputs something outside of that range,
      # Display an error and a hint message so they can get the input correct on the next iteration.
      unless (1..498).include?(product_selection.to_i)
        puts "[Invalid cat number, let's try that again.]".italic
        puts "HINT: Catalogue numbers are between 1 & 498\n".yellow
        next # Skip to next iteration
      end
      # Search through all records in STOCK to get the details of the record with the catalogue number typed by the user.
      for item in STOCK
        if item["Catno"] == product_selection
          @@cart << item # Add it to the user's cart & let them know what they added.
          puts "[You stash #{item["Artist"]}'s - #{item["Album"]} in your cart.]\n".light_blue.italic
        end
      end
      if PROMPT.yes?("Anything else?".green) == false # Continue the loop.
        done = true # Loop will end.
        system "clear"
      end
    end
    display_menu
  end
```

```ruby
def display_cart
  # If the cart is empty, display a message and return the user to the menu
  if @@cart.empty?
    puts "What a sad little empty cart. :(".light_red
    look_around
  else # Display all items in cart according to this aesthetic
    puts "****************************************\n"
    for vinyl in @@cart
      puts "#{vinyl["Catno"]}: ".magenta + "#{vinyl["Artist"]} - #{vinyl["Album"]} (#{vinyl["Year"]}) " + "$#{vinyl["Price"]}".yellow
    end
    puts "****************************************\n"
  end
  remove = PROMPT.yes?("Should I put anything back?".green)
  if remove
    remove_items_from_cart
  else
    system "clear"
    look_around
  end
end
```

```ruby
def remove_items_from_cart
  finished = false
  puts "[Type the ".italic + "catalogue number ".magenta.italic + "to remove the item from your cart.]".italic
  while !finished # Loop until a valid input matching a catalogue number in cart has been entered.
    product_selection = gets.chomp
    # If the input is equal to a catalogue number that is in the cart, delete it.
    if item = @@cart.detect { |item| item["Catno"] == product_selection }
      @@cart.delete(item)
      system "clear"
      puts "[You put #{item["Artist"]}'s - #{item["Album"]} back on the shelf.]".light_blue.italic
    else
      # Display an error to the user, letting them know their input was invalid.
      # Prints a hint about where to find catalogue numbers by referring to their colour on screen.
      # The hint is also highlighted in the same colour.
      system "clear"
      puts "INVALID CAT NO\n".red + "Hmm... my eyeballs must be playing up. " +
      "Let me peak at those catalogue numbers again. They're probably the ones written in " + "magenta..".magenta
      puts
    end
    display_cart
  end
  system "clear"
  display_menu
end

def total_cost_of_cart
  @@cart_total = @@cart.sum { |item| item["Price"] }.round(2) # Round cart total to 2 decimal places
end
end
```

IN APP

```
. . . ~ ~ ~ ~ ~ ~ ᕕ( ᐛ )ᕗ
****************************************
3: The Beatles - Revolver (1966) $89.99
488: ZZ Top - Tres hombres (1973) $89.99
199: Nine Inch Nails - The Downward Spiral (1994) $99.99
78: Led Zeppelin - Led Zeppelin II (1969) $99.99
45: The Band - The Band ("The Brown Album") (1969) $49.99
396: ZZ Top - Eliminator (1983) $39.95
462: Def Leppard - Hysteria (1987) $89.99
****************************************
Should I put anything back? (Y/n)
```

```
Should I put anything back? Yes
[Type the catalogue number to remove the item from your cart.]
488
```

```
[You put ZZ Top's - Tres hombres back on the shelf.]
****************************************
3: The Beatles - Revolver (1966) $89.99
199: Nine Inch Nails - The Downward Spiral (1994) $99.99
78: Led Zeppelin - Led Zeppelin II (1969) $99.99
45: The Band - The Band ("The Brown Album") (1969) $49.99
396: ZZ Top - Eliminator (1983) $39.95
462: Def Leppard - Hysteria (1987) $89.99
****************************************
Should I put anything back? (Y/n)
```

```
****************************************
Should I put anything back? (Y/n) fdgsdg
```

```
Should I put anything back? (Y/n)
>> Invalid input.
```

```
Should I put anything back? Yes
[Type the catalogue number to remove the item from your cart.]
aoisjf
```

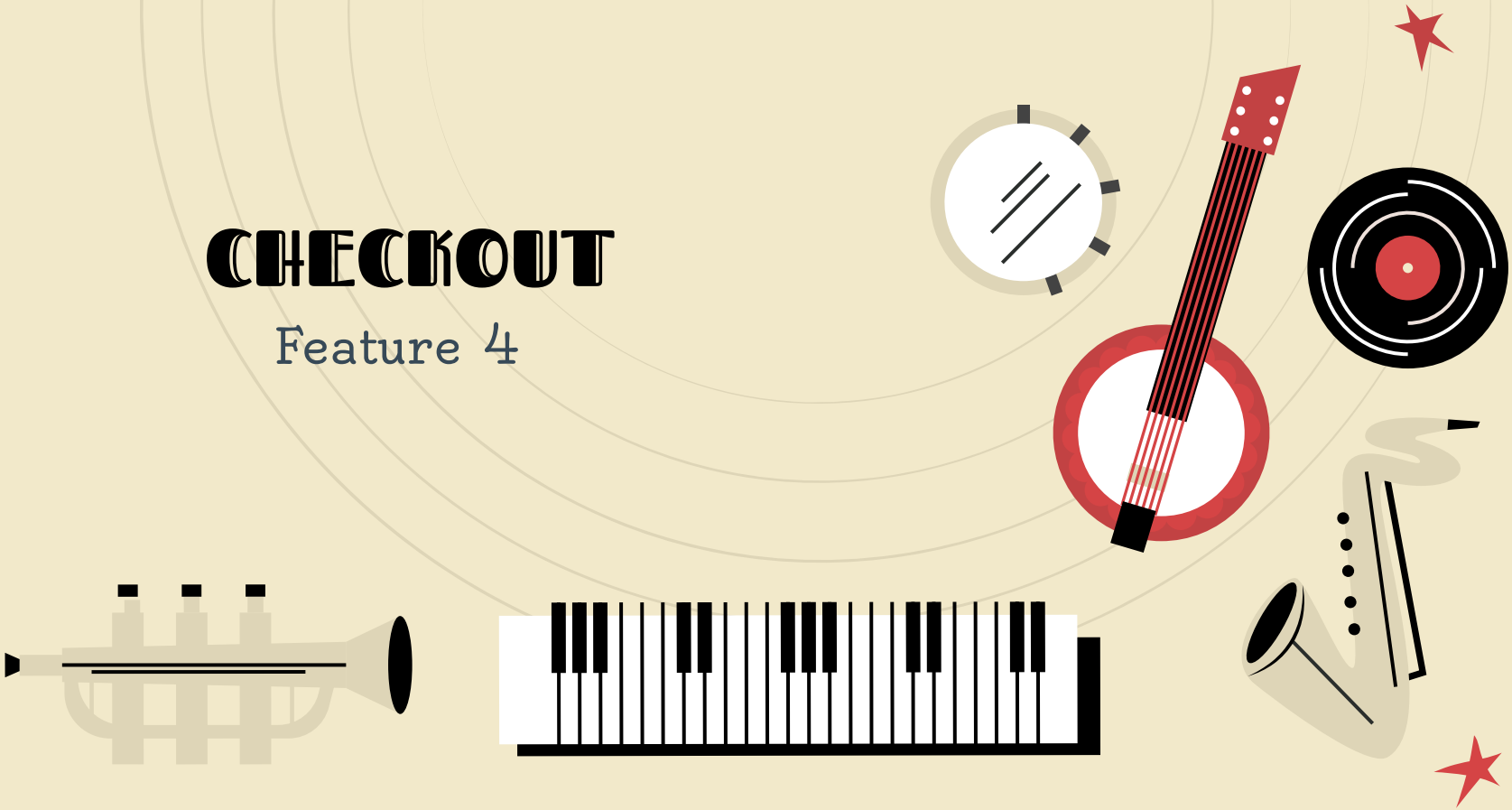
```
INVALID CAT NO
Hmm... my eyeballs must be playing up. Let me peak at those catalogue numbers again. They're probably the ones written in magenta..

****************************************
3: The Beatles - Revolver (1966) $89.99
488: ZZ Top - Tres hombres (1973) $89.99
199: Nine Inch Nails - The Downward Spiral (1994) $99.99
78: Led Zeppelin - Led Zeppelin II (1969) $99.99
45: The Band - The Band ("The Brown Album") (1969) $49.99
396: ZZ Top - Eliminator (1983) $39.95
462: Def Leppard - Hysteria (1987) $89.99
****************************************
Should I put anything back? (Y/n)
```

```
3: The Beatles - Revolver (1966) $89.99
****************************************
Should I put anything back? Yes
[Type the catalogue number to remove the item from your cart.]
3
```

```
[You put The Beatles's - Revolver back on the shelf.]
What a sad little empty cart. :(
Where should I go? (Press ↑/↓ arrow to move and Enter to select)
› Cruise over to the VINYL
  Review what's in your CART
  Head over to the COUNTER
  Turn around and leave. (EXIT)
```

CHECKOUT

Feature 4



PROCESS/CODE

- If user selects option 3 in look_around (store navigation menu), go_to_counter will run in module Counter
- Users can choose to pay, change mind and look around again, leave the store, or hit on the girl at the counter
- Choosing to pay will display a terminal-table of all items in the users cart
- Various difference interactions happen based on certain decisions/methods, e.g.
    - Using the .sample method the user can get_rejected by the girl 4 different ways
    - go_to_counter also uses .sample to randomise an introduction each time the user goes to the counter
    - If your cart is not empty and you tell her you want to leave, she will throw a table and get cranky at having to put all your records away

```ruby
require "json"
require "tty-prompt"

module Counter
  include Cart

  def get_rejected
    excuses = ["Oh, sorry. My sister's friend's fish died, and yes, it was tragic.", "That sounds really fun! " +
      "But sorry, I'm going to be busy not doing that.", "Oh what a shame, I actually have plans to teach my ferret to yodel. " +
      "Some other time maybe (not)?", "The voices in my head are telling me to say no. Sorry!"]
    puts puts "d_d #{excuses.sample}".light_cyan # Randomly print one of these strings in the array.
    counter_interaction
  end

  def order_summary(total)
    total_cost_of_cart
    headings = ["Catno", "Album", "Artist", "Year", "Price"]
    rows = []
    # Iterate through the hashes (i.e. records) in the user's cart, and add them to the rows for the Tax Invoice.
    @@cart.each do |hash|
      rows << ["#{hash["Catno"]}", "#{hash["Album"]}", "#{hash["Artist"]}", "#{hash["Year"]}", "$#{hash["Price"]}"]
    end
    # Create Tax Invoice with terminal-table so it displays in a readible/realstic way to the user.
    tax_invoice = Terminal::Table.new :title => "AJ's Records: Purchase Summary", :headings => headings, :rows => rows, :style => { :all_separators => true }
    puts "#{tax_invoice} \n"
    pay = PROMPT.yes?("Your total comes to $#{@@cart_total}, please! Did you want to go ahead and pay?".yellow)
    if !pay # Allow them a chance to back out
      system "clear"
      puts "No worries, come back once you're ready! (Please don't leave the store without paying...)".yellow
      look_around
    else
      system "clear"
    end
  end
```

```ruby
def counter_interaction
  counter_decision = PROMPT.select("How can I help?\n".yellow) do |menu|
    menu.choice "I'd like to pay for these", 1
    menu.choice "Errr actually, on second thought. I might have a second look around.", 2
    if @@cart.empty? == true
      menu.choice "I changed my mind and I'm just gonna head out.", 3
    else
      menu.choice "I changed my mind, and I'm going to head out. Can I just leave all these records with you?", 4
    end
    menu.choice "Actually... I wanted to see if you were busy later....want to grab a drink?", 5
  end
  system "clear"

  case counter_decision
  when 1
    if @@cart.empty?
      puts "(ง'̀-'́)ง [The girl gives you a very weird stare..]\n".italic
      puts "Ummm..you realise don't...have...anything yet?? Go check out our vinyl! \n" +
      "Just come back when you're done. " + "I'd give you a hand but I'm not a very good employee. Soz!\n".yellow
      look_around
    else
      puts "Awesome! I'll just tally that up for you...".yellow
      order_summary(@@cart_total) # Create a Tax Invoice table and calculate cart total
      puts "[You grimace at the total, and hand over your well-worn credit card]\n".italic.light_blue
      progressbar = ProgressBar.create
      10.times { progressbar.increment; sleep 0.1 }
      puts "\nd(^o^)b,,♬ Thanks for shopping at AJ's Records! Come back anytime.".yellow
      exit
    end
  when 2
    system "clear"
    puts "d(^o^)b,,♬\n"
    puts "No worries! Come over here when you're ready.".yellow
    look_around
  when 3
    system "clear"
    puts "d(^o^)b,,♬\n"
    puts "All right, see ya!".yellow
    exit
  when 4
    puts "Of coooooourse. Nothing I would enjoy more. So wonderful. I love my job. I love sorting. " +
    "EVERYTHING. IS. GREAT.\n (ﾉ°□°) ﾉ‿ ┻━┻\n".yellow
    puts "[...You back away slowly and exit the store]\n".italic
    exit
  when 5
    get_rejected
  end
end
```

```ruby
  def go_to_counter
    puts ["[An all too bubbly 5ft 4 blonde girl greets you with a massive smile.]\n".italic,
    "[The blonde's gaze at the counter is fixated on group of rebellious looking youths poking " +
    "around the store. After a moment she notices you are there.]\n".italic, "[The small blonde girl " +
    "is humming along to the Beatles song that's playing over the speakers]\n".italic].sample
    # Print a random string from this array each time this method is called
    counter_interaction
  end
end
```

IN APP

› Actually... I wanted to see if you were busy later....want to grab a drink?

ಠ_ಠ Oh, sorry. My sister's friend's fish died, and yes, it was tragic.

How can I help?
(Press ↑/↓ arrow to move and Enter to select)
› I'd like to pay for these
Errr actually, on second thought. I might have a second look around.
I changed my mind and I'm just gonna head out.
Actually... I wanted to see if you were busy later....want to grab a drink?

ಠ_ಠ The voices in my head are telling me to say no. Sorry!

ಠ_ಠ That sounds really fun! But sorry, I'm going to be busy not doing that.

ಠ_ಠ Oh what a shame, I actually have plans to teach my ferret to yodel. Some other time maybe (not)?

› I changed my mind, and I'm going to head out. Can I just leave all these records with you?

Of coooooourse. Nothing I would enjoy more. So wonderful. I love my job. I love sorting. EVERYTHING. IS. GREAT.
(╯°□°)╯ ┻┻

[...You back away slowly and exit the store]

angieloux@Angelas-MacBook-Pro src %

I changed my mind and I'm just gonna head out.

d(^o^)b ♬
All right, see ya!
angieloux@Angelas-MacBook-Pro src %

› I'd like to pay for these

Awesome! I'll just tally that up for you...

AJ's Records: Purchase Summary

Catno	Album	Artist	Year	Price
364	American Recordings	Johnny Cash	1994	$39.95
333	Superunknown	Soundgarden	1994	$55.95
287	Something Else by The Kinks	The Kinks	1967	$65.95
5	Rubber Soul	The Beatles	1965	$49.99
99	Odessey And Oracle	The Zombies	1968	$69.99

Your total comes to $281.83, please! Did you want to go ahead and pay? (Y/n) y

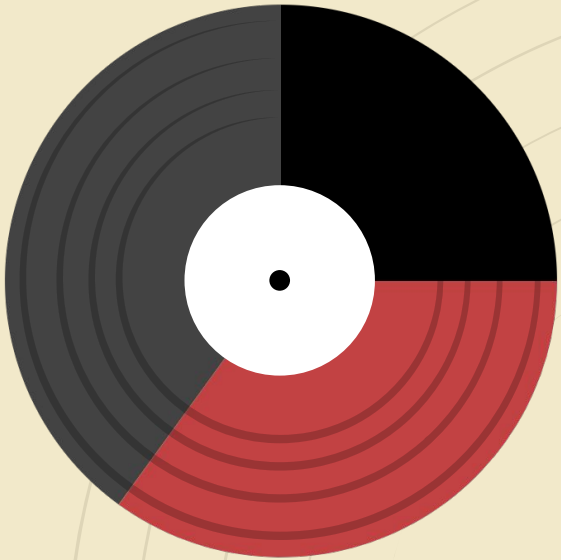
[You grimace at the total, and hand over your well-worn credit card]

Progress: |==================
d(^o^)b ♬ Thanks for shopping at AJ's Records! Come back anytime.
angieloux@Angelas-MacBook-Pro src %

› Errr actually, on second thought. I might have a second look around.

d(^o^)b ♬
No worries! Come over here when you're ready.
Where should I go? (Press ↑/↓ arrow to move and
› Cruise over to the VINYL
Review what's in your CART
Head over to the COUNTER
Turn around and leave. (EXIT)

CHALLENGES



SCOPE
How to work with multiple modules, understanding how they ‘talk’ to one another

DE-SPRINKLING
Creating what is realistic and achievable (originally had Games, Toys, Listening Station)

ARCHITECTURE
Structuring the entire app was without a doubt the most challenging aspect of all

ETHICAL CONSIDERATIONS

COPYRIGHT

- Music for the Listening Station (NOTE: this sprinkle was not used in final program, but may be added at a later date)
- Data libraries for products

INSPIRATION VS COPYING

- Using past examples to guide my understanding, but coding from scratch

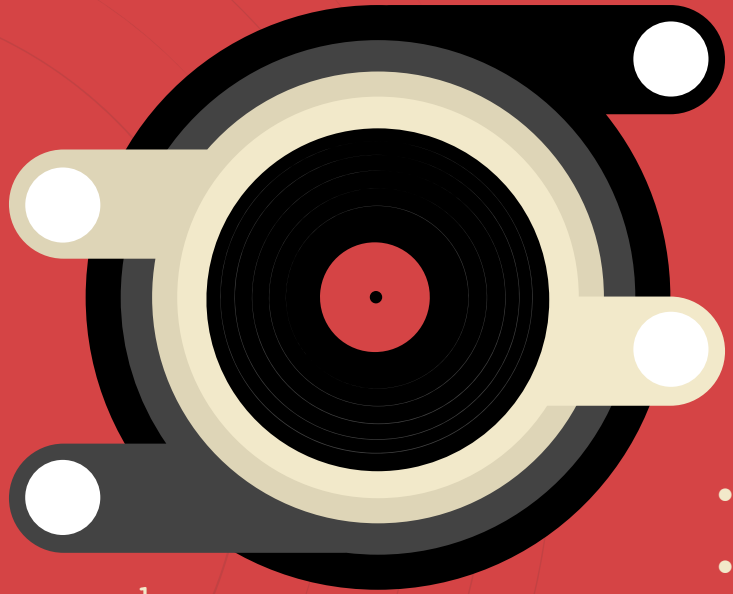
FAVOURITE PARTS

VIBE

Past experience in a record store, and I loooove music

SEARCH FILTER

Super flexible search functionality



PUSHING MYSELF

Challenged myself to go above and beyond, so I could learn as much as possible.

SPRINKLES/ GEMS

- Mimicking real store experience (artii)
- Terminal-table for checkout
- Using json to have a wealth of products for visitors to search through

THANKS!