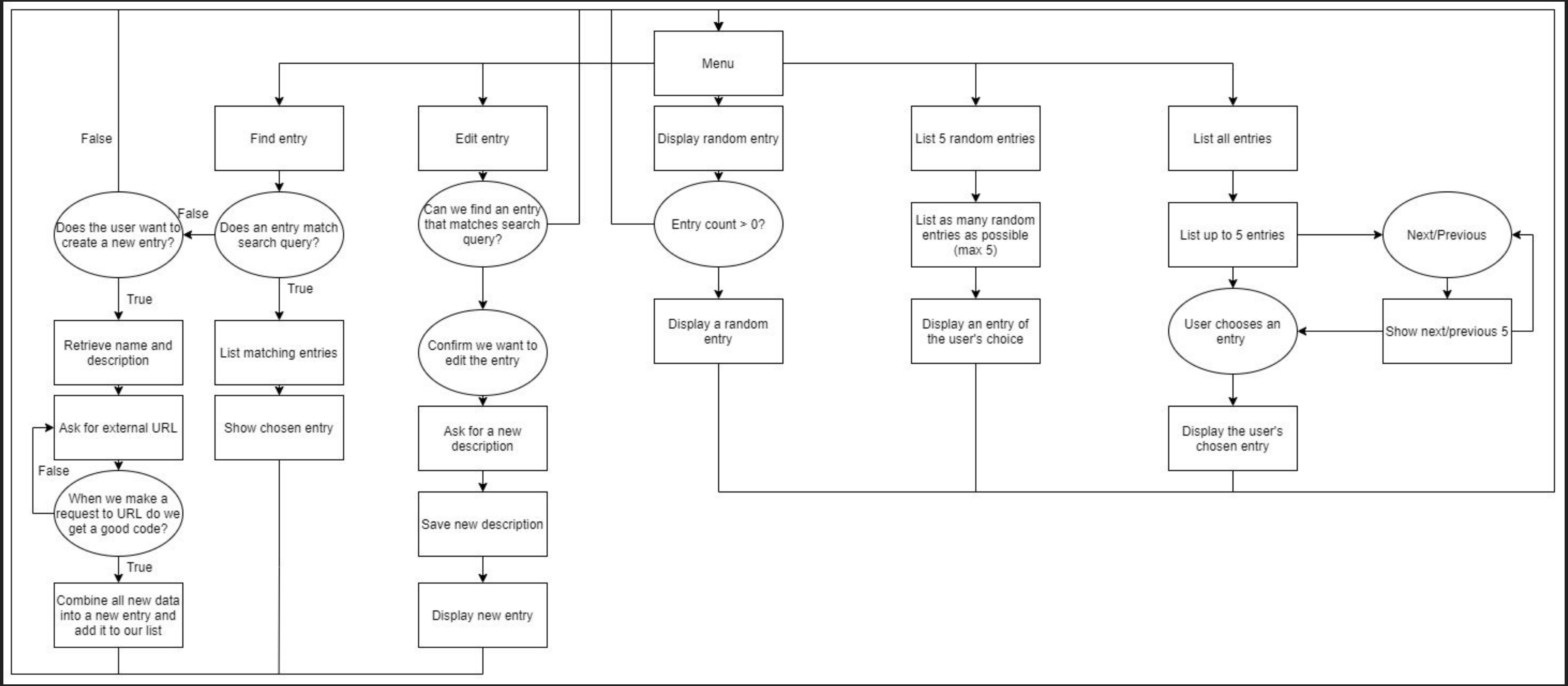
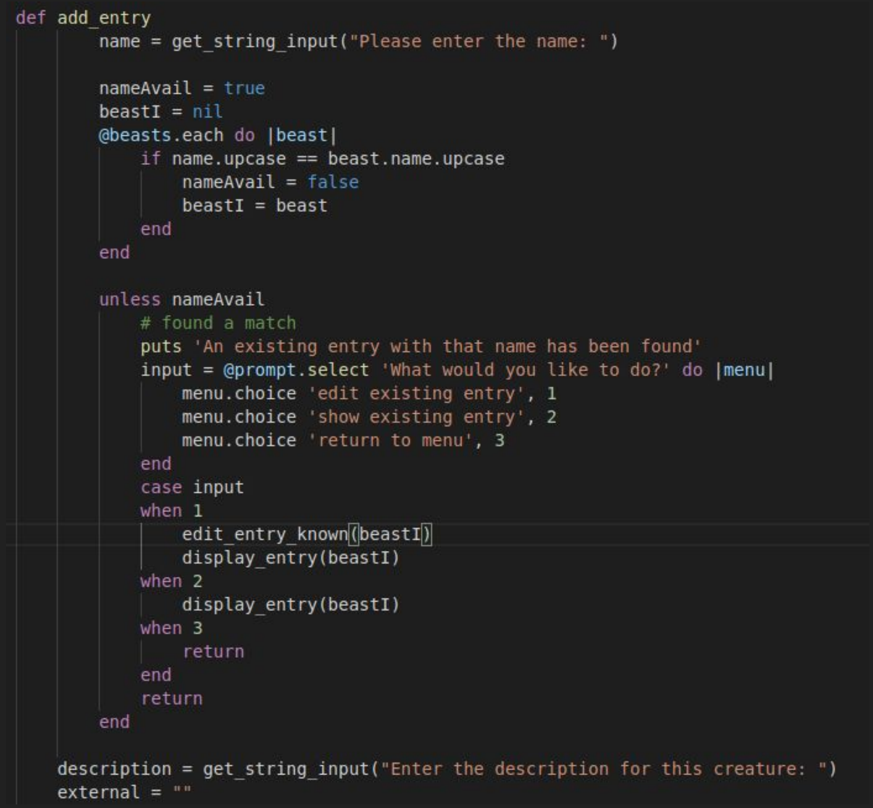
Bestiary terminal app

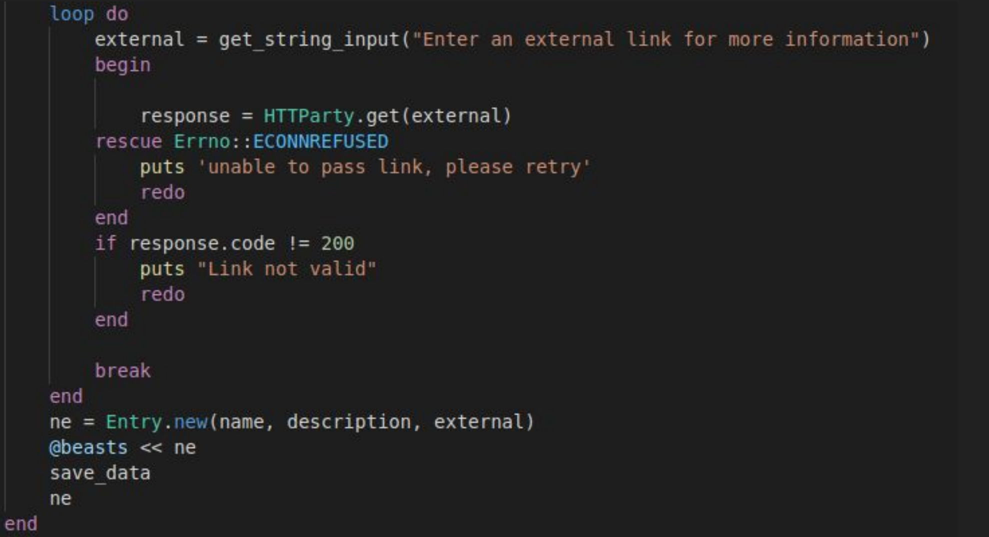
fancy


Menu
Find entry
Edit entry
Display random entry
List 5 random entries
List all entries
False
Does the user want to create a new entry?
False
Does an entry match search query?
Can we find an entry that matches search query?
Entry count > 0?
List as many random entries as possible (max 5)
List up to 5 entries
Next/Previous
True
True
Retrieve name and description
List matching entries
Confirm we want to edit the entry
Display a random entry
Display an entry of the user's choice
User chooses an entry
Show next/previous 5
Ask for external URL
Show chosen entry
Ask for a new description
Display the user's chosen entry
False
When we make a request to URL do we get a good code?
Save new description
True
Combine all new data into a new entry and add it to our list
Display new entry

Important parts of code

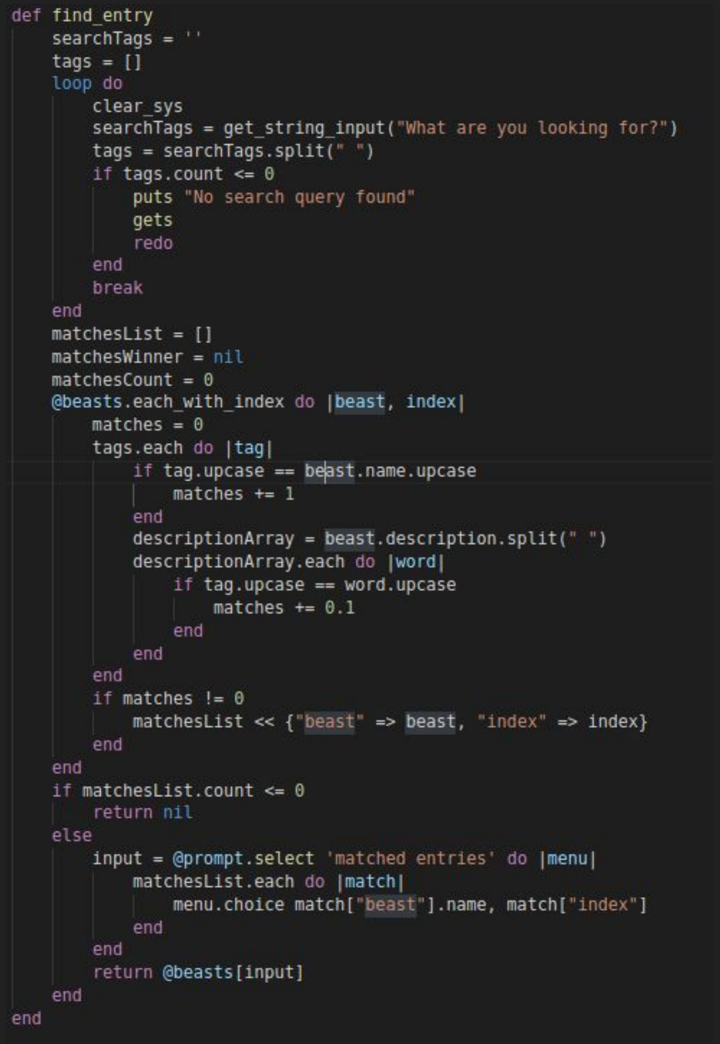


```ruby
def add_entry
        name = get_string_input("Please enter the name: ")

        nameAvail = true
        beastI = nil
        @beasts.each do |beast|
            if name.upcase == beast.name.upcase
                nameAvail = false
                beastI = beast
            end
        end

        unless nameAvail
            # found a match
            puts 'An existing entry with that name has been found'
            input = @prompt.select 'What would you like to do?' do |menu|
                menu.choice 'edit existing entry', 1
                menu.choice 'show existing entry', 2
                menu.choice 'return to menu', 3
            end
            case input
            when 1
                edit_entry_known(beastI)
                display_entry(beastI)
            when 2
                display_entry(beastI)
            when 3
                return
            end
            return
        end

    description = get_string_input("Enter the description for this creature: ")
    external = ""
```



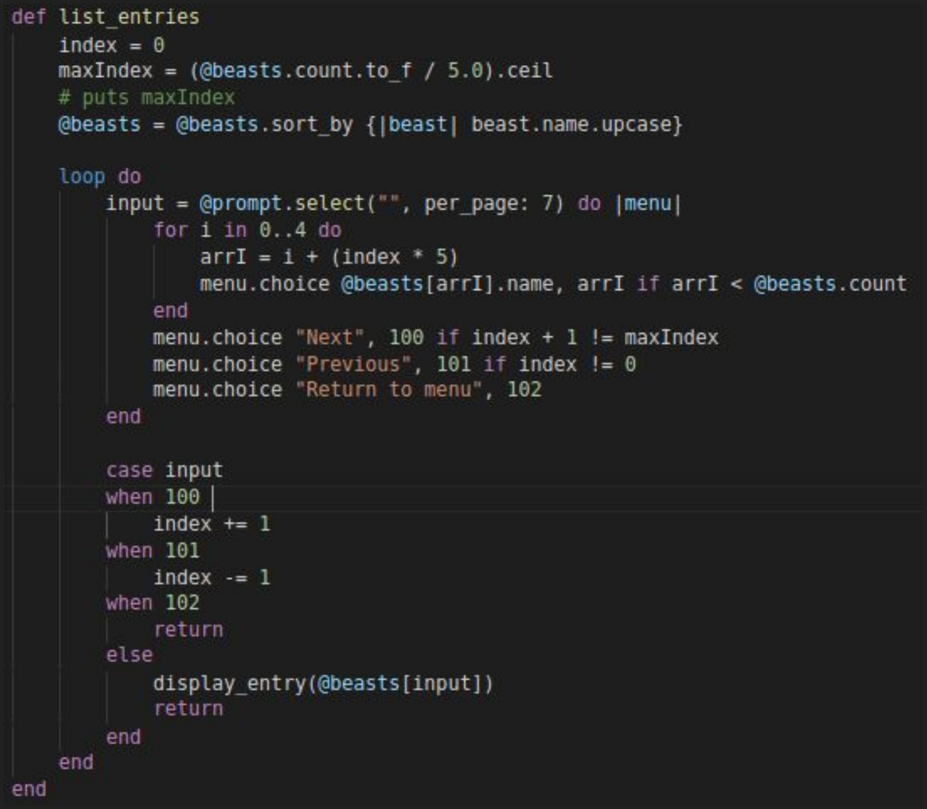
```ruby
    loop do
        external = get_string_input("Enter an external link for more information")
        begin

            response = HTTParty.get(external)
        rescue Errno::ECONNREFUSED
            puts 'unable to pass link, please retry'
            redo
        end
        if response.code != 200
            puts "Link not valid"
            redo
        end

        break
    end
    ne = Entry.new(name, description, external)
    @beasts << ne
    save_data
    ne
end
```

Important parts of code



```
def find_entry
    searchTags = ''
    tags = []
    loop do
        clear_sys
        searchTags = get_string_input("What are you looking for?")
        tags = searchTags.split(" ")
        if tags.count <= 0
            puts "No search query found"
            gets
            redo
        end
        break
    end
    matchesList = []
    matchesWinner = nil
    matchesCount = 0
    @beasts.each_with_index do |beast, index|
        matches = 0
        tags.each do |tag|
            if tag.upcase == beast.name.upcase
                matches += 1
            end
            descriptionArray = beast.description.split(" ")
            descriptionArray.each do |word|
                if tag.upcase == word.upcase
                    matches += 0.1
                end
            end
        end
        if matches != 0
            matchesList << {"beast" => beast, "index" => index}
        end
    end
    if matchesList.count <= 0
        return nil
    else
        input = @prompt.select 'matched entries' do |menu|
            matchesList.each do |match|
                menu.choice match["beast"].name, match["index"]
            end
        end
        return @beasts[input]
    end
end
```

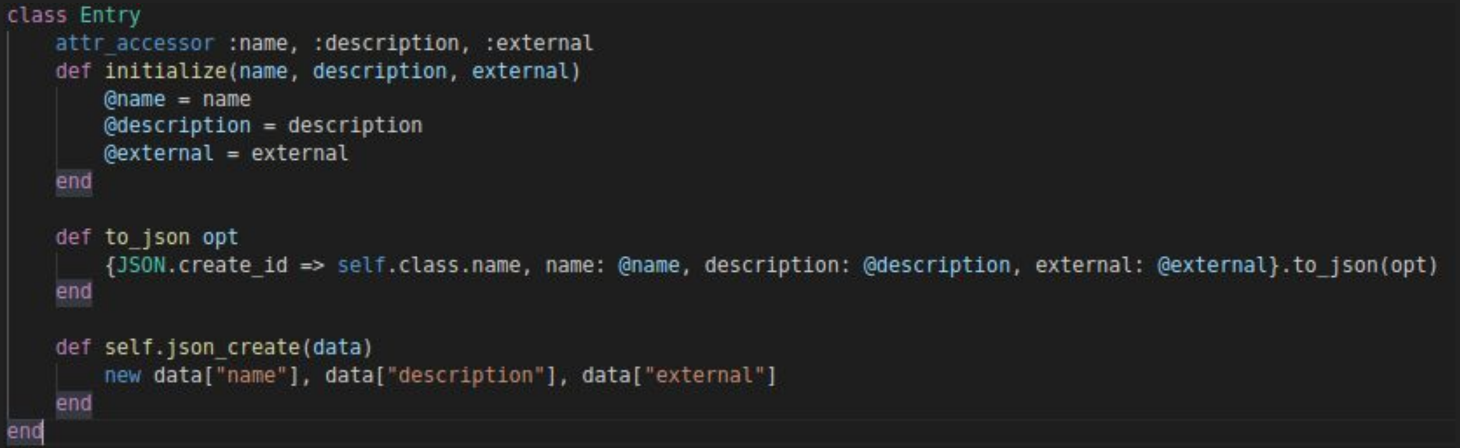
Important parts of code



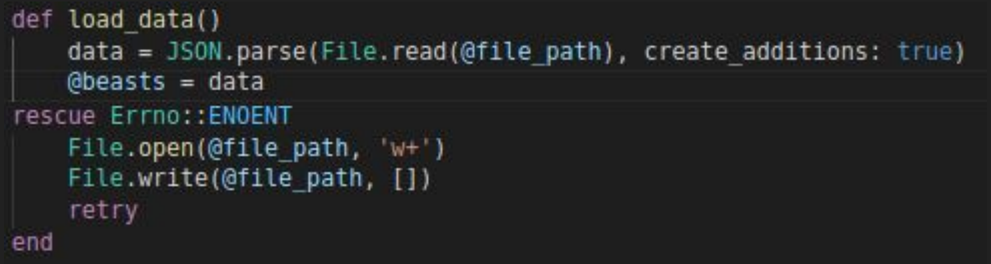
```ruby
def list_entries
    index = 0
    maxIndex = (@beasts.count.to_f / 5.0).ceil
    # puts maxIndex
    @beasts = @beasts.sort_by {|beast| beast.name.upcase}

    loop do
        input = @prompt.select("", per_page: 7) do |menu|
            for i in 0..4 do
                arrI = i + (index * 5)
                menu.choice @beasts[arrI].name, arrI if arrI < @beasts.count
            end
            menu.choice "Next", 100 if index + 1 != maxIndex
            menu.choice "Previous", 101 if index != 0
            menu.choice "Return to menu", 102
        end

        case input
        when 100
            index += 1
        when 101
            index -= 1
        when 102
            return
        else
            display_entry(@beasts[input])
            return
        end
    end
end
```

Challenges

Saving/Loading



```ruby
class Entry
    attr_accessor :name, :description, :external
    def initialize(name, description, external)
        @name = name
        @description = description
        @external = external
    end

    def to_json opt
        {JSON.create_id => self.class.name, name: @name, description: @description, external: @external}.to_json(opt)
    end

    def self.json_create(data)
        new data["name"], data["description"], data["external"]
    end
end
```



```ruby
def load_data()
    data = JSON.parse(File.read(@file_path), create_additions: true)
    @beasts = data
rescue Errno::ENOENT
    File.open(@file_path, 'w+')
    File.write(@file_path, [])
    retry
end
```