T1A3 – Terminal Application

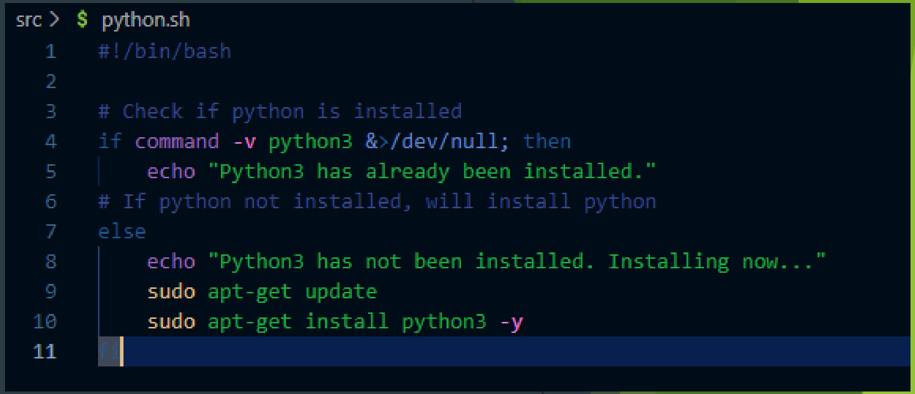
Jimmy Lam

Application overview

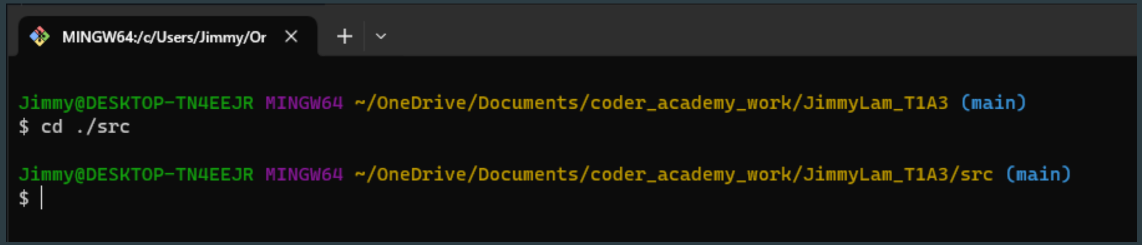
- Main features of the word guessing game:
- Word list and random word generator, a random word will be generated from a list of words for the players to guess from.
- A hint system that gives players a hint when they reach a certain number of guesses remaining
- A previous guess history feature that shows all previous guesses that a player has made and disallows duplicates.
- A way for players to view the word list and the rules on how to play.
- A greeting and start menu to welcome players.

Walk through of Application

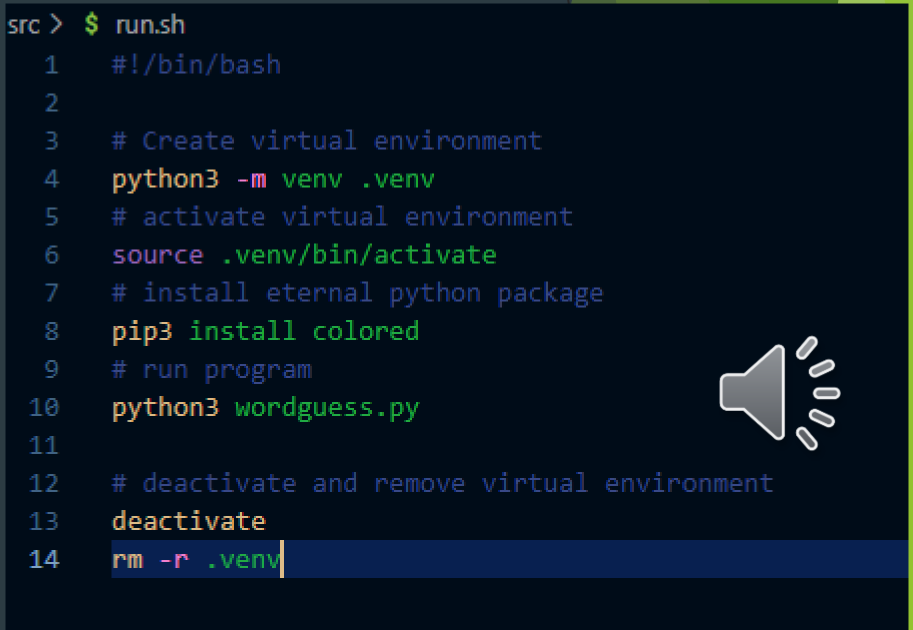
User opens terminal and types "./python.sh" to run the file that checks if python is installed. If not installed, then python will install itself. The second command "./run.sh" will open a virtual environment and download the necessary external packages and run the main code.



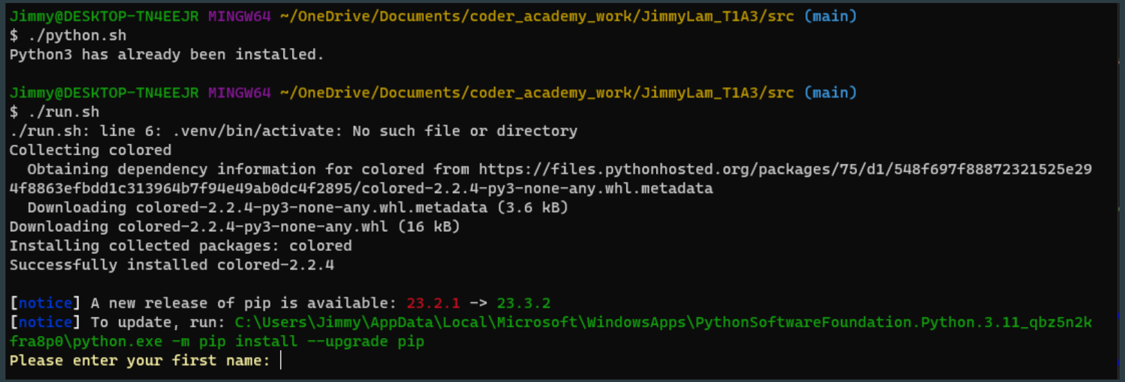
```bash
#!/bin/bash

# Check if python is installed
if command -v python3 &>/dev/null; then
    echo "Python3 has already been installed."
# If python not installed, will install python
else
    echo "Python3 has not been installed. Installing now..."
    sudo apt-get update
    sudo apt-get install python3 -y
fi
```



```
Jimmy@DESKTOP-TN4EEJR MINGW64 ~/OneDrive/Documents/coder_academy_work/JimmyLam_T1A3 (main)
$ cd ./src

Jimmy@DESKTOP-TN4EEJR MINGW64 ~/OneDrive/Documents/coder_academy_work/JimmyLam_T1A3/src (main)
$ |
```



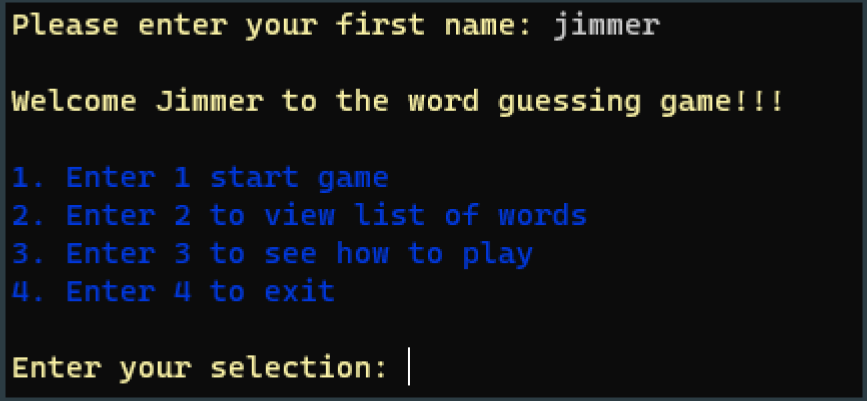
```bash
#!/bin/bash

# Create virtual environment
python3 -m venv .venv
# activate virtual environment
source .venv/bin/activate
# install eternal python package
pip3 install colored
# run program
python3 wordguess.py

# deactivate and remove virtual environment
deactivate
rm -r .venv
```



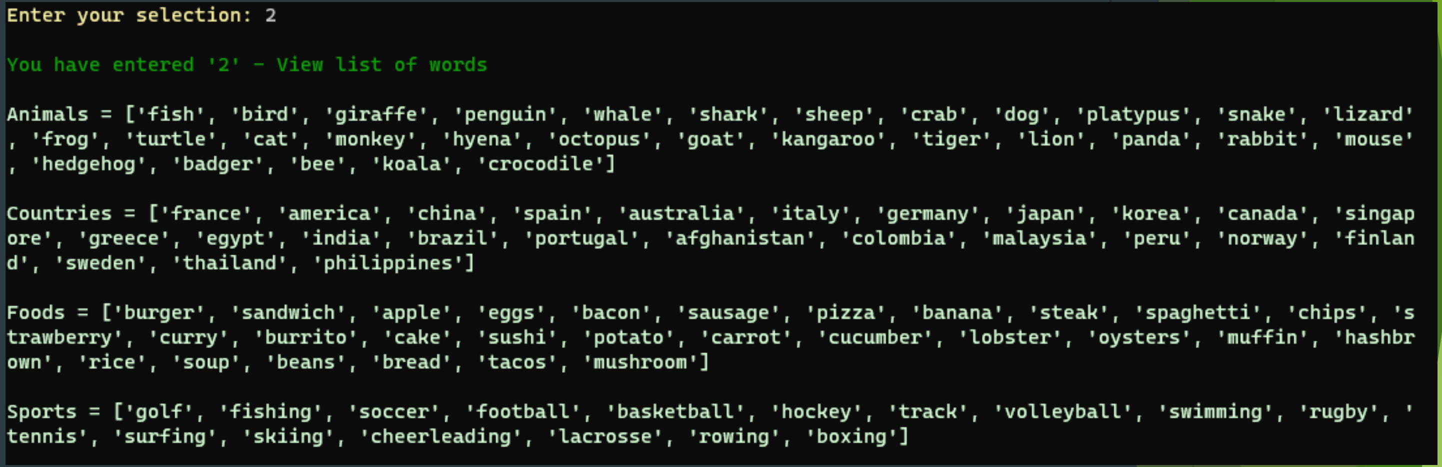
```
Jimmy@DESKTOP-TN4EEJR MINGW64 ~/OneDrive/Documents/coder_academy_work/JimmyLam_T1A3/src (main)
$ ./python.sh
Python3 has already been installed.

Jimmy@DESKTOP-TN4EEJR MINGW64 ~/OneDrive/Documents/coder_academy_work/JimmyLam_T1A3/src (main)
$ ./run.sh
./run.sh: line 6: .venv/bin/activate: No such file or directory
Collecting colored
  Obtaining dependency information for colored from https://files.pythonhosted.org/packages/75/d1/548f697f88872321525e29
4f8863efbdd1c313964b7f94e49ab0dc4f2895/colored-2.2.4-py3-none-any.whl.metadata
  Downloading colored-2.2.4-py3-none-any.whl.metadata (3.6 kB)
Downloading colored-2.2.4-py3-none-any.whl (16 kB)
Installing collected packages: colored
Successfully installed colored-2.2.4

[notice] A new release of pip is available: 23.2.1 -> 23.3.2
[notice] To update, run: C:\Users\Jimmy\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.11_qbz5n2k
fra8p0\python.exe -m pip install --upgrade pip
Please enter your first name: |
```

Walkthrough of game and features

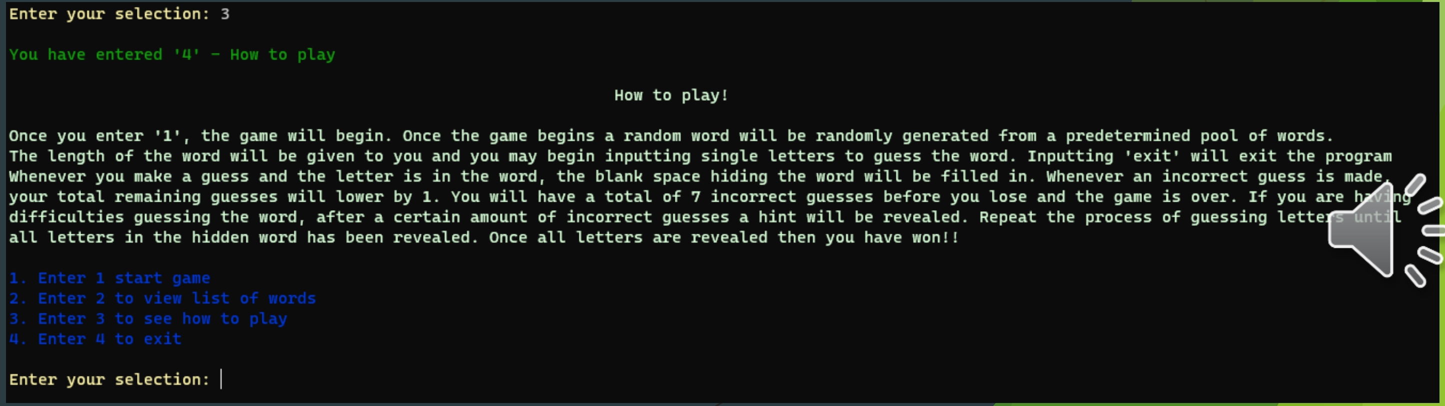
- First user will input their first name and then will shortly be greeted and met with a menu. Depending on the user’s next inputs the outcome will vary. “1” will start the game, “2” will show the list of words, “3” will show rules and “4” will exit the game completely.



```
Please enter your first name: jimmer

Welcome Jimmer to the word guessing game!!!

1. Enter 1 start game
2. Enter 2 to view list of words
3. Enter 3 to see how to play
4. Enter 4 to exit

Enter your selection: |
```



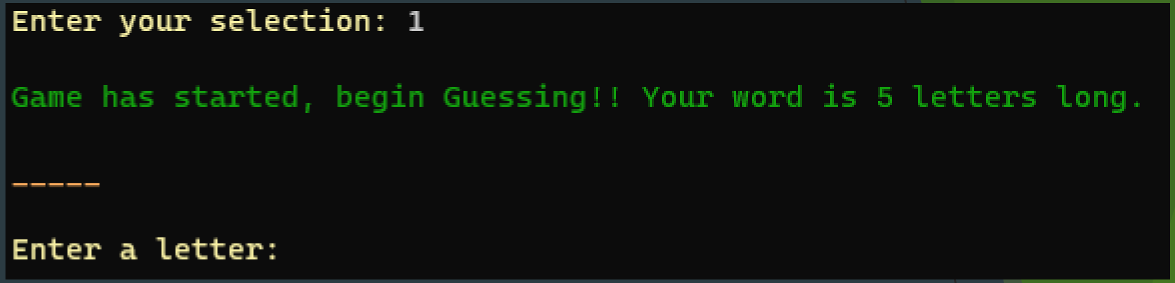
```
Enter your selection: 2

You have entered '2' - View list of words

Animals = ['fish', 'bird', 'giraffe', 'penguin', 'whale', 'shark', 'sheep', 'crab', 'dog', 'platypus', 'snake', 'lizard'
, 'frog', 'turtle', 'cat', 'monkey', 'hyena', 'octopus', 'goat', 'kangaroo', 'tiger', 'lion', 'panda', 'rabbit', 'mouse'
, 'hedgehog', 'badger', 'bee', 'koala', 'crocodile']

Countries = ['france', 'america', 'china', 'spain', 'australia', 'italy', 'germany', 'japan', 'korea', 'canada', 'singap
ore', 'greece', 'egypt', 'india', 'brazil', 'portugal', 'afghanistan', 'colombia', 'malaysia', 'peru', 'norway', 'finlan
d', 'sweden', 'thailand', 'philippines']

Foods = ['burger', 'sandwich', 'apple', 'eggs', 'bacon', 'sausage', 'pizza', 'banana', 'steak', 'spaghetti', 'chips', 's
trawberry', 'curry', 'burrito', 'cake', 'sushi', 'potato', 'carrot', 'cucumber', 'lobster', 'oysters', 'muffin', 'hashbr
own', 'rice', 'soup', 'beans', 'bread', 'tacos', 'mushroom']

Sports = ['golf', 'fishing', 'soccer', 'football', 'basketball', 'hockey', 'track', 'volleyball', 'swimming', 'rugby', '
tennis', 'surfing', 'skiing', 'cheerleading', 'lacrosse', 'rowing', 'boxing']
```



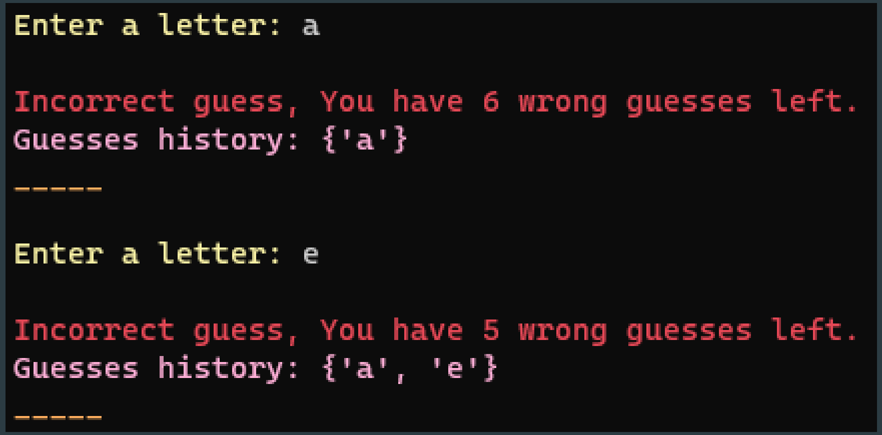
```
Enter your selection: 3

You have entered '4' - How to play

                                                  How to play!

Once you enter '1', the game will begin. Once the game begins a random word will be randomly generated from a predetermined pool of words.
The length of the word will be given to you and you may begin inputting single letters to guess the word. Inputting 'exit' will exit the program
Whenever you make a guess and the letter is in the word, the blank space hiding the word will be filled in. Whenever an incorrect guess is made,
your total remaining guesses will lower by 1. You will have a total of 7 incorrect guesses before you lose and the game is over. If you are having
difficulties guessing the word, after a certain amount of incorrect guesses a hint will be revealed. Repeat the process of guessing letters until
all letters in the hidden word has been revealed. Once all letters are revealed then you have won!!

1. Enter 1 start game
2. Enter 2 to view list of words
3. Enter 3 to see how to play
4. Enter 4 to exit

Enter your selection: |
```

Walkthrough of game and features

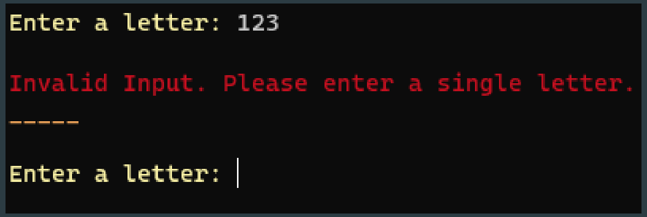
- When the game has been started a prompt providing the game has been started will appear and a hint to show how long your word is will also appear. The game will then ask for single letter inputs to begin guessing the word. Numbers, special characters and inputs longer than 1 letter will cause an error to pop up saying invalid input. Inputting “exit” at any time during the game will quit and close the game.



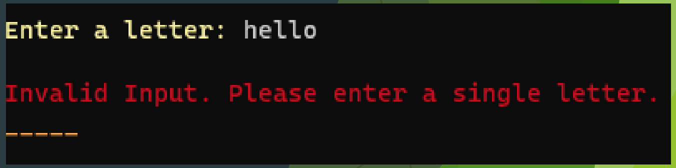
```
Enter your selection: 1

Game has started, begin Guessing!! Your word is 5 letters long.


-----

Enter a letter:
```



```
Enter a letter: a

Incorrect guess, You have 6 wrong guesses left.
Guesses history: {'a'}
-----

Enter a letter: e

Incorrect guess, You have 5 wrong guesses left.
Guesses history: {'a', 'e'}
-----
```



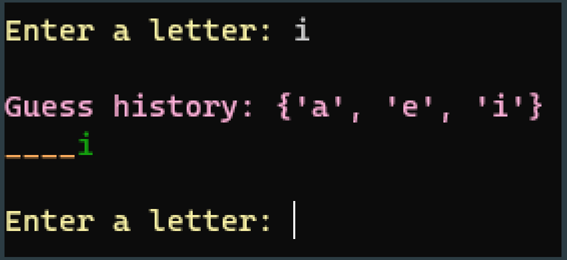
```
Enter a letter: 123

Invalid Input. Please enter a single letter.
-----

Enter a letter: |
```



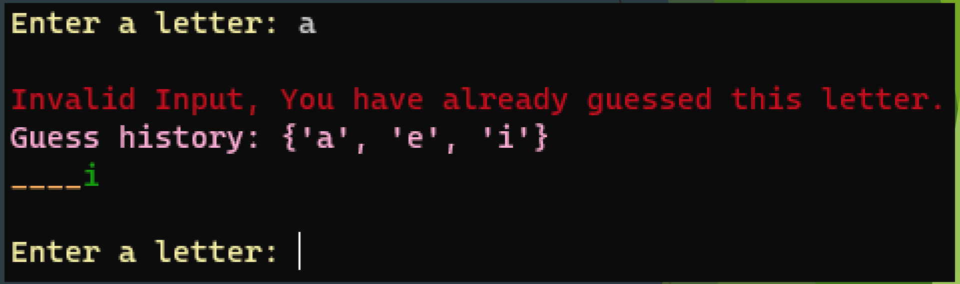
```
Enter a letter: hello

Invalid Input. Please enter a single letter.
-----
```

Walkthrough of game and features continued…

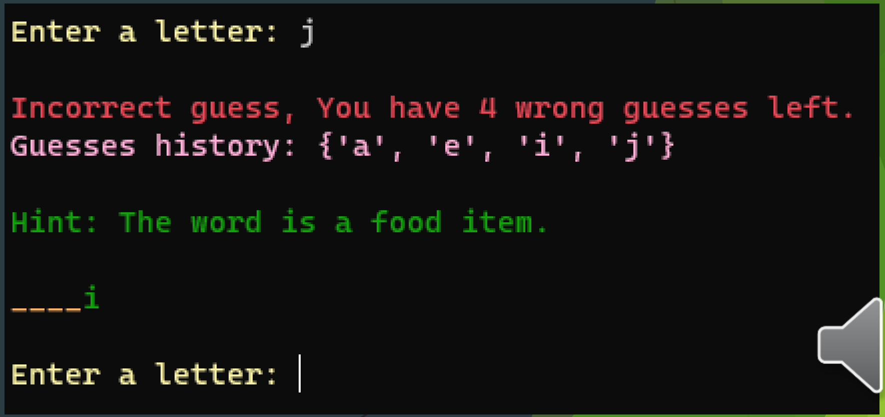
- When a letter is guessed correctly, It is displayed on screen for the rest of the game. Previous guesses are also displayed throughout the game for quality of life for the player. Duplicate guesses will not be allowed and will not lower guess count. Incorrect guesses will lower total guess count by 1 and at a certain amount of guesses a hint will be revealed to players to assist them.



```
Enter a letter: i

Guess history: {'a', 'e', 'i'}
____i

Enter a letter: 
```



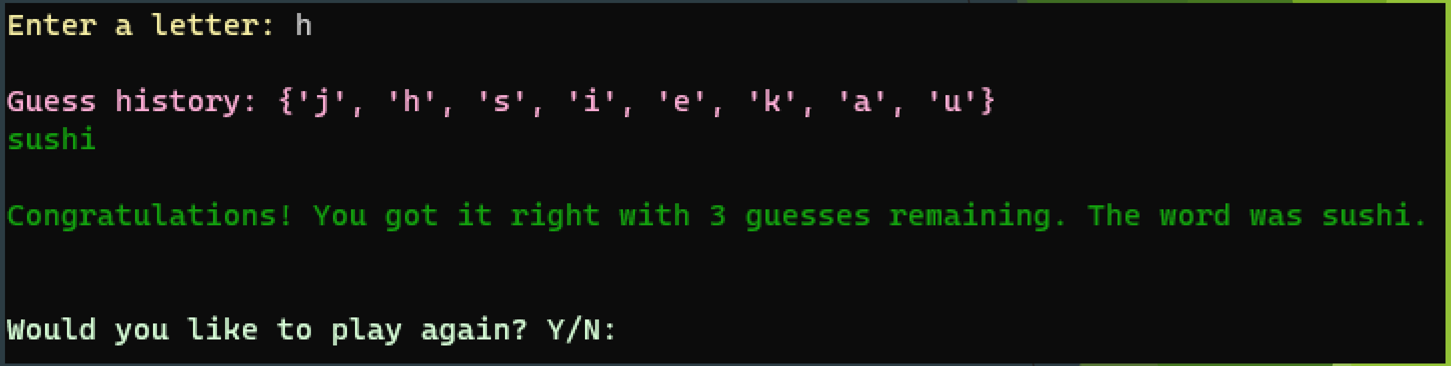
```
Enter a letter: a

Invalid Input, You have already guessed this letter.
Guess history: {'a', 'e', 'i'}
____i

Enter a letter: 
```



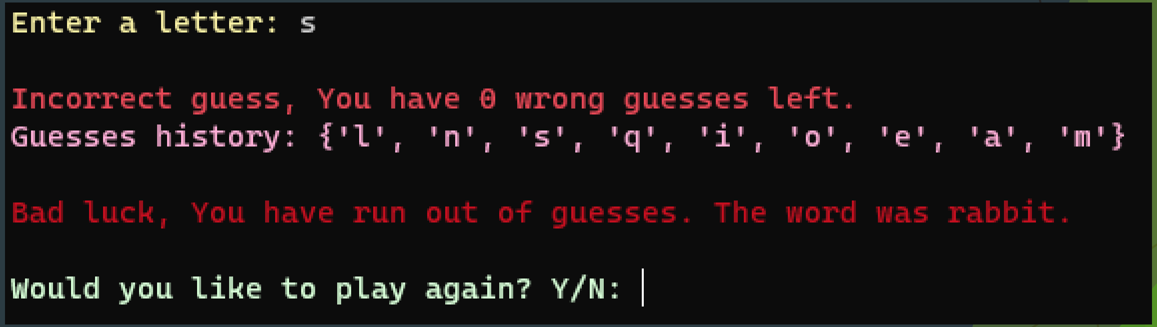
```
Enter a letter: j

Incorrect guess, You have 4 wrong guesses left.
Guesses history: {'a', 'e', 'i', 'j'}

Hint: The word is a food item.

____i

Enter a letter: 
```

Walkthrough of game and features continued…

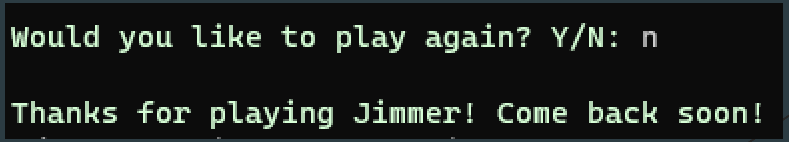
- When word is guessed correctly, You will be congratulated, and it will show up with the remaining guesses and the word given. A prompt to play again will also be given, typing Y or N respectively will either start a new game with a new word or quit. The same prompt will appear if the game is lost. When application is quit, the virtual environment will deactivate and then delete itself.



```
Enter a letter: h

Guess history: {'j', 'h', 's', 'i', 'e', 'k', 'a', 'u'}
sushi

Congratulations! You got it right with 3 guesses remaining. The word was sushi.


Would you like to play again? Y/N:
```



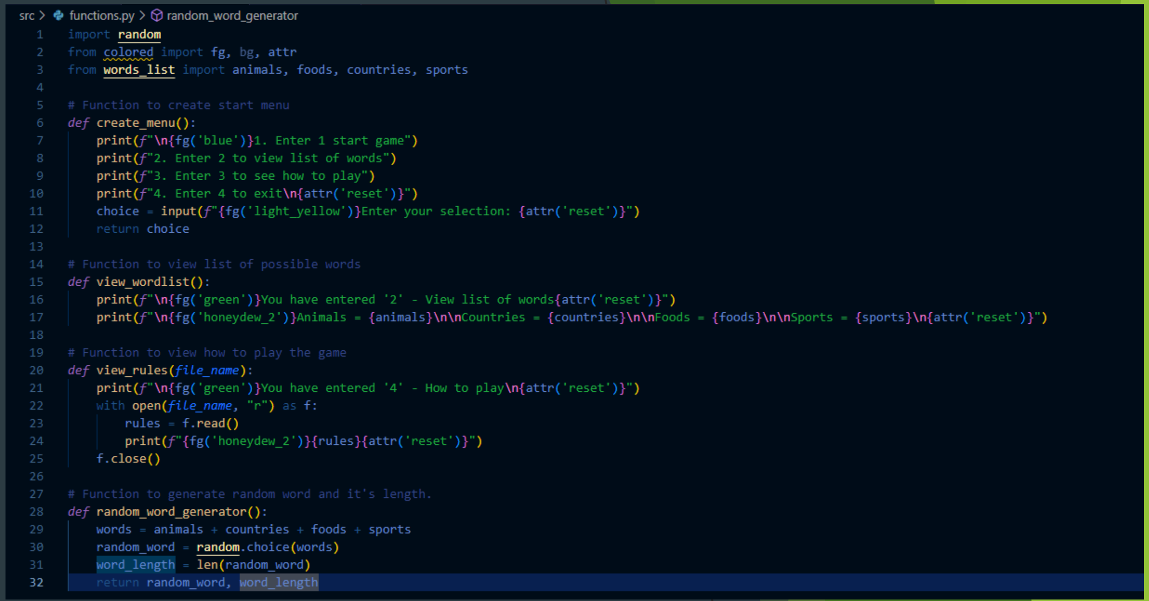
```
Enter a letter: s

Incorrect guess, You have 0 wrong guesses left.
Guesses history: {'l', 'n', 's', 'q', 'i', 'o', 'e', 'a', 'm'}

Bad luck, You have run out of guesses. The word was rabbit.

Would you like to play again? Y/N:
```



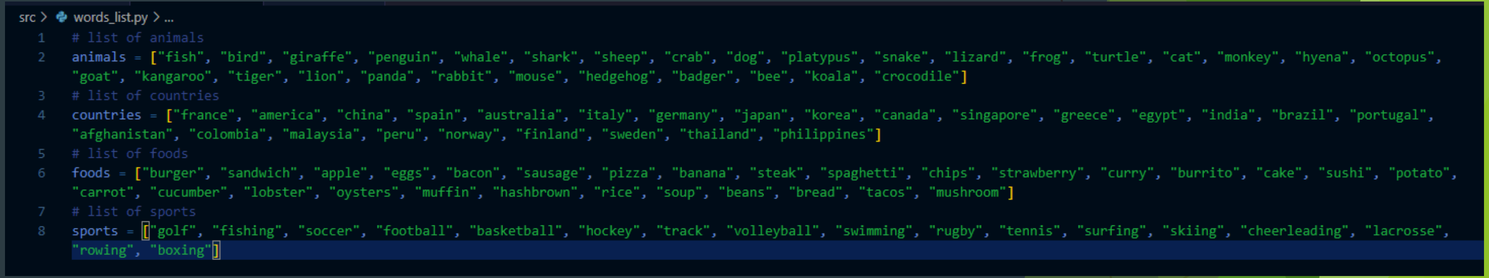
```
Would you like to play again? Y/N: n

Thanks for playing Jimmer! Come back soon!
```

Code Overview

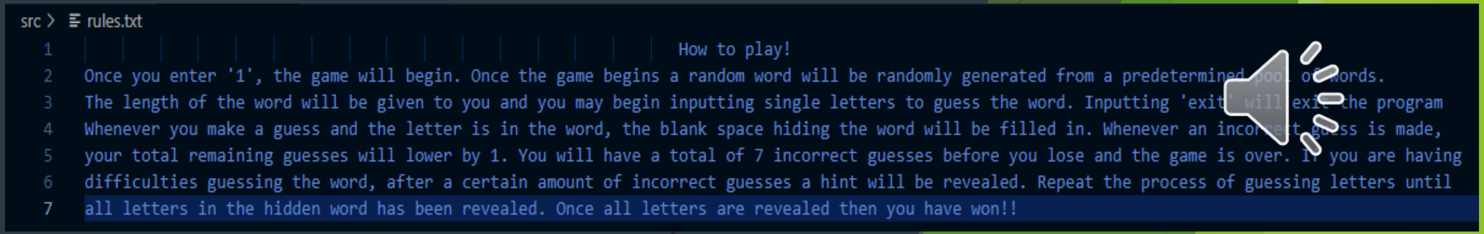
- This is my code. It shows all the functions and all the lists and files included within my source code.



```python
import random
from colored import fg, bg, attr
from words_list import animals, foods, countries, sports

# Function to create start menu
def create_menu():
    print(f"\n{fg('blue')}1. Enter 1 start game")
    print(f"2. Enter 2 to view list of words")
    print(f"3. Enter 3 to see how to play")
    print(f"4. Enter 4 to exit\n{attr('reset')}")
    choice = input(f"{fg('light_yellow')}Enter your selection: {attr('reset')}")
    return choice

# Function to view list of possible words
def view_wordlist():
    print(f"\n{fg('green')}You have entered '2' - View list of words{attr('reset')}")
    print(f"\n{fg('honeydew_2')}Animals = {animals}\n\nCountries = {countries}\n\nFoods = {foods}\n\nSports = {sports}\n{attr('reset')}")

# Function to view how to play the game
def view_rules(file_name):
    print(f"\n{fg('green')}You have entered '4' - How to play\n{attr('reset')}")
    with open(file_name, "r") as f:
        rules = f.read()
        print(f"{fg('honeydew_2')}{rules}{attr('reset')}")
    f.close()

# Function to generate random word and it's length.
def random_word_generator():
    words = animals + countries + foods + sports
    random_word = random.choice(words)
    word_length = len(random_word)
    return random_word, word_length
```

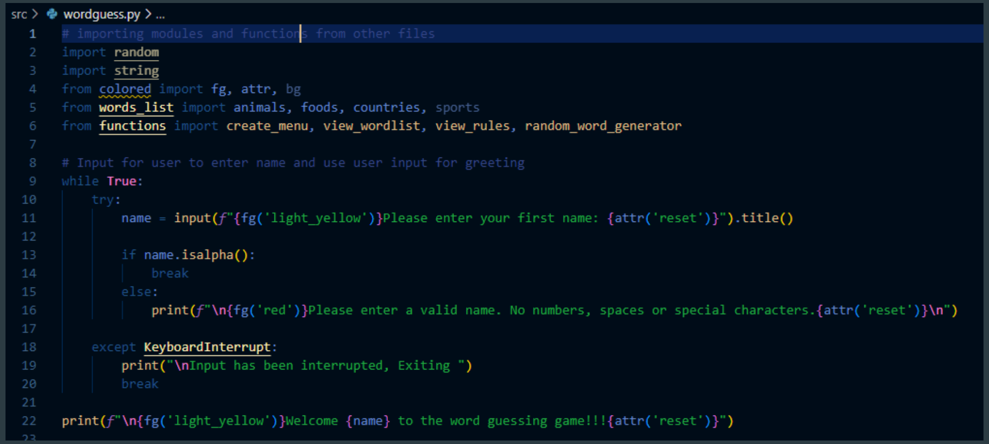


```python
# list of animals
animals = ["fish", "bird", "giraffe", "penguin", "whale", "shark", "sheep", "crab", "dog", "platypus", "snake", "lizard", "frog", "turtle", "cat", "monkey", "hyena", "octopus", "goat", "kangaroo", "tiger", "lion", "panda", "rabbit", "mouse", "hedgehog", "badger", "bee", "koala", "crocodile"]
# list of countries
countries = ["france", "america", "china", "spain", "australia", "italy", "germany", "japan", "korea", "canada", "singapore", "greece", "egypt", "india", "brazil", "portugal", "afghanistan", "colombia", "malaysia", "peru", "norway", "finland", "sweden", "thailand", "philippines"]
# list of foods
foods = ["burger", "sandwich", "apple", "eggs", "bacon", "sausage", "pizza", "banana", "steak", "spaghetti", "chips", "strawberry", "curry", "burrito", "cake", "sushi", "potato", "carrot", "cucumber", "lobster", "oysters", "muffin", "hashbrown", "rice", "soup", "beans", "bread", "tacos", "mushroom"]
# list of sports
sports = ["golf", "fishing", "soccer", "football", "basketball", "hockey", "track", "volleyball", "swimming", "rugby", "tennis", "surfing", "skiing", "cheerleading", "lacrosse", "rowing", "boxing"]
```



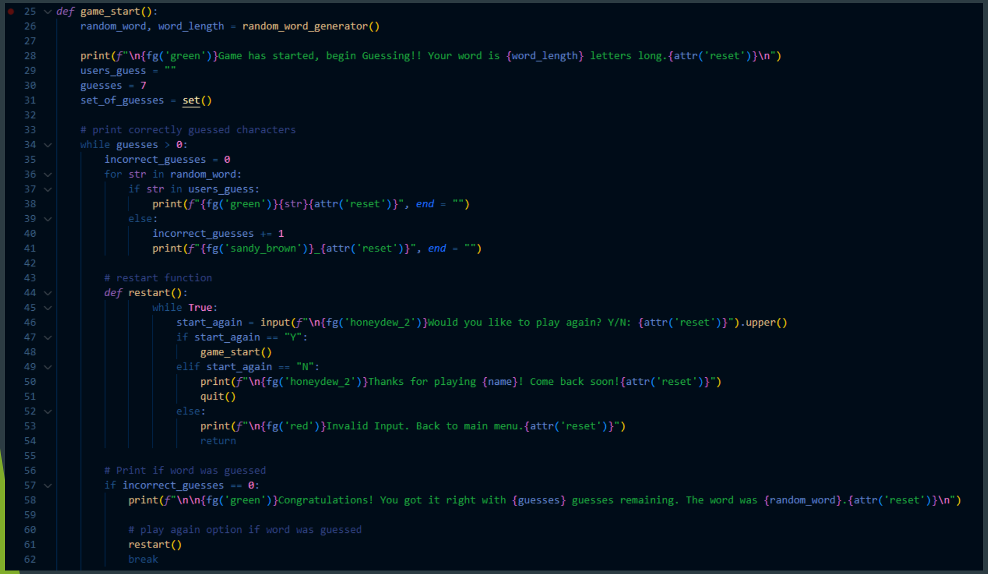
```
                                                How to play!
Once you enter '1', the game will begin. Once the game begins a random word will be randomly generated from a predetermined pool of words.
The length of the word will be given to you and you may begin inputting single letters to guess the word. Inputting 'exit' will exit the program
Whenever you make a guess and the letter is in the word, the blank space hiding the word will be filled in. Whenever an incorrect guess is made,
your total remaining guesses will lower by 1. You will have a total of 7 incorrect guesses before you lose and the game is over. If you are having
difficulties guessing the word, after a certain amount of incorrect guesses a hint will be revealed. Repeat the process of guessing letters until
all letters in the hidden word has been revealed. Once all letters are revealed then you have won!!
```

Code overview continued

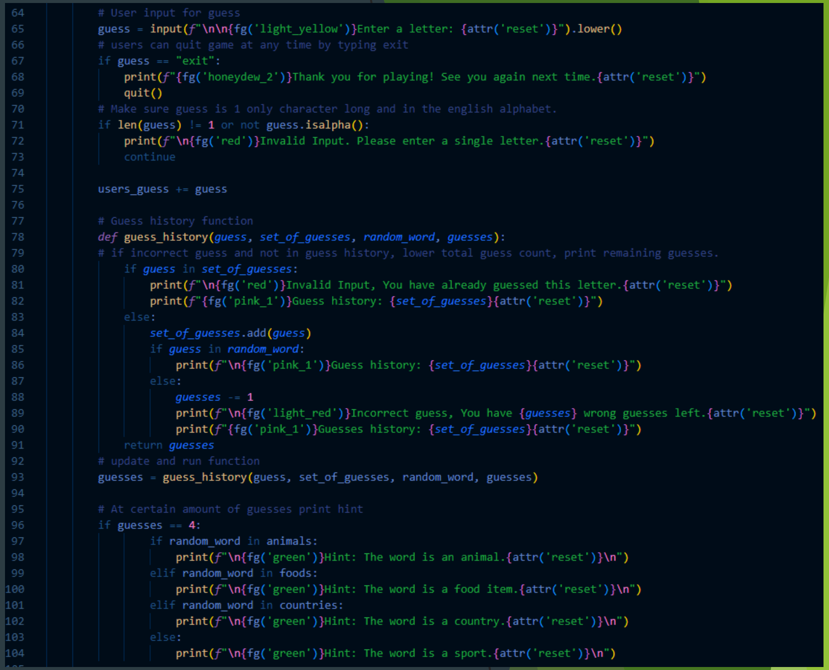


src > wordguess.py > ...

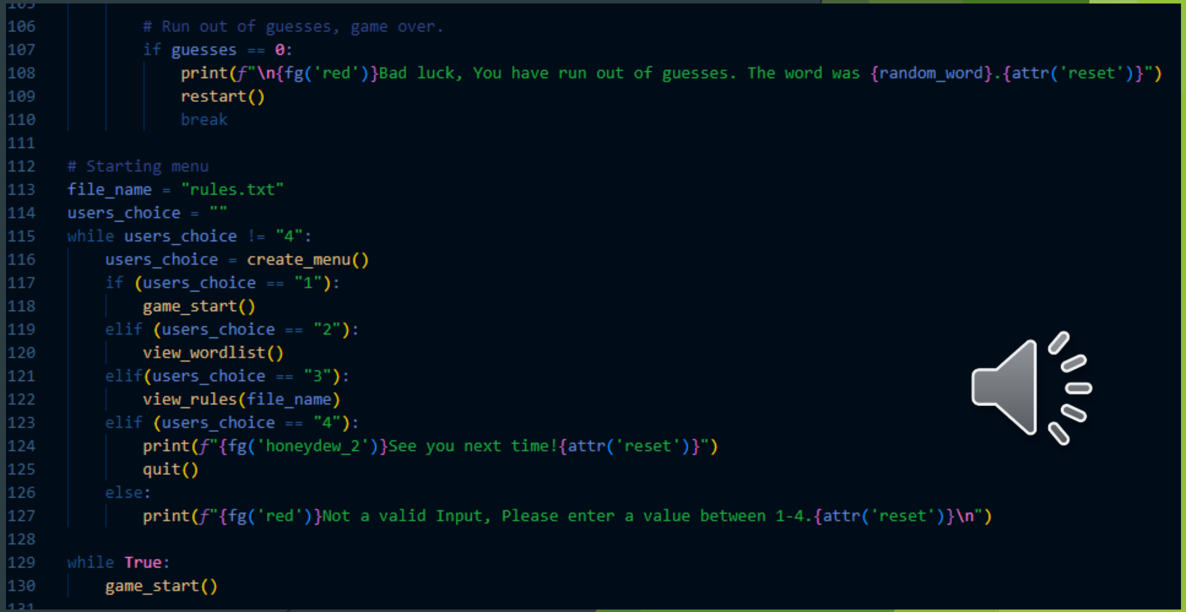
```python
# importing modules and functions from other files
import random
import string
from colored import fg, attr, bg
from words_list import animals, foods, countries, sports
from functions import create_menu, view_wordlist, view_rules, random_word_generator

# Input for user to enter name and use user input for greeting
while True:
    try:
        name = input(f"{fg('light_yellow')}Please enter your first name: {attr('reset')}").title()

        if name.isalpha():
            break
        else:
            print(f"\n{fg('red')}Please enter a valid name. No numbers, spaces or special characters.{attr('reset')}\n")

    except KeyboardInterrupt:
        print("\nInput has been interrupted, Exiting ")
        break

print(f"\n{fg('light_yellow')}Welcome {name} to the word guessing game!!!{attr('reset')}")
```



```python
def game_start():
    random_word, word_length = random_word_generator()

    print(f"\n{fg('green')}Game has started, begin Guessing!! Your word is {word_length} letters long.{attr('reset')}\n")
    users_guess = ""
    guesses = 7
    set_of_guesses = set()

    # print correctly guessed characters
    while guesses > 0:
        incorrect_guesses = 0
        for str in random_word:
            if str in users_guess:
                print(f"{fg('green')}{str}{attr('reset')}", end = "")
            else:
                incorrect_guesses += 1
                print(f"{fg('sandy_brown')}_{attr('reset')}", end = "")

        # restart function
        def restart():
                while True:
                    start_again = input(f"\n{fg('honeydew_2')}Would you like to play again? Y/N: {attr('reset')}").upper()
                    if start_again == "Y":
                        game_start()
                    elif start_again == "N":
                        print(f"\n{fg('honeydew_2')}Thanks for playing {name}! Come back soon!{attr('reset')}")
                        quit()
                    else:
                        print(f"\n{fg('red')}Invalid Input. Back to main menu.{attr('reset')}")
                        return

        # Print if word was guessed
        if incorrect_guesses == 0:
            print(f"\n\n{fg('green')}Congratulations! You got it right with {guesses} guesses remaining. The word was {random_word}.{attr('reset')}\n")

            # play again option if word was guessed
            restart()
            break
```



```python
        # User input for guess
        guess = input(f"\n\n{fg('light_yellow')}Enter a letter: {attr('reset')}").lower()
        # users can quit game at any time by typing exit
        if guess == "exit":
            print(f"{fg('honeydew_2')}Thank you for playing! See you again next time.{attr('reset')}")
            quit()
        # Make sure guess is 1 only character long and in the english alphabet.
        if len(guess) != 1 or not guess.isalpha():
            print(f"\n{fg('red')}Invalid Input. Please enter a single letter.{attr('reset')}")
            continue

        users_guess += guess

        # Guess history function
        def guess_history(guess, set_of_guesses, random_word, guesses):
        # if incorrect guess and not in guess history, lower total guess count, print remaining guesses.
            if guess in set_of_guesses:
                print(f"\n{fg('red')}Invalid Input, You have already guessed this letter.{attr('reset')}")
                print(f"{fg('pink_1')}Guess history: {set_of_guesses}{attr('reset')}")
            else:
                set_of_guesses.add(guess)
                if guess in random_word:
                    print(f"\n{fg('pink_1')}Guess history: {set_of_guesses}{attr('reset')}")
                else:
                    guesses -= 1
                    print(f"\n{fg('light_red')}Incorrect guess, You have {guesses} wrong guesses left.{attr('reset')}")
                    print(f"{fg('pink_1')}Guesses history: {set_of_guesses}{attr('reset')}")
            return guesses
        # update and run function
        guesses = guess_history(guess, set_of_guesses, random_word, guesses)

        # At certain amount of guesses print hint
        if guesses == 4:
                if random_word in animals:
                    print(f"\n{fg('green')}Hint: The word is an animal.{attr('reset')}\n")
                elif random_word in foods:
                    print(f"\n{fg('green')}Hint: The word is a food item.{attr('reset')}\n")
                elif random_word in countries:
                    print(f"\n{fg('green')}Hint: The word is a country.{attr('reset')}\n")
                else:
                    print(f"\n{fg('green')}Hint: The word is a sport.{attr('reset')}\n")
```



```python
        # Run out of guesses, game over.
        if guesses == 0:
            print(f"\n{fg('red')}Bad luck, You have run out of guesses. The word was {random_word}.{attr('reset')}")
            restart()
            break

# Starting menu
file_name = "rules.txt"
users_choice = ""
while users_choice != "4":
    users_choice = create_menu()
    if (users_choice == "1"):
        game_start()
    elif (users_choice == "2"):
        view_wordlist()
    elif(users_choice == "3"):
        view_rules(file_name)
    elif (users_choice == "4"):
        print(f"{fg('honeydew_2')}See you next time!{attr('reset')}")
        quit()
    else:
        print(f"{fg('red')}Not a valid Input, Please enter a value between 1-4.{attr('reset')}\n")

while True:
    game_start()
```

Review of development and build process

- Through the development and building process of my terminal application, I think I found the global and local variable aspect of things quite challenging to get a grasp on, but I think after building my app I think I have gained a deeper understanding of them. Circular imports was also an error I was constantly running into, but I did manage to overcome this challenge. There were no ethical issues I can think of that I faced whilst building the app. My favourite part of building the application was the feeling of being able to overcome the challenges that were thrown at me and the feeling of success and pride knowing I built a working game.