Problem Description:

The problem is to create a game of Hangman. The game should have a login system for both users and administrators, and should also keep track of high scores.

Distinguishing features of the project:

- Fully dedicated GUI for the game.
- Game has a Difficulty level ranging from Easy to Hard.
- The game has a login system for both users and administrators, allowing for different levels of access to the game.
- A feature of Suggested Guesses have been added where user will be prompted suggestion letters based on the likelihood of a letter being in the secret word.
- The game keeps track of high scores, and allows administrators to reset the high scores.
- The game has a user registration system, allowing new users to create an account and play the game.

Flow of Project

- The game starts with a login window for both users and administrators.
- For the user:
    - Before the login process user will be displayed choices on Difficulty. Ranging from Easy, Medium, Hard.
    - Once logged in, the user can play the game.
    - The user can guess a single alphabet at a time.
    - If the alphabet is in the secret word, it will be displayed on the screen.
    - If the alphabet is not in the secret word, the user will lose a guess.
    - The game will end when the user has guessed all the letters in the secret word or has run out of guesses.
    - If the user successfully guesses all the letters in the secret word before running out of guesses, they will be declared the winner and the game will end.
    - However, if user is unsuccessful in guessing the word before the guesses run out, they will lose.
    - After the game for both cases of win or lose It will ask the user if he/she wants to play again? If not then it will display a goodbye message screen.
    - If the user wants to play again the game will return to the game window. Where a new letter will be generated.
- For the admin:
    - After Authentication of admin with relevant password and username.
    - A window specific to the admin will open up and there will reside methods to Add words and reset the high scores.
    - The game also keeps track of high scores.

REPORT

Most challenging parts while working on the project:

There were two parts that would be classified as Challenging:

1. The Most challenging part was actually implementing the Tkinter module as it was very lengthy and I had to be very careful in defining certain objects.
2. Developing some good algorithms for data processing was also very challenging.

Things learnt in Python while working on the project:

I learnt a lot during the process of making this project:

1. During the project, I learned how to use the Tkinter library to create GUI in Python.
2. We also learned how to work with file I/O to keep track of high scores and user login information.
3. Many new ways to shorten lines of code were also learned like Lambda, if expression, Dictionary comprehension.
4. Use of GitHub was also employed to share versions of code.

Individual contributions of each group member:

1. **Muneeb Ahmed (CS-22048):**
    a. Group Leader
    b. Lead Programmer.
    c. Asset Design.
    d. Report.

2. **Ahzam Rehan (CS-22046):**
    a. Research
    b. GUI concept.
    c. Game Tester.

3. **Abdullah Athar (CS-22041):**
    a. Game Tester.
    b. GUI concept.
    c. Ideas for improvement.

REPORT

Future expansions:

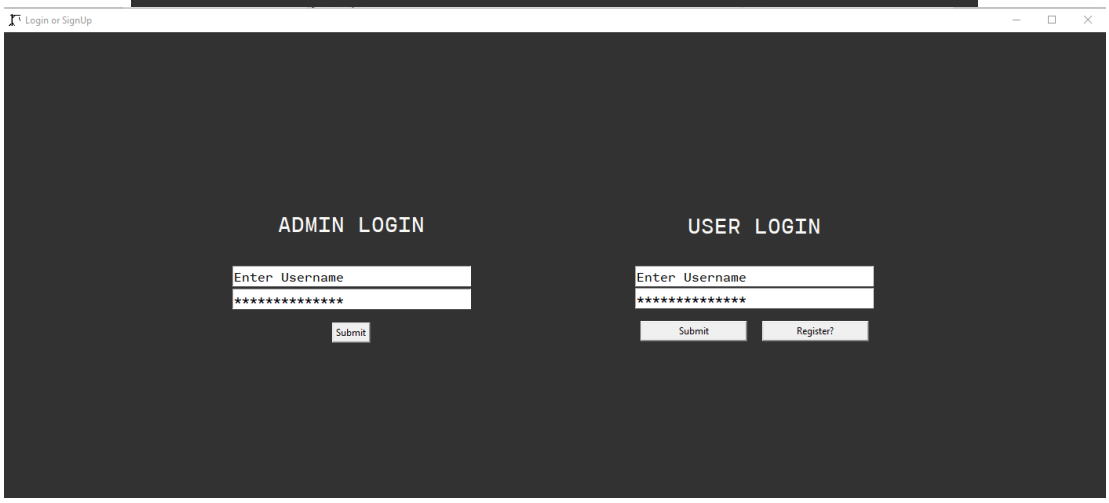
- Expanding the game to the web for better reach.
- Adding a feature for users to play with friends by sharing a game code.
- Adding a feature for users to change their password.
- [i]Adding a feature that would allow admins to manipulate the words (deleting or replacing words).
- Adding a feature for users to play the game with different rule sets, such as allowing or disallowing multiple guesses for a single letter.
- Adding a feature for users to play the game with different time limits, such as 60 seconds, 90 seconds, and 120 seconds.
- Adding a feature for users to play the game with different point systems, such as points for each correct letter and penalty points for each incorrect letter.
- Adding a feature for users to play the game with different scoring systems, such as time-based scoring, word-based scoring, and letter-based scoring.
- Adding a feature for users to play the game with different hints, such as giving the first letter of the secret word or giving a definition of the secret word.
- Adding a feature for users to play the game with different power-ups, such as extra guesses, extra time, and extra points.

List of references:

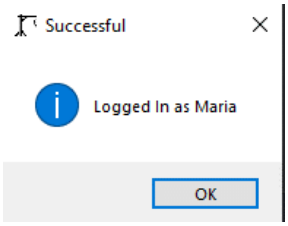
- Tkinter documentation:
    - *https://docs.python.org/3/library/tk.html*
- High Scores and File I/O in Python:
    - *https://realpython.com/working-with-files-in-python/*
- Tkinter Tutorial:
    - *https://www.youtube.com/watch?v=TuLxsvK4svQ&t=3143s&ab_channel=BroCode*
- W3Schools Docs:
    - *https://www.w3schools.com/python/*

LOGIN PROCESS:

(COMMON FOR EVERY TEST)



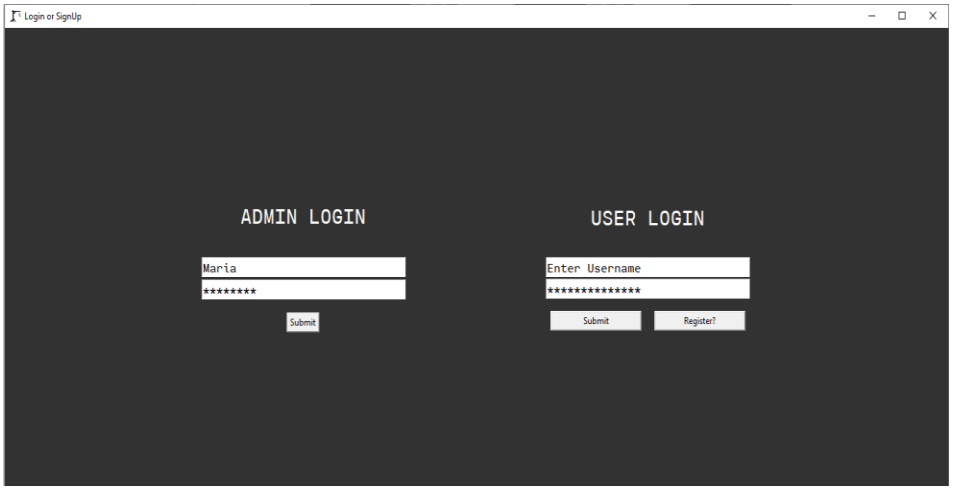
If incorrect login information



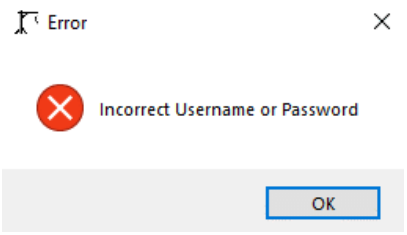
if correct login information

REPORT

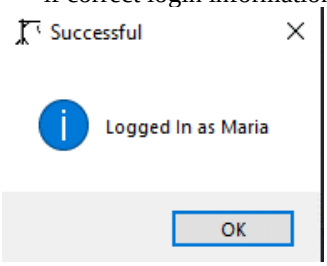
ADMIN LOGIN:



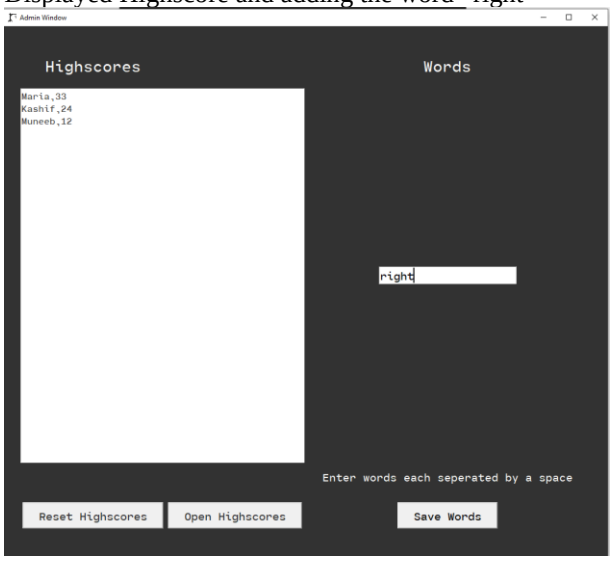
If incorrect login information

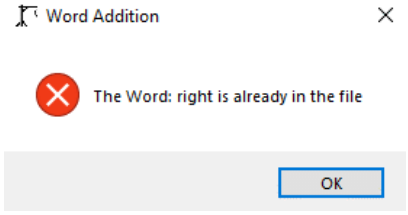


if correct login information

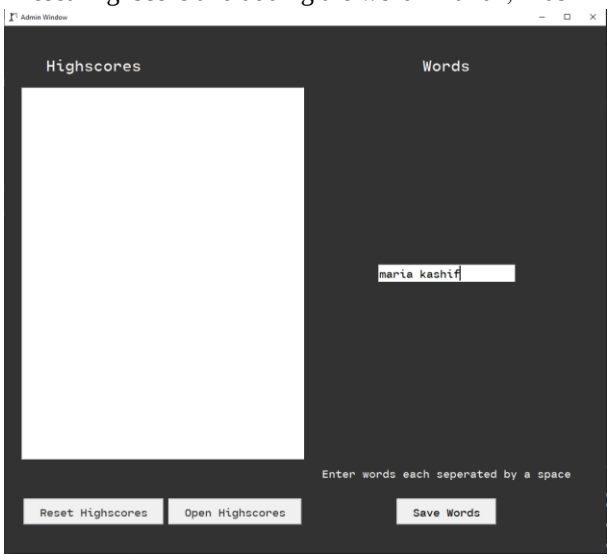


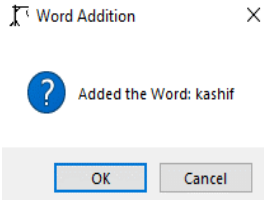
Displayed Highscore and adding the word “right”

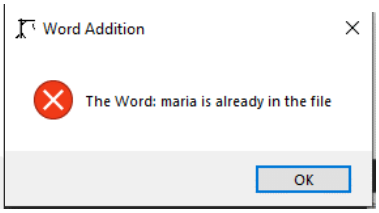




Reset Highscore and adding the word “maria”, “kashif”







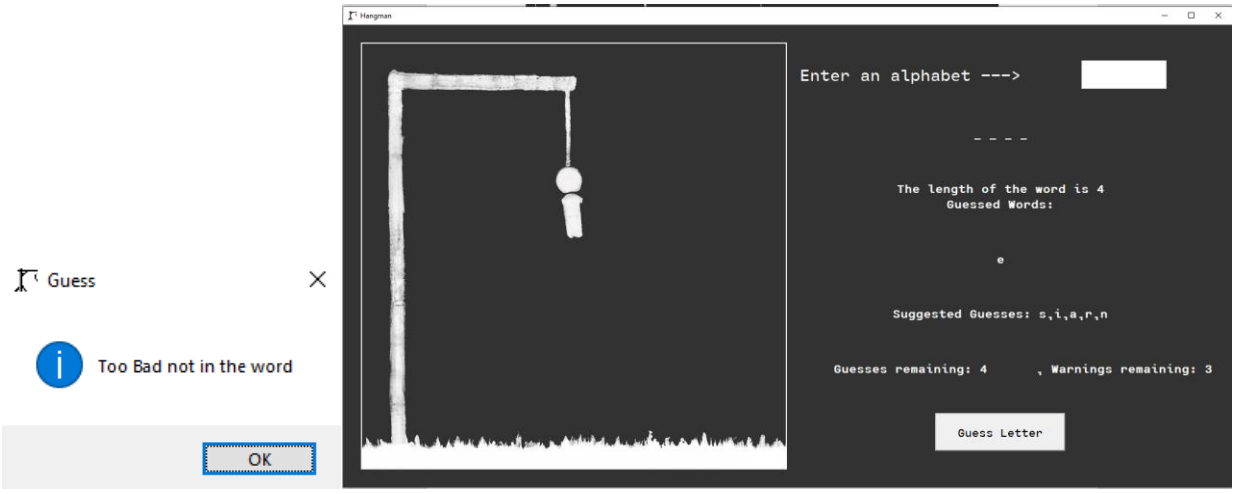
GAMEPLAY:

WIN CASE:

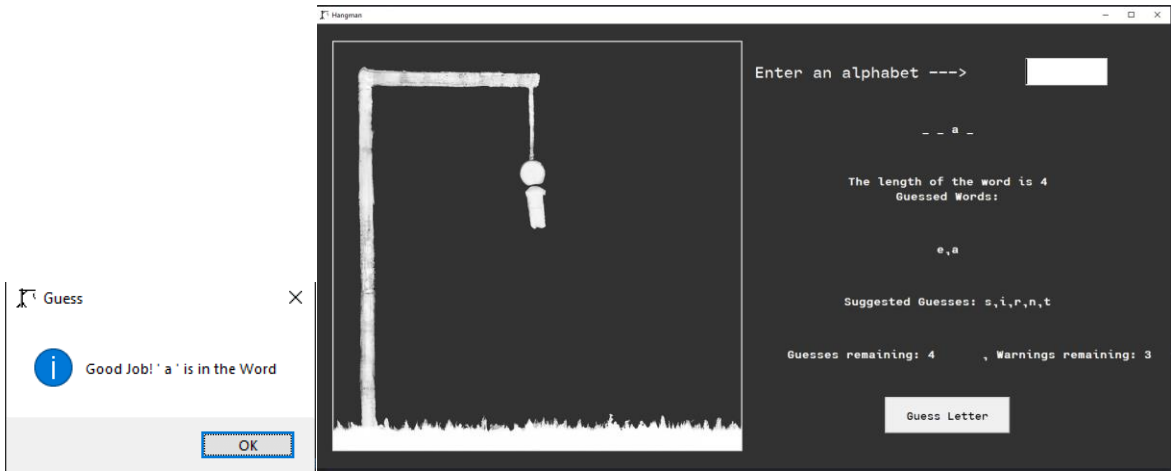
GUI OF GAME

SECRET WORD: "utah"

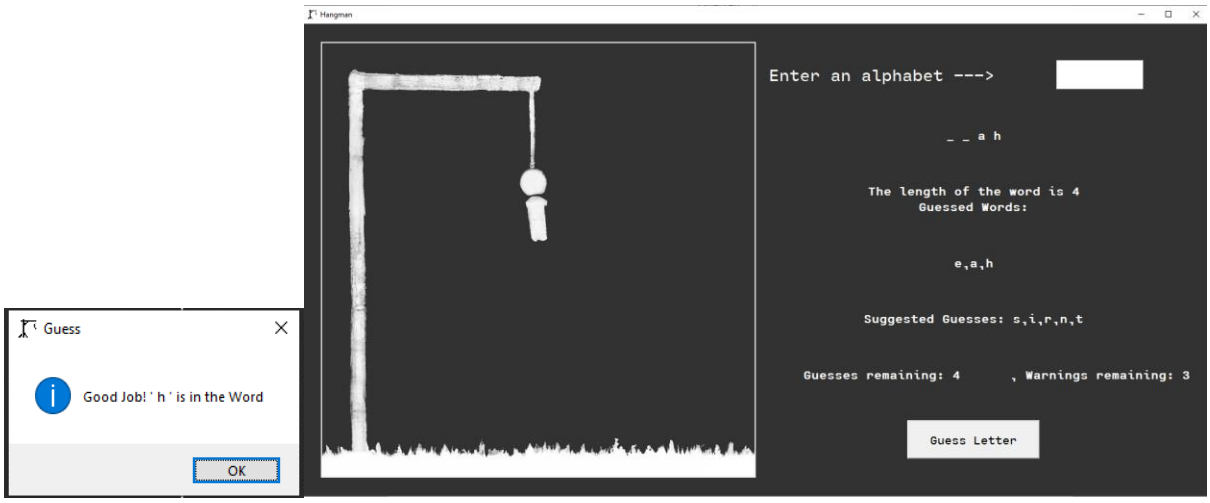
Guessed: "e":



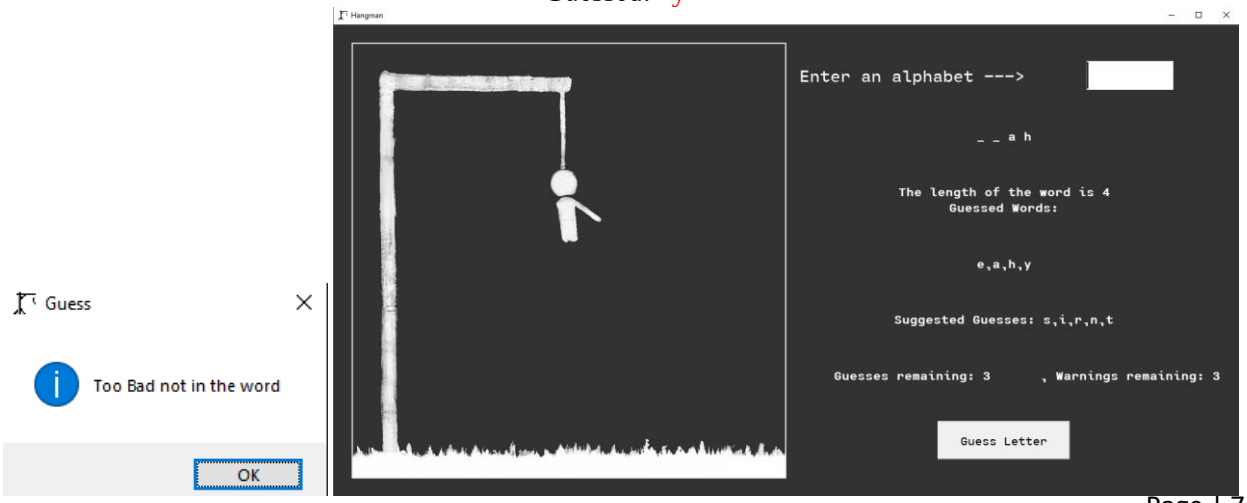
Guessed: “a”



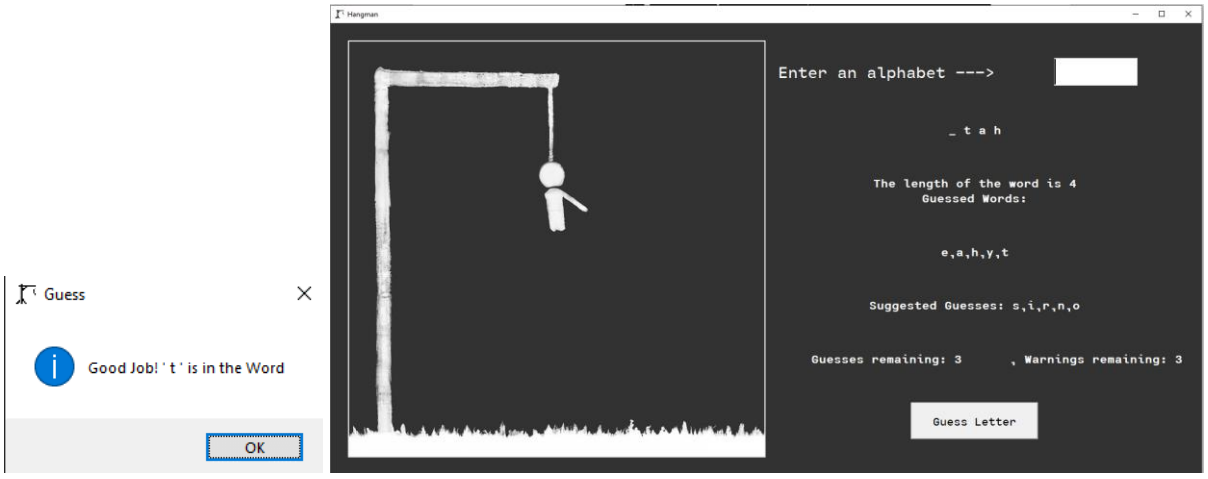
Guessed: “h”



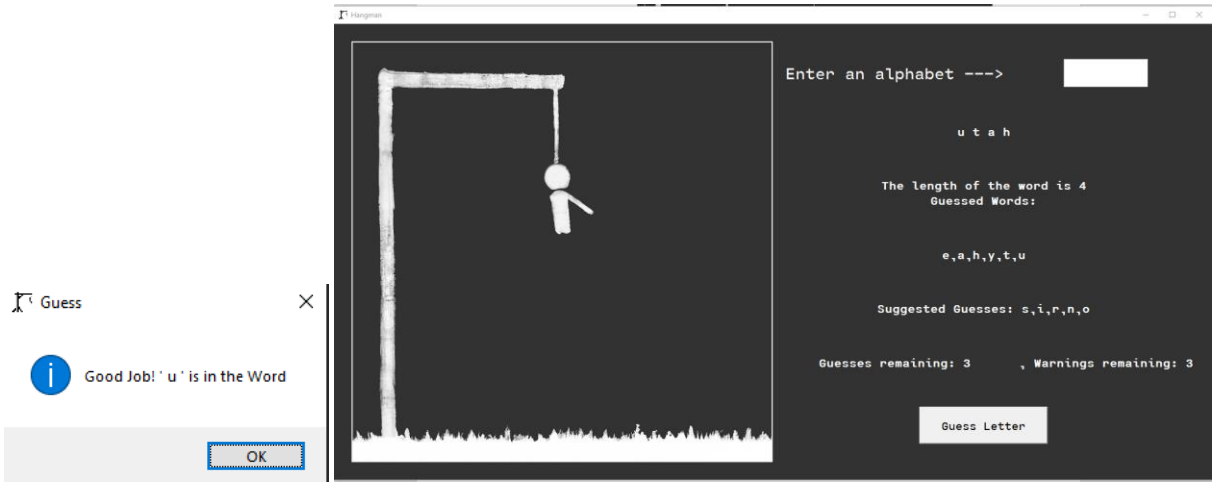
Guessed: “y”



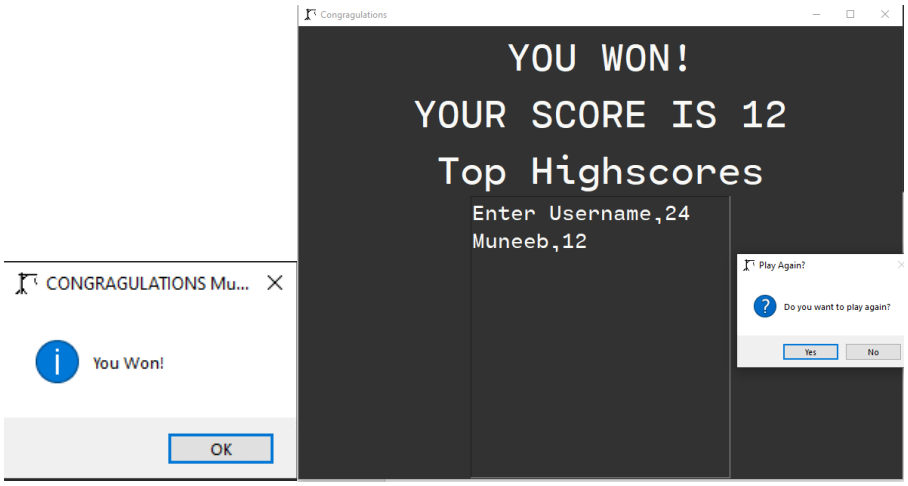
Guessed: “t”



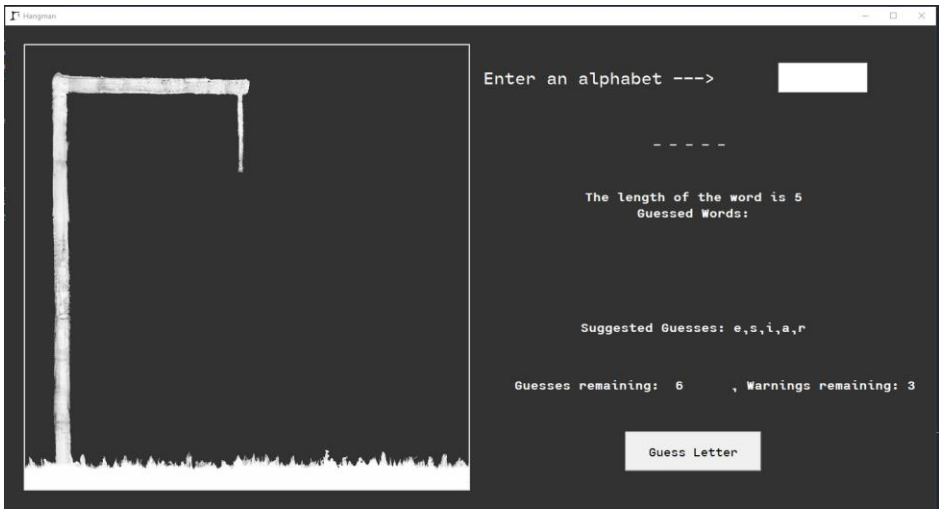
Guessed: “u”



After Guessing the word correctly

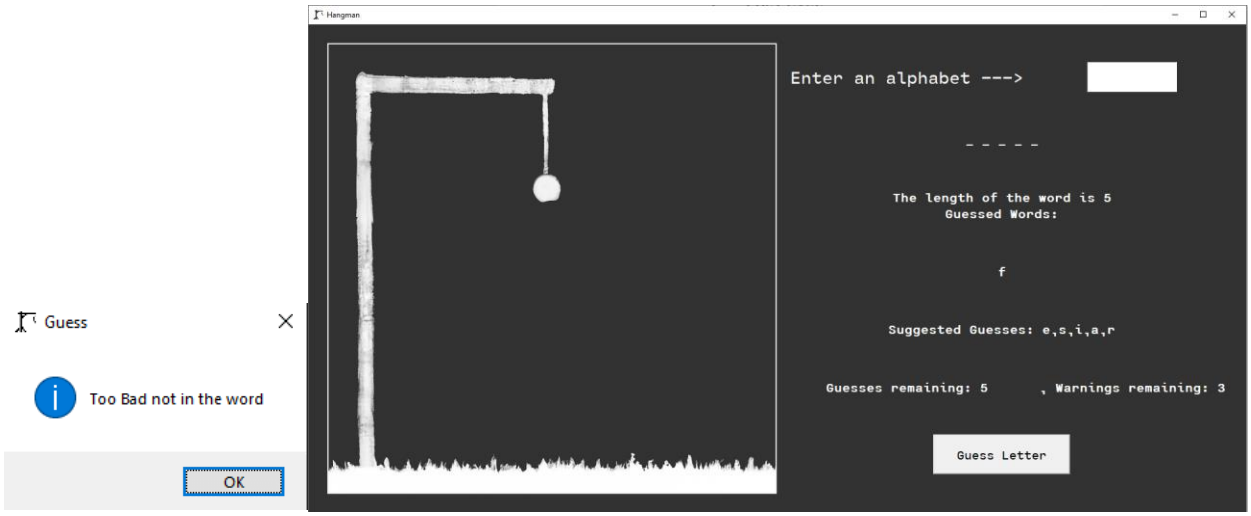


LOST CASE:
GUI OF GAME

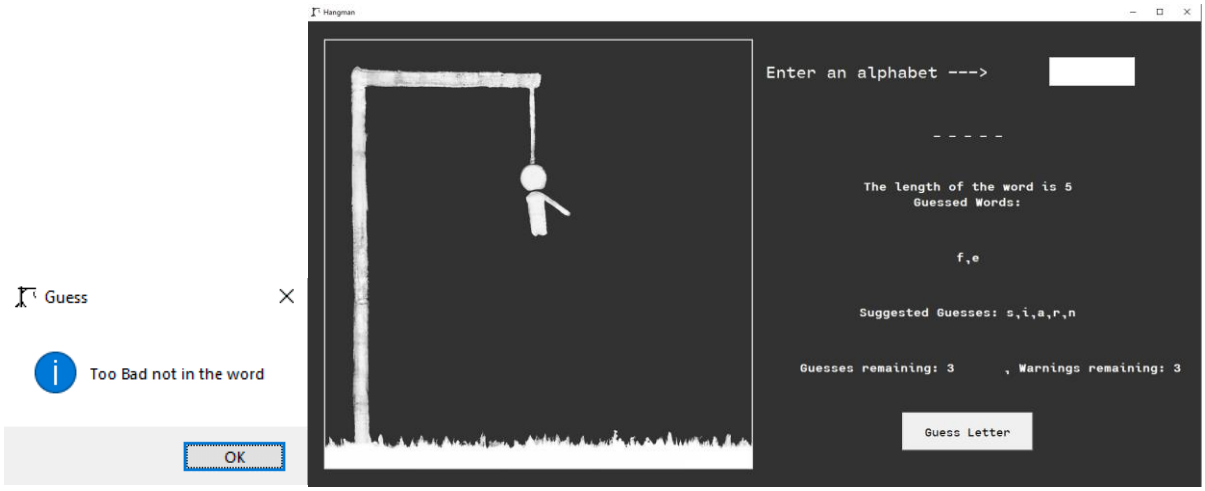


SECRET WORD: "wilds"

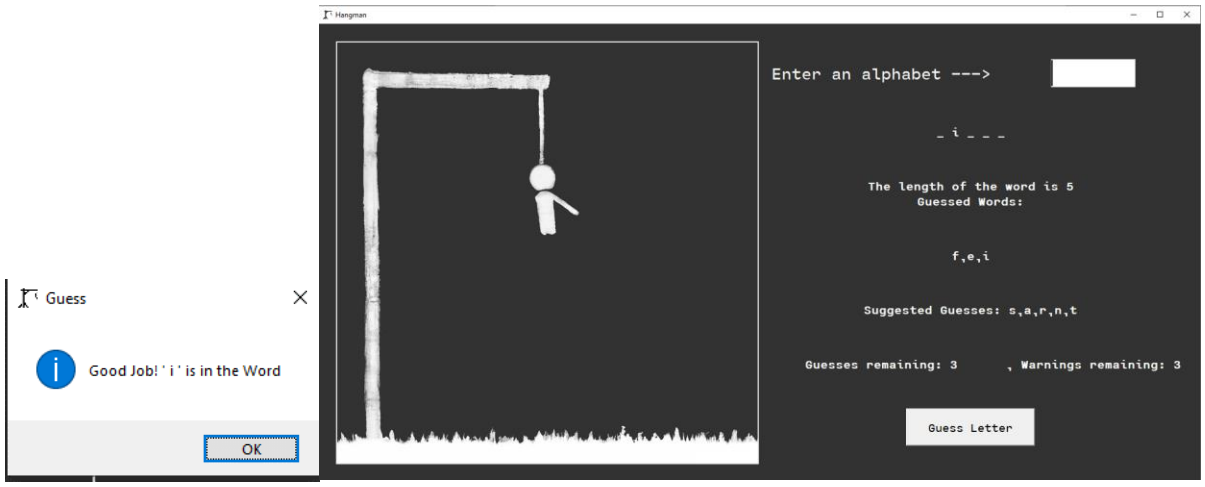
Guessed "f":



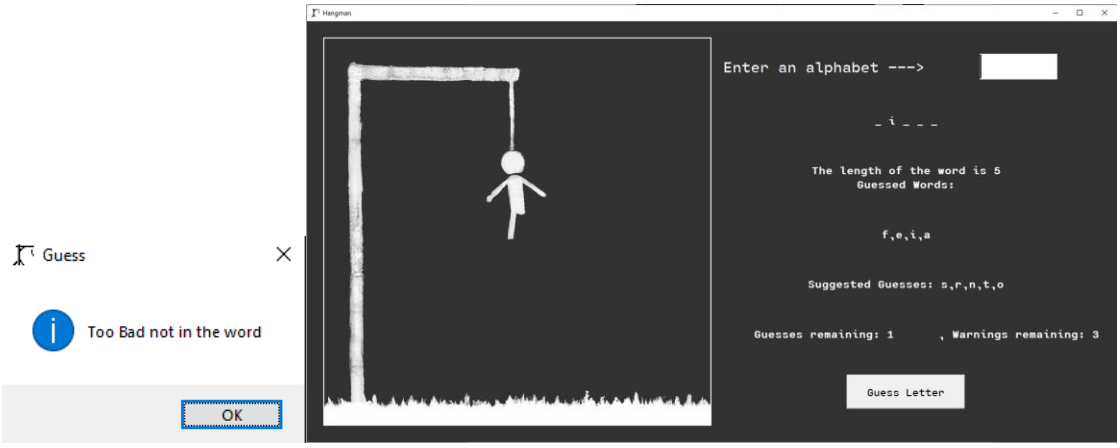
Guessed "e":



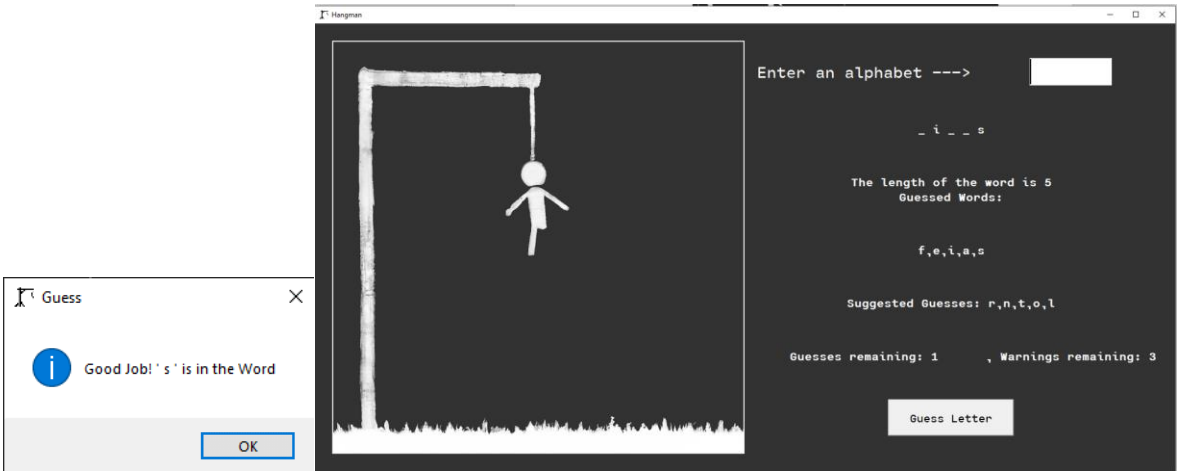
Guessed "i":



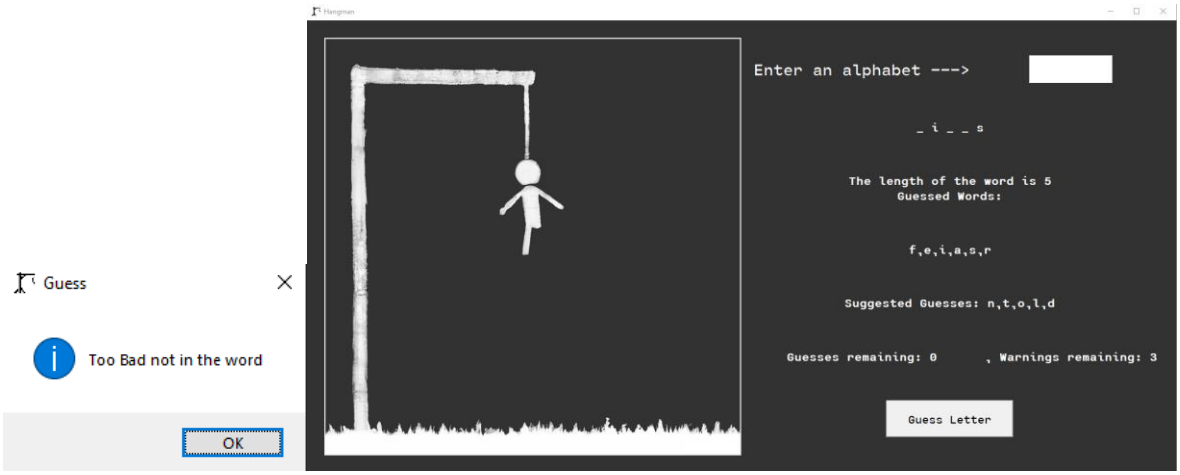
Guessed "a":



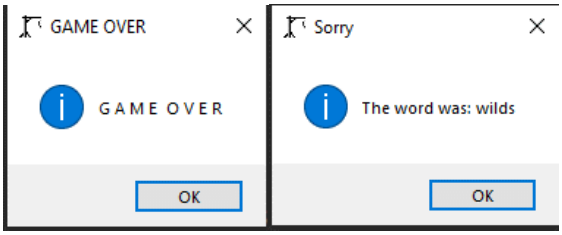
Guessed "s":

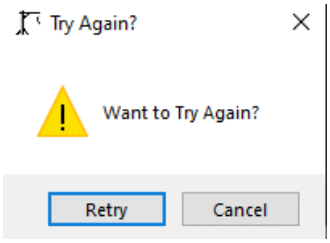


Guessed "r":



AFTER REMAINING GUESSES ARE GONE:





GOODBYE SCREEN:

(COMMON FOR WIN AND LOSS CASE)