By : Rahul Khandebharad

VIT Bhopal University

Typing Speed Game

March 17, 2021

Overview

As personal computers have appeared in every industry, typing skills have become a fundamental part of “computer literacy”. From a typical office setting person who types the memos, financial reports etc to programmers who type codes and make application softwares, typing has been the key factor to save time, increase work efficiency and to increase brian-hand coordination.

Goals

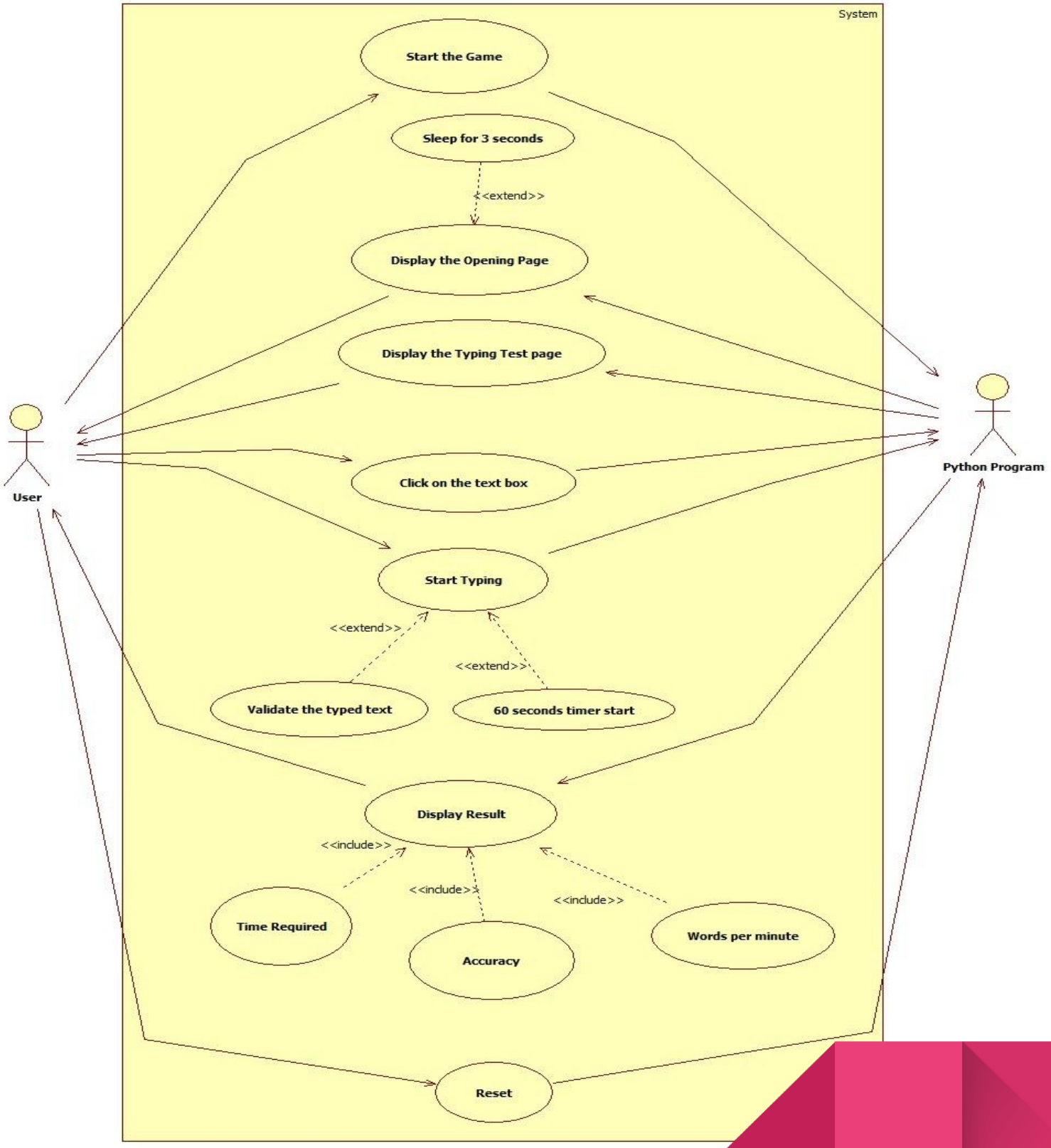
1. **Drawing Use Case Diagram (UML):** To understand the dynamic behavior of the system (i.e the behavior of the game when it is running), users interaction with the system and the external and internal factors influencing the system.
2. **Developing in Python:** Install and import the pygame library used to work with graphics, creating the Game class and creating the draw_text() method, get_sentence() method, show_results() method, run() method, reset_game() method for the development.

Specifications

The game will display the user a random sentence from a set of sentences that are already provided to the game, Game will also display a text box where the user needs to click and start typing the sentence, as user finishes typing and clicks enter the user is displayed with the result, the result contains the time required by the user to type the sentence, his accuracy and words per minute.

Milestones

1. Use Case Diagram: (Note: If diagram is not clear you may refer to it in the repository)



2. Developing the Game

1. Import the libraries:

```
import pygame
from pygame.locals import *
import sys
import time
import random
```

2. Create the game class:

```
class Game:

    def __init__(self):
        self.w = 900
        self.h = 600
        self.reset = True
        self.active = False
        self.input_text = ''
        self.word = ''
        self.time_start = 0
        self.total_time = 0
        self.accuracy = '0%'
        self.results = 'Time:0 Accuracy:0 % Wpm:0 '
        self.wpm = 0
        self.end = False
        self.HEAD_C = (192, 192, 192)
        self.TEXT_C = (255, 255, 0)
        self.RESULT_C = (255, 165, 0)
```

```

        pygame.init()
        self.open_img = pygame.image.load('opening-back.png')
        self.open_img = pygame.transform.scale(self.open_img, (self.w, self.h))

        self.bg = pygame.image.load('background.jpg')
        self.bg = pygame.transform.scale(self.bg, (600, 900))

        self.screen = pygame.display.set_mode((self.w, self.h))
        pygame.display.set_caption('Typing Speed Test')

```

3. draw_text() method:

```
        def draw_text(self, screen, msg, y, fsize, color):
            font = pygame.font.SysFont("Times New Roman", fsize)
            text = font.render(msg, 1, color)
            text_rect = text.get_rect(center=(self.w/2, y))
            screen.blit(text, text_rect)
            pygame.display.update()
```

4. get_sentence() method:

```
        def get_sentence(self):
            f = open('sentences.txt').read()
            sentences = f.split('\n')
            sentence = random.choice(sentences)
            return sentence
```

5. show_results() method:

```
        def show_results(self, screen):
            if(not self.end):
                # Calculate time
                self.total_time = time.time() - self.time_start

                # Calculate accuracy
                count = 0
                for i, c in enumerate(self.word):
                    try:
                        if self.input_text[i] == c:
                            count += 1
                    except:
                        pass
                self.accuracy = count/len(self.word)*100
```

```

    # Calculate words per minute
    self.wpm = len(self.input_text)*60/(5*self.total_time)
    self.end = True
    print(self.total_time)

    self.results = 'Time: '+str(round(self.total_time)) + " secs   Accuracy: " \
                + str(round(self.accuracy)) + "%" + '   Wpm: ' + str(round(self.wpm))
```

```
73      # draw icon image
74      self.time_img = pygame.image.load('icon.png')
75      self.time_img = pygame.transform.scale(self.time_img, (150, 150))
76
77      screen.blit(self.time_img, (self.w/2-75, self.h-140))
78      self.draw_text(screen, "Reset", self.h - 70, 26, (100, 100, 100))
79
80      print(self.results)
81      pygame.display.update()
```

6. run() method:

```
83          def run(self):
84              self.reset_game()
85
86              self.running = True
87              while(self.running):
88                  clock = pygame.time.Clock()
89                  self.screen.fill((0, 0, 0), (50, 250, 800, 50))
90                  pygame.draw.rect(self.screen, self.HEAD_C, (50, 250, 800, 50), 2)
```

```
92              # update the text of user input
93              self.draw_text(self.screen, self.input_text, 274, 26, (250, 250, 250))
94              pygame.display.update()
95              for event in pygame.event.get():
96                  if event.type == QUIT:
97                      self.running = False
98                      sys.exit()
99                  elif event.type == pygame.MOUSEBUTTONUP:
100                     x, y = pygame.mouse.get_pos()
```

```
102                 # position of input box
103                 if(x>=200 and x<=800 and y>=250 and y<=300):
104                     self.active = True
105                     self.input_text = ''
106                     self.time_start = time.time()
107
108                 # position of reset box
109                 if(x>=310 and x<=660 and y>=390 and self.end):
110                     self.reset_game()
111                     x, y = pygame.mouse.get_pos()
```

```
113         elif event.type == pygame.KEYDOWN:
114             if self.active and not self.end:
115                 if event.key == pygame.K_RETURN:
116                     print(self.input_text)
117                     self.show_results(self.screen)
118                     print(self.results)
119                     self.draw_text(self.screen, self.results, 350, 28, self.RESULT_C)
120                     self.end = True
```

```
122                             elif event.key == pygame.K_BACKSPACE:
123                                 self.input_text = self.input_text[:-1]
124                             else:
125                                 try:
126                                     self.input_text += event.unicode
127                                 except:
128                                     pass
129 
130                 pygame.display.update()
131 
132             clock.tick(60)
```

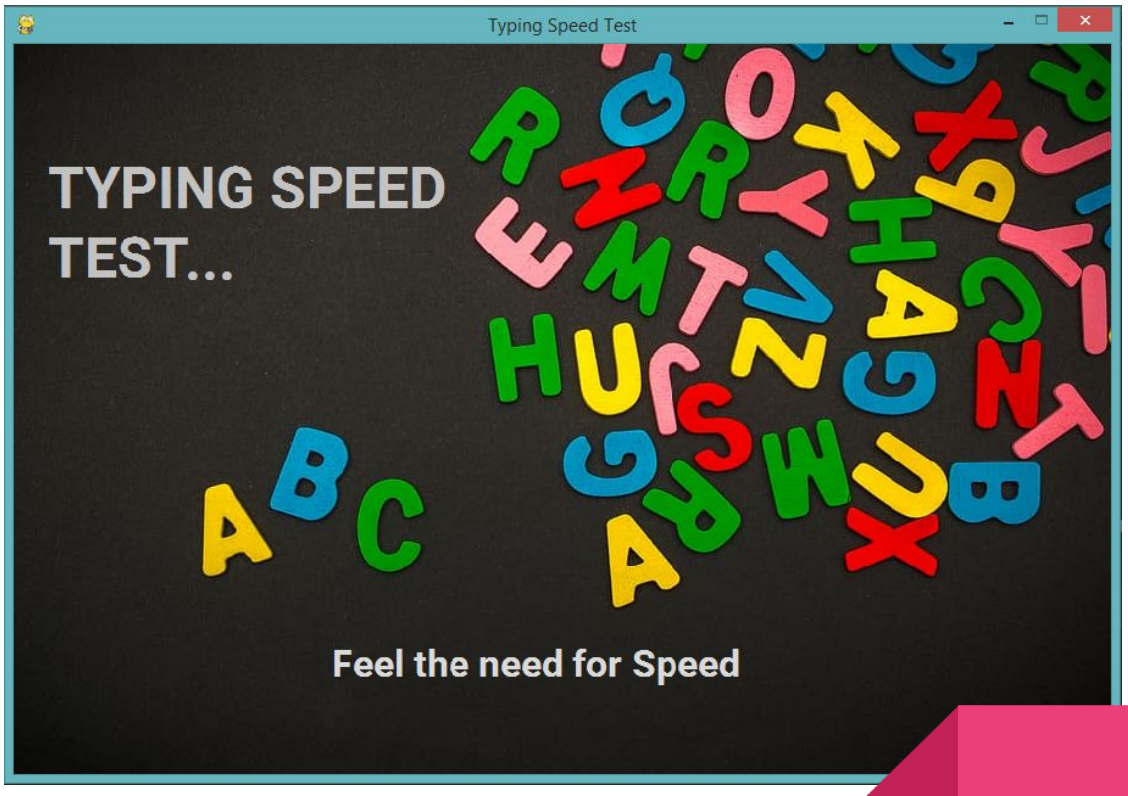
7. reset_game() method:

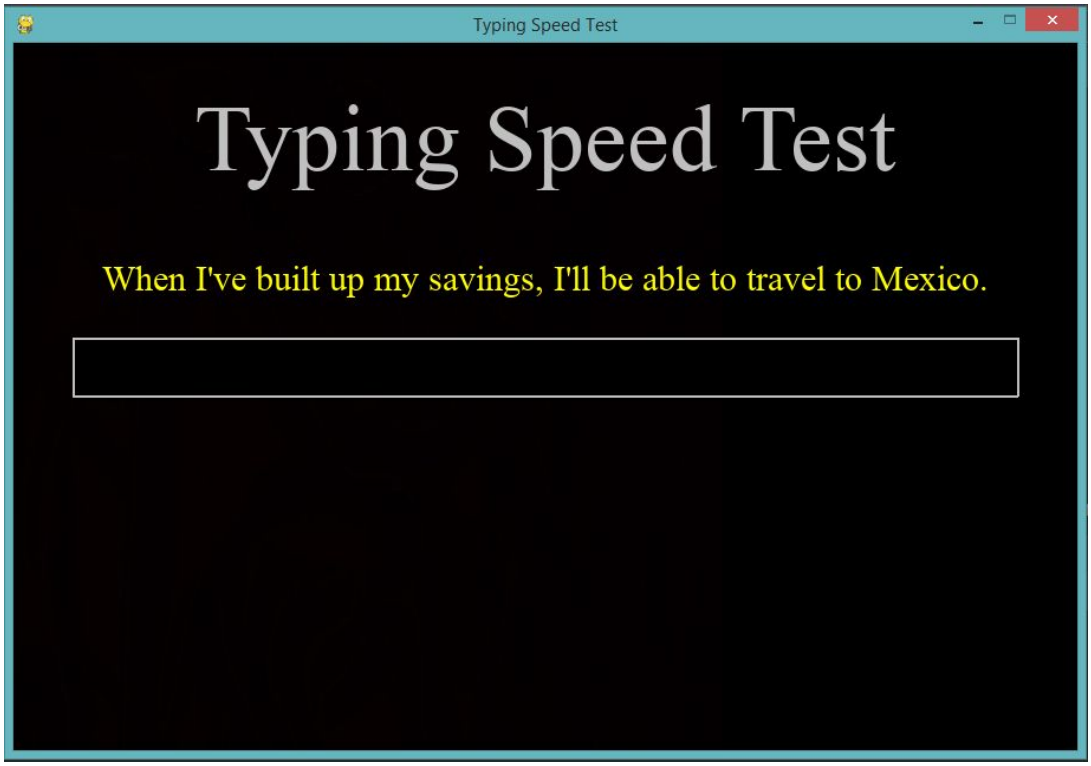
```
134         def reset_game(self):
135             self.screen.blit(self.open_img, (0, 0))
136 
137             pygame.display.update()
138             time.sleep(3)
139 
140             self.reset = False
141             self.end = False
142 
143             self.input_text = ''
144             self.word = ''
145             self.time_start = 0
146             self.total_time = 0
147             self.wpm = 0
148 
```

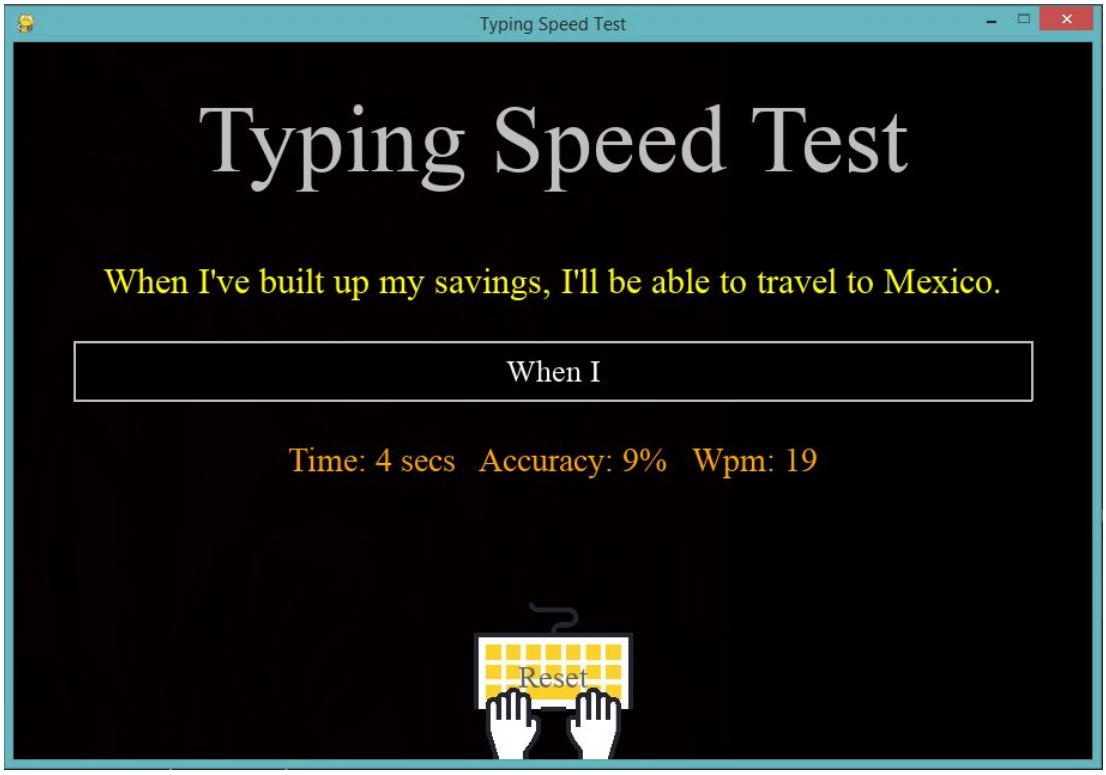
```
149             # Get random sentence
150             self.word = self.get_sentence()
151             if (not self.word): self.reset_game()
152 
153             # drawing heading
154             self.screen.fill((0, 0, 0))
155             self.screen.blit(self.bg, (0, 0))
156             msg = "Typing Speed Test"
157             self.draw_text(self.screen, msg, 80, 80, self.HEAD_C)
158 
```

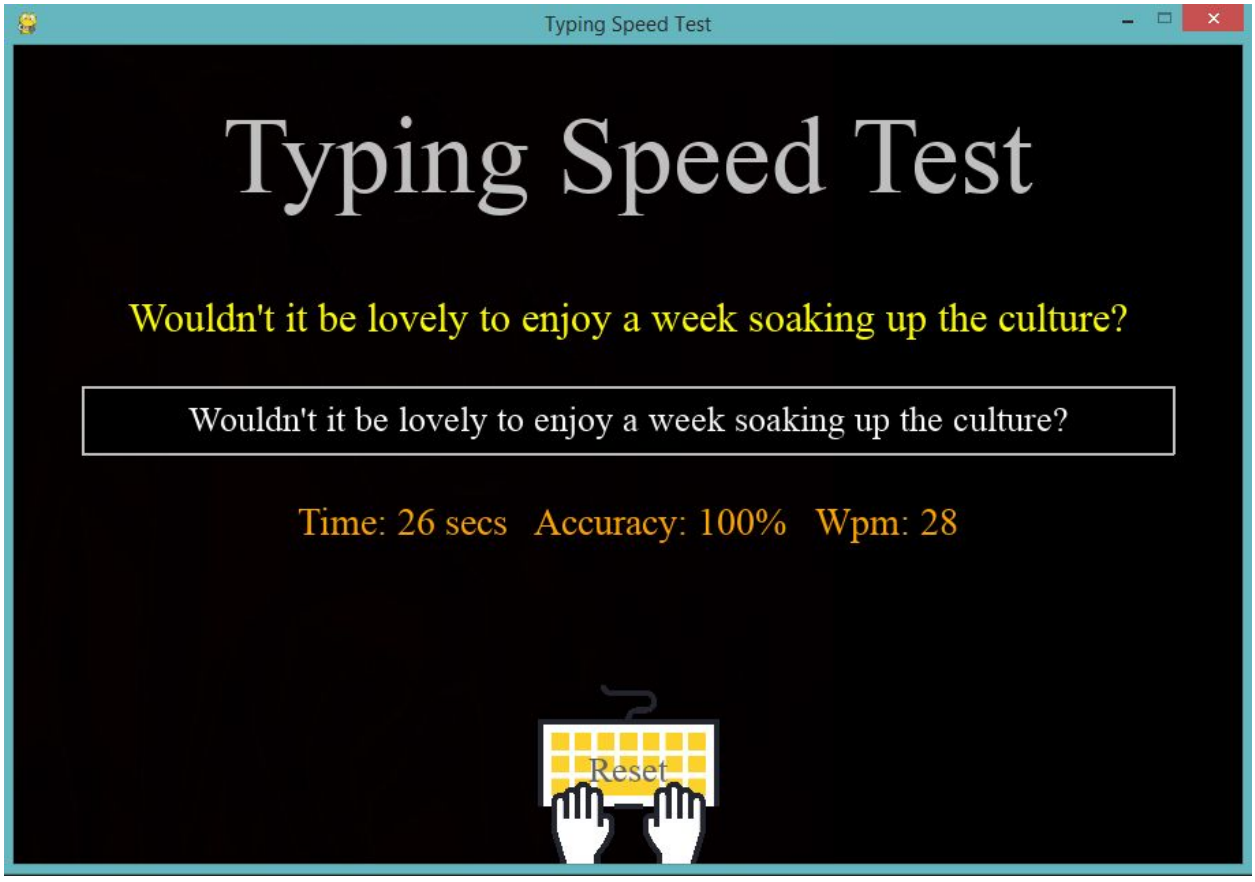
```
159             # draw the rectangle for input box
160             pygame.draw.rect(self.screen, (255, 192, 25), (50, 250, 650, 50), 2)
161 
162             # draw the sentence string
163             self.draw_text(self.screen, self.word, 200, 30, self.TEXT_C)
164 
165             pygame.display.update()
166 
167 
168 
169     Game().run()
```

3. Outcome ScreenShots:




Typing Speed Test
Typing Speed Test
When I've built up my savings, I'll be able to travel to Mexico.


Typing Speed Test
Typing Speed Test
When I've built up my savings, I'll be able to travel to Mexico.
When I
Time: 4 secs Accuracy: 9% Wpm: 19
Reset



Note : ER Diagram is not drawn because database connectivity is not required in this basic version of the game. Further Login and Signup can be added to the game to allow the user to track their typing speed on a daily basis and can compare it to that of all typers playing that game, This can be implemented by connecting it to Relational Database.