Proyecto “Adivina palabras (Juego)”

11 de diciembre 2020

Integrantes

Bracco Lucas Oscar

Bracco Heredia Ignacio Agustin

OBJETIVOS

Adivina Palabras

Como dueño de una empresa que desarrolla juegos necesitaría un programa que permite adivinar palabras en un tiempo determinado

Como criterios de aceptación para el proyecto, se deberá tener en cuenta lo siguiente :
- Permitir ingresar el nombre del jugador y sino seleccionar alguno que ya se ingresó.
- Se deberá mostrar una palabra desordenada y el jugador deberá adivinarla en menos de 20 segundos.
- Por palabra acertada cada jugador suma 10 puntos.
- Mostrar una tabla con las posiciones de los jugadores.
- El jugador en cada partida tiene 3 vidas.
- El Diccionario de palabras es a elección.
- Se deben seguir mostrando palabras al azar hasta que el jugador pierda las 3 vidas.

Tutorial

Como primer paso para poder jugar a este juego se necesita descargar los archivos del repositorio:

https://gitlab.com/braccoig1501/proyectofinalap

descomprimimos los archivos en el lugar deseado, ingresamos a la carpeta database borrando el archivo en caso de que ya esté creado. y ejecutamos el archivo “game.py” (Sin comillas) ubicándonos con la consola de comandos en el directorio donde se encuentra el archivo, con el siguiente comando:

directorio “python game.py”

en mi caso por ejemplo sería: “/home/braccoignacio/Documentos/proyectoProg python game.py”

Al ejecutar el comando, esto automáticamente se crearán las tablas con la base de datos correspondiente, luego se abrirá una ventana de Login, bastante intuitiva donde si no tenemos un usuario nos dejará generarlo, los nombres de usuarios son únicos y las claves están encriptadas con la libreria bcrypt.

Una vez creado el usuario procedemos a ingresar al juego.

donde tenemos la posibilidad de ver los Puntajes almacenados y comenzar el juego,

tenemos 20 segundos para adivinar la palabra desordenada que se muestra en pantalla, colocamos la palabra correcta o que se piensa que es correcta dentro del casillero (input) luego hacemos click en intentar. esto resetea el contador a 20 segundos nuevamente si la palabra es correcta caso contrario seguirá descontando tiempo mostrando el mensaje de que la palabra es incorrecta.

Al acertar palabras sumaremos el puntaje correspondiente, el juego acaba al quedarnos sin vidas las cuales se consumen al pasar los 20 segundos de no adivinar la palabra.

¡A JUGAR!

SOBRE EL DESARROLLO

El lenguaje utilizado para este proyecto es python, se generaron las vistas con QT designer

se las nombro de la siguiente manera:

```
V_Login.ui
```

cambiando según la pantalla en donde estamos parados (Login, Register, Score, EndGame, Menú y Game(pantalla de juego)).

Los botones y labels se nombraron de la siguiente manera:

```
b_Register

lb_Name

le_User
```

primero el tipo y luego el nombre del datos que trabajamos, en este caso es un botón, luego un label y por último un LineEdit.

Además de la utilización de las librerías necesarias para el funcionamiento de PyQt5 siendo estas para el manejo de los labels, inputs, pantallas, mensajes.

```
PyQt5.QtWidgets QApplication, QMainWindow, QLineEdit, QLabel, QMessageBox

PyQt5 uic QtCore QtGui
```

utilizamos las siguientes librerías:

```
bcrypt # para la encriptación de los datos sensibles como contraseñas

time # para el manejo de los tiempo dentro del código, y poder utilizar

     # el temporizador en la función del juego.

sqlite3 # para tener una mejor consistencia en los datos y poder

        # almacenarlos y leerlos cuando sea necesario

random # utilizamos la librería random para seleccionar las palabras, y

       # para desordenarlas
```

Tambien importamos:

```
conexion # archivo externo donde manejamos algunos datos y funciones en la
         # BD
Lista_Palabras # Archivo externo donde manejamos las funciones para la
               # Selección de palabras y donde tenemos almacenadas las
               # palabras que se utilizaran en el juego
```

Funciones más importantes utilizadas:

Función para generar la BD y Tablas a utilizar:

Por medio de una función, generamos la BD y Las tablas a utilizar en el caso de que NO existan ya previamente.

Función de login:

Se realiza la comprobación de las variables ingresadas en los inputs (LineEdit), donde controlamos primero que el usuario exista, para luego comprobar si la contraseña coincide con la ya almacenada en ese nombre de usuario.

Función de registro de usuario:

Se pide un nombre de usuario y dos contraseñas para validar la información y evitar errores, primero comprobamos si el nombre de usuario ya se encuentra en uso, luego de no existir encriptamos la variable con la contraseña con la librería bcrypt. si las dos contraseñas coinciden, se procede almacenando los datos en la BD.

Función selección de palabras:

Para generar las palabras lo que hacemos es seleccionar una palabra aleatoria de la lista ya establecida. dividiéndola en 3 niveles de dificultad, la seleccionamos con la función de la librería random “choice()” y la pasamos por la función para desordenar esa palabra, manteniendo la palabra seleccionada para luego ser comparada con la palabra que ingrese el jugador.

Función para desordenar palabra:

Para desordenar la palabra seleccionada utilizamos la librería random la cual tiene una función llamada “sample()” la cual nos entrega una lista con las letras desordenadas aleatoriamente y luego utilizamos la función “join()” para unir esas letras y poder mostrarlas

```
#While que verifica que si la p_desordenada es igual a p_elegida vuelve a
desordenar.

while self.p_desordenada == self.p_elegida:

            aux = random.sample(self.p_elegida,len(self.p_elegida))

            self.p_desordenada = ''.join(aux)
```

Función para mostrar scores (puntajes almacenados):

Para hacer eso realizamos una consulta a la BD donde ya nos entrega un array con los datos requeridos para poder mostrarlos en pantalla, la consulta utilizada es:

```
SELECT * FROM score ORDER BY puntaje DESC LIMIT 3
```

¿QUE CONSIDERAMOS QUE NOS FALTA AGREGAR O MODIFICAR? (OPCIONAL)

Estas son funciones que pensamos que nos falto agregar o modificar al programa:

- Comparar si la palabra seleccionada, no fue la anterior, esto ocasiona, que se seleccione la misma palabra con distinto orden hasta 3 o 4 veces seguidas.
- Utilizamos palabras en inglés, pero en varias partes escribimos en español.
- Una pantalla con todos los scores Individuales de los jugadores.
- Mostrar los Scores en una tabla, mostrando más cantidad de datos.
- Mejoras visuales, imágenes, botones con diseños y otros tipos de letras.

HITOS

- 11/11 - Prototipado de las Pantallas (Subir archivos .ui) <Prototipos_Pantalla>
- 18/11 - Por lo menos 1 Criterio de Aceptación Funcionando
- 25/11 - Por lo menos 2 Criterios de Aceptación Funcionando
- 02/12 - Por lo menos 3 Criterios de Aceptación Funcionando
- 09/12 - Todo el Software Debería estar funcionando

11/11

Se entrego lo pedido con una demora de un dia.

18/11

Ya se cumplian 2 criterios (Usuarios, y generacion de palabras)

25/11

Todos los criterios cumplidos, solo restaba mostrar en pantalla las scores (puntaje)

02/12

Se completaron todos los criterios de aceptación, se siguió mejorando código y funciones.

09/12

Se completaron todos los criterios de aceptación, se siguió mejorando código y funciones, solucionando “bugs”.