# I.E.S LAS SALINAS


## PROYECTO FINAL FIN DE GRADO

## CURSO 22-23

FormWithMe

CICLO FORMATIVO GRADO SUPERIOR

DESARROLLO DE APLICACIONES MULTIPLATAFORMA

**Autor: Alex Asavei**

**Tutor: Oscar Lillo Díaz**

# 1.- Resumen:

El proyecto consistirá en el desarrollo de una aplicación de escritorio para la creación de pequeños formularios de forma sencilla, como fue el desarrollo de toda la aplicación, complejidad, código, programas usados, el lenguaje usado...

La función principal será generar un código html de formularios de forma rápida, mediante pocos pasos y campos a rellenar, para ser introducidos en páginas web creadas desde 0, o en páginas de diseño web online (que permita implementar código propio). Tales como WordPress, Wix... La aplicación mostrará el código <form> o el <html> completo a elección. Diseñada para los diseñadores webs, y toda empresa que se dedique a ello.

Dispone también de una versión mejorada y otra simple, siendo la versión mejorada la implementación de pago a futuro, que se conseguiría al registrarte y canjear el código/clave (el pago no está en la versión actual del proyecto, pero el registro si tiene su funcionalidad) y la versión básica, llamada “guest”, una versión a la que no hay que registrarte o iniciar sesión, y tiene limitaciones en selección de algunos campos.

Summary

The project will consist of the development of a desktop application for the creation of small forms in a simple way, such as the development of the entire application, complexity, code, programs and language used...

The main function will be to generate a html code of forms quickly, through a few steps and fields to be filled, to be introduced in web pages created from 0, or in online web design pages (that allows you to implement your own code). Such as WordPress, Wix... The application will display the code <form> or the <html> complete, depending of your choices. Designed for web designers, and any company that is dedicated to it.

It also has an improved version and a simple one, the improved version being the implementation of future payment, which would be achieved by registering and redeeming the code / key (the payment is not in the current version of the project, but the registration has its functionality) and the basic version, called "guest", it’s a version which you do not have to register or log in, and has limitations in selecting some fields.

## 2.- Justificación:

El tema del proyecto se tomó cuando en el periodo de prácticas, se necesitó de una herramienta de formularios fácil de usar para los nuevos en el mundo de diseño web, y a la falta de esta, se inició con pequeñas ideas para crear una aplicación que cumpla con esa función y permita abarcar más campos.

El objetivo de la aplicación buscada era poder crear formularios sin la necesidad de entrar en códigos fuentes de la página sin tener mucho conocimiento de manejo de código, cuando algunos de los editores web, tales como (Wix, WordPress...) es algo complicado de tocar para los nuevos, se necesitaba algo más interactivo.

A lo largo del proyecto, la idea se fue torciendo cuando se investigó y se decantó de que la complejidad no estaba en la implementación del código, si no en la creación de dichos formularios una y otra vez...

Para el proyecto, se hizo más investigación en WordPress y se mencionará más adelante.

# 3.- DEFINICION DEL PROYECTO

## 3.1- Introducción

En el actual documento, se está presentando todo acerca del TFG de "Desarrollo de Aplicaciones Multiplataforma" y sobre el código de la aplicación desarrollada.

Tal TFG consiste en desarrollar una aplicación de escritorio, Android... usando Java, Python... junto a un manual y presentación.

La idea elegida para este proyecto es la siguiente:

Tener una página web para tu empresa, para darte a conocer más es algo indispensable hoy en día, y tener algún contacto de los usuarios/clientes con la empresa, así como un feedback de ellos para mejoras en la empresa, también lo es. Y una de las mejores formas es mediante pequeños formularios fáciles de entender y rellenar.

El diseño web es bastante abierto en cuanto a cómo hacerlo, ya que se puede crear desde cero, con un editor de texto simple, así como usar páginas e instalaciones ya preparadas para un manejo básico más fácil.

Sin embargo, no está todo tan completo y fácil, por lo que, la comunidad en constante creación y desarrollo de addons/plugins, que ayude aún más, que sean más flexibles a ciertas necesidades. Y estás pueden ser la creación de formularios.

Siguiendo con el caso de Wordpress, permite introducir código propio dentro de la página, perfecto para la implementación de formularios creados con HTML/CSS/JS... Pero ¿y si el usuario no está familiarizado con creación de formularios y pueda cometer fallos de código? ¿O si se necesitan muchos formularios y poco tiempo disponible? Pues aquí es donde entra la importancia de tener una aplicación que permita generar el código completo de un formulario, con unos simples clics de forma muy rápida y eficaz.

Así que el proyecto se basará en crear la aplicación que permita crear esos formularios con una interfaz sencilla y guiada con un resultado exitoso.

## 3.2- Objetivos:

- Mejorar la eficacia de la creación de formularios.

Gracias a que la aplicación será compuesta por botones y pequeños campos a rellenar, no tomará tanto tiempo y se evitarán errores en código.

- Satisfacer con las necesidades de los clientes en torno a los formularios.
Que los usuarios puedan elegir todo tipo de campos necesarios sin que les falte información.

- Aprender y comprender más del lenguaje usado frente a posibles futuros problemas y mejoras en funcionalidades y rendimiento.

Investigar, buscar, usar y comprender más sobre Python usando código y funciones no vistas en clase, y aprender más sobre los funcionamientos y como poder implementarlos.

- Comprobar el funcionamiento de implementación de addons y plugins en distintos sitios e instaladores de diseño web.

La implementación de productos similares a páginas de diseño web para tener un conocimiento frente a la competencia en dichas páginas.

Estos son los objetivos principales que se tomaron en cuenta con la idea de la aplicación. Y se tomarán en cuenta en todo momento para crear la aplicación y poder cumplirlos sin faltas.

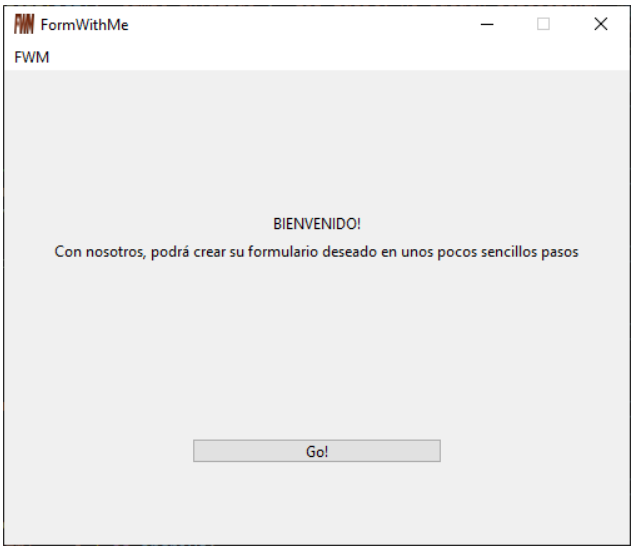
## 4- Aplicación:

La aplicación es llamada “FormWithMe”, donde su función es generar un código de formulario en HTML, usando unos campos a rellenar. Todo esto se crea mediante unos simples clics.

### 4.1- Logo:

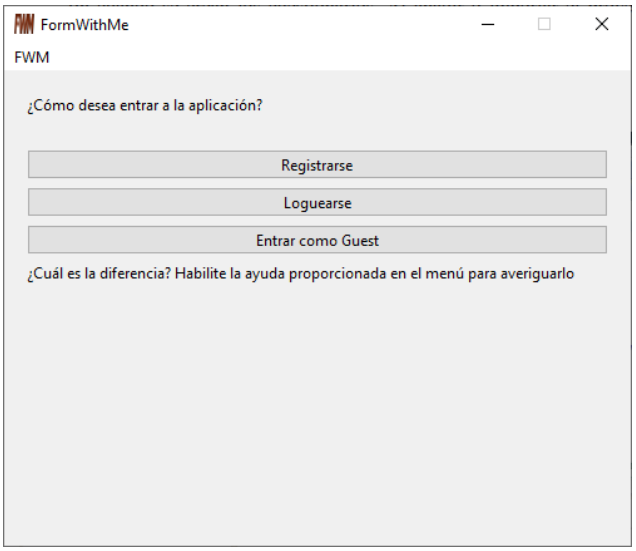

### 4.2- Ventanas:

La aplicación se dispone de 6 ventanas, siendo estas:



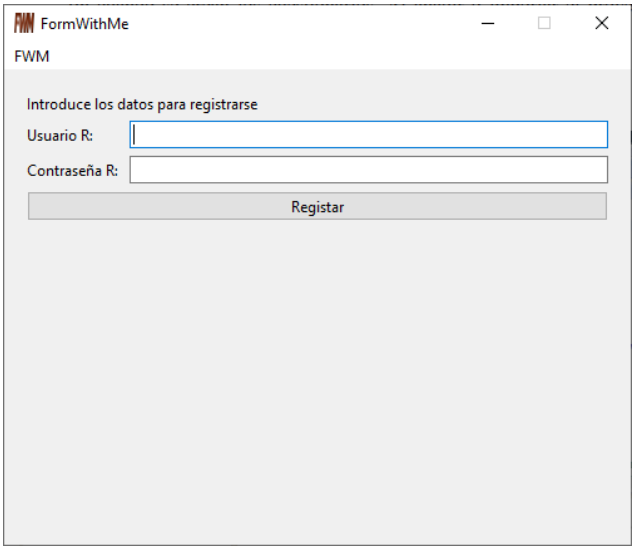

Ventana 1- Bienvenida

Nada más abrir la aplicación, se mostrará una ventana de bienvenida con un pequeño mensaje, y un botón para continuar con el programa




Ventana 2- Inicio

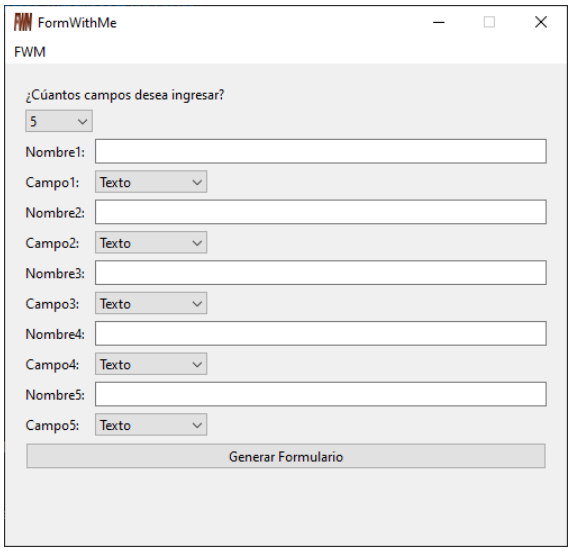
Seguidamente hay una ventana que indica que tiene que iniciar sesión o entrar como "Guest" sin tener que introducir datos algunos, así como poder registrarte y luego iniciar la sesión.




Ventana 3 y 4 – Registro e Inicio de sesión

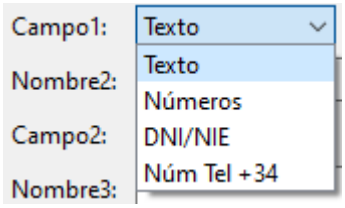
Las ventanas de inicio y registro son iguales, dispone de los campos necesarios a rellenar y continuar.

La ventana de registro está pensada en tener un campo adicional a la hora de la venta comercial del programa, donde los clientes, tendrán que introducir una clave que se adquiriría pagando desde una web para obtener las ventajas "premium".



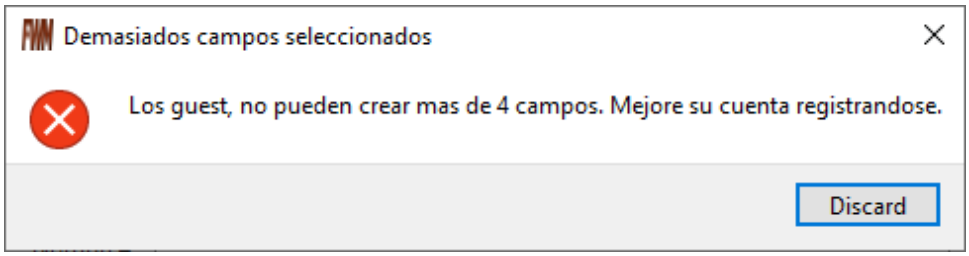

Ventana 5- Recogida de datos

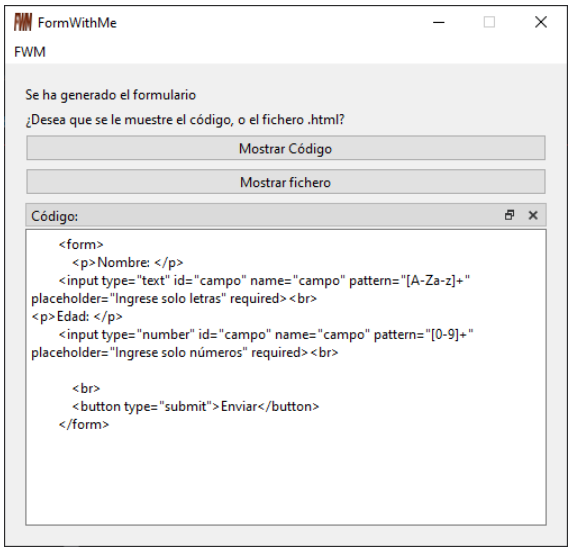
Al iniciar sesión o entrar como “guest”, los usuarios, deberán elegir cuantos campos quiere generar (con un máximo de 7, 4 para los “guest”), así como introducir el nombre que tendría y el tipo de campo que desea. Los campos serían los siguientes:




Texto- donde es un campo que permite letras sin límite. Números- campo que permite solamente números, sin límite. DNI/NIE- (solo para usuarios registrados) campo con una plantilla formada de 8 números y una letra al final (excluyendo la i, o y ñ). Núm. Tel +34- (solo para usuarios registrados) campo con una plantilla formada de 9 dígitos.

Existe una ventana emergente para el “guest” al intentar usar las funciones “premium”.





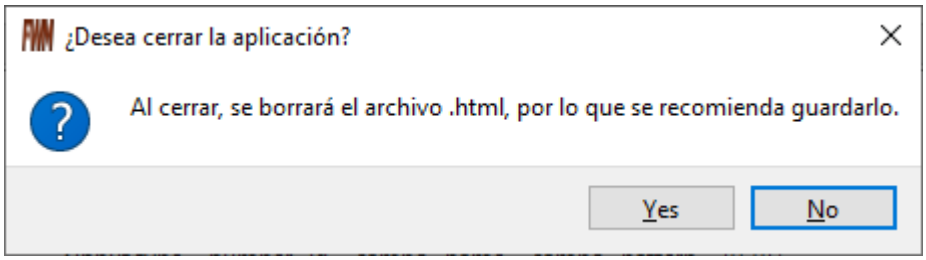

Ventana 6- Resultado Final

Finalmente, al darle al botón de generar código, se mostrará una última ventana con 2 botones y opciones, mostrar el código o mostrar el fichero (en dicha imagen, ya se pulsó el botón de mostrar código).

Mostrar código hará que aparezca un campo inferior, con únicamente el código de un formulario (excluyendo las etiquetas <html><body>).

Mostrar fichero hará que se abra el explorador de windows con el fichero .html seleccionado para poder copiar y trabajar con él.

Al cerrar la aplicación, se dará un aviso de que el fichero .html será borrado, por lo que se recomienda cambiarlo de sitio, copiarlo o cambiar el nombre. Se crea en el Escritorio de forma predeterminada.



# 5.- Desarrollo:

## 5.1- Primeros pasos:

Tras tener pensada la idea principal, se fue detallando lo esencial de la aplicación, su principal función: crear formularios a base de botones y generar el código.

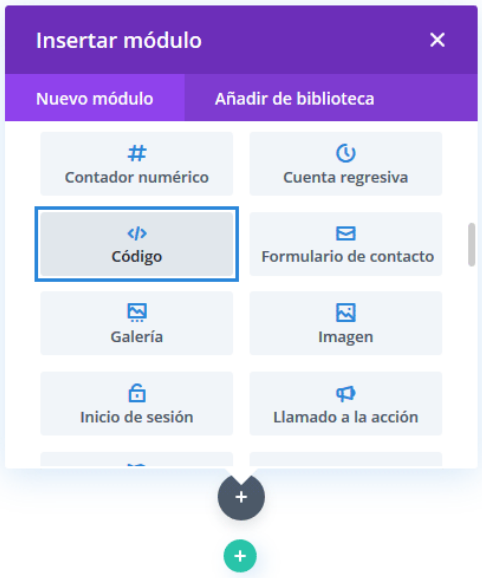
La idea es que al final de la creación, que el programa muestre el código ya hecho, e implementarlo en donde necesite el cliente.

Aparte del código, el usuario también puede recoger un fichero .html donde estará todo el código del formulario, para que pueda ser usado e implementado mediante un iframe, ser modificarlo a su gusto, implementar diseño CSS o funciones JavaScript.

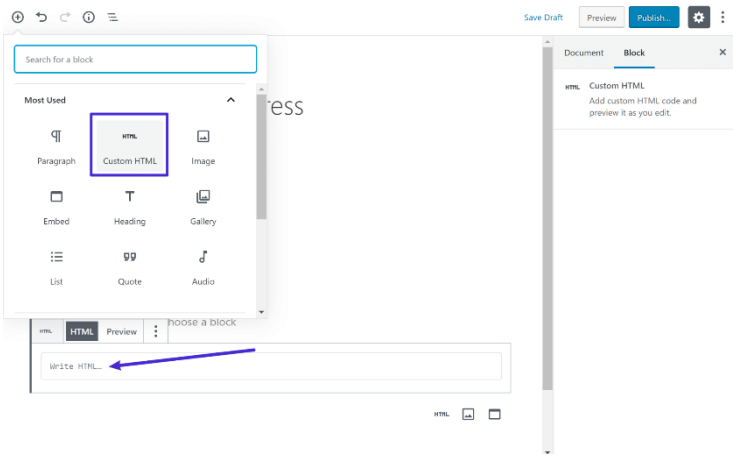
El código se puede usar en la plataforma WordPress, donde está la función de implementar código HTML, que hará fácil la implementación. Para el CSS o JS se habrá que usar la base de datos instalada y cambiar desde ahí.

Temas para WordPress, como Divi, dispone de un módulo único de código, fácil de usar, y también permite meter CSS de forma muy sencilla.

Módulo de código en Divi, para una implementación fácil del código generado con la aplicación.



Opción del propio WordPress de implementar código.




[Fuente]

## 5.2- Metodología:

Tras conocer la idea principal del proyecto, se fue tomando la siguiente metodología:

**-Diseño conceptual-** Al ser una aplicación cuya función sea mejorar la eficacia en la creación de formularios, su diseño será sencillo, fácil de entender y visualizar los componentes y que tenga una navegación sin caminos cortados.

Lo importante para el usuario es no hacerle perder el tiempo con cosas que podrían confundirle, por lo que se dispondrá solamente de elementos necesarios, caminos y seguimientos únicos para que los usuarios no se pierdan y tengan una continuidad hasta el resultado final.

**-Elegir tecnología (lenguaje)-** Luego se empezó a pensar en el lenguaje de programación que se usaría, así como otras tecnologías (base de datos).

El lenguaje elegido y usado es Python, siendo Python3

Se ha usado el PySide6 para el tema general y los widgets.

> Se ha usado subprocess, para permitir que salga una ventana emergente al darle a la X de cerrar la aplicación y dar un aviso de que el fichero se borrará.

Se ha usado el os para conseguir abrir el explorador de windows con una ruta específica para mostrar al usuario el fichero generado.

Se ha usado el Visual Estudio Code para la escritura de todo el código junto a extensiones de Python.

Información VSC

Versión: 1.78.2 (user setup)

Commit: b3e4e68a0bc097f0ae7907b217c1119af9e03435

Fecha: 2023-05-10T14:39:26.248Z

Electrón: 22.5.2

Chromium: 108.0.5359.215

Node.js: 16.17.1

V8: 10.8.168.25-electron.0

OS: Windows_NT x64 10.0.19045

En espacio aislado: Yes

Información Python

Python 3.11.3

Se eligió el Python como lenguaje gracias a su fácil entendimiento de código, una gran cantidad de librerías que permite crear muchísimos módulos y funciones, por su popularidad y con eso la rapidez de encontrar ayuda online y por ser un lenguaje portátil (aunque el proyecto final está hecho para Windows)

## 5.3- Base de datos

Como base de datos, en el proyecto se usa un editor de texto sencillo para guardar información de los usuarios. Pero claro está que, en versiones finales, dependiendo del cliente, y del servidor que usará para guardar el programa.

Por lo qué, está a disposición del cliente de la empresa decidir dónde quiere guardar los datos. En el caso actual, en un servidor comprado que disponen.

**-Desarrollo aplicación-** Se fue desarrollando la aplicación con el lenguaje finalmente elegido.

Seguidamente se dará paso al desarrollo del código y función para la aplicación.

La aplicación consta de una clase principal QMainWindow, el init, y varios métodos con su respectiva función.

Como importación y uso de librerías hay las siguientes:

```
from PySide6.QtWidgets import *
from PySide6.QtCore import *
from PySide6.QtGui import *
import subprocess
import os
```

Los métodos son los siguientes:

```
  6  class VentanaPrincipal(QMainWindow):
  7  >    def __init__(self):···
315
316  >    def mwhatsthis(self):···
318
319  >    def hLogOut(self):···
321
322  >    def mostrarCodigo(self):···
334
335  >    def mostrarFichero(self):···
342
343  >    def recogidaDatos(self):···
595
596  >    def setvalue(self):···
884
885  >    def iniciar(self):···
887
888  >    def irRegistro(self):···
890
891  >    def irLogueo(self):···
893
894  >    def irCreacion(self):···
896
897  >    def registrar(self):···
904
905  >    def comprobar(self):···
925
926  >    def closeEvent(self, event):···
943
```

## 5.4- Explicación del código

### Menú

```
#-------------------------------------Menus-------------------------------------------
bMenu = self.menuBar()
menu = bMenu.addMenu("FWM")

aLO = QAction("Log out", self)
aLO.triggered.connect(self.hLogOut)
aLO.setWhatsThis("Hacer LogOut de tu cuenta y volver")
menu.addAction(aLO)


wt = QAction("¿Ayuda?", self)
wt.triggered.connect(self.mwhatsthis)
wt.setWhatsThis("Haciendo click se le mostrará información sobre algunos campos que necesite ayuda")
menu.addAction(wt)

iGH = QAction("GitHub", self)
iGH.triggered.connect(self.mwhatsthis)
iGH.setWhatsThis("Se le llevará al GITHUB del proyecto/código")
menu.addAction(iGH)
```

Se crea un menú con 3 opciones, siendo estás: la opción de hacer LogOut y que el siguiente a usar la aplicación, no tenga acceso a las funciones limitadas si no tiene cuenta; la opción de habilitar el ”WhatsThis” y que el usuario se guie mejor con algo de ayuda proporcionada; y la opción de GitHub, para acceder al código fuente de la aplicación, así como el tutorial, readme, etc.

### Inicio

```
#-------------------------------------Inicio-------------------------------------------
layoutP = QVBoxLayout()
VP = QWidget()
VP.setLayout(layoutP)
self.setCentralWidget(VP)

layoutInicio = QVBoxLayout()
inicio = QWidget()
inicio.setLayout(layoutInicio)

self.bienvenida = QLabel("BIENVENIDO!", self)
self.bienvenida.setAlignment(Qt.AlignmentFlag.AlignCenter | Qt.AlignmentFlag.AlignBottom)
layoutInicio.addWidget(self.bienvenida)

self.texto = QLabel("Con nosotros, podrá crear su formulario deseado en unos pocos sencillos pasos", self)
self.texto.setAlignment(Qt.AlignmentFlag.AlignHCenter)
layoutInicio.addWidget(self.texto)

self.botonIniciar = QPushButton("Go!")
self.botonIniciar.setFixedSize(200, 20)
layoutInicio.addWidget(self.botonIniciar)
layoutInicio.setAlignment(self.botonIniciar, Qt.AlignHCenter | Qt.AlignVCenter)
self.botonIniciar.clicked.connect(self.iniciar)
```

Aquí se define la estructura principal de la ventana, y contiene 2 textos que dan la bienvenida al usuario, y un único botón para continuar con el programa.

**Acceso**

```
#---------------------------------------Entrar-------------------------------------------------
layoutPRE = QFormLayout()
Pre = QWidget()
Pre.setLayout(layoutPRE)

self.textoIniciar = QLabel("¿Cómo desea entrar a la aplicación?",self)
self.textoRelleno = QLabel("")
layoutPRE.addRow(self.textoIniciar)
layoutPRE.addRow(self.textoRelleno)

self.botonIR = QPushButton("Registrarse")
layoutPRE.addRow(self.botonIR)
self.botonIR.setWhatsThis("Se deberá ingresar un usuario y una contrasña que desea para introducirlo en la base de datos")
self.botonIR.clicked.connect(self.irRegistro)

self.botonIL = QPushButton("Loguearse")
layoutPRE.addRow(self.botonIL)
self.botonIL.setWhatsThis("Ingrese el usuario y la contraseña válida para prodecer a loguearse y tener algunas ventajas")
self.logedin = False
self.botonIL.clicked.connect(self.irLogueo)


self.botonIG = QPushButton("Entrar como Guest")
layoutPRE.addRow(self.botonIG)
self.botonIG.setWhatsThis("Entrará sin necesidad de registro o inicio de sesión, pero se le limitará algunas acciones")
self.botonIG.clicked.connect(self.irCreacion)

self.premium = QLabel("¿Cuál es la diferencia? Habilite la ayuda proporcionada en el menú para averiguarlo")
layoutPRE.addRow(self.premium)
self.premium.setWhatsThis("Al iniciar con el Guest, se le limitará algunas acciones tales como: No podrá implementar mas de 4 campos;
```

La siguiente parte seria ya dar acceso a la creación de formularios, usando una cuenta, creándola al momento (junto a la clave comprada), o usando un acceso limitado en algunas funciones, llamada ”guest”.

Todos los campos necesarios de explicación tienen puesto una línea con un texto de ayuda para entenderlo mejor.

También hay un texto siempre visible que indica sobre esta herramienta de ayuda, por lo que los usuarios estarán ya al tanto de le ayuda, y no queda como un elemento oculto.

Posteriormente se añadió la opción del uso de la tecla F1 (generalmente usada en muchas otras aplicaciones para obtener ayuda) para habilitar el Whats this sin ir al menú.

```
def keyPressEvent(self, event):
    if event.key() == Qt.Key.Key_F1:
        self.mwhatsthis()
```

**Registro/Inicio**

```
#-------------------------------------------Registro-----------------------------------
layoutFormR = QFormLayout()
register = QWidget()
register.setLayout(layoutFormR)
expl = QLabel("Introduce los datos para registrarse", self)
layoutFormR.addRow(expl)
usuario = QLabel("Usuario R: ", self)
clave = QLabel("Contraseña R: ", self)
self.usuarioQLr = QLineEdit()
layoutFormR.addRow(usuario, self.usuarioQLr)
self.passwordr = QLineEdit()
self.passwordr.setEchoMode(QLineEdit.Password)
layoutFormR.addRow(clave, self.passwordr)

self.botonR = QPushButton("Registar")
layoutFormR.addRow(self.botonR)
self.botonR.clicked.connect(self.registrar)

#-------------------------------------------Login-------------------------------------
layoutFormL = QFormLayout()
login = QWidget()
login.setLayout(layoutFormL)
usuario = QLabel("Usuario: ", self)
clave = QLabel("Contraseña: ",self)
self.usuarioQL = QLineEdit("")
layoutFormL.addRow(usuario, self.usuarioQL)
self.password = QLineEdit("")
self.password.setEchoMode(QLineEdit.Password)
layoutFormL.addRow(clave, self.password)


self.botonL = QPushButton("Login")
layoutFormL.addRow(self.botonL)
self.botonL.clicked.connect(self.comprobar)
```

El código entre el registro y el inicio de sesión es muy similar, comparten los campos a rellenar, lo diferenciado es la función a la que llama. Nota: son 2 ventanas distintas.

Recogida de datos

```
#-----------------------------------------------Recogida de datos--------------------
layoutPCantidad = QFormLayout()
cantidad = QWidget()
cantidad.setLayout(layoutPCantidad)


self.textoIniciar2 = QLabel("¿Cúantos campos desea ingresar?",self)
layoutPCantidad.addRow(self.textoIniciar2)

self.cantidadItems = QComboBox()
layoutPCantidad.addRow(self.cantidadItems)
self.cantidadItems.setFixedSize(60,20)
self.cantidadItems.setCurrentIndex(200)
self.cantidadItems.addItems(["None","1", "2", "3", "4", "5", "6", "7"])

self.cantidadItems.currentIndexChanged.connect(self.setvalue)
```

Lo siguiente ya es la ventana posterior al acceso elegido, donde hay un texto que nos pregunta la cantidad de campos, y un QComboBox con los números de campo que se puede elegir, y que al elegir cualquiera, el resto de la página mostrará o quitará los campos según los que se elijan.

**Recogida datos - campos**

```
                #-------------------------------------------------
                                #CAMPOS
                #---------------------1---------------------------
        layoutPC1 = QFormLayout()
        self.cantidad1 = QWidget()
        self.cantidad1.setLayout(layoutPC1)

        self.nombreC1 = QLabel("Nombre1: ", self)
        self.tipoC1 = QLabel("Campo1: ", self)
        self.nombreC1L = QLineEdit()
        layoutPC1.addRow(self.nombreC1, self.nombreC1L)
        self.tipoC1L = QComboBox()
        layoutPC1.addRow(self.tipoC1L)
        self.tipoC1L.setFixedSize(100,20)
        self.tipoC1L.setCurrentIndex(200)
        self.tipoC1L.addItems(["Texto", "Números", "DNI/NIE", "Núm Tel +34"])

        layoutPC1.addRow(self.tipoC1, self.tipoC1L)

        self.cantidad1.hide()

        layoutPCantidad.addRow(self.cantidad1)
                #-------------------------------------------------
```

Este es el código de los campos a rellenar por el usuario para la recogida de datos y luego escribir el código HTML. Dicho código se repita 7 veces, 1 vez por cada campo a rellenar.

```
            #-------------------------------------------------

self.botonGF = QPushButton("Generar Formulario")
layoutPCantidad.addRow(self.botonGF)
self.botonGF.clicked.connect(self.recogidaDatos)
```

Finalmente, en esta ventana está el botón que hará continuar con el programa con los datos recopilados.

**Ventana Final**

```
#-------------------------------------Formulario Final---------------------------------
layoutFormFf = QFormLayout()
final = QWidget()
final.setLayout(layoutFormFf)

self.text1QLF = QLabel("Se ha generado el formulario")
layoutFormFf.addRow(self.text1QLF)
self.text2QLF = QLabel("¿Desea que se le muestre el código, o el fichero .html?")
layoutFormFf.addRow(self.text2QLF)
self.botonMC = QPushButton("Mostrar Código")
layoutFormFf.addRow(self.botonMC)
self.botonMC.clicked.connect(self.mostrarCodigo)
self.botonMF = QPushButton("Mostrar fichero")
layoutFormFf.addRow(self.botonMF)
self.botonMF.clicked.connect(self.mostrarFichero)


self.dock = QDockWidget()
self.dock.setWindowTitle("Código:")
self.campoC = QTextEdit("")
self.dock.setWidget(self.campoC)
self.dock.setHidden(True)
layoutFormFf.addRow(self.dock)
```

La última ventana, es ya donde se mostraría el producto final, pudiendo elegir entre el código en pantalla en un DockWidget, o que se muestre en el explorador de archivos, el fichero seleccionado, para poder usarlo como se desea.

**Código visual final**

```
#-----------------------------------------Stacked Layout-----
self.capa = QStackedLayout()
self.capa.addWidget(inicio)
self.capa.addWidget(Pre)
self.capa.addWidget(cantidad)
self.capa.addWidget(register)
self.capa.addWidget(login)
self.capa.addWidget(final)

layoutP.addLayout(self.capa)
```

Por último, en cuanto al código que hace referencia a lo visual, está la estructura de las ventanas, usando un Stacked Layout, para el orden y el cambio entre pantallas.

## 5.5- Explicación código métodos

**Menú**

```
def mwhatsthis(self):
    QWhatsThis.enterWhatsThisMode()

def hLogOut(self):
    self.logedin = False
    self.capa.setCurrentIndex(0)

def keyPressEvent(self, event):
    if event.key() == Qt.Key.Key_F1:
        self.mwhatsthis()

def irGitHub(self):
    webbrowser.open('http://github.com')
```

Los siguientes métodos, representan las acciones que tomarán las opciones del menú superior, así como la función de la tecla F1 para activar el Whats this sin tener que interactuar con el menú.

**Método del registro**

```
def registrar(self):
    registros = open ('usuariosRegistrados.txt', "a")
    textR =  self.usuarioQLr.text() + ";" + self.passwordr.text()
    registros.write(textR)
    registros.write("\n")
    registros.close()
    self.capa.setCurrentIndex(1)
```

El método de registrar recoge los datos de los campos "usuario" y "contraseña", los QLineEdit, y los junta con un carácter ";" entre medias, para que luego la lectura sea más fácil. Luego, los escribe en un fichero .txt.

Recordemos que, para este proyecto, simulamos la base de datos con un editor de texto.

**Método de LogIn**

```
def comprobar(self):
    comprobar = 0
    registros = open ('usuariosRegistrados.txt', "r+")
    registros.seek(0)

    #comprobación
    for linea in registros:
        name, passwd = linea.split(';')
        if self.usuarioQL.text() == name and self.password.text() + "\n" == passwd :
            comprobar = 1

    #If para seguir adelante o recibir un mensaje de Error
    if comprobar == 1:
            print("Usuario_login_correcto")
            self.logedin = True
            self.capa.setCurrentIndex(2)
    else:
            QMessageBox.critical(self, "Error Usuario/Contraseña", "El usuario y/o contrase
            buttons=QMessageBox.Discard , defaultButton=QMessageBox.Discard)
    registros.close()
```

Para el método de inicio de sesión, primero se tendrá que abrir el archivo que guarda los usuarios y contraseñas del método registro, recorre cada línea, separando el usuario y contraseña gracias al punto y coma (;) antes puesto, y hace la comparación con los campos que el usuario ha introducido en la ventana. Si la comparación es correcta, pasa a mostrar la siguiente ventana, y se cambia un valor "logedin" a True, para un futuro uso con las opciones especiales al ser registrado.

Si la comparación es errónea, mostrará un mensaje tipo crítico, avisando que algún campo está mal escrito, pudiendo volver a intentarlo.

**Método mostrar campos**

```
def setvalue(self):
   getvalue = self.cantidadItems.currentIndex()

   if getvalue == 0:
    self.cantidad1.hide()
    self.cantidad2.hide()
    self.cantidad3.hide()
    self.cantidad4.hide()
    self.cantidad5.hide()
    self.cantidad6.hide()
    self.cantidad7.hide()

   elif getvalue == 1:
    self.cantidad1.show()
    self.cantidad2.hide()
    self.cantidad3.hide()
    self.cantidad4.hide()
    self.cantidad5.hide()
    self.cantidad6.hide()
    self.cantidad7.hide()
```

Este método consiste de un if, y varios elif, que lee la cantidad seleccionada en el ComboBox, y muestra y quita los elementos según el número elegido. Llega hasta 7 veces más, una por cada opción, además de la opción NONE, que quita todas.

**Método recogida datos**

```
def recogidaDatos(self):
    getvalue = self.cantidadItems.currentIndex()
    if getvalue > 4 and self.logedin == False:
        QMessageBox.critical(self, "Demasiados campos seleccionados", "Los guest, no pueden
            buttons=QMessageBox.Discard , defaultButton=QMessageBox.Discard)
    else:
        contenido_html = """
<!DOCTYPE html>
<html>
<head>
    <title>Formulario</title>
</head>
<body>
    <h1>FORMULARIO</h1>
    <form>
        """
        rutaEF = os.path.join(os.path.join(os.environ['USERPROFILE']), 'Desktop')
        with open(rutaEF +"/formulario.html", "a") as archivo:
                archivo.write(contenido_html)
```

La primera parte de este método hace una comprobación con la variable “logedin” indicada anteriormente, con la cual detecta si el número de campos elegido es mayor o menor al permitido por un “guest”, y si se inició como “guest” o como usuario registrado. En caso de que se haya elegido más campos de los permitidos siendo “guest”, saltará el mensaje indicando que no puede continuar con dichos campos.

Una vez que todo está adecuado para seguir sin mensajes de error, empieza creando un fichero en el escritorio del usuario, llamado “formulario.html”. Y dentro se escribirá el inicio del código html, código que nunca cambiará y será el mismo.

```
for i in range(getvalue):
    if i == 0:
        campo1 = self.nombreC1L.text()
        textF1 = '<p>' + campo1 + ": </p>"

        tipo1 = self.tipoC1L.currentIndex()
        if tipo1 == 0:
            tipo1campo = '        <input type="text" id="campo" name="campo" pattern="[A-Za-z]+" placeholder="Ingrese solo letras" required>
        elif tipo1 == 1:
            tipo1campo = '        <input type="number" id="campo" name="campo" pattern="[0-9]+" placeholder="Ingrese solo números" required>
        elif tipo1 == 2:
            if self.logedin == False:
                QMessageBox.critical(self, "Campo inválido", "Los guest, no pueden seleccionar esta plantilla. Mejore su cuenta registrandos
    buttons=QMessageBox.Discard , defaultButton=QMessageBox.Discard)
            else:
                tipo1campo = '        <input type="text" pattern="^\d{8}[A-HJ-NP-TV-Za-hj-np-tv-z]$" placeholder="Ingrese un DNI válido (8 d
        elif tipo1 == 3:
            if self.logedin == False:
                QMessageBox.critical(self, "Campo inválido", "Los guest, no pueden seleccionar esta plantilla. Mejore su cuenta registrandos
    buttons=QMessageBox.Discard , defaultButton=QMessageBox.Discard)
            else:
                tipo1campo = '        <input type="tel" pattern="^\d{9}$" placeholder= "Ingrese un Número de telefono" required><br>'
        else:
            print("metaestable")

        rutaEF = os.path.join(os.path.join(os.environ['USERPROFILE']), 'Desktop')
        with open(rutaEF +"/formulario.html", "a") as archivo:
            archivo.write(textF1)
            archivo.write("\n")
            archivo.write(tipo1campo)
            archivo.write("\n")
            archivo.close()
```

La siguiente parte se vuelve a repetir una vez por cada campo, un total de 7.

Lo que se hace, es leer los campos introducidos por el usuario, y asigna una variable con un texto. Texto que será introducido como línea en el HTML, el  con el nombre del campo, y los <input type>, que según el campo elegido es uno u otro. Con dichas variables, se van escribiendo en el fichero html antes creado.

En los <input type>, se puede ver que hay varios, con patrones únicos y restricciones, estando 2 de ellos, con una comprobación de “logedin”, ya que los “guest” no pueden usar la plantilla ya creada.

```
            contenido_html_final = """
            <br>
            <button type="submit">Enviar</button>
        </form>
    </body>
    </html>
                """
            rutaEF = os.path.join(os.path.join(os.environ['USERPROFILE']), 'Desktop')
            with open(rutaEF +"/formulario.html", "a") as archivo:
                    archivo.write(contenido_html_final)

            self.capa.setCurrentIndex(5)
```

Y, por último, solo queda cerrar el formulario con los cierres de las etiquetas, escribirlo todo en el fichero, y saltar ya a la ventana final.

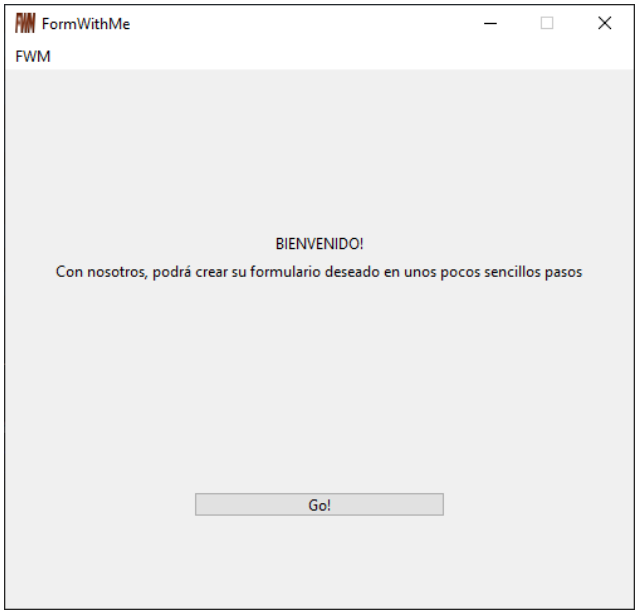
**Métodos mostrar código y/o fichero**

```
def mostrarCodigo(self):
    rutaEF = os.path.join(os.path.join(os.environ['USERPROFILE']), 'Desktop')
    rutaC = rutaEF + "/formulario.html"
    with open(rutaC, "r") as archivo:
        lineas = archivo.readlines()

    lineas_deseadas = lineas[8:-3]

    for lineas in lineas_deseadas:
        contenido = ''.join(lineas_deseadas) #solución encontrada en internet || lee cada l
    self.campoC.setText(contenido)
    self.dock.setHidden(False)

def mostrarFichero(self):
    rutaF = os.path.join(os.path.join(os.environ['USERPROFILE']), 'Desktop')
    archivoF = 'formulario.html'
    print(rutaF)
    rutaFC = os.path.join(rutaF, archivoF)
    subprocess.Popen(['explorer', '/select,', rutaFC])
```

El método de mostrar código abre el fichero para leerlo, se salta las líneas que no son necesarias, ya que se quiere mostrar solo lo que abarca el formulario, y lo muestra en un DockWidget, pudiendo copiarlo desde ahí.

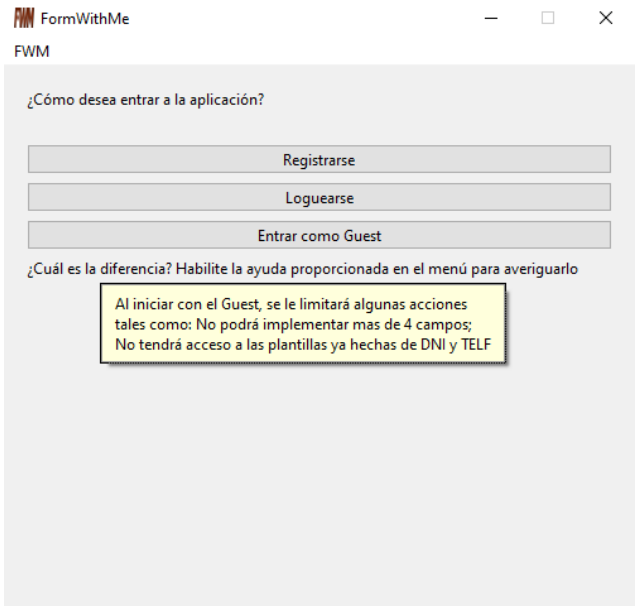
Por otro lado, el otro método, busca la ruta del escritorio y el fichero creado, ahí gracias al subprocess, abre con el explorador de archivos, y selecciona el propio fichero indicado.

**-Interfaz de usuario-** Sencilla y con elementos bien definidos y destacados para evitar confusiones y conseguir que el usuario no se pierda ningún elemento.




La interfaz es simple y sencilla para que el cliente pueda entenderlo y adaptarse fácilmente, dispone de elementos muy básicos y familiarizados.

También se toma en cuenta las dudas que pueda haber, por lo que hay un botón de ayuda que implemente el “What’s this”, un sistema de ayuda que al pinchar sobre un campo con el modo What’s this activado, se le proporciona información adicional para un mejor entendimiento.



**-Pruebas y depuración-** Mediante todo el desarrollo se realiza varias pruebas, y con la versión final, hacer pruebas más exigentes y depuraciones para evitar todos los problemas que puedan surgir.

Durante todo el proceso, se fue ejecutando el código en busca de errores y complicaciones, siendo solucionados bajo ayudas en la red o ayuda de tutores.

La ayuda en red se basó en foros de programación Python tales como:

https://stackoverflow.com/

https://www.reddit.com/

Hay más pasos que seguir, que implica desde la realización de la documentación de la propia aplicación, hasta el lanzamiento y mantenimiento y actualización de la misma. Pero son pasos a futuros después de la entrega.

## 5.6- Conclusión metodología

La metodología vista anteriormente puede verse como una adaptación de varias metodologías conocidas en el mundo del desarrollo del software. Tales como la SCRUM, que durante el código se fue centrando en cumplir con la funcionalidad que más se requería al momento.

También se puede ver algo de la metodología Kanban, que se usó para completar una funcionalidad para que funcione en su totalidad y a la perfección, sin distraerse otras funciones que se podrían implementar después (por ejemplo, el sistema registro/inicio sesión).

Dichas metodologías permitieron completar paso por paso las funciones necesarias e incluso se pudo permitir dar a conocer otras funciones, o cambios en dichas, que mejore el funcionamiento de la aplicación.

## 6.- Acerca de la organización:

La aplicación fue hecha por Alex Asavei, trabajador autónomo que dio sus servicios a Blue Web Service, una empresa que se encarda de hacer páginas y servicio de diseño web a clientes. Necesitando así una aplicación que permita introducir formularios sencillos en sus creaciones de las páginas.

Blue Web Service es una empresa situada en Seseña Viejo, Calle la Vega 24, que proporciona un servicio de creación de páginas web a los clientes. Dispone de 2 empleados dedicados a el diseño web.

El coste fue un único pago inicial de 950 €, con un acuerdo de más pagas por mantenimiento.

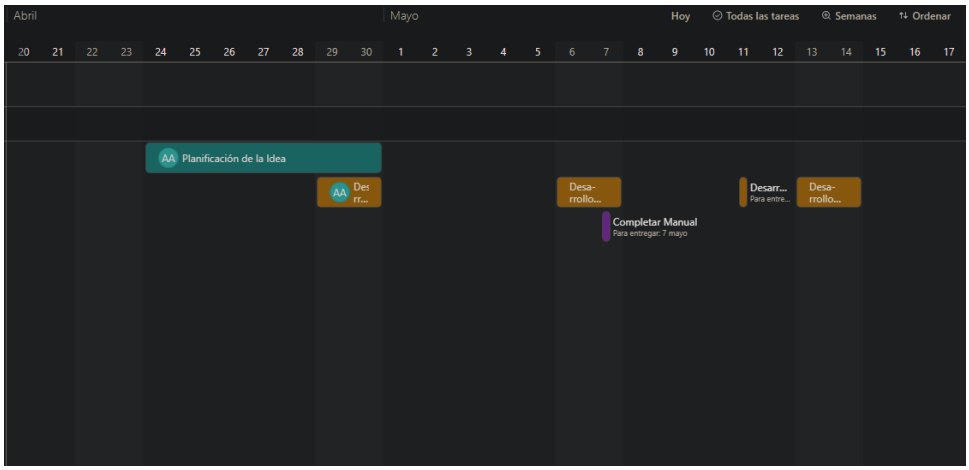
El precio fue decidido tras el análisis del mercado, los costes y gastos por el tiempo invertido, los beneficios que la aplicación proporcionará a la empresa y negociación sobre el mantenimiento y futuras actualizaciones e implementaciones.

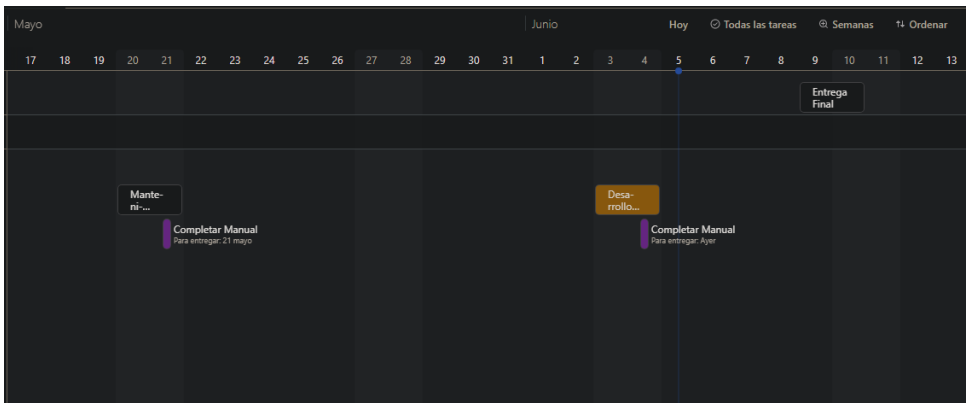
El encargo de la app se propuso por el constante trabajo con páginas web que la empresa recibía, y con los pocos trabajadores que había, para disminuir el tiempo de creación por cada web, e incrementar las ganancias por trabajos/tiempo.

Con la aplicación en sus manos, el tiempo empleado para cada página es muy reducido, por lo que, en un futuro, se hablará de más aplicaciones que podrían servir para incrementar la eficacia de creación de páginas, por mucho menos esfuerzo.

## 7.- Otros detalles:

**Diagrama de Gantt** periodo proyecto




Leyenda:

Azul: Planificación idea: 24-abril ~ 30-abril

Naranja: Desarrollo del código: 29 y 30 abril, 6 y 7 mayo, 11 mayo, 13 y 14 de mayo, 3 y 4 de junio (4 horas de media/día)

Morado: Completar Manual/Documentación: 7 y 21 mayo, 4 junio

Extras: Mantenimiento y búsqueda de errores: 20 y 21 mayo

Entrega final: 9 junio

**Hardware, software y presupuesto**

Sin uso de programas de pago o subscripciones.

**Aplicaciones:**

Visual Studio Code - Programación del código

Python 3.11 - Lenguaje instalado

Gimp 2.10 - Diseño de logotipo

Google Chrome – Búsqueda de dudas e información // Pruebas HTML

**Ordenador Personal** propio (~0€ coste adicional por proyectos)

Ordenador CPU-Ryzen 5 5600X; GPU-RX 580; RAM-16GB 1300MHz

Sistema operativo - Windows 10 Pro

**Enlaces:**

Se usó principalmente, Wikipedia, e información sacada de las páginas web de cada software o contenido.

**Base de datos:**

Editor de textos en la misma ruta del ejecutable, permitiendo almacenar y leer los usuarios con su respectiva contraseña, para el uso posterior.

Para el correcto uso de una base de datos, se usaría un servidor en la red, y conectado con MySQL/MariaDB. Se tendria que acceder con una conexión ftp para modificarlo a distancia, o mediante phpmyadmin, y dentro del código, hacer la conexión con la importación del conector mysql.

No se dispone de un backend, no hay cuenta administradora en este caso. La única vista es la del user, igual para todos.

**Traducción Ingles**

Todo el programa podría ser traducido al inglés sin ningún problema o punto que pueda afectar al mal funcionamiento del código.

# 8.- Conclusión

Conclusión acerca del proyecto:

**¿Se consiguió el objetivo propuesto?**

Como ya se comentó, la idea se fue ajustando a medida que se fue investigando y realizando la aplicación, y al darse cuenta de lo realmente necesario para los usuarios. Si bien la idea más antigua fue el sacar los datos de un formulario e introducirlos en un Excel u otra hoja de cálculos, se vio que la necesidad principal era crear esos formularios. Sin embargo, toda la aplicación funciona dando el resultado que se propuso finalmente, generar el código para una implementación.

**¿Se realizó a tiempo?**

Efectivamente el tiempo fue suficiente para el trabajo y un desarrollo más que suficiente para el correcto funcionamiento y el diseño sencillo.

**¿Hubo algún problema en el desarrollo?**

A medida que se creaba el código, si hubo problemas. Se puede destacar problemas resueltos como que, al hacer logout desde el menú, no se cambiaba la variable que detectaba el logueo, y sin cerrar la aplicación, al volver, pero esta vez con “guest”, el usuario podía usar campos que no debería. Esto de detectó a la hora de hacer pruebas y fue exitosamente resuelto.

Otro error puntual descubierto, es qué al iniciar el login incorrecto, no mostraba mensaje alguno. Dicho error solo ocurrió una vez, aún después de múltiples pruebas. Por lo que se decantó que fue un error en la compilación y ejecución en ese momento. Ya sea por error en la CPU, RAM u otro elemento externo al código.

**¿Qué hay del futuro de la aplicación?**

Como también se mencionó antes, la aplicación puede tener un potencial de venta gracias a las plantillas ya creadas y la mayor cantidad de campos que se puede seleccionar iniciando sesión. Por lo que, una idea sería la venta del producto “premium”, ya sea un único pago o subscripción, y mediante una clave, activar la cuenta en la ventana de registro.

# 9.- Anexo

Html/Form

HTML es el lenguaje de marcado estándar utilizado para crear páginas web. Permite estructurar y presentar el contenido de una página web, definiendo elementos y etiquetas que representan diferentes partes del contenido. Suelen estar en conjunto con CSS y JS.

HTML, CSS y JavaScript son los tres lenguajes fundamentales utilizados en el desarrollo web. HTML se encarga de estructurar y organizar el contenido de una página web mediante etiquetas que describen elementos como encabezados, párrafos, imágenes y formularios. CSS se utiliza para diseñar y estilizar el aspecto visual de la página, controlando colores, fuentes, tamaños y disposición de elementos. JavaScript agrega interactividad y funcionalidad al permitir la manipulación dinámica del contenido y el comportamiento de los elementos HTML y CSS. Con JavaScript, se pueden realizar acciones como validación de formularios, animaciones y actualizaciones en tiempo real. Estos tres lenguajes trabajan juntos para crear experiencias web atractivas y funcionales.

La etiqueta <form> es una de las etiquetas fundamentales en HTML y se utiliza para crear un formulario en una página web. Un formulario permite a los usuarios enviar datos al servidor web para su procesamiento. Dentro de la etiqueta <form>, se pueden agregar diferentes elementos de entrada, como campos de texto, casillas de verificación, botones y más.

Muchas empresas con páginas web disponen con estos formularios para recibir feedback y contacto con los usuarios que visitan la página, y es un ayuda para la empresa.

Enlace:

https://es.wikipedia.org/wiki/HTML

WordPress

WordPress es un sistema de gestión de contenido de código abierto muy popular y ampliamente utilizado para crear y administrar sitios web. Fue lanzado por primera vez en 2003 y se ha convertido en una plataforma versátil y escalable para todo tipo de sitios, desde blogs personales hasta sitios web corporativos.

Es una plataforma de gestión de contenido flexible, fácil de usar y altamente personalizable que permite a los usuarios crear y administrar sitios web de manera eficiente y efectiva, sin requerir conocimientos avanzados de programación o diseño web. Todo gracias a la interfaz intuitiva que trae, así como amplia gama de temas y complementos.

WordPress también cuenta con una gran comunidad de desarrolladores y usuarios que contribuyen activamente a su crecimiento y mejora. Esto significa que hay una amplia variedad de complementos disponibles para agregar funcionalidades adicionales a un sitio web, como formularios de contacto, galerías de imágenes, integración con redes sociales, optimización de motores de búsqueda (SEO) y más.

Enlace:

https://wordpress.com/es/

https://wordpress.org/es/

Plugins y Addons

Los plugins o addons son complementos que se pueden instalar en un sistema de gestión de contenido, como WordPress, para agregar funcionalidades adicionales y personalizadas a un sitio web. Estos complementos son desarrollados por terceros y se pueden instalar y activar fácilmente sin necesidad de modificar el núcleo del CMS.

Algunas páginas de ejemplo que usan WordPress como editor principal son:

Mercedes-Benz, Sony Music, El País, El Corte Inglés, RTVE...

Divi

Divi es un tema de WordPress altamente popular y ampliamente utilizado que se caracteriza por su enfoque en la creación de sitios web visualmente impresionantes y altamente personalizados. Desarrollado por Elegant Themes, Divi ofrece una amplia gama de características y herramientas de diseño que facilitan la construcción de sitios web atractivos y funcionales sin necesidad de conocimientos avanzados de programación.

Uno de los elementos destacados de Divi es su sistema de construcción de páginas, que utiliza una interfaz de arrastrar y soltar para crear y organizar el contenido del sitio. Esto permite a los usuarios diseñar páginas personalizadas de manera fácil e intuitiva, sin tener que escribir código manualmente. El constructor de páginas de Divi ofrece una amplia selección de módulos predefinidos que se pueden personalizar y ajustar según las necesidades del sitio web.

Sin embargo, trae un módulo llamado “código” que es una de las características más poderosas para aquellos que tienen conocimientos de programación o desean agregar personalizaciones avanzadas a su sitio web. Este módulo permite a los usuarios agregar su propio código HTML, CSS o JavaScript dentro del constructor de páginas de Divi, ofreciendo flexibilidad adicional al contenido ya existente.

Enlace:

https://www.elegantthemes.com/gallery/divi/

VSC

Visual Studio Code es un editor de código fuente desarrollado por Microsoft que se ha vuelto muy popular entre los desarrolladores. Es un entorno de desarrollo integrado (IDE) ligero y

altamente personalizable que ofrece una amplia gama de características y extensiones para facilitar la escritura de código. Proporciona resaltado de sintaxis, autocompletado, depuración integrada y control de versiones con Git. Tiene una amplia gama de extensiones que agregan funcionalidades adicionales, como soporte para diferentes lenguajes de programación y herramientas de formateo de código. Se integra con servicios populares como Azure Cloud, Docker, GitHub y más. También cuenta con una terminal integrada, snippets, plantillas y la capacidad de colaborar en tiempo real con Live Share. La comunidad activa contribuye con extensiones y mejoras continuas. En resumen, VSC es un editor versátil con herramientas poderosas que facilitan el desarrollo de software.

Enlace:

https://code.visualstudio.com/

Whats this

La función Whats this es una función personalizada que se utiliza en determinados contextos dentro de programas específicos. Su propósito principal es proporcionar información adicional o aclaraciones sobre un elemento o concepto en particular dentro del código.

La función Whats this se utiliza como una herramienta de documentación interna para los desarrolladores, permitiéndoles agregar comentarios o notas explicativas directamente en el código fuente. Esto puede ser especialmente útil cuando el código es complejo o cuando se trabaja en un proyecto colaborativo.

El funcionamiento de Whats this puede variar según cómo se haya implementado en el programa en el que se utiliza. En general, se puede utilizar de la siguiente manera:

Llamada a la función: En algún lugar del código, se realiza una llamada a la función Whats this. Esto puede ser en una línea específica o dentro de un bloque de código.

Parámetros de entrada: La función Whats this puede recibir parámetros de entrada que proporcionan información sobre el elemento al que se refiere. Estos parámetros pueden incluir el nombre del elemento, su propósito, su uso recomendado u otra información relevante.

Salida o visualización: Dependiendo de la implementación, la función Whats this puede mostrar la información proporcionada en la consola, en una ventana emergente o en otro formato adecuado para el entorno de desarrollo en el que se está utilizando.

Es importante destacar que la función Whats this no forma parte de la sintaxis básica de Python y su existencia depende del programa o proyecto específico en el que se esté trabajando. Por lo tanto, es recomendable consultar la documentación o los recursos relacionados con el programa en cuestión para obtener más detalles sobre el uso y las capacidades de la función Whats this en ese contexto específico.

Enlace:

https://doc.qt.io/qtforpython-5/PySide2/QtWidgets/QWhatsThis.html

PySide6

PySide6 es una biblioteca de Python que permite la creación de aplicaciones de escritorio con interfaces gráficas de usuario (GUI) utilizando el framework Qt. Es una versión para Python del popular conjunto de herramientas Qt, que ofrece una amplia gama de componentes y funcionalidades para crear interfaces gráficas atractivas y interactivas. PySide6 es compatible con varias plataformas, incluyendo Windows, macOS, Linux y Android, lo que permite desarrollar aplicaciones multiplataforma sin la necesidad de realizar cambios significativos en el código. Proporciona una amplia variedad de widgets y elementos de interfaz gráfica personalizables, permitiendo la construcción de interfaces atractivas y funcionales. Utiliza el mecanismo de señales y slots para facilitar la comunicación entre los componentes de la interfaz gráfica, lo que permite una programación reactiva y una interacción fluida. PySide6 también ofrece soporte para modelos y vistas, lo que facilita la gestión y visualización de datos tabulares y jerárquicos. Además, proporciona la capacidad de personalizar la apariencia de la interfaz gráfica utilizando hojas de estilo CSS. PySide6 se integra con otras bibliotecas populares de Python, como NumPy y Pandas, lo que permite combinar las capacidades de manipulación de datos con la interfaz gráfica generada por PySide6. Con lo que se resume en una biblioteca para aplicaciones de GUI con mucha compatibilidad, elementos, soportes, vistas, modelos...

Enlace:

https://es.wikipedia.org/wiki/PySide

Asana

Asana es una aplicación web y móvil que ofrece servicios de gestión de proyectos y colaboración en equipo. Con Asana, puedes organizar y supervisar tareas, proyectos y equipos de manera eficiente. Permite crear proyectos, asignar tareas, establecer fechas de vencimiento y definir hitos clave. También facilita la comunicación entre los miembros del equipo, el seguimiento del progreso y la colaboración en tiempo real. Asana es una herramienta versátil que ayuda a mejorar la productividad y la eficacia en la gestión de proyectos y tareas de forma colaborativa.

Enlace:

https://app.asana.com/