README

Views

Función perfil_usuario

Esta función devuelve a todos los usuarios registrados por medio de la consulta PerfilDelUsuario.objects.all(), y los envía a través de un contexto al template usuarios.html, donde será recibida por función for para ser listados.

```
def perfil_usuario(request):
    usuarios = PerfilDeUsuario.objects.all()
    return render(request, 'usuarios.html', {'usuarios': usuarios})
from django.shortcuts import render
```

Función ingresar_datos_view

Esta función devuelve a un usuario seleccionado por medio de un id, el cual será obtenido por medio de consulta(PerfilDelUsuario.objects.get(user_id=user_id), este registro es enviado al template por medio de un contexto.

```
def ingresar_datos_view(request, user_id):

        usuario = PerfilDeUsuario.objects.get(user_id=user_id)
        context = {'usuario' : usuario}

        return render(request, 'perfil_usuario.html', cont
```

Función home

Esta función solo re direcciona a la página de inicio del sistema.

```
def home(request):
    return render(request,'inicio.html')
```

Función Carrusel

Esta función solo re direcciona a una prueba de boostrap, donde se agregó una estructura carrusel.

```
def carrusel(request):
    return render(request,'carrusel.html')
```

Función register

Esta podría ser la función más importante de los views, en esta función se discrimina entre dos tipos de consulta, si la consulta fue enviada por el método post agregar los datos aun registro, en específico, los de usuarios. Si fuera get enviaría un formulario del archivo **forms.py** que se relacione con esta función. En esta función se utilizó la función **make_password** para poder codificarla y la clave fuera compatible con la función authenticate de la función usuario_login.

```
def register(request):
    if request.method == 'POST':
        formulario_usuario = UserForm(request.POST)
        formulario_perfil_usuario = PerfilUsuarioForm(request.POST)
        print()

        if formulario_usuario.is_valid() and
formulario_perfil_usuario.is_valid():
            print(formulario_usuario)
            user= formulario_usuario.save(commit=False)
            user.password= make_password(request.POST['password'])
            user.save()
           # perfil = formulario_perfil_usuario.save()
            perfil = formulario_perfil_usuario.save(commit=False)

            perfil.user=user
            perfil.save()

            return redirect('/usuarios/')



    else:
        formulario_usuario = UserForm()
        formulario_perfil_usuario=PerfilUsuarioForm()


    context = {
        'formulario_usuario': formulario_usuario,
        'formulario_perfil_usuario':formulario_perfil_usuario


        }

    return render(request, 'register.html', context)
```

Función login

Esta función es la función es la que lanza la vista del login, tiene una mecánica similar a la de register, si es post, autentifica , por medio de la función athenticate, si los datos ingresados son correctos. Y si no lanza la página del login.

```
def usuario_login(request):
        if request.method=='POST':
            form = LoginUsuarioForm(request.POST)
            if form.is_valid():
                usuario= form.cleaned_data['usuario']
                password = form.cleaned_data['password']
                print(usuario)
                print(password)
                user = authenticate(request, username=usuario,
password=password)
                print(user)
                if user is not None:
                    if user.is_active:
                        login(request,user)
                        return  render(request,'logueo_exitoso.html')
                    else:
                        return  render(request,'cuenta_desactivada.html')
                else:
                    return  render(request,'logueo_rechazada.html')
        else:
            form=LoginUsuarioForm()
        return  render(request,'login.html',{'form':form})
```

Función logout

Esta función cierra la sesión de un usuario ,anterior mente inicializada, se puede comprobar, utilizando {{user}} en los template, si se estuviera logueado, debería aparecer su nombre en la interfaz.

```
def logout_view(request):
    print(request.user)
    print('logout')
    logout(request)
    return redirect('/')
```

Templates

Base Este archivo es la base de toda la interfaz.	**Inicio** Es la interfaz de la página principal o inicio del sist3ma.
Carrusel Es solo un prueba de la estructura carrusel de boostrap 5.	**Cuenta desactivada** Es la interfaz para avisar que la cuenta esta desactivada.
Logueo exitoso Es la interfaz que avisa que el logue fue exitoso.	**Logue rechazado** Es la interfaz que avisa que el logue fue rechazado.
Perfil_usuario Es la página que muestra el perfil de cada usuario del aplicación .	**Register** Register es la interfaz de para registrar a los usuarios.
Usuarios Es la interfaz que lista todos los usuarios de la aplicación.	

Forms.py

En este archivo se definen las clases utilizadas para generar formularios, en este caso se desarrollaron los siguientes:

UserForm: formulario para registrar los datos principales de cada usuario.

PerfilUsuarioForm: formulario que registra datos de usuarios que no han sido registrados por el formulario de UserForm, una extensión de user, sus modelos poseen una relación de uno es a uno.

LoginUsuarioForm: es utilizado para loguear a los usuarios.

***Los widget**, son sentencias que representan al campo en la interfaz de forma visual y en su comportamiento frente a ciertos eventos. En los códigos de este sistema se agregar para que fueran compatibles con boostrap 5

Models.py

Acá solo se creó una clase "**PerfilDeUsuario",** este modelo tiene una relación de uno es a uno con "User", ya que este posee los atributos que user no posee, una extensión de user, se agregaron campos como dirección, profesión, etc,

Url.py

Es código es el que relaciona las peticiones con las vistas, se importan las vistas desde el archivo views y se agregan a las vistas.

Básicamente esto

```
urlpatterns = [
    path("admin/", admin.site.urls),
    path("usuarios/", perfil_usuario, name="usuarios"),
    path("ingresar-datos/<int:user_id>/", ingresar_datos_view,
name="ingresar-datos"),
    path("register/", register, name="register"),
    path("login/", usuario_login, name="login"),
    path("out/", logout_view, name="logout"),
    path("", home,name="home"),
    path("carrusel/", carrusel,name="carrusel"),
]
```

URL – > Es la lista de urls.

La sintaxis de un path es:

Función Path()	Dirección	Vista	Nombre de la path