



UNIVERSIDAD TÉCNICA DE MANABÍ
FACULTAD DE CIENCIAS INFORMÁTICA
INGENIERÍA EN SISTEMAS DE INFORMACIÓN



DESARROLLO DE SISTEMAS DE INFORMACIÓN

PROYECTO FIN DE CICLO:

App de Mensajería

INTEGRANTES:

Carranza Moreira Luis.
Cedeño Vélez Génesis.
Navas Zambrano Diego.
Guaranda Pin Jonathan.
Pico Mera Armando.

DOCENTE:

Ing. José Párraga.

NIVEL:

4° Semestre “B”

PERIODO ACADÉMICO:

MAYO 2020 - OCTUBRE 2020

Proyecto Webplayground

App Messenger

Para el desarrollo de este proyecto, usamos ya la base trabajada durante todo este fin de ciclo.

Este proyecto consiste en crear una app de mensajería que permita intercambiar mensajes entre diferentes usuarios registrados e identificados. La funcionalidad de este proyecto es que se pueda establecer un chat privado entre el usuario y otros usuarios.

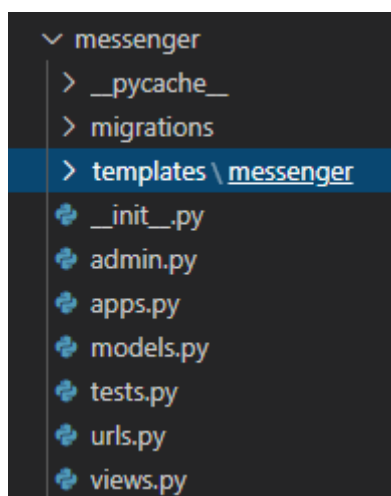
Para ello necesitamos ya estar en nuestro ambiente de trabajo creado durante las sesiones de clases:

```
C:\Proyecto\webplayground>C:/Users/USUARIO/anaconda3/Scripts/activate  
  
(base) C:\Proyecto\webplayground>conda activate django  
  
(django) C:\Proyecto\webplayground>
```

Una vez dentro de nuestro ambiente de trabajo, ahora vamos al desarrollo de la nueva app de mensajería primero debemos crearla como ya es de conocimiento en el terminal de Python de la siguiente forma:

```
(django) C:\Proyecto\webplayground>python manage.py startapp messenger
```

Al ejecutar el comando, se crearán los siguientes archivos lo que indica que ya tenemos creada la app denominada **‘messenger’**



Nota: La carpeta templates/messenger y el archivo urls.py se crean manualmente más adelante.

Una vez creada la app, se procede a registrarla en el archivo **settings.py** del webplayground:

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'ckeditor',  
    'core',  
    'messenger',  
]
```

Una vez registrada nuestra app, procedemos ir al **models.py** de messenger para establecer los modelos de la app:

```
messenger > 📁 models.py > 📁 messages_changed  
1  from django.contrib.auth.models import User  
2  from django.db.models.signals import m2m_changed  
3  from django.db import models  
4  
5  # Create your models here.  
6  class Message(models.Model):  
7      user = models.ForeignKey(User, on_delete=models.CASCADE)  
8      content = models.TextField()  
9      created = models.DateTimeField(auto_now_add=True)  
10  
11     class Meta:  
12         ordering = ['created']  
13  
14  
15     class ThreadManger(models.Manager):  
16         def find(self, user1, user2):  
17             queryset = self.filter(users=user1).filter(users=user2)  
18             if len(queryset) > 0:  
19                 return queryset[0]  
20             return None  
21  
22         def find_or_create(self, user1, user2):  
23             thread = self.find(user1, user2)  
24             if thread is None:  
25                 thread = Thread.objects.create()  
26                 thread.users.add(user1, user2)  
27             return thread  
28
```

```

messenger > models.py > messages_changed
29
30 class Thread(models.Model):
31     users = models.ManyToManyField(User, related_name='threads')
32     messages = models.ManyToManyField(Message)
33     updated = models.DateTimeField(auto_now=True)
34
35     objects = ThreadManger()
36
37     class Meta:
38         ordering = ['-updated']
39
40 def messages_changed(sender, **kwargs):
41     instance = kwargs.pop('instance', None)
42     action = kwargs.pop('action')
43     pk_set = kwargs.pop('pk_set')
44
45     false_pk_set = set()
46     if action is "pre_add":
47         for msg_pk in pk_set:
48             msg = Message.objects.get(pk=msg_pk)
49             if msg.user not in instance.users.all():
50                 false_pk_set.add(msg_pk)
51
52     pk_set.difference_update(false_pk_set)
53
54     instance.save()
55
56
57
58 m2m_changed.connect(messages_changed, sender=Thread.messages.through)

```

Una vez realizado los modelos para la app, se procede hacer la migración de esos modelos, esto se lo puede desarrollar varias veces dependiendo de los cambios que se le vayan ir dando a futuro al modelo, para realizar la migración escribimos el siguiente comando:

```

(django) C:\Proyecto\webplayground>python manage.py makemigrations messenger
Migrations for 'messenger':
  messenger\migrations\0002_auto_20201013_1421.py
    - Change Meta options on message
    - Change Meta options on thread
    - Add field updated to thread

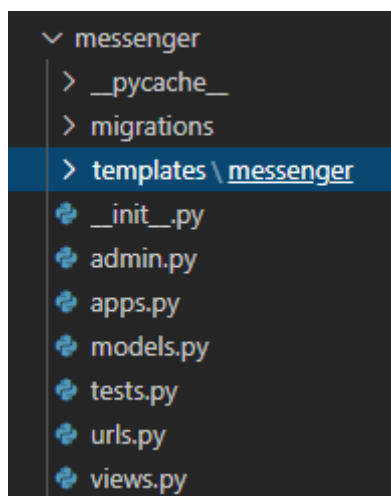
```

Como podemos observar, se nos modifica ya los cambios hechos en el **models.py**

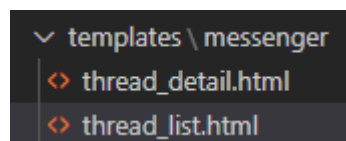
Una vez hecha la migración debemos aplicar los cambios, es decir migrar esos cambios, para ello se lo realiza con el siguiente comando:

```
(django) C:\Proyecto\webplayground>python manage.py migrate messenger
Operations to perform:
  Apply all migrations: messenger
Running migrations:
  Applying messenger.0002_auto_20201013_1421... OK
```

Ahora ya podemos avanzar con la app, necesitamos de unas vistas para poder visualizar el contenido que acabamos de crear, para ello dentro de la app messenger creamos los templates que guardarán nuestros archivos .html



Dentro de los templates desarrollamos dos archivos .html que nos servirán para el enmaquetado de nuestra app



thread_detail.html

```
messenger > templates > messenger > <> thread_detail.html
1  {% extends 'core/base.html' %}
2  {% load static %}
3  {% block title %}Hilo{% endblock %}
4  {% block content %}
5  <style>
6      .avatar { width:50px; height:50px; float:left; margin-right:10px; }
7      .thread { max-height:300px; overflow-y:auto; padding:0 0.5em;}
8      .mine   { padding:0 0.5em 0.25em; background-color:rgba(230,242,245,.5); width:92%; margin-left:8%; }
9      .other  { padding:0 0.5em 0.25em; background-color:#f2f3f5; width:92%; }
10 </style>
11 <main role="main">
12     <div class="container">
13         <div class="row mt-3">
14             <div class="col-md-9 mx-auto mb-5">
15                 <div class="row">
16                     <!-- Hilos de conversación -->
17                     <div class="col-md-4">
18                         <!-- Con una búsqueda inversa user.threads también podemos conseguir los hilos de un usuario -->
19                         {% for thread in request.user.threads.all %}
20                             <!-- Sólo mostraremos un Thread si tiene como mínimo 1 mensaje -->
21                             {% if thread.messages.all|length > 0 %}
22                                 <div class="mb-3">
23                                     <!-- Recorremos los miembros del hilo menos el propio request.user -->
24                                     {% for user in thread.users.all %}
25                                         {% if user != request.user %}
26                                             <!-- Mostramos el avatar del miembro -->
27                                             {% if user.profile.avatar %}
28                                                 
29                                             {% else %}
30                                                 
31                                             {% endif %}
32                                     <!-- Mostramos la información del miembro -->
33                                     <div>
34                                         <a href="{% url 'messenger:detail' thread.pk %}">{{user}}</a><br>
35                                         <small><i>Hace {{thread.messages.last.created|timesince}}</i></small>
36                                     </div>
37                                     {% endif %}
38                                 {% endfor %}
39                             </div>
40                             {% endif %}
41                         {% endfor %}
42                     </div>
43                     <!-- Hilo de conversación -->
44                     <div class="col-md-8">
45                         <!-- Recorremos los miembros del hilo menos el propio request.user -->
46                         {% for user in thread.users.all %}
47                             {% if user != request.user %}
48                                 <h4 class="mb-4">Mensajes con <a href="{% url 'profiles:detail' user %}">{{user}}</a></h4>
49                             {% endif %}
50                         {% endfor %}
51                         <!-- Mostramos los mensajes en una capa que tiene un overflow vertical de 300 píxeles -->
52                         <div class="thread" id="thread">
53                             {% for message in object.messages.all %}
54                                 <!-- Dependiendo del usuario asignamos una clase con un color de fondo u otro en el mensaje -->
55                                 <div {% if request.user == message.user %}class="mine mb-3"{% else %}class="other mb-3"{% endif %}>
56                                     <small><i>Hace {{thread.messages.last.created|timesince}}</i></small><br>
57                                     {{message.content}}
58                                 </div>
59                             {% endfor %}
60                         </div>
```

```

61 <!-- Aquí crearemos el formulario -->
62 <textarea id="content" class="form-control mb-2" rows="2" placeholder="Escribe tu mensaje aquí"></textarea>
63 <button id="send" class="btn btn-primary btn-block btn-sm" disabled>Enviar mensaje</button>
64 <script type="text/javascript">
65
66     var send = document.getElementById('send');
67     send.addEventListener('click', function(){
68         var content = encodeURIComponent(document.getElementById('content').value);
69         if(content.length > 0){
70             const url = "{% url 'messenger:add' thread.pk %}" + '?content=' + content;
71             fetch(url, {'credentials':'include'}).then(response => response.json()).then(function (data) {
72                 // Si el mensaje se ha creado correctamente...
73                 if (data.created) {
74                     document.getElementById('content').value = '';
75                     send.disabled = true;
76                     var message = document.createElement('div');
77                     message.classList.add('mine', 'mb-3');
78                     message.innerHTML = '<small><i>Hace unos segundos</i></small><br>' + decodeURIComponent(content);
79                     document.getElementById('thread').appendChild(message);
80                     scrollTopInThread();
81                 } else {
82                     console.log("Algo ha fallado y el mensaje no se ha podido añadir.")
83                 }
84
85                 if(data.first){
86                     window.location.href = "{% url 'messenger:detail' thread.pk %}"
87                 }
88             })
89         }
90     });

```

```

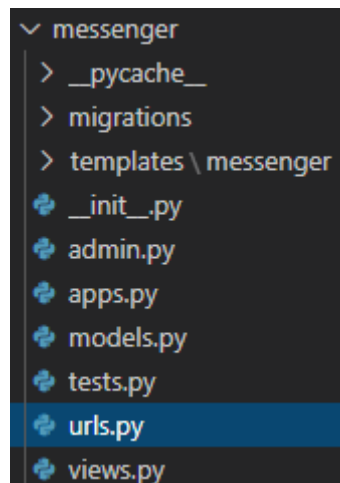
91
92     var content = document.getElementById('content');
93     content.addEventListener('keyup', function () {
94         if(!this.checkValidity() || !this.value){
95             send.disabled = true;
96         }else{
97             send.disabled = false;
98         }
99     })
100
101     function scrollTopInThread() {
102         var thread = document.getElementById('thread');
103         thread.scrollTop = thread.scrollHeight;
104     }
105
106     scrollTopInThread();
107
108 </script>
109 </div>
110 </div>
111 </div>
112 </div>
113 </main>
114 {% endblock %}

```

thread_list.html

```
messenger > templates > messenger > <> thread_list.html
1  {% extends 'core/base.html' %}
2  {% load static %}
3  {% block title %}Mensajes{% endblock %}
4  {% block content %}
5  <style>
6  | .avatar { width:50px; height:50px; float:left; margin-right:10px; }
7  </style>
8  <main role="main">
9  | <div class="container">
10 | | <div class="row mt-3">
11 | | | <div class="col-md-9 mx-auto mb-5">
12 | | | | <div class="row">
13 | | | | | <!-- Hilos de conversación -->
14 | | | | | <div class="col-md-4">
15 | | | | | | <!-- Recorremos los Threads y sólo mostramos los que tienen como mínimo 1 mensaje -->
16 | | | | | | {% for thread in request.user.threads.all %}
17 | | | | | | | {% if thread.messages.all|length > 0 %}
18 | | | | | | | <div class="mb-3">
19 | | | | | | | | <!-- Recorremos los miembros del hilo menos el propio request.user -->
20 | | | | | | | | {% for user in thread.users.all %}
21 | | | | | | | | | {% if user != request.user %}
22 | | | | | | | | | | <!-- Mostramos el avatar del miembro -->
23 | | | | | | | | | | {% if user.profile.avatar %}
24 | | | | | | | | | | | 
25 | | | | | | | | | | | {% else %}
26 | | | | | | | | | | | 
27 | | | | | | | | | | | {% endif %}
28 | | | | | | | | | | <!-- Mostramos la información del miembro -->
29 | | | | | | | | | | <div>
30 | | | | | | | | | | | <a href="{% url 'messenger:detail' thread.pk %}">{{user}}</a><br>
31 | | | | | | | | | | | <small><i>Hace {{thread.messages.last.created|timesince}}</i></small>
32 | | | | | | | | | | </div>
33 | | | | | | | | | | {% endif %}
34 | | | | | | | | | | {% endfor %}
35 | | | | | | | | | | </div>
36 | | | | | | | | | | {% endif %}
37 | | | | | | | | | | {% endfor %}
38 | | | | | | | | </div>
39 | | | | | | | <!-- Hilos de conversación -->
40 | | | | | | | <div class="col-md-8">
41 | | | | | | | | <p><i>Selecciona un hilo de conversación de tu panel izquierdo.</i></p>
42 | | | | | | | </div>
43 | | | | | | </div>
44 | | | | </div>
45 | </div>
46 </div>
47 </main>
48 {% endblock %}
```


Ahora procedemos a crear el archivo **urls.py** dentro de nuestra app messenger



Una vez creado el archivo, establecemos los path de nuestra app

```
messenger > urls.py > ...
1
2 from django.urls import path
3 from .views import ThreadList, ThreadDetail, add_message, start_thread
4
5 messenger_patterns = [
6     path('', ThreadList.as_view(), name='list'),
7     path('thread/<int:pk>/', ThreadDetail.as_view(), name='detail'),
8     path('thread/<int:pk>/add/', add_message, name='add'),
9     path('thread/start/<username>/', start_thread, name='start'),
10 ], 'messenger']
```

Ahora nos dirigimos al archivo **views.py** de nuestra app, procedemos a crear las vistas

```
messenger > views.py > ...
1 from django.contrib.auth.decorators import login_required
2 from django.utils.decorators import method_decorator
3 from django.views.generic.detail import DetailView
4 from django.views.generic import TemplateView
5 from django.http import Http404, JsonResponse
6 from django.shortcuts import get_object_or_404, redirect
7 from django.urls import reverse_lazy
8 from django.contrib.auth.models import User
9 from .models import Thread, Message
10
11 # Create your views here.
12 @method_decorator(login_required, name='dispatch')
13 class ThreadList(TemplateView):
14     template_name = 'messenger/thread_list.html'
15
16 @method_decorator(login_required, name='dispatch')
17 class ThreadDetail(DetailView):
18     model = Thread
19
20     def get_object(self):
21         obj = super(ThreadDetail, self).get_object()
22         if self.request.user not in obj.users.all():
23             raise Http404()
24         return obj
```

```

25
26 def add_message(request, pk):
27     json_response = {'created': False}
28
29     if request.user.is_authenticated:
30         content = request.GET.get('content', None)
31         if(content):
32             thread = get_object_or_404(Thread, pk=pk)
33             message = Message.objects.create(user=request.user, content=content)
34             thread.messages.add(message)
35             json_response['created'] = True
36
37             if(len(thread.messages.all()) is 1):
38                 json_response['first'] = True
39         else:
40             raise Http404('User is not authenticated')
41
42     return JsonResponse(json_response)
43
44 @login_required
45 def start_thread(request, username):
46     user = get_object_or_404(User, username=username)
47     thread = Thread.objects.find_or_create(request.user, user)
48     return redirect( reverse_lazy('messenger:detail', args=[thread.pk]) )
49

```

Ahora nos dirigimos a la **urls.py** de nuestro proyecto principal, es decir del webplayground, procedemos a registrar el path del messenger

```

webplayground > 📄 urls.py > {} settings
11     2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13     1. Import the include() function: from django.urls import include, path
14     2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16 from django.contrib import admin
17 from django.urls import path, include
18 from pages.urls import pages_patterns
19 from messenger.urls import messenger_patterns
20 from profiles.urls import profiles_patterns
21 from django.conf import settings
22
23 urlpatterns = [
24     path('',include('core.urls')),
25     path('pages/',include(pages_patterns)),
26     path('admin/', admin.site.urls),
27     path('messenger/', include(messenger_patterns)),

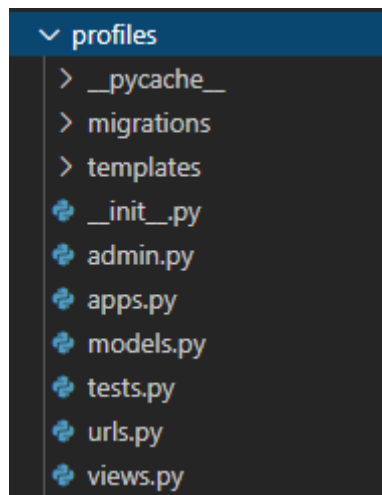
```

Bien hasta aquí ya tenemos la mensajería hecha, pero necesitamos de los users para poder usar esa mensajería, ya que el objetivo es que diferentes usuarios envíen mensajes entre sí.

Para esto, tenemos que crear otra app para los perfiles de los usuarios, y prácticamente los pasos son los mismos cuando se creó la app de messenger, creamos la app profiles:

```
(django) C:\Proyecto\webplayground>python manage.py startapp profiles
```

Se nos crean los archivos profiles:

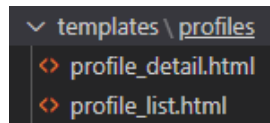


Nota: La carpeta templates y el archivo urls.py se crean manualmente más adelante.

Registramos la nueva app en el settings.py del proyecto principal:

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'ckeditor',  
    'core',  
    'messenger',  
    'profiles',  
]
```

Creamos una carpeta **templates/profiles** para guardar los archivos .html de nuestros perfiles



profile_detail.html

```
profiles > templates > profiles > <> profile_detail.html
1  {% extends 'core/base.html' %}
2  {% load static %}
3  {% block title %}{{profile.user}}{% endblock %}
4  {% block content %}
5  <main role="main">
6      <div class="container">
7          <div class="row mt-3">
8              <div class="col-md-9 mx-auto mb-5">
9                  <form action="" method="post" enctype="multipart/form-data">{% csrf_token %}
10                     <div class="row">
11                         <!-- Avatar -->
12                         <div class="col-md-2">
13                             {% if profile.avatar %}
14                             
15                             {% else %}
16                             
17                             {% endif %}
18                             {% if request.user != profile.user %}
19                             <a href="{% url 'messenger:start' profile.user.username %}" class="btn btn-primary btn-sm btn-block mt-3">
20                                 Enviar mensaje
21                             </a>
22                             {% endif %}
23                         </div>
24                         <!-- Campos -->
25                         <div class="col-md-10">
26                             <h3>{{profile.user}}</h3>
27                             {% if profile.bio %}<p>{{profile.bio}}</p>{% endif %}
28                             {% if profile.link %}<p><a href="{{profile.link}}" target="_blank">{{profile.link}}</a></p>{% endif %}
29                         </div>
30                     </div>
31                 </form>
32             </div>
33 </div>
```

profile_list.html

```
profiles > templates > profiles > <> profile_list.html
1  {% extends 'core/base.html' %}
2  {% load static %}
3  {% block title %}Perfiles{% endblock %}
4  {% block content %}
5  <style>.profile-avatar{float:left;width:4rem;height:4rem}.profile-data{padding-left:4.5rem;padding-top:.4rem}</style>
6  <main role="main">
7      <div class="container">
8          <div class="row mt-3">
9              <div class="col-md-9 mx-auto mb-5">
10                 <h2>Perfiles</h2>
11                 <div class="row">
12                     {% for profile in profile_list %}
13                     <div class="col-md-4 mt-2 mb-3">
14                         <div class="row p-1">
15                             <div class="col-md-12">
16                                 {% if profile.avatar %}
17                                 
18                                 {% else %}
19                                 
20                                 {% endif %}
21                                 <p class="profile-data">
22                                     <b>{{profile.user|truncatechars:"16"}}</b><br>
23                                     <a href="{% url 'profiles:detail' profile.user %}">Ver perfil</a>
24                                 </p>
25                             </div>
26                         </div>
27                     </div>
28                     {% endfor %}
29                 </div>
30             </div>
31         </div>
32     </div>
```

```

30     <!-- Menú de paginación -->
31     {% if is_paginated %}
32         <nav aria-label="Page navigation">
33             <ul class="pagination justify-content-center">
34                 {% if page_obj.has_previous %}
35                     <li class="page-item">
36                         <a class="page-link" href="?page={{ page_obj.previous_page_number }}">&laquo;</a>
37                     </li>
38                 {% else %}
39                     <li class="page-item disabled">
40                         <a class="page-link" href="#" tabindex="-1">&laquo;</a>
41                     </li>
42                 {% endif %}
43                 {% for i in paginator.page_range %}
44                     <li class="page-item {% if page_obj.number == i %}active{% endif %}">
45                         <a class="page-link" href="?page={{ i }}">{{ i }}</a>
46                     </li>
47                 {% endfor %}
48                 {% if page_obj.has_next %}
49                     <li class="page-item">
50                         <a class="page-link" href="?page={{ page_obj.next_page_number }}">&raquo;</a>
51                     </li>
52                 {% else %}
53                     <li class="page-item disabled">
54                         <a class="page-link" href="#" tabindex="-1">&raquo;</a>
55                     </li>
56                 {% endif %}
57             </ul>
58         </nav>
59     {% endif %}
60 </div>
61 </div>

```

Ahora nos dirigimos al archivo **views.py** de nuestra app profiles, para crear las vistas

```

profiles > views.py > ...
1  from django.shortcuts import render
2  from django.views.generic.detail import DetailView
3  from django.shortcuts import get_object_or_404
4  from django.views.generic.list import ListView
5  from registration.models import Profile
6
7  # Create your views here.
8  class ProfileListView(ListView):
9      model = Profile
10     template_name = 'profiles/profile_list.html'
11     paginate_by = 3
12
13     class ProfileDetailView(DetailView):
14         model = Profile
15         template_name = 'profiles/profile_detail.html'
16
17         def get_object(self):
18             return get_object_or_404(Profile, user__username=self.kwargs['username'])

```

Y luego hacemos el llamado de nuestras vistas a las urls.py de nuestra app profiles

```
profiles > urls.py > ...
2  from .views import ProfileListView, ProfileDetailView
3
4  profiles_patterns = [
5      path('', ProfileListView.as_view(), name="list"),
6      path('<username>/', ProfileDetailView.as_view(), name="detail"),
7  ], 'profiles']
```

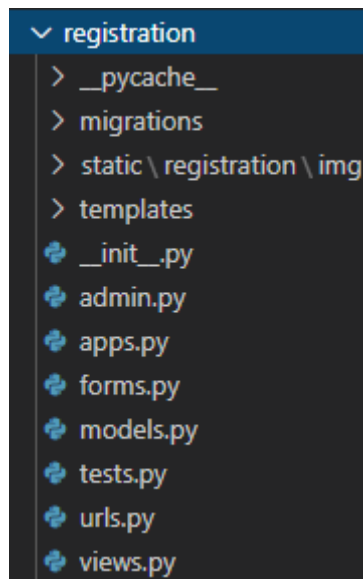
Ahora vamos a la **urls.py** de nuestro proyecto principal a registrar el path de profiles

```
webplayground > urls.py > ...
14      2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15      """
16  from django.contrib import admin
17  from django.urls import path, include
18  from pages.urls import pages_patterns
19  from messenger.urls import messenger_patterns
20  from profiles.urls import profiles_patterns
21  from django.conf import settings
22
23  urlpatterns = [
24      path('', include('core.urls')),
25      path('pages/', include(pages_patterns)),
26      path('admin/', admin.site.urls),
27      path('messenger/', include(messenger_patterns)),
28      path('profiles/', include(profiles_patterns)),
```

Pues bien, ahora ya tenemos nuestros perfiles para los usuarios, pero aún no tenemos un parámetro definido para registrarlos, es decir cuando un usuario quiera enviarle un mensaje a otro usuario, no podrá porque no está registrado en la base de datos, para ello entonces necesitamos registrar esos usuarios creando una app denominada registration.

```
(django) C:\Proyecto\webplayground>python manage.py startapp registration
(django) C:\Proyecto\webplayground>
```

Se nos crean los archivos de la nueva app



*Nota: La carpeta templates, static y los archivos forms.py
Y urls.py se crean manualmente más adelante.*

Registramos la nueva app registration en el settings.py de nuestro proyecto principal

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'ckeditor',  
    'core',  
    'messenger',  
    'profiles',  
    'registration',  
]
```

En esta app si necesitamos crear los modelos para el registro de los usuarios, para ello nos dirigimos al models.py de nuestra app registration y creamos los modelos que están indexados con una clase Profile

```
registration > models.py > custom_upload_to
1  from django.db.models.signals import post_save
2  from django.contrib.auth.models import User
3  from django.dispatch import receiver
4  from django.db import models
5
6  def custom_upload_to(instance, filename):
7      old_instance = Profile.objects.get(pk=instance.pk)
8      old_instance.avatar.delete()
9      return 'profiles/'+filename
10
11  # Create your models here.
12  class Profile(models.Model):
13      user = models.OneToOneField(User, on_delete=models.CASCADE)
14      avatar = models.ImageField(upload_to=custom_upload_to, null=True, blank=True)
15      bio = models.TextField(null=True, blank=True)
16      link = models.URLField(max_length=200, null=True, blank=True)
17
18      class Meta:
19          ordering = ['user__username']
20
21  @receiver(post_save, sender=User)
22  def ensure_profile_exists(sender, instance, **kwargs):
23      if kwargs.get('created', False):
24          Profile.objects.get_or_create(user=instance)
```

Una vez creados los modelos para la app registration, procedemos a hacer la migración

```
(django) C:\Proyecto\webplayground>python manage.py makemigrations registration
Migrations for 'registration':
  registration\migrations\0001_initial.py
  - Create model Profile
```

Y luego guardar los cambios, es decir migrarlos

```
(django) C:\Proyecto\webplayground>python manage.py migrate registration
Operations to perform:
  Apply all migrations: registration
Running migrations:
  Applying registration.0001_initial... OK
```


Ahora creamos una carpeta **templates/registration** donde alojaremos todos nuestros archivos .html para el registro y login

```
▼ templates \ registration
  <> login.html
  <> profile_email_form.html
  <> profile_form.html
  <> signup.html
```

profile_form.html

```
registration > templates > registration > <> profile_form.html
1  {% extends 'core/base.html' %}
2  {% load static %}
3  {% block title %}Perfil{% endblock %}
4  {% block content %}
5  <style>.errorlist{color:red;} label{display:none}</style>
6  <main role="main">
7      <div class="container">
8          <div class="row mt-3">
9              <div class="col-md-9 mx-auto mb-5">
10                 <form action="" method="post" enctype="multipart/form-data">{% csrf_token %}
11                     <div class="row">
12                         <!-- Previa del avatar -->
13                         <div class="col-md-2">
14                             {% if request.user.profile.avatar %}
15                                 
16                                 <p class="mt-1">¿Borrar? <input type="checkbox" id="avatar-clear" name="avatar-clear" /></p>
17                             {% else %}
18                                 <img src='{% static "registration/img/no-avatar.jpg" %}' class="img-fluid">
19                             {% endif %}
20                         </div>
21                         <!-- Formulario -->
22                         <div class="col-md-10">
23                             <h3>Perfil</h3>
24                             <input type="file" name="avatar" class="form-control-file mt-3" id="id_avatar">
25                             {{ form.bio }}
26                             {{ form.link }}
27                             <input type="email" value="{{ request.user.email }}" class="form-control mt-3" readonly>
28                             <p class="mt-3">
29                                 Si deseas editar tu email haz click <a href="{% url 'profile_email' %}">aquí.</a> <br>
30                                 Y si quieres cambiar tu contraseña haz click <a href="{% url 'password_change' %}">aquí.</a>
31                             </p>
32                             <input type="submit" class="btn btn-primary btn-block mt-3" value="Actualizar">
33                         </div>
```

profile_email_form.html

```
registration > templates > registration > <> profile_email_form.html
1  {% extends 'core/base.html' %}
2  {% load static %}
3  {% block title %}Email{% endblock %}
4  {% block content %}
5  <style>.errorlist{color:red;} label{display: none;}</style>
6  <main role="main">
7      <div class="container">
8          <div class="row mt-3">
9              <div class="col-md-9 mx-auto mb-5">
10                 <form action="" method="post">{% csrf_token %}
11                     <h3 class="mb-4">Email</h3>
12                     {{ form.as_p }}
13                     <p><input type="submit" class="btn btn-primary btn-block" value="Actualizar"></p>
14                 </form>
15             </div>
16         </div>
17     </div>
18 </main>
19 {% endblock %}
```

Signup.html

```
registration > templates > registration > <> signup.html
1  {% extends 'core/base.html' %}
2  {% load static %}
3  {% block title %}Registro{% endblock %}
4  {% block content %}
5  <style>.errorlist{color:red;} label{display: none;}</style>
6  <main role="main">
7      <div class="container">
8          <div class="row mt-3">
9              <div class="col-md-9 mx-auto mb-5">
10                 <form action="" method="post">{% csrf_token %}
11                     <h3 class="mb-4">Registro</h3>
12                     {{ form.as_p }}
13                     <p><input type="submit" class="btn btn-primary btn-block" value="Confirmar"></p>
14                 </form>
15             </div>
16         </div>
17     </div>
18 </main>
19 {% endblock %}
```

login.html

```
registration > templates > registration > login.html
1  {% extends 'core/base.html' %}
2  {% load static %}
3  {% block title %}Iniciar sesión{% endblock %}
4  {% block content %}
5  <style>.errorlist{color:red;}</style>
6  <main role="main">
7      <div class="container">
8          <div class="row mt-3">
9              <div class="col-md-9 mx-auto mb-5">
10                 {% if 'register' in request.GET %}
11                     <p style="color:green;">
12                         Usuario registrado correctamente, ya puedes identificarte.
13                     </p>
14                 {% endif %}
15                 <form action="" method="post">{% csrf_token %}
16                     <h3 class="mb-4">Iniciar sesión</h3>
17                     {% if form.non_field_errors %}
18                         <p style="color:red">Usuario o contraseña incorrectos, prueba de nuevo.</p>
19                     {% endif %}
20                     <p>
21                         <input type="text" name="username" autofocus maxlength="254" required
22                         id="id_username" class="form-control" placeholder="Nombre de usuario"/>
23                     </p>
24                     <p>
25                         <input type="password" name="password" required
26                         id="id_password" class="form-control" placeholder="Contraseña"/>
27                     </p>
28                     <p><input type="submit" class="btn btn-primary btn-block" value="Acceder"></p>
29                 </form>
30                 <p>
31                     ¿Ha olvidado su clave? puede restaurarla <a href="{% url 'password_reset' %}">aquí.</a>
32                 </p>
33             </div>
```

Ahora nos dirigimos al **views.py** de nuestra app a crear las vistas de nuestro register

```
registration > views.py > ...
1  from .forms import UserCreationFormWithEmail, ProfileForm, EmailForm
2  from django.views.generic import CreateView
3  from django.views.generic.edit import UpdateView
4  from django.utils.decorators import method_decorator
5  from django.contrib.auth.decorators import login_required
6  from django.urls import reverse_lazy
7  from django import forms
8  from .models import Profile
9
10 # Create your views here.
11 class SignUpView(CreateView):
12     form_class = UserCreationFormWithEmail
13     template_name = 'registration/signup.html'
14
15     def get_success_url(self):
16         return reverse_lazy('login') + '?register'
17
18     def get_form(self, form_class=None):
19         form = super(SignUpView, self).get_form()
20         # Modificar en tiempo real
21         form.fields['username'].widget = forms.TextInput(attrs={'class':'form-control mb-2', 'placeholder':'Nombre de U
22         form.fields['email'].widget = forms.EmailInput(attrs={'class':'form-control mb-2', 'placeholder':'Dirección ema
23         form.fields['password1'].widget = forms.PasswordInput(attrs={'class':'form-control mb-2', 'placeholder':'Contra
24         form.fields['password2'].widget = forms.PasswordInput(attrs={'class':'form-control mb-2', 'placeholder':'Repite
25
26         return form
```

Aquí mismo en views.py de la app registration creamos clases para editar los perfiles en caso de que el usuario desee modificarlos, editar un correo, siempre y cuando el usuario esté logeado, para eso se hace un método que hace el llamado a esa función de logeado.

```
28 @method_decorator(login_required, name='dispatch')
29 class ProfileUpdate(UpdateView):
30     form_class = ProfileForm
31     # model = Profile
32     # fields = ['avatar', 'bio', 'link']
33     success_url = reverse_lazy('profile')
34     template_name = 'registration/profile_form.html'
35
36     def get_object(self):
37         profile, created = Profile.objects.get_or_create(user=self.request.user)
38         return profile
39
40 @method_decorator(login_required, name='dispatch')
41 class EmailUpdate(UpdateView):
42     form_class = EmailForm
43     # model = Profile
44     # fields = ['avatar', 'bio', 'link']
45     success_url = reverse_lazy('profile')
46     template_name = 'registration/profile_email_form.html'
47
48     def get_object(self):
49         return self.request.user
50
51     def get_form(self, form_class=None):
52         form = super(EmailUpdate, self).get_form()
53         # Modificar en tiempo real
54         form.fields['email'].widget = forms.EmailInput(attrs={'class': 'form-control mb-2', 'placeholder': 'Dirección e
55
56         return form
```

En urls.py de la app registration hacemos el llamado de nuestros modelos de clases creadas anteriormente, cabe recalcar que este archivo **urls.py** nosotros mismos lo debemos crear manualmente dentro de la app registration al igual que lo hicimos en las anteriores apps.

```
registration > urls.py > ...
1 from django.urls import path
2 from .views import SignUpView, ProfileUpdate, EmailUpdate
3
4 urlpatterns = [
5     path('signup/', SignUpView.as_view(), name="signup"),
6     path('profile/', ProfileUpdate.as_view(), name="profile"),
7     path('profile/email/', EmailUpdate.as_view(), name="profile_email"),
8
9 ]
```

Dentro de la app registration creamos un nuevo archivo llamado **forms.py** donde haremos el llamado de los modelos creados para el usuario y registro, es aquí donde se hacen las validaciones

```
registration > forms.py > ...
1  from django.contrib.auth.forms import UserCreationForm
2  from django.contrib.auth.models import User
3  from .models import Profile
4  from django import forms
5
6  class UserCreationFormWithEmail(UserCreationForm):
7      email = forms.EmailField(required=True, help_text='Requerido, 254 caracteres como máximo y debe ser válido')
8
9      class Meta:
10         model = User
11         fields = ('username', 'email', 'password1', 'password2')
12
13
14     def clean_email(self):
15         email = self.cleaned_data.get('email')
16
17         if User.objects.filter(email=email).exists():
18             raise forms.ValidationError('El email ya está registrado, prueba con otro.')
19         return email
20
21 class ProfileForm(forms.ModelForm):
22     class Meta:
23         model = Profile
24         fields = ['avatar', 'bio', 'link']
25         widgets = {
26             'avatar': forms.ClearableFileInput(attrs={'class': 'form-control-file mt-3'}),
27             'bio': forms.Textarea(attrs={'class': 'form-control mt-3', 'rows': 3, 'placeholder': 'Biografía'}),
28             'link': forms.URLInput(attrs={'class': 'form-control mt-3', 'placeholder': 'Enlace'}),
29         }
```

```
31 class EmailForm(forms.ModelForm):
32     email = forms.EmailField(required=True, help_text='Requerido, 254 caracteres como máximo y debe ser válido')
33
34     class Meta:
35         model = User
36         fields = ['email']
37
38
39     def clean_email(self):
40         email = self.cleaned_data.get('email')
41
42         if 'email' in self.changed_data:
43             if User.objects.filter(email=email).exists():
44                 raise forms.ValidationError('El email ya está registrado, prueba con otro.')
45         return email
```

Creamos otro nuevo archivo dentro de nuestra app registration llamado **test.py** donde se testearán cada perfil que un usuario cree.

```
registration > tests.py > ...
1  from django.test import TestCase
2  from .models import Profile
3  from django.contrib.auth.models import User
4
5  # Create your tests here.
6  class ProfileTestCase(TestCase):
7      def setUp(self):
8          User.objects.create_user('test', 'test@test.com', 'test1234')
9
10     def test_profile_exists(self):
11         exists = Profile.objects.filter(user__username='test').exists()
12         self.assertEqual(exists, True)
```

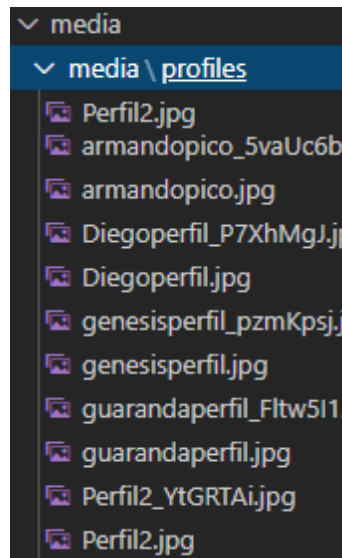
Ahora nos vamos a la urls.py de nuestro proyecto principal para hacer el llamado al path de nuestra app registration

```
16 from django.contrib import admin
17 from django.urls import path, include
18 from pages.urls import pages_patterns
19 from messenger.urls import messenger_patterns
20 from profiles.urls import profiles_patterns
21 from django.conf import settings
22
23 urlpatterns = [
24     path('', include('core.urls')),
25     path('pages/', include(pages_patterns)),
26     path('admin/', admin.site.urls),
27     path('messenger/', include(messenger_patterns)),
28     path('profiles/', include(profiles_patterns)),
29     path('accounts/', include('django.contrib.auth.urls')),
30     path('accounts/', include('registration.urls')),
31 ]
```

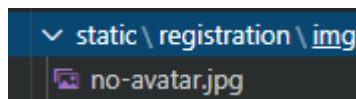
Bien ya para la parte final solo tenemos que modificar nuestro base.html para los nuevos campos de las apps que creamos anteriormente, como los Perfiles, Mensajes, Perfil y Salir

```
core > templates > core > <> base.html
18 <!-- Navegación -->
19 <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
20     <div class="container">
21         <a class="navbar-brand" href="{% url 'home' %}">Playground</a>
22         <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#ned" aria-controls="ned" ar
23             <span class="navbar-toggler-icon"></span>
24         </button>
25         <div class="collapse navbar-collapse" id="ned">
26             <ul class="navbar-nav mr-auto">
27                 <li class="nav-item">
28                     <a class="nav-link" href="{% url 'home' %}">Inicio</a>
29                 </li>
30                 <li class="nav-item">
31                     <a class="nav-link" href="{% url 'pages:pages' %}">Página</a>
32                 </li>
33                 <li class="nav-item">
34                     <a class="nav-link" href="{% url 'profiles:list' %}">Perfiles</a>
35                 </li>
36             </ul>
37             <ul class="navbar-nav">
38                 {% if request.user.is_authenticated %}
39                 <li class="nav-item">
40                     <a class="nav-link" href="{% url 'messenger:list' %}">Mensajes</a>
41                 </li>
42                 <li class="nav-item">
43                     <a class="nav-link" href="{% url 'profile' %}">Perfil</a>
44                 </li>
45                 <li class="nav-item">
46                     <a class="nav-link" href="{% url 'logout' %}">Salir</a>
47                 </li>
48             </ul>
49         </div>
50     </div>
51 </nav>
```

Creamos una carpeta afuera de nuestro proyecto que aloje imágenes para nuestros perfiles de usuario



Y en nuestra app registration creamos una carpeta static que aloje una imagen tipo usuario por defecto que se visualizará para cuando un nuevo user se registre



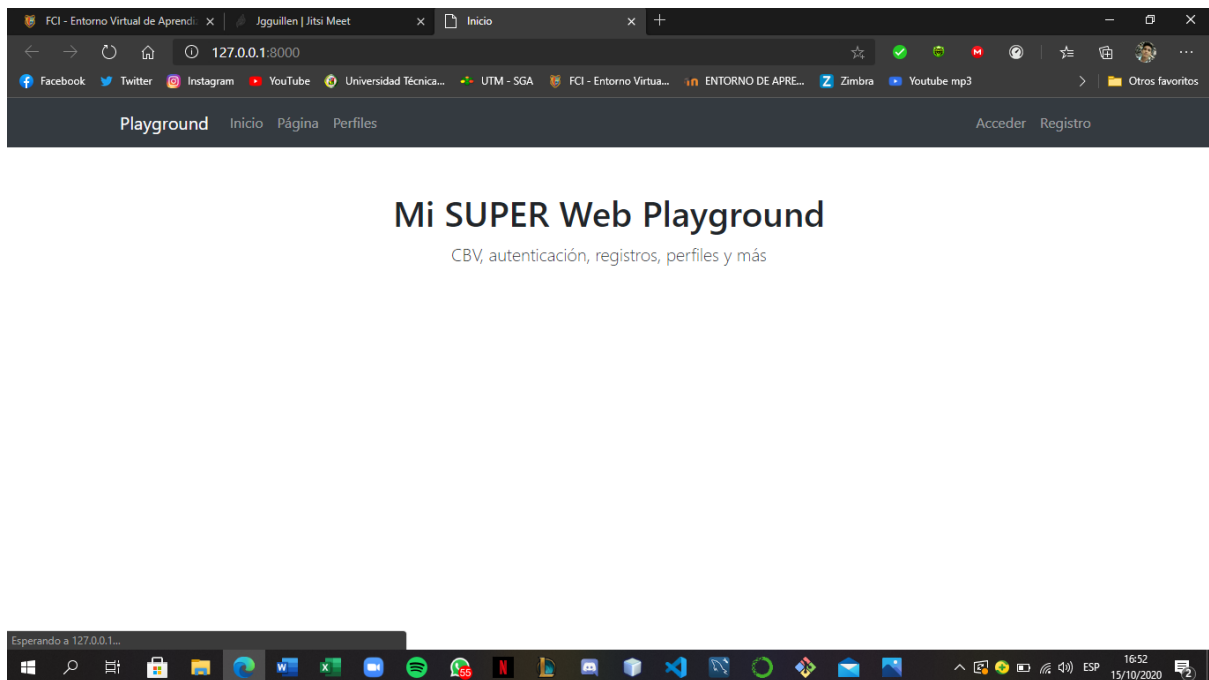
Luis

Bien ahora vamos a ejecutar el proyecto! Desde el terminal de Python escribimos el siguiente comando

```
(django) C:\Proyecto\webplayground>python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
October 15, 2020 - 16:50:01
Django version 3.0.7, using settings 'webplayground.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Vista principal



Campos del Registro

[Playground](#) [Inicio](#) [Página](#) [Perfiles](#) [Acceder](#) [Registro](#)

Registro

Requerido. 150 caracteres como máximo. Únicamente letras, dígitos y @/./+/-/_

Requerido, 254 caracteres como máximo y debe ser válido

- Su contraseña no puede asemejarse tanto a su otra información personal.
- Su contraseña debe contener al menos 8 caracteres.
- Su contraseña no puede ser una clave utilizada comúnmente.
- Su contraseña no puede ser completamente numérica.

Para verificar, introduzca la misma contraseña anterior.

Confirmar

Campos del Login

Playground

Inicio

Página

Perfiles

Acceder

Registro

Iniciar sesión

Nombre de usuario

Contraseña

Acceder

¿Ha olvidado su clave? puede restaurarla [aquí](#).

Iniciamos una sección

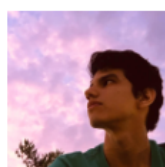
Iniciar sesión

Luis

.....

Acceder

Aquí ya nos accede a los campos de profiles



¿Borrar? ☐

Perfil

Elegir archivo

 No se eligió ningún archivo

Estudiante de informática, apasionado por el encebollado con doble porción de chifle <3

<https://www.facebook.com/LuisCarranza.404/>

fernanduiz987@gmail.com

Si deseas editar tu email haz click [aquí](#).

Y si quieres cambiar tu contraseña haz click [aquí](#).

Actualizar

Aquí tenemos ya los diferentes perfiles en una lista

Perfiles



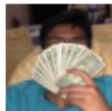
Armando
[Ver perfil](#)



Diego
[Ver perfil](#)



Genesis
[Ver perfil](#)

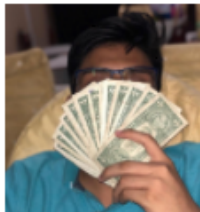


Guaranda
[Ver perfil](#)



Luis
[Ver perfil](#)

Revisión de cada perfil junto con la opción ya de poder enviarle un mensaje



Guaranda

Real GT For Life Baby Todo lo puedo en Cristo que me fortalece <3

<https://www.facebook.com/jonathan.guaranda.18>

Enviar mensaje

Como aquí podemos apreciar ya está la mensajería siendo en uso



Armando
Hace 1 día, 18 horas



Genesis
Hace 1 día, 22 horas



Diego
Hace 1 día, 22 horas



Guaranda
Hace 2 días, 1 hora

Mensajes con Guaranda

Hace 2 días, 1 hora

Hola Luis! Un saludo fraterno, te aviso que el día viernes vamos a tener una parrillada en mi casa, quedas invitado mi prro. Saludos y Bendiciones <3

Hace 2 días, 1 hora

Hola compa, que tal? chévere loco ahi estaré entonces, nos vemos.

Escribe tu mensaje aquí

Enviar mensaje

También hay la opción entrar por Mensajes para iniciar una nueva conversación con cualquiera de los usuarios que están registrados

Playground

Inicio


Página

Perfiles

Mensajes


Perfil

Salir




Armando

Hace 1 día, 18 horas




Genesis

Hace 1 día, 22 horas



Diego


Hace 1 día, 22 horas



Guaranda


Hace 2 días, 1 hora

Selecciona un hilo de conversación de tu panel izquierdo.




Armando

Hace 1 día, 18 horas




Genesis

Hace 1 día, 22 horas



Diego

Hace 1 día, 22 horas



Guaranda

Hace 2 días, 1 hora

Mensajes con Armando

Hace 1 día, 18 horas


Hola Armando! Oye cuando me regresas los \$20 que te presté??
No te me hagas el loco

Hace 1 día, 18 horas

No tengo dinero...


Escribe tu mensaje aquí

Enviar mensaje




Armando

Hace 1 día, 18 horas




Genesis

Hace 1 día, 22 horas



Diego

Hace 1 día, 22 horas



Guaranda

Hace 2 días, 1 hora

Mensajes con Genesis

Hace 1 día, 22 horas

Hola ñaño, como estás, oe adivina quien me volvió a escribir?
xD

Hace 1 día, 22 horas

Hola ñañita :3 Bien y tú? Quien te escribió? :0

Hace 1 día, 22 horas

El Joao xd

Hace 1 día, 22 horas

Me dijo que quiere volver conmigo

Escribe tu mensaje aquí

Enviar mensaje

Vamos a probar logeando un usuario ya registrado

Iniciar sesión

Genesis

.....



Acceder

¿Ha olvidado su clave? puede restaurarla [aquí](#).

Podemos borrar la foto y subir una nueva, siempre y cuando esté dentro de la carpeta media
Así mismo actualizar nuestros datos.



¿Borrar? ☐

Perfil

Elegir archivo

No se eligió ningún archivo

Soy madre luchona, Bendecida con mi pequeño <3

<https://www.facebook.com/genesis.velez.58118>

auxiliadoraxd@gmail.com

Si deseas editar tu email haz click [aquí](#).

Y si quieres cambiar tu contraseña haz click [aquí](#).

Actualizar

Aquí observamos que Génesis solo tuvo conversación con dos usuarios, al contrario del usuario Luis que tuvo conversaciones con todos los usuarios registrados



Luis

Hace 1 día, 22 horas



Diego

Hace 1 día, 22 horas

Selecciona un hilo de conversación de tu panel izquierdo.

Como observamos son los mismos mensajes que tuvo con el usuario Luis, lo que significa que funciona la mensajería



Conclusiones

- Para el desarrollo de este proyecto, se tuvieron que crear adicional dos más apps nuevas para la app messenger ya que teníamos que tener usuarios que se registren para luego logearse y poder intercambiar mensajes entre si.
- Se hizo más interactiva las vistas para los perfiles de usuario y la mensajería.
- Se trabajó como plus adicional con las vistas CRUD para la modificación y eliminación de páginas, no fue requisito para este proyecto pero se lo hizo como nuevo aprendizaje.