

Informe de Laboratorio 10

Tema: Proyecto Final

Nota

Estudiante(s)	Escuela	Asignatura
Christian Raul Mestas Zegarra cmestasz@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Programación Web 2 Semestre: I Código: 1702122

Laboratorio	Tema	Duración
10	Proyecto Final	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 24 Junio 2024	Al 06 Julio 2024

Laboratorio 10

1	Tarea	3
2	Equipos y materiales utilizados	4
3	Desarrollo	5
3.1	Backend en Django	5
3.2	Frontend en Angular	16
3.2.1	Aplicación base	16
3.2.2	dynamic-form	17
3.2.3	home	20
3.2.4	login	25
3.2.5	register	26
3.2.6	services	28
3.2.7	models	29
3.3	Demostración	30
3.4	Lista de commits	31
4	URL del repositorio en GitHub	35
5	Enlace al video	35
6	Estructura de laboratorio 10	35
7	Referencias	36

1 Tarea

- **Enunciado:**

- Este trabajo debe ser hecho en grupos de 3 a 4. Escoger una empresa, escoger el tema a trabajar y definir los Requerimientos funcionales.
- El trabajo en equipo, debe ser hecho de modo que todos los participantes del grupo conozcan todo el código.
 - * La programación debe ser hecha en pares, es decir que durante una videollamada, dos o más integrantes (al menos dos, pueden ser todos los integrantes del grupo) programen un mismo código en un repositorio central, turnándose la labor de observador y codificar a intervalos de tiempo equitativos.
 - * No se debe dividir el trabajo en partes y que los integrantes hagan su parte sin que los demás conozcan de su trabajo o él conozca y participe en el trabajo de los demás.
 - * Se deben crear ramas en sus proyectos de git para cada una de las funcionalidades, una vez alcanzada con éxito la funcionalidad, esta se debe integrar a la rama principal.
- La aplicación será calificada sobre 20 y deberá incluir:
 - * URLs propios, usando reverse (2 puntos)
 - * Plantillas propias de la aplicación (1 puntos)
 - * Que usen widgets de manera elegante (1 puntos)
 - * Vistas de Listado, Detalle, Crear, Actualizar y Borrar (4 puntos)
 - * Formulario con restricciones de seguridad adicionales (2 puntos)
 - * Vista de consultas que devuelva Json (3 puntos)
 - * Programa cliente para hacer y consumir las consultas con Ajax (2 puntos)
 - * Programa cliente para hacer y consumir las consultas con Framework de JavaScript (3 puntos)
 - * Al menos dos modelos (2 puntos)
 - * Modelo con clave externa: foreign key (+2 puntos, opcional)
 - * CSS o Bootstrap (2 puntos)
 - * Publicó su aplicación en el web (+3 puntos, opcional)
 - * Descargar un informe como archivos pdf (+2 puntos, opcional)
 - * Enviar correo (+2 puntos, opcional)
- Trabajo en equipo, los 20 puntos serán distribuidos de manera equitativa entre los siguientes criterios
 - * Responsabilidad: cumplió puntualmente con todo
 - * Proactividad: hizo más de lo que se pidió
 - * Aporte al grupo
 - * Calificó a sus compañeros (Si no lo hizo 50% de la nota de trabajo individual)
- La calificación final será un promedio entre la nota individual 40% y grupal 60%
 - * La nota grupal será por el producto calificado por el profesor, en base a sus avances, exposición e informe.
 - * La nota individual será calificada por sus propios compañeros. Los que no califiquen a sus compañeros tendrán la mitad de la nota en este componente.

2 Equipos y materiales utilizados

- Cuenta en GitHub con el correo institucional.
- Sistema Operativo Microsoft Windows 10
- Visual Studio Code
- Git
- Windows PowerShell
- Navegador Mozilla Firefox
- Angular CLI
- Django

3 Desarrollo

3.1 Backend en Django

models.py

```
1  from django.db import models
2  from django.contrib.auth.models import AbstractUser, BaseUserManager
3
4  # Create your models here.
5
6
7  class UsuarioManager(BaseUserManager):
8      def create_user(self, email, telefono, password):
9          if not email:
10             raise ValueError('Ingresa un correo electrónico')
11          if not telefono:
12             raise ValueError('Ingresa un número de teléfono')
13          if not password:
14             raise ValueError('Ingresa una contraseña')
15          n_email = self.normalize_email(email)
16          usuario = self.model(
17              username=n_email,
18              telefono=telefono,
19              email=n_email
20          )
21          usuario.set_password(password)
22          usuario.save(using=self._db)
23          return usuario
24
25      def create_superuser(self, email, password=None):
26          usuario = self.create_user(email, '00000000', password)
27          usuario.is_admin = True
28          usuario.is_staff = True
29          usuario.is_superuser = True
30          usuario.email = email
31          usuario.tipo = Usuario.Types.ADMIN
32          usuario.save(using=self._db)
33          return usuario
34
35
36  class Usuario(AbstractUser):
37      class Types(models.TextChoices):
38          USUARIO = 'US', 'Usuario'
39          DUEÑO = 'DU', 'Dueño'
40          ADMIN = 'AD', 'Administrador'
41
42      email = models.EmailField(unique=True)
43      telefono = models.CharField(max_length=9)
44      yape_qr = models.ImageField(upload_to='yape_qrs/', blank=True, null=True)
45      tipo = models.CharField(
46          max_length=2, choices=Types.choices, default=Types.USUARIO)
47
48      objects = UsuarioManager()
49
50      REQUIRED_FIELDS = []
51      USERNAME_FIELD = 'email'
```

```
52
53
54 class Tienda(models.Model):
55     nombre = models.CharField(max_length=100)
56     descripcion = models.TextField()
57     categoria = models.CharField(max_length=100)
58     dueño = models.ForeignKey(Usuario, on_delete=models.CASCADE, limit_choices_to={
59         'tipo': Usuario.Types.DUEÑO})
60     latitud = models.DecimalField(max_digits=9, decimal_places=6)
61     longitud = models.DecimalField(max_digits=9, decimal_places=6)
62
63
64 class Producto(models.Model):
65     nombre = models.CharField(max_length=100)
66     descripcion = models.TextField()
67     precio = models.DecimalField(max_digits=6, decimal_places=2)
68     imagen = models.ImageField(upload_to='productos/', blank=True, null=True)
69     tienda = models.ForeignKey(Tienda, on_delete=models.CASCADE)
70     stock = models.PositiveIntegerField()
71
72
73 class Venta(models.Model):
74     usuario = models.ForeignKey(Usuario, on_delete=models.CASCADE, limit_choices_to={
75         'tipo': Usuario.Types.USUARIO})
76     producto = models.ForeignKey(Producto, on_delete=models.CASCADE)
77     fecha = models.DateTimeField(auto_now_add=True)
78     cantidad = models.PositiveIntegerField()
```

- Se crean los modelos **Usuario**, **Tienda**, **Producto** y **Venta**.
- Se crea la clase **UsuarioManager** para poder usar la autenticación de Django.

forms.py

```
1 from django import forms
2 from django.forms import ModelForm
3 from .models import Usuario, Tienda, Producto, Venta
4
5 class LoginForm(forms.Form):
6     username = forms.CharField(label='Nombre de usuario')
7     password = forms.CharField(label='Contraseña', widget=forms.PasswordInput)
8
9 class LogoutForm(forms.Form):
10     pass
11
12 class UsuarioForm(ModelForm):
13     password1 = forms.CharField(label='Contraseña', widget=forms.PasswordInput)
14     password2 = forms.CharField(label='Confirmar contraseña', widget=forms.PasswordInput)
15
16     class Meta:
17         model = Usuario
18         fields = ['email', 'telefono', 'password1', 'password2']
19
20     def clean_telefono(self):
21         telefono = self.cleaned_data.get('telefono')
22         if not telefono:
```

```
23         raise forms.ValidationError('Ingrese un número de teléfono')
24     if len(telefono) != 9:
25         raise forms.ValidationError('Ingrese un número de teléfono válido')
26     return telefono
27
28     def clean_password2(self):
29         password1 = self.cleaned_data.get('password1')
30         password2 = self.cleaned_data.get('password2')
31         if password1 and password2 and password1 != password2:
32             raise forms.ValidationError('Las contraseñas no coinciden')
33         return password2
34
35     def save(self, commit=True):
36         user = super().save(commit=False)
37         user.username = self.cleaned_data['email']
38         user.set_password(self.cleaned_data['password1'])
39         if commit:
40             user.save()
41         return user
42
43     class DueñoForm(UsuarioForm):
44         class Meta:
45             model = Usuario
46             fields = ['email', 'telefono', 'password1', 'password2', 'yape_qr']
47
48         def save(self, commit=True):
49             user = super().save(commit=False)
50             user.tipo = Usuario.Types.DUEÑO
51             if commit:
52                 user.save()
53             return user
54
55         def clean_yape_qr(self):
56             yape_qr = self.cleaned_data.get('yape_qr')
57             if not yape_qr:
58                 instance = self.instance
59                 if instance.pk:
60                     yape_qr = instance.yape_qr
61             return yape_qr
62
63     class AdminForm(UsuarioForm):
64         class Meta:
65             model = Usuario
66             fields = ['username', 'telefono', 'password1', 'password2']
67
68         def save(self, commit=True):
69             user = super().save(commit=False)
70             user.tipo = Usuario.Types.ADMIN
71             if commit:
72                 user.save()
73             return user
74
75     class TiendaForm(ModelForm):
76         class Meta:
77             model = Tienda
78             fields = ['nombre', 'descripcion', 'categoria', 'latitud', 'longitud']
```

```
79
80 class ProductoForm(ModelForm):
81     class Meta:
82         model = Producto
83         fields = '__all__'
84
85
86 class VentaForm(ModelForm):
87     class Meta:
88         model = Venta
89         fields = ['producto', 'cantidad']
```

- Se crean los formularios **UsuarioForm**, **DueñoForm**, **AdminForm**, **TiendaForm**, **ProductoForm** y **VentaForm**.
- Además se crean los formularios **LoginForm** y **LogoutForm** para el inicio y cierre de sesión.
- Los formularios tienen validaciones personalizadas para los campos.

serializers.py

```
1 from django import forms
2 from django.forms import ModelForm
3 from django.core import validators
4 from rest_framework import serializers, viewsets
5 from .models import Usuario, Tienda, Producto, Venta
6
7
8 class UsuarioSerializer(serializers.ModelSerializer):
9     class Meta:
10         model = Usuario
11         fields = ['id', 'username', 'password', 'telefono', 'tipo', 'yape_qr']
12
13
14 class TiendaSerializer(serializers.ModelSerializer):
15     class Meta:
16         model = Tienda
17         fields = ['id', 'nombre', 'descripcion',
18                 'categoria', 'dueño', 'latitud', 'longitud']
19
20
21 class ProductoSerializer(serializers.ModelSerializer):
22     class Meta:
23         model = Producto
24         fields = ['id', 'nombre', 'descripcion',
25                 'precio', 'imagen', 'tienda', 'stock']
26
27
28 class VentaSerializer(serializers.ModelSerializer):
29     class Meta:
30         model = Venta
31         fields = ['id', 'usuario', 'producto', 'fecha', 'cantidad']
32
33
34 def form_serializer(form: ModelForm):
35     fields = []
```



```
36     for name, field in form.fields.items():
37         field_data = {
38             'tipoCampo': get_tipo_campo(field),
39             'name': name,
40             'label': field.label or name.capitalize(),
41             'validators': get_validators(field),
42             'attributes': get_attributes(field),
43         }
44         if field_data['tipoCampo'] == 'select':
45             field_data['options'] = get_options(field)
46         fields.append(field_data)
47     return fields
48
49
50 def get_tipo_campo(field):
51     if isinstance(field, forms.FileField) or isinstance(field, forms.ImageField):
52         return 'input'
53     if isinstance(field, forms.ChoiceField):
54         return 'select'
55     if isinstance(field, forms.Textarea):
56         return 'textarea'
57     return 'input'
58
59
60 def get_validators(field):
61     validators_dict = {}
62     for validator in field.validators:
63         if isinstance(validator, validators.MaxLengthValidator):
64             validators_dict['maxlength'] = str(validator.limit_value)
65         elif isinstance(validator, validators.MinLengthValidator):
66             validators_dict['minlength'] = str(validator.limit_value)
67         elif isinstance(validator, validators.EmailValidator):
68             validators_dict['email'] = 'true'
69
70     if field.required:
71         validators_dict['required'] = 'true'
72
73     if isinstance(field, forms.DecimalField):
74         validators_dict['min'] = str(
75             field.min_value) if field.min_value is not None else ''
76         validators_dict['max'] = str(
77             field.max_value) if field.max_value is not None else ''
78         validators_dict['decimal'] = 'true'
79
80     if isinstance(field, forms.IntegerField):
81         validators_dict['min'] = str(
82             field.min_value) if field.min_value is not None else ''
83         validators_dict['max'] = str(
84             field.max_value) if field.max_value is not None else ''
85
86     return validators_dict
87
88
89 def get_attributes(field):
90     attributes = {}
91     if hasattr(field.widget, 'input_type'):
```

```
92         attributes['type'] = field.widget.input_type
93
94     if isinstance(field, forms.FileField) or isinstance(field, forms.ImageField):
95         attributes['type'] = 'file'
96         if isinstance(field, forms.ImageField):
97             attributes['accept'] = 'image/*'
98
99     if isinstance(field, forms.DateTimeField):
100         attributes['type'] = 'datetime-local'
101
102     if isinstance(field, forms.DecimalField):
103         attributes['step'] = str(10 ** -field.decimal_places)
104
105     return attributes
106
107
108 def get_options(field):
109     if isinstance(field, forms.ModelChoiceField):
110         return [{'label': str(obj), 'value': obj.pk} for obj in field.queryset]
111     elif hasattr(field, 'choices'):
112         return [{'label': label, 'value': serialize_value(value)} for value, label in
113                 ↪ field.choices]
114     return []
115
116 def serialize_value(value):
117     if hasattr(value, 'pk'):
118         return value.pk
119     elif isinstance(value, (list, tuple)) and len(value) == 2:
120         return value[0]
121     return value
```

- Se usan los serializadores de Django Rest Framework para los modelos **Usuario, Dueño, Admin, Tienda, Producto y Venta**.
- Adicionalmente, se crea un serializador de formularios para poder enviar los datos de los formularios por api.

views.py

```
1  from django.shortcuts import render
2  from django.urls import reverse
3  from django.http import JsonResponse
4  from .forms import UsuarioForm, DueñoForm, TiendaForm, ProductoForm, VentaForm, LoginForm,
   ↪ LogoutForm
5  from .models import Tienda, Producto, Venta, Usuario
6  from .serializers import UsuarioSerializer, TiendaSerializer, ProductoSerializer,
   ↪ VentaSerializer, form_serializer
7  from django.contrib.auth import authenticate, login, logout
8  from django.middleware.csrf import get_token
9  from rest_framework.views import APIView
10 import requests
11 # Create your views here.
12
13 MESSAGES = {
14     'correct': {'status': 200, 'message': 'Correcto'},
```

```
15     'created': {'status': 201, 'message': 'Creado'},
16     'no_login': {'status': 401, 'message': 'No permitido (inicia sesión para continuar)'},
17     'unallowed': {'status': 401, 'message': 'No permitido'},
18     'fields_error': {'status': 406, 'message': 'Error en los campos'},
19     'wrong_password': {'status': 406, 'message': 'Contraseña incorrecta'}
20 }
21
22
23 def verify_login(request):
24     if request.user.is_authenticated:
25         return True
26     return False
27
28
29 def process_create_form(self, request, form):
30     if form.is_valid():
31         form.save()
32         return JsonResponse(MESSAGES['created'])
33     return JsonResponse(MESSAGES['fields_error'])
34
35
36 def send_create_form(self, request, form):
37     response = JsonResponse({
38         'status': 200,
39         'campos': form_serializer(form)
40     })
41     return response
42
43 def is_user(request):
44     return request.user.tipo == 'US'
45
46 def is_owner(request):
47     return request.user.tipo == 'DU'
48
49 def is_admin(request):
50     return request.user.tipo == 'AD'
51
52
53 # TODO :admin forms
54
55
56
57 class IniciarSesion(APIView):
58     def post(self, request, format=None):
59         form = LoginForm(request.data)
60         if form.is_valid():
61             user = authenticate(
62                 request, username=form.cleaned_data['username'],
63                 ↪ password=form.cleaned_data['password'])
64             if user:
65                 login(request, user)
66                 get_token(request)
67                 return JsonResponse(MESSAGES['correct'])
68             return JsonResponse(MESSAGES['wrong_password'])
69         return JsonResponse(MESSAGES['fields_error'])
```

```
70     def get(self, request, format=None):
71         response = JsonResponse({
72             'status': 200,
73             'campos': form_serializer(LoginForm())
74         })
75         return response
76
77
78 class CerrarSesion(APIView):
79     def post(self, request, format=None):
80         if verify_login(self, request):
81             logout(request)
82             return JsonResponse(MESSAGES['correct'])
83         return JsonResponse(MESSAGES['no_login'])
84
85     def get(self, request, format=None):
86         if verify_login(self, request):
87             response = JsonResponse({
88                 'status': 200,
89                 'campos': form_serializer(LogoutForm())
90             })
91             return response
92         return JsonResponse(MESSAGES['no_login'])
93
94
95 class CreateUsuario(APIView):
96     def post(self, request, format=None):
97         form = UsuarioForm(request.data)
98         return process_create_form(self, request, form)
99
100     def get(self, request, format=None):
101         return send_create_form(self, request, UsuarioForm())
102
103
104 class CreateDueño(APIView):
105     def post(self, request, format=None):
106         form = DueñoForm(request.data)
107         return process_create_form(self, request, form)
108
109     def get(self, request, format=None):
110         return send_create_form(self, request, DueñoForm())
111
112
113 class CreateTienda(APIView):
114     def post(self, request, format=None):
115         if (not is_owner(request)):
116             return JsonResponse(MESSAGES['unallowed'])
117         form = TiendaForm(request.data)
118
119         if form.is_valid():
120             instance = form.save(commit=False)
121             instance.dueño = request.user
122             instance.save()
123             return JsonResponse(MESSAGES['created'])
124         return JsonResponse(MESSAGES['fields_error'])
125
```

```
126     def get(self, request, format=None):
127         if (not is_owner(request)):
128             return JsonResponse(MESSAGES['unallowed'])
129         return send_create_form(self, request, TiendaForm())
130
131
132 class CreateProducto(APIView):
133     def post(self, request, format=None):
134         if (not is_owner(request)):
135             return JsonResponse(MESSAGES['unallowed'])
136         form = ProductoForm(request.data)
137         return process_create_form(self, request, form)
138
139     def get(self, request, format=None):
140         if (not is_owner(request)):
141             return JsonResponse(MESSAGES['unallowed'])
142         form = ProductoForm()
143         form.fields['tienda'].queryset = Tienda.objects.filter(
144             dueño=request.user)
145         return send_create_form(self, request, form)
146
147
148 class CreateVenta(APIView):
149     def post(self, request, format=None):
150         if (not is_user(request)):
151             return JsonResponse(MESSAGES['unallowed'])
152         form = VentaForm(request.data)
153
154         if form.is_valid():
155             instance = form.save(commit=False)
156             instance.usuario = request.user
157             instance.save()
158             return JsonResponse(MESSAGES['created'])
159         return JsonResponse(MESSAGES['fields_error'])
160
161     def get(self, request, format=None):
162         if (not is_user(request)):
163             return JsonResponse(MESSAGES['unallowed'])
164         form = VentaForm()
165         form.fields['producto'].queryset = Producto.objects.filter(
166             tienda=request.json()['tienda'])
167         return send_create_form(self, request, VentaForm())
168
169
170 class GetUsuarios(APIView):
171     def post(self, request, format=None):
172         usuarios = Usuario.objects.all()
173         serializer = UsuarioSerializer(
174             usuarios, many=True, context={'request': request})
175         return JsonResponse({'status': 200, 'usuarios': serializer.data}, safe=False)
176
177
178 class GetTiendas(APIView):
179     def post(self, request, format=None):
180         if (not is_owner(request)):
181             return JsonResponse(MESSAGES['unallowed'])
```

```
182     tiendas = Tienda.objects.filter(dueño=request.user)
183     serializer = TiendaSerializer(
184         tiendas, many=True, context={'request': request})
185     return JsonResponse({'status': 200, 'tiendas': serializer.data}, safe=False)
186
187 class GetProductos(APIView):
188     def post(self, request, format=None):
189         if (is_user(request)):
190             productos = Producto.objects.filter(tienda=request.json()['tienda'])
191         elif (is_owner(request)):
192             productos = Producto.objects.filter(tienda__dueño=request.user)
193         else:
194             productos = Producto.objects.all()
195
196         serializer = ProductoSerializer(
197             productos, many=True, context={'request': request})
198         return JsonResponse({'status': 200, 'productos': serializer.data}, safe=False)
199
200 class GetVentas(APIView):
201     def post(self, request, format=None):
202         if (is_user(request)):
203             ventas = Venta.objects.filter(usuario=request.user)
204         elif (is_owner(request)):
205             ventas = Venta.objects.filter(producto__tienda__dueño=request.user)
206         else:
207             ventas = Venta.objects.all()
208
209         serializer = VentaSerializer(
210             ventas, many=True, context={'request': request})
211         return JsonResponse({'status': 200, 'ventas': serializer.data}, safe=False)
```

- Se crean las vistas para los formularios de **Usuario**, **Dueño**, **Admin**, **Tienda**, **Producto** y **Venta**.
- También se crean las vistas que hacen uso de los viewsets para listar los objetos de los modelos.
- Estas vistas retornan un JSON con los datos del formulario tras ser serializados y reciben las respuestas del form.
- Se crean las vistas **iniciar_sesion** y **cerrar_sesion** para el inicio y cierre de sesión.

viewsets.py

```
1 from rest_framework import viewsets
2 from .models import Usuario, Tienda, Producto, Venta
3 from .serializers import UsuarioSerializer, TiendaSerializer, ProductoSerializer, VentaSerializer
4
5
6 class UsuarioViewSet(viewsets.ModelViewSet):
7     queryset = Usuario.objects.all()
8     serializer_class = UsuarioSerializer
9
10 class TiendaViewSet(viewsets.ModelViewSet):
11     queryset = Tienda.objects.all()
12     serializer_class = TiendaSerializer
13
14 class ProductoViewSet(viewsets.ModelViewSet):
```

```
15     queryset = Producto.objects.all()
16     serializer_class = ProductoSerializer
17
18 class VentaViewSet(viewsets.ModelViewSet):
19     queryset = Venta.objects.all()
20     serializer_class = VentaSerializer
21
```

- Se crean los viewset para listar los objetos de los modelos **Usuario**, **Dueño**, **Admin**, **Tienda**, **Producto** y **Venta**.

urls.py

```
1  from django.urls import path
2  from . import views
3  from rest_framework import routers
4  from .viewsets import UsuarioViewSet, TiendaViewSet, ProductoViewSet, VentaViewSet
5  from django.urls import include
6
7  router = routers.DefaultRouter()
8  router.register(r'usuarios', UsuarioViewSet)
9  router.register(r'tiendas', TiendaViewSet)
10 router.register(r'productos', ProductoViewSet)
11 router.register(r'ventas', VentaViewSet)
12
13
14 urlpatterns = [
15     path('rest/', include(router.urls)),
16     path('api-auth/', include('rest_framework.urls', namespace='rest_framework')),
17     path('iniciar_sesion/', views.IniciarSesion.as_view()),
18     path('cerrar_sesion/', views.CerrarSesion.as_view()),
19     path('crear_usuario/', views.CreateUsuario.as_view()),
20     path('crear_dueño/', views.CreateDueño.as_view()),
21     path('crear_tienda/', views.CreateTienda.as_view()),
22     path('crear_producto/', views.CreateProducto.as_view()),
23     path('crear_venta/', views.CreateVenta.as_view()),
24     path('get_usuarios/', views.GetUsuarios.as_view()),
25     path('get_tiendas/', views.GetTiendas.as_view()),
26     path('get_productos/', views.GetProductos.as_view()),
27     path('get_ventas/', views.GetVentas.as_view()),
28 ]
```

- Se crean las rutas para las vistas de los formularios y los listados de los modelos.
- Se crean las rutas para las vistas de inicio y cierre de sesión.

3.2 Frontend en Angular

3.2.1 Aplicación base

```
app.routes.ts
1 import { Routes } from '@angular/router';
2 import { DinamicFormComponent } from '../dinamic-form/dinamic-form.component';
3 import { HomeComponent } from '../home/home.component';
4 import { HeaderComponent } from '../home/header/header.component';
5 import { FooterComponent } from '../home/footer/footer.component';
6 import { LoginComponent } from '../login/login.component';
7 import { RegisterComponent } from '../register/register.component';
8 import { TestComponent } from '../test/test.component';
9
10 export const routes: Routes = [
11   {
12     path: '', component: HomeComponent
13   },
14   {
15     path: 'login', component: LoginComponent
16   },
17   {
18     path: 'register', component: RegisterComponent
19   },
20   {
21     path: 'test', component: TestComponent
22   }
23 ];
```

- Archivo de rutas de la aplicación.
- Se crean las rutas para los componentes **home**, **login** y **register**.

styles.css

```
1 /* You can add global styles to this file, and also import other style files */
2 html, body {
3   font-family: Arial, sans-serif;
4   margin: 0;
5   padding: 0;
6   display: flex;
7   flex-direction: column;
8 }
9
10 html, body {
11   height: 100%;
12 }
13
14 .container {
15   display: flex;
16   flex-direction: column;
17   min-height: 100vh;
18 }
19
20 header {
21 }
```



```
22
23 main {
24     flex: 1;
25     padding: 20px;
26 }
27
28 footer {
29     text-align: center;
30     padding: 10px 0;
31 }
```

- Estilos de la aplicación.
- Se encarga de dar estilos generales a la aplicación.

3.2.2 dynamic-form

```
                                loader-form.component.ts
1  import { Component, ComponentRef, ViewContainerRef } from '@angular/core';
2  import { DinamicFormComponent } from '../dinamic-form.component';
3  import { TYPE_FORMS } from '../../constants';
4
5  @Component({
6      selector: 'loader-form',
7      standalone: true,
8      imports: [],
9      template: `
10 `
11 })
12 export class LoaderFormComponent {
13     private componentRef!: ComponentRef<DinamicFormComponent>;
14
15     constructor(
16         private viewContainer: ViewContainerRef,
17     ){
18
19     }
20
21     createForm(typeForm: string): void {
22         if(this.componentRef)
23             this.componentRef.destroy();
24         this.componentRef = this.viewContainer.createComponent(DinamicFormComponent);
25         if(this.componentRef && this.componentRef.instance){
26             this.componentRef.instance.loadSchema(typeForm);
27         }else{
28             console.log("El componente no se ha podido cargar correctamente");
29         }
30     }
31
32 }
```

- Componente que se encarga de cargar el formulario dinámico.
- Se encarga de recibir los datos del formulario y enviarlos al componente **dynamic-form**.

dynamic-form.component.ts

```
1  import { Component} from '@angular/core';
2  import { FormBuilder, FormGroup, ReactiveFormsModule, Validators } from '@angular/forms';
3  import { FormField } from '../models/form-field';
4  import { ApiService } from '../services/api.service';
5  import { response } from 'express';
6
7  @Component({
8    selector: 'dinamic-form',
9    standalone: true,
10   imports: [ReactiveFormsModule],
11   templateUrl: './dinamic-form.component.html',
12   styleUrls: ['./dinamic-form.component.css']
13 })
14 export class DinamicFormComponent{
15
16   form: FormGroup;
17   fields: FormField[] = [];
18   model!: string;
19   csrf!: string;
20
21   constructor(
22     private formBuilder: FormBuilder,
23     private api: ApiService
24   ){
25     this.form = this.formBuilder.group({
26     });
27   }
28
29   loadSchema(model: string): void{
30     this.model = model;
31     this.form = this.formBuilder.group({});
32     this.api.getFormSchema(model, ).subscribe(fieldsReceived => {
33       this.fields = fieldsReceived;
34       this.buildForm();
35     });
36   }
37
38   buildForm(){
39     for(const field of this.fields) {
40       const control = this.formBuilder.control('', this.getValidators(field));
41       this.form.addControl(field.name, control);
42     }
43   }
44
45   getValidators(field: FormField){
46     const validators = [];
47     if(field.validators){
48       for(const [key, value] of Object.entries(field.validators)){
49         switch(key){
50           case 'required':
51             validators.push(Validators.required); break;
52           case 'maxlength':
53             validators.push(Validators.maxLength(Number(value))); break;
54           case 'minlength':
55             validators.push(Validators.minLength(Number(value))); break;
56           case 'email':
```

```
57         validators.push(Validators.email); break;
58     default:
59         console.warn(`This validator is not supported: ${key}`);
60     }
61 }
62 }
63 return validators;
64 }
65
66 onSubmit(): void {
67     if (this.form.valid) {
68         console.log('Formulario enviado:', this.form.value);
69         // Enviar datos a la API
70         this.api.postForm(this.form.value, this.model, this.csrf).subscribe(response => {
71             console.log(response);
72             alert(response.message);
73         });
74
75     } else {
76         this.form.markAllAsTouched();
77         console.error('Formulario no válido');
78     }
79 }
80
81 getValidatorMessage(fieldName: string) {
82     const control = this.form.controls[fieldName];
83     if (control.errors) {
84         return Object.keys(control.errors)
85             .map(key => {
86                 switch (key) {
87                     case 'required':
88                         return 'Este campo es obligatorio';
89                     case 'minlength':
90                         return `Debe tener al menos ${control.errors?.['minlength'].requiredLength}
91                             ↪ caracteres`;
92                     case 'maxlength':
93                         return `Debe tener como máximo ${control.errors?.['maxlength'].requiredLength}
94                             ↪ caracteres`;
95                     case 'email':
96                         return 'Ingrese un email correcto';
97                     default:
98                         return '';
99                 }
100             })
101             .join(', ');
102     }
103     return '';
```

- Componente que se encarga de crear un formulario dinámico.
- Recibe un JSON con los campos del formulario y se encarga de deserializar los datos enviados desde Django.

dynamic-form.component.html

```

1  <form [formGroup]="form" [ngSubmit]="onSubmit()">
2    @for (field of fields; track $index) {
3      <label [for]="field.name">{{ field.label }}</label>
4      @switch (field.tipoCampo) {
5        @case ('input') {
6          <input [type]="field.attributes?.['type']"
7                [formControlName]="field.name"
8                [attr.placeholder]="field.attributes?.['placeholder'] || field.name"
9          >
10         }
11        @case ('select') {
12          <select [formControlName]="field.name"
13          >
14
15          <option value="" disabled selected>{{ field.attributes?.['placeholder'] ||
16            ↳ 'Seleccione una opción' }}</option>
17          @for (option of field.options; track $index) {
18            <option [value]="option.value">
19              {{ option.label }}
20            </option>
21          }
22          </select>
23        }
24        @case ('textarea') {
25          <br>
26          <textarea [formControlName]="field.name"
27                    [attr.placeholder]="field.attributes?.['placeholder'] || field.name"
28                    [attr.rows]="field.attributes?.['rows'] || 10"
29                    [attr.cols]="field.attributes?.['cols'] || 10"
30          ></textarea>
31        }
32      }
33      <br>
34      @if (form.get(field.name)?.invalid && form.get(field.name)?.touched) {
35        <div>
36          <small>
37            {{ getValidatorMessage(field.name) }}
38          </small>
39        </div>
40      }
41    }
42    <button>Guardar</button>
43  </form>

```

- Plantilla del componente **dynamic-form**.
- Se crean los campos del formulario de acuerdo al JSON recibido.

3.2.3 home

footer.component.ts

```

1  import { Component } from '@angular/core';
2  import { FaIconLibrary, FontAwesomeModule } from '@fortawesome/angular-fontawesome';
3  import { faFacebook, faInstagram, faWhatsapp } from '@fortawesome/free-brands-svg-icons';
4

```

```

5
6 @Component({
7   selector: 'home-footer',
8   standalone: true,
9   imports: [FontAwesomeModule],
10  templateUrl: './footer.component.html',
11  styleUrls: ['./footer.component.css'],
12 })
13 export class FooterComponent {
14   constructor(library: FaIconLibrary) {
15     library.addIcons(faFacebook, faWhatsapp, faInstagram);
16   }
17 }

```

- Componente que se encarga de mostrar el footer de la página.

footer.component.html

```

1 <footer class="footer">
2   <div class="container-footer">
3     <div class="row">
4       <div>
5         <h2 class="footer-heading logo">Kios Kios</h2>
6         <p class="menu">
7           <a href="#">Home</a>
8           <a href="#">About</a>
9           <a href="#">Coments</a>
10          <a href="#">All</a>
11          <a href="#">Contact</a>
12        </p>
13        <ul class="ftco-footer-social ">
14          <li class="ftco-animate"><a href="#" data-toggle="tooltip"
15            ↳ data-placement="top" title="Whatsapp"><fa-icon [icon]="['fab',
16              ↳ 'whatsapp']"></fa-icon></a></li>
17          <li class="ftco-animate"><a href="#" data-toggle="tooltip"
18            ↳ data-placement="top" title="Facebook"><fa-icon [icon]="['fab',
19              ↳ 'facebook']"></fa-icon></a></li>
20          <li class="ftco-animate"><a href="#" data-toggle="tooltip"
21            ↳ data-placement="top" title="Instagram"><fa-icon [icon]="['fab',
22              ↳ 'instagram']"></fa-icon></a></li>
23        </ul>
24      </div>
25    </div>
26    <div class="row">
27      <p>
28        Kioskios 2024 | Sigue disfrutando de tus compras en KiosKios.com
29      </p>
30    </div>
31  </div>
32 </footer>

```

- Plantilla del componente **footer**.

header.component.ts

```

1  import { Component } from '@angular/core';
2  import { RouterLink } from '@angular/router';
3  import { FaIconLibrary, FontAwesomeModule } from '@fortawesome/angular-fontawesome';
4  import { faCartShopping, faMagnifyingGlass, faBars } from
    ↳ '@fortawesome/free-solid-svg-icons';
5
6
7  @Component({
8    selector: 'home-header',
9    standalone: true,
10   imports: [RouterLink, FontAwesomeModule],
11   templateUrl: './header.component.html',
12   styleUrls: ['./header.component.css'],
13 })
14
15 export class HeaderComponent {
16   isMenuOpen: boolean = false;
17   constructor(library: FaIconLibrary){
18     library.addIcons(faCartShopping, faMagnifyingGlass);
19   }
20
21   toggleMenu() {
22     this.isMenuOpen = !this.isMenuOpen;
23   }
24 }

```

- Componente que se encarga de mostrar el header de la página.

header.component.html

```

1  <header>
2    <div class="header-left">
3      <div class="logo">
4        <a routerLink="/">
5          
6        </a>
7      </div>
8      <div class="search">
9        <section class="search-bar">
10         <form action="." method="GET">
11           <label for="q"></label>
12           <input type="text" name="q" placeholder="Busca un artículo">
13           <button class="busqueda"><fa-icon [icon]="['fas',
    ↳ 'magnifying-glass']"></fa-icon></button>
14         </form>
15       </section>
16     </div>
17
18   </div>
19
20   <nav class="header-right">
21     <button class="menu-toggle" id="menu-toggle">&#9776;</button>
22     <ul class="nav-list" id="nav-list">
23       <li><a routerLink="/cart"><fa-icon [icon]="['fas', 'cart-shopping']"></fa-icon>
    ↳ Carrito</a></li>
24       <li><a routerLink="/login">Iniciar Sesión</a></li>

```

```
25         <li><a routerLink="/register" class="register">Regístrate</a></li>
26     </ul>
27 </nav>
28
29 </header>
```

- Plantilla del componente **header**.

nav.component.ts

```
1 import { Component } from '@angular/core';
2
3 @Component({
4     selector: 'home-nav',
5     standalone: true,
6     imports: [],
7     template: `
8         <nav></nav>
9     `,
10    styles: ``
11 })
12 export class NavComponent {
13
14 }
```

- Componente que se encarga de mostrar la barra de navegación de la página.

banner.component.ts

```
1 import { Component, ElementRef, ViewChildren, QueryList } from '@angular/core';
2 import { RouterLink } from '@angular/router';
3
4
5 @Component({
6     selector: 'home-banner',
7     standalone: true,
8     imports: [RouterLink],
9     templateUrl: './banner.component.html',
10    styleUrls: ['./banner.component.css'],
11 })
12
13 export class BannerComponent {
14
15     @ViewChildren('slide') slides!: QueryList<ElementRef>;
16     private currentIndex: number = 0;
17
18     nextSlide() {
19         this.slides.toArray()[this.currentIndex].nativeElement.classList.remove('active');
20         this.currentIndex = (this.currentIndex + 1) % this.slides.length;
21         this.slides.toArray()[this.currentIndex].nativeElement.classList.add('active');
22     }
23
24     prevSlide() {
25         this.slides.toArray()[this.currentIndex].nativeElement.classList.remove('active');
26         this.currentIndex = (this.currentIndex - 1 + this.slides.length) % this.slides.length;
```

```

27     this.slides.toArray()[this.currentIndex].nativeElement.classList.add('active');
28   }
29 }
30

```

- Componente que se encarga de mostrar el banner de la página.

banner.component.html

```

1  <section class="banner">
2    <div class="slider">
3      <div #slide class="slide active" style="background-image: url('banner.png');">
4        <div class="banner-text">
5          <h1>Revisa los nuevos productos disponibles</h1>
6          <p>Encuentra los mejores precios de tus productos y ahorra
7            ↪ inteligentemente.</p>
8          <button routerLink="/register">Regístrate Ya</button>
9        </div>
10     </div>
11     <div #slide class="slide" style="background-image:
12       ↪ url('banner2.png');justify-content:flex-end;">
13       <div class="banner-text" style="color: black">
14         <h1>Revisa los nuevos productos disponibles</h1>
15         <p>Encuentra los mejores precios de tus productos y ahorra
16           ↪ inteligentemente.</p>
17         <button routerLink="/register">Regístrate Ya</button>
18       </div>
19     </div>
20     <div #slide class="slide" style="background-image: url('banner3.png');">
21       <div class="banner-text" style="color: black;">
22         <h1>Revisa los nuevos productos disponibles</h1>
23         <p>Encuentra los mejores precios de tus productos y ahorra
24           ↪ inteligentemente.</p>
25         <button routerLink="/register" class="register">Regístrate</button>
26       </div>
27     </div>
28   </div>
29   <div class="controls">
30     <button [(click)="prevSlide()]"></button>
31     <button [(click)="nextSlide()]"></button>
32   </div>
33 </section>

```

- Plantilla del componente **banner**.

home.component.ts

```

1  import { Component } from '@angular/core';
2  import { HeaderComponent } from '../header/header.component';
3  import { NavComponent } from '../nav/nav.component';
4  import { FooterComponent } from '../footer/footer.component';
5  import { RouterOutlet } from '@angular/router';
6  import { BannerComponent } from '../banner/banner.component';
7
8  @Component({

```



```
9 selector: 'app-home',
10 standalone: true,
11 imports: [HeaderComponent, NavComponent, FooterComponent, RouterOutlet, BannerComponent],
12 templateUrl: './home.component.html',
13 styleUrls: ['./home.component.css']
14 })
15 export class HomeComponent {
16
17 }
```

- Componente que se encarga de mostrar la página principal.
- Posee redirecciones a las demás páginas de la aplicación.

home.component.html

```
1 <home-header />
2 <home-nav />
3 <main>
4   <router-outlet />
5   <home-banner />
6 </main>
7 <home-footer />
8
```

- Plantilla del componente **home**.
- Desde aquí se puede acceder a las demás páginas de la aplicación.

3.2.4 login

login.component.ts

```
1 import { AfterViewInit, Component, OnInit, ViewChild } from '@angular/core';
2 import { LoaderFormComponent } from '../dinamic-form/loader-form/loader-form.component';
3 import { TYPE_FORMS } from '../constants';
4
5
6
7 @Component({
8   selector: 'app-login',
9   standalone: true,
10  imports: [LoaderFormComponent],
11  templateUrl: './login.component.html',
12  styleUrls: ['./login.component.css']
13 })
14 export class LoginComponent implements AfterViewInit {
15
16   @ViewChild(LoaderFormComponent) loaderForm!: LoaderFormComponent;
17
18   ngAfterViewInit(): void {
19     if(this.loaderForm)
20       this.loaderForm.createForm(TYPE_FORMS.LOGIN);
21     else
22       console.log("No se pudo cargar el formulario de login");
23   }
24
25 }
```

- Componente que se encarga de mostrar el formulario de inicio de sesión.
- Se encarga de enviar los datos al backend y recibir la respuesta.

login.component.html

```
1 <main>
2   <loader-form />
3 </main>
```

- Plantilla del componente **login**.
- Se muestra el formulario de inicio de sesión.

3.2.5 register

register.component.ts

```
1 import { AfterViewInit, Component, ViewChild } from '@angular/core';
2 import { LoaderFormComponent } from '../dinamic-form/loader-form/loader-form.component';
3 import { TYPE_FORMS } from '../constants';
4 import { FooterComponent } from '../home/footer/footer.component';
5 import { HeaderComponent } from '../home/header/header.component';
6
7 @Component({
8   selector: 'app-register',
9   standalone: true,
10  imports: [LoaderFormComponent, FooterComponent, HeaderComponent],
11  templateUrl: './register.component.html',
12  styleUrls: ['./register.component.css']
13 })
14 export class RegisterComponent {
15
16   @ViewChild(LoaderFormComponent) loaderForm!: LoaderFormComponent;
17
18   loadOwner() : void {
19     this.loaderForm.createForm(TYPE_FORMS.CREATE_OWNER);
20   }
21
22   loadUser() : void {
23     this.loaderForm.createForm(TYPE_FORMS.CREATE_USER);
24   }
25
26 }
```

- Componente que se encarga de mostrar el formulario de registro.
- Se encarga de enviar los datos al backend y recibir la respuesta.

register.component.html

```
1 <home-header />
2 <br><br>
3 <main>
4   <div class="container">
5     <div class="buttons">
```

```
6     <button (click)="loadOwner()" class="button" >Soy Dueño</button>
7     <button (click)="loadUser()" class="button" >Soy Comprador</button>
8   </div>
9   <div class="form-container">
10     <loader-form />
11   </div>
12 </div>
13
14
15 </main>
16
17 <home-footer />
```

- Plantilla del componente **register**.
- Se muestra el formulario de registro.

register.component.css

```
1 .container{
2   display: flex;
3   flex-direction: column;
4   justify-content: flex-start;
5   align-items: center ;
6 }
7 .buttons{
8   display: flex;
9   flex-direction: row;
10  justify-content: center;
11  gap: 20px;
12  height: 100px;
13 }
14
15 .button{
16   color: #ffffff;
17   background-color: #ff9900 ;
18   border-radius: 10%;
19   padding: 10px;
20   width: 180px;
21   max-width: 180px;
22   max-height: 40px;
23   border-color: #ff9900;
24 }
25 .form-container {
26   background-color: rgb(233, 233, 233);
27   padding: 20px;
28   border-radius: 10px;
29   box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
30   width: 100%;
31   max-width: 400px;
32 }
```

- Estilos del componente **register**.
- Se encarga de dar estilos al formulario de registro.

3.2.6 services

```
api.service.ts
1 import { Injectable } from '@angular/core';
2 import { HttpClient, HttpHeaders } from '@angular/common/http';
3 import { Observable, catchError, map, of } from 'rxjs';
4 import { FormField } from '../models/form-field';
5 import { Form } from '../form';
6 @Injectable({
7   providedIn: 'root',
8 })
9 export class ApiService {
10
11   private urlBaseApi : string = 'http://localhost:8000/api';
12
13   constructor(private http: HttpClient) { }
14
15   postForm(formtoSend: Form, to: string, token: string):Observable<any>{
16     const url = this.urlBaseApi + `/${to}`;
17     const httpOptions = {
18       headers: new HttpHeaders({
19         'Content-Type': 'application/json',
20         //'Authorization': 'authkey',
21         //'userid': '1'
22       })
23     };
24     return this.http.post<Form>(url, formtoSend, httpOptions);
25   }
26
27   getFormSchema(formToGet : string) : Observable<FormField[]> {
28     const url = this.urlBaseApi + `/${formToGet}`;
29     const httpOptions = {
30       headers: new HttpHeaders({
31         'Content-Type': 'application/json',
32
33         //'Authorization': 'authkey',
34         //'userid': '1'
35       })
36     };
37     return this.http.get<{status: number, campos: FormField[]}>(url, httpOptions).pipe(
38       map(response => {
39         if(response.status != 200)
40           throw new Error("No autorizado")
41         return response.campos;
42       }),
43       catchError(error => {
44         throw error;
45       })
46     );
47   }
48 }
```

- Servicio que se encarga de realizar las peticiones al backend.
- Se encarga de enviar los datos al backend y recibir la respuesta.

auth-interceptor.service.ts

```
1 import { HttpResponse, HttpEvent, HttpHandler, HttpInterceptor, HttpRequest } from
  ↳ '@angular/common/http';
2 import { Injectable } from '@angular/core';
3 import { Router } from '@angular/router';
4 import { error } from 'console';
5 import { Observable, catchError } from 'rxjs';
6
7 @Injectable({
8   providedIn: 'root'
9 })
10 export class AuthInterceptorService implements HttpInterceptor {
11
12   constructor(private router: Router) {}
13
14   intercept(req: HttpRequest<any>, next: HttpHandler): Observable<HttpEvent<any>> {
15     return next.handle(req).pipe(
16       catchError((error: HttpResponse) => {
17         if(error.status === 401) {
18           const returnUrl = this.router.url;
19           this.router.navigate(['/login'], { queryParams: { returnUrl } })
20         }
21         throw error;
22       })
23     )
24   }
25 }
```

- Servicio que se encarga de interceptar las peticiones al backend.

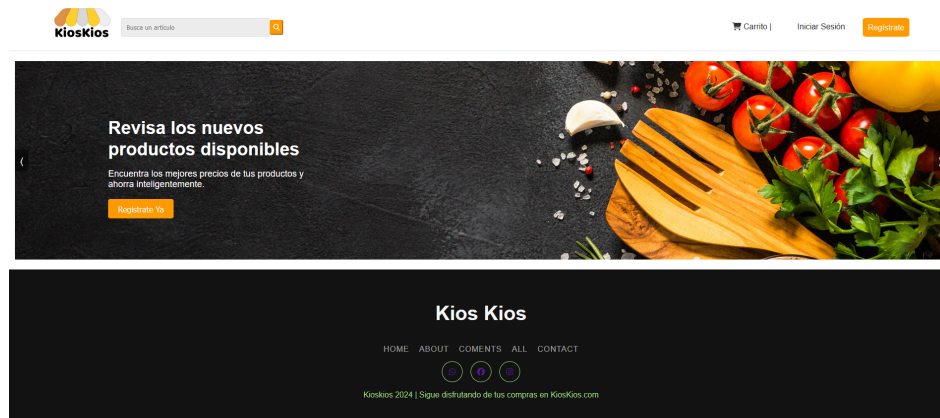
3.2.7 models

form-field.ts

```
1 export interface FormField {
2   tipoCampo: 'input' | 'select' | 'textarea';
3   name: string;
4   label: string;
5   validators?: { [key: string]: string };
6   attributes?: { [key: string]: string };
7   options?: { label: string; value: any }[];
8 }
```

- Modelo que se encarga de definir los campos de un formulario.
- Usado para la creación de formularios dinámicos.

3.3 Demostración



Página principal.

Nombre de usuario

Contraseña

Inicio de sesión.

Username

Telefono

Contraseña

Confirmar contraseña
















































































































Yape qr

Registro.



3.4 Lista de commits

Se manejan de forma más sólida las validaciones de los campos en el formulario	4854761	🔗	↗
🔗 guoCreator committed last week			
Se añade un campo atributos opcional para los atributos de los elementos de entrada del formulario	5694f6a	🔗	↗
🔗 guoCreator committed last week			
Se	18a1181	🔗	↗
🔗 guoCreator committed last week			
🔗 Commits on Jun 29, 2024			
Merge branch 'main' of https://github.com/cmestaz/Kiosko	03a076d	🔗	↗
🔗 cmestaz committed last week			
Se abandona el soporte para windows	4b909f6	🔗	↗
🔗 cmestaz committed last week			
🔗 Commits on Jun 28, 2024			
Merge pull request #4 from cmestaz/angularforms	894f094	🔗	↗
🔗 cmestaz committed last week			
Se reusa el componente para testarlo en la web, asimismo, se provee del HttpClient, con fetch ya que es recomendado para site server rendering	a76a8b7	🔗	↗
🔗 guoCreator committed last week			
Se genera algunos campos de prueba para probar la funcionalidad de la construcción de formularios	3a65d9e	🔗	↗
🔗 guoCreator committed last week			
Se construye en la estructura del componente el formulario con la construcción previa generada con los métodos en el componente	0a4a323	🔗	↗
🔗 guoCreator committed last week			
Se crean los metodos del componente que permitieran construirlo con validaciones	0916361	🔗	↗
🔗 guoCreator committed last week			
El componente FormBuilder fue reemplazado por el componente DynamicForm	47d2577	🔗	↗
🔗 guoCreator committed last week			
Se crea el componente dynamicForm que construye los formularios que recibe de la api	0e48f1a	🔗	↗
🔗 guoCreator committed last week			
Se modifica el modelo de los campos a recibir de la api para renderizar los formularios	2d90b07	🔗	↗
🔗 guoCreator committed last week			
Se modifican las vistas para retomar json y servir como api	03d3338	🔗	↗
🔗 cmestaz committed last week			
Merge pull request #3 from cmestaz/forms-api	8073c19	🔗	↗
🔗 cmestaz committed last week			
Se completa la funcionalidad de elegir una ubicación al momento de crear la tienda	7946033	🔗	↗
🔗 cmestaz committed last week			
Se solucionan los conflictos para añadir el campo de ubicación a la base de datos	8350338	🔗	↗
🔗 cmestaz committed last week			
Se modifica los modelos de usuarios para cumplir con las normas de la autenticación de django	46c5aea	🔗	↗
🔗 cmestaz committed last week			
Se modifican todos los modelos y formularios para ajustarse al sistema de autenticación de django	4d5c33a	🔗	↗
🔗 cmestaz committed last week			
Se crean formularios base para los modelos	c7a23a6	🔗	↗
🔗 cmestaz committed last week			
Merge pull request #2 from cmestaz/modelos	172b633	🔗	↗
🔗 cmestaz committed last week			
Se añade un modelo dentro de los campos del formulario que permite colocar contenido en caso el elemento lo tenga	7d72796	🔗	↗
🔗 guoCreator committed last week			
Merge pull request #1 from cmestaz/angularforms	3d0ca71	🔗	↗
🔗 cmestaz committed last week			
Se reea el servicio de la API con un método inicial para hacer peticiones de los formularios	3d4b083	🔗	↗
🔗 guoCreator committed last week			
Se migran los modelos y añade una guía de instalación	5b53936	🔗	↗
🔗 cmestaz committed last week			
Se crea un modelo de campo del formulario para el envío de datos desde la api	6d6b198	🔗	↗
🔗 guoCreator committed last week			
Creando el componente que construyó los formularios a partir de peticiones a la API	950a1a8	🔗	↗
🔗 guoCreator committed last week			
Se crea la aplicación y los modelos	c11c9d0	🔗	↗
🔗 cmestaz committed last week			
Ignorando los archivos de configuración de vscode	a90b33f	🔗	↗
🔗 guoCreator committed last week			
Se elimina la página por defecto de angular	8d7a48b	🔗	↗
🔗 guoCreator committed last week			
🔗 Commits on Jun 26, 2024			
Instalación de los paquetes npm	8958b1c	🔗	↗
🔗 guoCreator committed last week			
Añadiendo en el README la manera de instalar los requerimientos	35c38d0	🔗	↗
🔗 guoCreator committed last week			
🔗 Commits on Jun 25, 2024			
Se inicializa el proyecto de angular	4803ae0	🔗	↗
🔗 cmestaz committed 2 weeks ago			
Se inicializa el proyecto de django	08aef0a	🔗	↗
🔗 cmestaz committed 2 weeks ago			
Se crean los requerimientos y el .gitignore	2463897	🔗	↗
🔗 cmestaz committed 2 weeks ago			
Inicial commit	2d48766	🔗	↗
🔗 cmestaz committed 2 weeks ago			

Lista de commits.

Se mueve el loader del formulario a un componente externo. Se crea un método en el mismo componente que permite cargar el formulario según un tipo.	626b7f51		
 guoCenator committed 4 days ago			
Se lanza una excepción cuando no se recibe una respuesta esperada de esquema de formulario para que el interceptor pueda manejar el error.	04ae325c		
 guoCenator committed 4 days ago			
Se crea un interceptor de las solicitudes para que redirija al login en caso el usuario no tenga los permisos para acceder a una parte de la página web	c803e10b		
 guoCenator committed 4 days ago			
Se cambia de nombre a las constantes para que sean más descriptivas	08b1c17e		
 guoCenator committed 4 days ago			
Se agrega una constante de formulario para el login	08a7c1d7		
 guoCenator committed 4 days ago			
Se cambia ligeramente el mensaje enviado en caso de error de autenticación	6c2d314		
 guoCenator committed 4 days ago			
Merge pull request #7 from cmetaszc/home	75f6304e		
 cmetaszc committed 4 days ago			
Merge branch 'main' into home	f4b758ac		
 cmetaszc committed 4 days ago			
Se realizan las peticiones a la api desde el servicio	e1a75080		
 guoCenator committed 4 days ago			
Se utiliza la constante creada para abstraer los tipos de formularios que se obtendrán	5994c182		
 guoCenator committed 4 days ago			
Se modifica llamada a los archivos de acuerdo a la modificación del modelo del esquema del formulario	001910a		
 guoCenator committed 4 days ago			
Se simplifica el modelo del esquema de los formularios	1a45a50a		
 guoCenator committed 4 days ago			
Se crea un archivo de constantes para manejar de mejor manera las peticiones	5a6620e		
 guoCenator committed 4 days ago			
Se actualizan los requerimientos	c012257e		
 guoCenator committed 4 days ago			
Se cambia las respuestas de la api a las respuestas esperadas y un poco más adecuadas	0191137		
 guoCenator committed 4 days ago			
Se corrigen algunos problemas con el envío de la serialización	41097153		
 cmetaszc committed 4 days ago			
Merge pull request #6 from cmetaszc/forms-fix	710a208f		
 cmetaszc committed 4 days ago			
Se verifica el funcionamiento de los serializadores	0077504f		
 cmetaszc committed 4 days ago			
Se añade la verificación del CORS	b012204b		
 cmetaszc committed 4 days ago			
Se crean serializadores para los modelos y formularios (no probado)	25c69f9b		
 cmetaszc committed 4 days ago			
Se corrigen algunos errores de validación en los formularios	e5d85311		
 cmetaszc committed 4 days ago			
Se implementa el sistema de inicio de sesión y se añade un decorador para verificar si el usuario ha iniciado sesión	79b1390b		
 cmetaszc committed 4 days ago			
Se corrige la subida de imágenes al almacenamiento	05a4870e		
 cmetaszc committed 4 days ago			
Se deshace la integración de servicios de geolocalización en el backend	c998a527		
 cmetaszc committed 4 days ago			
 Commit on Jul 1, 2024			
Merge pull request #5 from cmetaszc/home	0e776e6c		
 cmetaszc committed 5 days ago			
Se cambian de campos para probar la funcionalidad del formulario del login	056a403e		
 guoCenator committed 5 days ago			
Se coloca el loader del en el componente del login	a0c8f06f		
 guoCenator committed 5 days ago			
Se elimina la carga de un formulario por defecto	2a8651a1		
 guoCenator committed 5 days ago			
Se crea un cargador del formulario para que permita cargarlo dinámicamente. Luego lo referencia y le aplico el método que necesita para cargar de acuerdo al formulario requerido.	e7120ca		
 guoCenator committed 5 days ago			
Se añade una ruta para el componente <code>register</code>	71a7a2a		
 guoCenator committed 5 days ago			
Se crea el componente <code>register</code> , que como su nombre lo indica, será para el registro de usuarios	4a67a76b		
 guoCenator committed 5 days ago			
Se coloca un <code>router-outlet</code> en el componente principal y se delega los componentes de lo que llamamos la "página principal" al componente <code>main</code> . De esta manera, se podrán manejar las rutas adecuas...	9559c08		
 guoCenator committed 5 days ago			
Se añaden los componentes <code>nav</code> y <code>footer</code> al componente principal.	1a2a2a00		
 guoCenator committed 5 days ago			
Se une el componente header al home con el contenido que tenía anteriormente	711a5527		
 guoCenator committed 5 days ago			
 Commit on Jul 1, 2024			
Se crea un componente para la página principal llamado <code>main</code> y subcomponentes que son <code>header</code> , <code>nav</code> y <code>footer</code> . Adicionalmente, se creo un componente <code>login</code> para el inicio de sesión de los usuarios.	077c197e		
 guoCenator committed 5 days ago			
 Commit on Jun 30, 2024			
Se crea una estructura para la página principal de la aplicación, sin embargo, luego se componentizará la página en footer, header, nav, etc	4889717f		
 guoCenator committed 5 days ago			

Lista de commits.

Se realizan unas mejoras al diseño. Posicion del header como sticky y slider principal con animación guicentro committed 45 hours ago	676a145	🔗 ↩
Se envía la verificación del formulario del login en caso de que no sea de método post guicentro committed 1 hour ago	3c43a18	🔗 ↩
Se mejora el diseño de la lupa de la sección de búsqueda dmeduc1 committed 1 hour ago	68ee4d0	🔗 ↩
Se completo el style del formulario de registro dmeduc1 committed 2 hours ago	71ea2a7	🔗 ↩
Se implemento el diseño del contenedor del formulario dmeduc1 committed 2 hours ago	8f55913	🔗 ↩
Se estilizo el formulario de registro , falta terminar dmeduc1 committed 2 hours ago	8d3526c	🔗 ↩
Se convirtió el banner en un componente dmeduc1 committed 14 hours ago	4b33275	🔗 ↩
Se hizo el diseño del banner responsive dmeduc1 committed 20 hours ago	3d6d680	🔗 ↩
Se implemento mas componentes en el diseño del banner dmeduc1 committed 20 hours ago	77a3548	🔗 ↩
Se implemento el diseño base de home del banner dmeduc1 committed 20 hours ago	8f73985	🔗 ↩
Se implemento el diseño base de home del banner dmeduc1 committed 20 hours ago	4ac479f	🔗 ↩
Se implemento la estructura html para el banner dmeduc1 committed yesterday	6a772a8	🔗 ↩
Se volvio responsive la barra de búsqueda dmeduc1 committed yesterday	8d5fcea	🔗 ↩
Se realizaron modificaciones en el componente header para que no se desborde y sea adaptable dmeduc1 committed yesterday → Commits on Jul 5, 2024	ac2088d	🔗 ↩
Se importo icons de una libreria y se uso para los icons del footer dmeduc1 committed yesterday	6a397d0	🔗 ↩
Se mejora el style y se agrega componentes para que sea responsive dmeduc1 committed yesterday	8851586	🔗 ↩
Se implementa el style base del footer dmeduc1 committed yesterday	15ff498	🔗 ↩
Se agrega la estructura html con clases en el componente footer dmeduc1 committed yesterday → Commits on Jul 4, 2024	e8f0a0c	🔗 ↩
Merge pull request #9 from cmetasz/home  Verify guicentro committed 2 days ago	388a4c1	🔗 ↩
Se regenera el archivo de requerimientos cmetasz committed 2 days ago	4381c15	🔗 ↩
Se hicieron algunas mejoras en el header dmeduc1 committed 2 days ago	88f53ae	🔗 ↩
Se estilizo la barra de búsqueda del header dmeduc1 committed 2 days ago	64d8f8d	🔗 ↩
Se agrega diseño a la sección de url de login , register en el header dmeduc1 committed 2 days ago	89ad7d0	🔗 ↩
Se agrega el diseño de la barra de header y el logo dmeduc1 committed 2 days ago	63804ca	🔗 ↩
Se realizo la estructura del header dmeduc1 committed 2 days ago → Commits on Jul 3, 2024	5aaf73b	🔗 ↩
Se añaden los requerimientos para Node en el README guicentro committed 3 days ago	2a7f70a	🔗 ↩
Se crean las funciones que cargan los formularios de registro dependiendo de la persona que quiera registrarse guicentro committed 3 days ago	a31f488	🔗 ↩
Se envía una cabecera en la petición con el formato de respuesta esperado guicentro committed 3 days ago	3c8d8d3	🔗 ↩
Se colocan botones que llaman funciones y que cargan el formulario dependiendo de cual se requiera guicentro committed 3 days ago	92d4179	🔗 ↩
El componente que crea los formularios ahora destruye los que ya ha creado. Asimismo, los construye solo cuando se le pide que lo cree. guicentro committed 3 days ago	12a0582	🔗 ↩
Se revisó como está enviada la validación de email y se colocó en el apartado de validaciones guicentro committed 3 days ago	7f46371	🔗 ↩
Se coloca al servidor de Angular como origen permitido para hacer peticiones a la api guicentro committed 3 days ago → Commits on Jul 2, 2024	3afba29	🔗 ↩
Merge branch 'main' of https://github.com/cmetasz/kioticoe cmetasz committed 4 days ago	378a326	🔗 ↩
Se añaden estados a las vistas para encajar con el frontend de angular cmetasz committed 4 days ago	395294c	🔗 ↩
Merge pull request #8 from cmetasz/home  Verify cmetasz committed 4 days ago → Commits on Jul 2, 2024	77801a8	🔗 ↩
En el login se creó el formulario del login. Aun no se ha implementado la capacidad de redirigir o de hacer submit. guicentro committed 4 days ago	4a822d4	🔗 ↩

Lista de commits.

Se agrega enlaces al register desde el banner	85a44ee	C3
Se limpia y corrige un error con la bd	415a780	C3
Se arregla el problema de los métodos de las vista de Django	686a277	C3
Se crea una alerta al enviar el formulario	9237a18	C3
Se cambia la posición del header a sticky	878771c	C3
Se modifican las url con las que se hacen peticiones a la API	c871425	C3
Se ajusto el css del header	a97a124	C3
Comitted on Jul 9, 2024		
Angular ya no necesita el token, por lo que se elimina. Asimismo, se arreglaron algunos errores de sintaxis en las vista de Django	775a988	C3
Se reagrega el token csrf	88c046d	C3
Merge pull request #12 from cmestasz/rest-api	87a4a73	C3
Se corrigen las url	ca33248	C3
Se recorren todos los formularios, vistas y serializadores para funcionar correctamente con el framework de rest	8223421	C3
Se hace funcionar el sistema de envio de formularios	b25a67c	C3
Merge branch 'main' of https://github.com/cmestasz/KioskoKios into rest-api	f258748	C3
Se cambia el sistema de backend a django rest framework	358c78d	C3
Merge pull request #11 from cmestasz/infome-01	18bc13b	C3
Se añaden los enlaces	a24467d	C3
Se configura el csrf token para el envio del formulario a la api	c58b1a4	C3
Merge pull request #10 from cmestasz/views-update	5d5c310	C3
Se añaden los tokens csrf a los forms	74a089d	C3
Se modifica el sistema de acceso	a88a018	C3

Lista de commits.

4 URL del repositorio en GitHub

- <https://github.com/cmestasz/KiosKios.git>

5 Enlace al video

- <https://youtu.be/SwXIicDCeu4>

6 Estructura de laboratorio 10

El contenido que se entrega en este laboratorio es el siguiente:

```
1 KiosKios/
2 |--- Informe.pdf
3 |--- Informe.tex
4 |--- README.md
5 |--- kioskios_api
6     |--- api
7         |--- __init__.py
8         |--- admin.py
9         |--- apps.py
10        |--- forms.py
11        |--- migrations
12        |--- models.py
13        |--- serializers.py
14        |--- tests.py
15        |--- urls.py
16        |--- views.py
17        |--- viewsets.py
18    |--- kioskios_api
19        |--- __init__.py
20        |--- settings.py
21        |--- urls.py
22        |--- wsgi.py
23    |--- manage.py
24 |--- kioskios_web
25     |--- src
26         |--- app
27             |--- app.routes.ts
28             |--- app.component.html
29             |--- app.component.css
30             |--- dinamic-form
31                 |--- loader-form
32                     |--- loader-form.component.ts
33             |--- dynamic-form
34                 |--- dynamic-form.component.ts
35                 |--- dynamic-form.component.html
36                 |--- dynamic-form.component.spec.ts
37                 |--- dynamic-form.component.css
38         |--- home
39             |--- footer
40                 |--- footer.component.ts
41                 |--- footer.component.html
42         |--- header
43             |--- header.component.ts
```

```
44         |--- header.component.html
45     |--- nav
46         |--- nav.component.ts
47     |--- banner
48         |--- banner.component.ts
49         |--- banner.component.html
50     |--- home.component.ts
51     |--- home.component.html
52     |--- home.component.css
53 |--- login
54     |--- login.component.ts
55     |--- login.component.html
56     |--- login.component.css
57 |--- register
58     |--- register.component.ts
59     |--- register.component.html
60     |--- register.component.css
61 |--- services
62     |--- api.service.ts
63     |--- auth-interceptor.service.ts
64 |--- models
65     |--- form-field.ts
66 |--- (otros archivos de configuración)
67 |--- index.html
68 |--- styles.css
69 |--- main.ts
70 |--- main.server.ts
71 |--- (otros archivos de configuración)
72 |--- img
73     |--- logo_abet.png
74     |--- logo_episunsa.png
75     |--- logo_unsa.jpg
76     |--- E1.png
77     |--- E2.png
78     |--- E3.png
79     |--- commits1.png
80     |--- commits2.png
81     |--- commits3.png
82     |--- commits4.png
```

7 Referencias

- <https://angular.dev/tutorials/learn-angular>
- <https://www.djangoproject.com/>