

ASUNTO:	<b>REPORTE TÉCNICO</b>		
SPRINT:	<b>3</b>	FECHA:	<b>1 DE JUNIO AL 31 DE OCTUBRE 2022</b>
DESCRIPCION	Programar los servicios web en la capa SOA		
OBJETIVO	Obtener los recursos de servicios web en la capa Back-End SOA y programar las primeras pantallas del FRONT-END		

## TECNOLOGIA IMPLEMENTADA

**Python, Django, Rest\_Framework, CORSHEADERS**

```
# Application definition
```

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'backend',  
    'corsheaders',  
    'rest_framework',  
]
```

## RECURSOS API CONSTRUIDOS

```
urlpatterns = [  
    path('', inicio),  
    path('inicio/', inicio),  
    path('noticias/', noticias),  
    path('noticias/<int:anio>/<int:mes>', noticias),  
    path('acercade/', acercaDe),  
  
    path('estado/', estado),  
    path('version/', version),  
    path('orden/', orden),  
    path('proceso/', proceso),  
    path('palabra/', palabra),  
    path('palabra/<str:proceso>', palabra),  
    path('palabra/<str:proceso>/<str:tipo>', palabra),  
    path('aula/', construccion),  
    path('actividad/', construccion),  
    # path('galeria/', galeria),  
    path('pdf/', pdf),  
  
    path('api/estado/', estados),  
    path('api/estado/<int:id>', estados),  
    path('api/version/', versiones),  
    path('api/version/<int:id>', versiones),  
    path('api/orden/', ordenes),  
    path('api/orden/<int:id>', ordenes),
```

```

path('api/proceso/' , procesos),
path('api/proceso/<int:id>' , procesos),
path('api/palabra/' , palabras),
path('api/palabra/<str:proceso>' , palabras),
path('api/palabra/<str:proceso>/<str:tipo>' , palabras),
path('api/aula/' , aulas),
path('api/actividad/' , actividades),
path('api/recurso/' , recursos),
path('api/instrumento/' , instrumentos),

path('api/auth/login' , login),
path('api/auth/logout' , logout),
path('api/auth/refresh' , refresh),
#path('api/auth/user' , user),

#path('rest/' , include('rest_framework.urls'))

path('admin/' , admin.site.urls),
]+static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)

```

RECURSO	/inicio
URI	<a href="https://flicclassroom.com/api/inicio">https://flicclassroom.com/api/inicio</a>

```

def inicio(request):
    fecha=datetime.datetime.now()
    return render(request, 'inicio.html', {"titulo":"Inicio", "fecha":fecha})

```

RECURSO	/noticias
URI	<a href="https://flicclassroom.com/api/noticias">https://flicclassroom.com/api/noticias</a>

```

def noticias(request,anio=0,mes=0):
    if anio==0: anio=datetime.datetime.now().year
    if mes ==0: mes=datetime.datetime.now().month
    fecha=datetime.datetime.now()
    return render(request, 'noticias.html', {"titulo":"Noticias",
"fecha":fecha,"anio":anio,"mes":mes})

```

RECURSO	/acercade
URI	<a href="https://flicclassroom.com/api/acercade">https://flicclassroom.com/api/acercade</a>

```

def acercaDe(request):
    return render(request, 'acercaDe.html', {"titulo":"Acerca de..."})

```

RECURSO	/estado
URI	<a href="https://flicclassroom.com/api/estado">https://flicclassroom.com/api/estado</a>

```

def estado(request):
    listado = models.Estado.objects.all

    return render(request, 'estado.html', {"titulo":"Estado",
"listado":listado})

```

RECURSO	/estados
URI	https://fliclassroom.com/api/estados

```
def estados(request, id=''):
    listado =
models.Estado.objects.all().filter(Q(est_estado=models.EEstado.ACTIVO))
    if id!='':
        #print(id)
        listado = listado.filter(Q(est_id=id))

    data = []
    for lis in listado:
        data.append(lis.toJson())

    return JsonResponse(data, safe=False)
```




[[{"id": 0, "nombre": "Inactivo", "descripcion": "Estado Inactivo usado en todas las tablas", "tipo": "Todas"}, {"id": 1, "nombre": "Activo", "descripcion": "Estado Activo usado en todas las tablas", "tipo": "Todas"}, {"id": 2, "nombre": "Iniciada", "descripcion": "Estado Iniciada usado en todas las tablas", "tipo": "Todas"}, {"id": 3, "nombre": "Programada", "descripcion": "Actividad Programada", "tipo": "Aula"}, {"id": 4, "nombre": "Calificado", "descripcion": "Actividad Calificado", "tipo": "Aula"}]]

RECURSO	/versiones
URI	https://fliclassroom.com/api/versiones

```
def versiones(request, id=''):
    listado =
models.Version.objects.all().filter(Q(ver_estado=models.EEstado.ACTIVO))
    if id!='':
        #print(id)
        listado = listado.filter(Q(ver_id=id))

    data = []
    for lis in listado:
        data.append(lis.toJson())

    return JsonResponse(data, safe=False)
```



[[{"id": 2, "nombre": "Versi\u00f3n 2", "descripcion": "Segunda Versi\u00f3n Registrada", "a\u00f1o": 2001}]]

RECURSO	/orden
URI	https://fliclassroom.com/api/orden

```
def orden(request):
    listado = models.Orden.objects.all

    return render(request, 'orden.html', {"titulo":"Orden",
"listado":listado})
```

RECURSO	/ordenes
URI	https://fliclassroom.com/api/ordenes

```
def ordenes(request, id=''):
    listado = models.Orden.objects.all()
    data = []
    for lis in listado:
        data.append(lis.toJson())

    return JsonResponse(data, safe=False)
```



RECURSO	/proceso
URI	https://fliclassroom.com/api/proceso

```
def proceso(request):
    listado = models.Proceso.objects.all

    return render(request, 'proceso.html', {"titulo":"Proceso",
"listado":listado})
```



RECURSO	/procesos
URI	https://fliclassroom.com/api/procesos

```
def procesos(request, id=''):
```

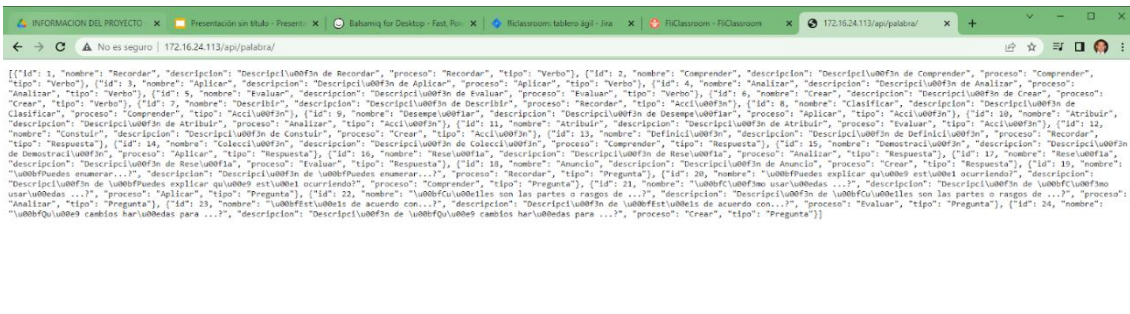
```
listado =
models.Proceso.objects.all().filter(Q(pro_estado=models.EEstado.ACTIVO)).orde
r_by('-pro_posicion')
data = []
for lis in listado:
    data.append(lis.toJson())

return JsonResponse(data, safe=False)
```

RECURSO	/palabras
URI	https://fliclassroom.com/api/palabras

```
def palabras(request, proceso='', tipo=''):
    listado =
models.Palabra.objects.all().filter(Q(pal_estado=models.EEstado.ACTIVO))
    if proceso!='':
        if len(proceso) > 1:
            proceso=models.Proceso.objects.filter(Q(pro_nombre=proceso))
            if len(proceso) > 0:
                proceso = proceso[0].pro_id
            else:
                proceso = '0'
        listado = listado.filter(Q(pal_fkproceso=proceso))
    if tipo!='':
        if len(tipo) > 1:
            try:
                tipo=models.EPalabra[tipo.upper()]
            except:
                pass
        listado = listado.filter(pal_tipo=tipo)
    data = []
    for lis in listado:
        data.append(lis.toJson())

    return JsonResponse(data, safe=False)
```



[[{"id": 1, "nombre": "Recordar", "descripcion": "Descripci\u00f3n de Recordar", "proceso": "Recordar", "tipo": "Verbo"}, {"id": 2, "nombre": "Comprender", "descripcion": "Descripci\u00f3n de Comprender", "proceso": "Comprender", "tipo": "Verbo"}, {"id": 3, "nombre": "Aplicar", "descripcion": "Descripci\u00f3n de Aplicar", "proceso": "Aplicar", "tipo": "Verbo"}, {"id": 4, "nombre": "Analizar", "descripcion": "Descripci\u00f3n de Analizar", "proceso": "Analizar", "tipo": "Verbo"}, {"id": 5, "nombre": "Evaluar", "descripcion": "Descripci\u00f3n de Evaluar", "proceso": "Evaluar", "tipo": "Verbo"}, {"id": 6, "nombre": "Crear", "descripcion": "Descripci\u00f3n de Crear", "proceso": "Crear", "tipo": "Verbo"}, {"id": 7, "nombre": "Describir", "descripcion": "Descripci\u00f3n de Describir", "proceso": "Recordar", "tipo": "Acci\u00f3n"}, {"id": 8, "nombre": "Clasificar", "descripcion": "Descripci\u00f3n de Clasificar", "proceso": "Comprender", "tipo": "Acci\u00f3n"}, {"id": 9, "nombre": "Desempe\u00f1ar", "descripcion": "Descripci\u00f3n de Desempe\u00f1ar", "proceso": "Aplicar", "tipo": "Acci\u00f3n"}, {"id": 10, "nombre": "Atribuir", "descripcion": "Descripci\u00f3n de Atribuir", "proceso": "Analizar", "tipo": "Acci\u00f3n"}, {"id": 11, "nombre": "Evaluar", "descripcion": "Descripci\u00f3n de Evaluar", "proceso": "Evaluar", "tipo": "Acci\u00f3n"}, {"id": 12, "nombre": "Construir", "descripcion": "Descripci\u00f3n de Construir", "proceso": "Crear", "tipo": "Acci\u00f3n"}, {"id": 13, "nombre": "Definir", "descripcion": "Descripci\u00f3n de Definir", "proceso": "Recordar", "tipo": "Acci\u00f3n"}, {"id": 14, "nombre": "Colectar", "descripcion": "Descripci\u00f3n de Colectar", "proceso": "Comprender", "tipo": "Respuesta"}, {"id": 15, "nombre": "Demostrar", "descripcion": "Descripci\u00f3n de Demostrar", "proceso": "Aplicar", "tipo": "Respuesta"}, {"id": 16, "nombre": "Resolver", "descripcion": "Descripci\u00f3n de Resolver", "proceso": "Analizar", "tipo": "Respuesta"}, {"id": 17, "nombre": "Rese\u00f1ar", "descripcion": "Descripci\u00f3n de Rese\u00f1ar", "proceso": "Evaluar", "tipo": "Respuesta"}, {"id": 18, "nombre": "Anunciar", "descripcion": "Descripci\u00f3n de Anunciar", "proceso": "Crear", "tipo": "Respuesta"}, {"id": 19, "nombre": "Puedes enumerar...", "descripcion": "Descripci\u00f3n de Puedes enumerar...", "proceso": "Recordar", "tipo": "Pregunta"}, {"id": 20, "nombre": "Puedes explicar qu\u00e9 est\u00e1 ocurriendo", "descripcion": "Descripci\u00f3n de Puedes explicar qu\u00e9 est\u00e1 ocurriendo", "proceso": "Comprender", "tipo": "Pregunta"}, {"id": 21, "nombre": "Puedes usar palabras...", "descripcion": "Descripci\u00f3n de Puedes usar palabras...", "proceso": "Aplicar", "tipo": "Pregunta"}, {"id": 22, "nombre": "Puedes explicar las partes o rangos de...", "descripcion": "Descripci\u00f3n de Puedes explicar las partes o rangos de...", "proceso": "Analizar", "tipo": "Pregunta"}, {"id": 23, "nombre": "Puedes explicar qu\u00e9 est\u00e1 ocurriendo", "descripcion": "Descripci\u00f3n de Puedes explicar qu\u00e9 est\u00e1 ocurriendo", "proceso": "Evaluar", "tipo": "Pregunta"}, {"id": 24, "nombre": "Puedes explicar qu\u00e9 est\u00e1 ocurriendo", "descripcion": "Descripci\u00f3n de Puedes explicar qu\u00e9 est\u00e1 ocurriendo", "proceso": "Crear", "tipo": "Pregunta"}]]

RECURSO	/login
URI	https://fliclassroom.com/api/login

```
@api_view(['POST']) #@csrf_exempt #@require_http_methods(["POST"])
#@csrf_exempt #@api_view(['POST']) #@csrf_exempt
#@permission_classes([AllowAny, ])
def login(request):
    # user = User.objects.create_user('usuario@correo.com',
    'usuario@correo.com', '12341234')
    # user.first_name = 'Usuario'
    # user.last_name = 'Usuario'
    # user.save()

    # u = User.objects.get(username='jona')
    # u.set_password('12341234')
    # u.save()

    # print("W"*20)
    # print(request.META)
    # print("W"*20)
    data = []
    if request.method=='POST' and request.data:
        data = [{'status' : 400}]
        username = '*'
        password = '*'
        roles = 'alumno'
        rdata = request.data
        if "username" in rdata:
            username = rdata['username']
        if "password" in rdata:
            password = rdata['password']
        if "roles" in rdata and rdata['roles']=='docente':
            roles = rdata['roles']
        # print(username, password)

    try:
        addr = request.META.get('REMOTE_ADDR')
        ip = models.Fail.objects.get(ip=addr)
        diff = datetime.datetime.now() - ip.date #hrs =
divmod(diff.total_seconds(), 3600)[0]
        mins = divmod(diff.total_seconds(), 60)[0]
        if mins >= 1.0:
            ip.attempts = 1
        else:
            ip.attempts = ip.attempts + 1
        ip.save()
        data = [{'intentos' : ip.attempts}]
    except:
        ip = models.Fail.create(addr, username)

    if ip.attempts<=3 and username!='*' and password!='*':
        user = authenticate(username=username, password=password)
        if user is not None:
            ip.attempts = 0
            ip.save()
            token = models.Token.objects.filter(Q(username=username, ip=addr,
roles=roles)).last()
            if token is not None:
                diff = datetime.datetime.now() - token.date
                hrs = divmod(diff.total_seconds(), 3600)[0]
```



```

        if hrs>=3:
            token = token.update()
        else:
            token = models.Token.create(username, roles, addr)
            data = [{'token' : token.token, 'username' : username,
'fullName': user.first_name + ' ' + user.last_name, 'roles': roles}]

            #print(data)

        return JsonResponse(data, safe=False)

```

RECURSO	/logout
URI	https://fliclassroom.com/api/logout

```

@api_view(['POST'])
def logout(request):
    data = [{'status' : 400}]
    if request.method=='POST' and request.data:
        username = '*'
        token = '*'
        rdata = request.data
        if "username" in rdata:
            username = rdata['username']
        if "token" in rdata:
            token = rdata['token']

        try:
            addr = request.META.get('REMOTE_ADDR')
            ip = models.Fail.objects.get(ip=addr)
            diff = datetime.datetime.now() - ip.date #hrs =
divmod(diff.total_seconds(), 3600)[0]
            mins = divmod(diff.total_seconds(), 60)[0]
            if mins >= 1.0:
                ip.attempts = 1
            else:
                ip.attempts = ip.attempts + 1
            ip.save()
            data = [{'intentos' : ip.attempts}]
        except:
            ip = models.Fail.create(addr, username)

        if ip.attempts<=3 and username!='*' and token!='*':
            token = models.Token.objects.filter(Q(username=username,
token=token, ip=addr, active=1)).last()
            if token is not None:
                data = []
                token.logout()

        return JsonResponse(data, safe=False)

```

RECURSO	/aulas
URI	https://fliclassroom.com/api/aulas



```
@api_view(['POST'])
def aulas(request):
    data = [{'status' : 400}]
    if request.method=='POST' and request.data:
        username = '*'
        token = '*'
        aula = '*'
        nueva = '*'
        edita = '*'
        rdata = request.data
        if "username" in rdata: username = rdata['username']
        if "token" in rdata: token = rdata['token']
        if "aula" in rdata: aula = rdata['aula']
        if "nueva" in rdata: nueva = rdata['nueva']
        if "edita" in rdata: edita = rdata['edita']

        if username!='*' and token!='*':
            addr = request.META.get('REMOTE_ADDR')
            token = models.Token.objects.filter(Q(username=username,
            token=token, ip=addr, active=1)).last()
            if token is not None:
                data = []
                user = User.objects.get(username=username)

                #Cuando es nueva aula
                if nueva!='*' and "nombre" in nueva and "descripcion" in nueva
                and "proceso" in nueva and "estado" in nueva:
                    nueva = models.Aula.create(nueva, user)

                #Cuando edita un aula
                if edita!='*' and "nombre" in edita and "descripcion" in edita
                and "estado" in edita:
                    edita = models.Aula.update(edita, user)

                parts =
                models.Participante.objects.all().filter(Q(par_fkusuario=user.id,
                par_estado=models.EEstado.ACTIVO), Q(par_rol='Creador') |
                Q(par_rol='Colaborador') | Q(par_rol='Participante'))
                aulas = []
                for part in parts:
                    aulas.append(part.par_fkaula.aul_id)
                if len(aulas)>0:
                    if aula!="*":
                        listado = models.Aula.objects.all().filter(Q(aul_id=aula))
                    else:
                        listado =
                        models.Aula.objects.all().filter(Q(aul_id__in=aulas))
                        for lis in listado:
                            data.append(lis.toJson(username))
            #print(data)
            return JsonResponse(data, safe=False)
```



URI	<a href="https://flicclassroom.com/api/actividades">https://flicclassroom.com/api/actividades</a>
-----	---

```
@api_view(['POST'])
def actividades(request):
    data = [{'status' : 400}]
    if request.method=='POST' and request.data:
        username = '*'
        token = '*'
        aula = '*'
        nueva = '*'
        edita = '*'
        actividad = '*'
        recurso = '*'
        instrumento = '*'
        galeria = '*'
        galeriaInst = '*'
        rdata = request.data
        if "username" in rdata: username = rdata['username']
        if "token" in rdata: token = rdata['token']
        if "aula" in rdata: aula = rdata['aula']
        if "nueva" in rdata: nueva = rdata['nueva']
        if "edita" in rdata: edita = rdata['edita']
        if "actividad" in rdata: actividad = rdata['actividad']
        if "recurso" in rdata: recurso = rdata['recurso']
        if "instrumento" in rdata: instrumento = rdata['instrumento']
        if "galeria" in rdata: galeria = rdata['galeria']
        if "galeriaInst" in rdata: galeriaInst = rdata['galeriaInst']

        if username!='*' and token!='*' and aula!='*':
            addr = request.META.get('REMOTE_ADDR')
            token = models.Token.objects.filter(Q(username=username, token=token,
ip=addr, active=1)).last()
            if token is not None:
                data = []
                user = User.objects.get(username=username)

                #Cuando es nueva actividad
                if nueva!='*' and "nombre" in nueva and "descripcion" in nueva and
"inicio" in nueva and "fin" in nueva and "integrantes" in nueva and "puntaje" in
nueva and "estado" in nueva:
                    nueva = models.Actividad.create(aula, nueva, user)

                #Cuando edita una actividad
                if edita!='*' and "nombre" in edita and "descripcion" in edita and
"inicio" in edita and "fin" in edita and "integrantes" in edita and "puntaje" in
edita and "estado" in edita:
                    edita = models.Actividad.update(aula, edita, user)

                #Cuando edita un recurso
                if recurso!='*' and actividad!='*' and "nombre" in rdata and
"descripcion" in rdata and "publico" in rdata and "file" in rdata and "estado" in
rdata:
                    recurso = {"nombre": rdata['nombre'], "descripcion":
rdata['descripcion'], "publico": rdata['publico'], "file": rdata['file'], "estado":
rdata['estado']}
                    recurso = models.Recurso.create(recurso, user)
                    edita = models.Actividad.recurso(aula, actividad, recurso, user)

                #Cuando edita un instrumento
                if instrumento!='*' and actividad!='*' and "nombre" in rdata and
"descripcion" in rdata and "file" in rdata and "estado" in rdata:
                    instrumento = {"nombre": rdata['nombre'], "descripcion":
rdata['descripcion'], "file": rdata['file'], "estado": rdata['estado']}
```



```
instrumento = models.Archivo.create(instrumento, user)
edita = models.Actividad.instrumento(aula, actividad, instrumento,
user)

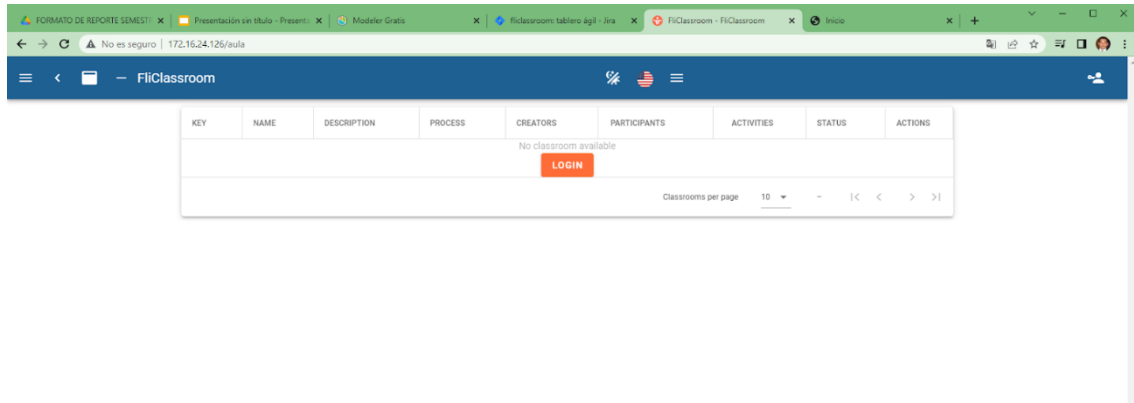
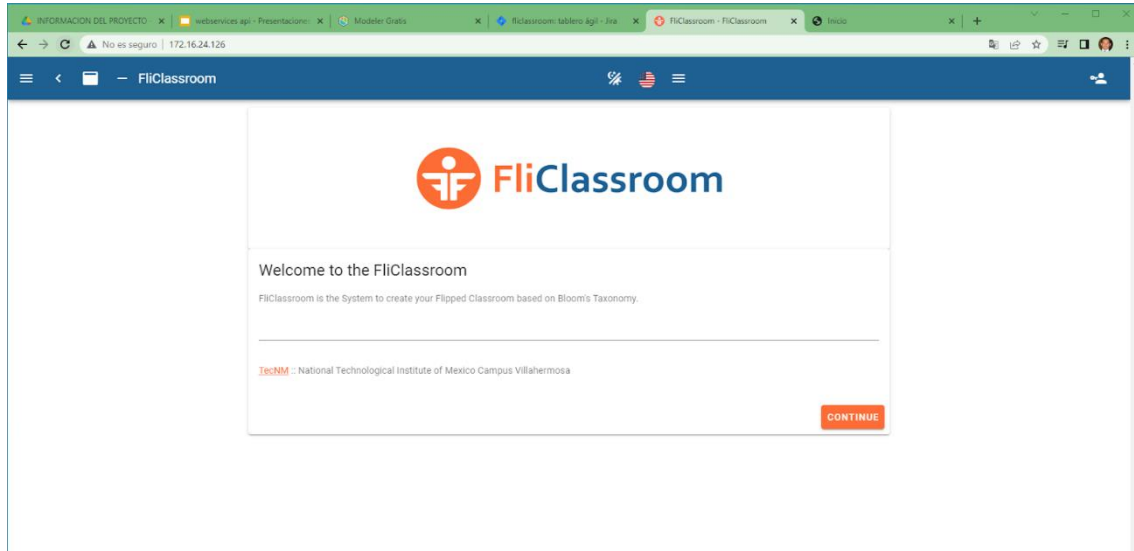
#Cuando es la galeria de recursos
if galeria!='*' and actividad!='*' and "galeriaId" in rdata:
    try:
        recurso =
models.Recurso.objects.get(rec_id=int(rdata['galeriaId']))
        if recurso is not None:
            actividad = models.Actividad.objects.get(act_id=actividad)
            if actividad is not None:
                actividad.act_fkrecurso = recurso
                actividad.act_modificador = user.id
                actividad.save()
    except:
        pass

#Cuando es la galeria de instrumentos
if galeriaInst!='*' and actividad!='*' and "galeriaId" in rdata:
    try:
        instrumento =
models.Archivo.objects.get(arc_id=int(rdata['galeriaId']))
        if instrumento is not None:
            actividad = models.Actividad.objects.get(act_id=actividad)
            if actividad is not None:
                actividad.act_fkinstrumento= instrumento
                actividad.act_modificador = user.id
                actividad.save()
    except:
        pass

parts = models.Participante.objects.all().filter(Q(par_fkusuario=user.id,
par_estado=models.EEstado.ACTIVO), Q(par_rol='Creador') |
Q(par_rol='Colaborador') | Q(par_rol='Participante'))
aulas = []
for part in parts:
    aulas.append(part.par_fkaula.aul_id)
aula = int(rdata['aula'])
if len(aulas)>0 and aula in aulas:
    listado = models.Actividad.objects.all().filter(Q(act_fkaula=aula))
    for lis in listado:
        data.append(lis.toJson(user.id))
#print(data)
return JsonResponse(data, safe=False)
```

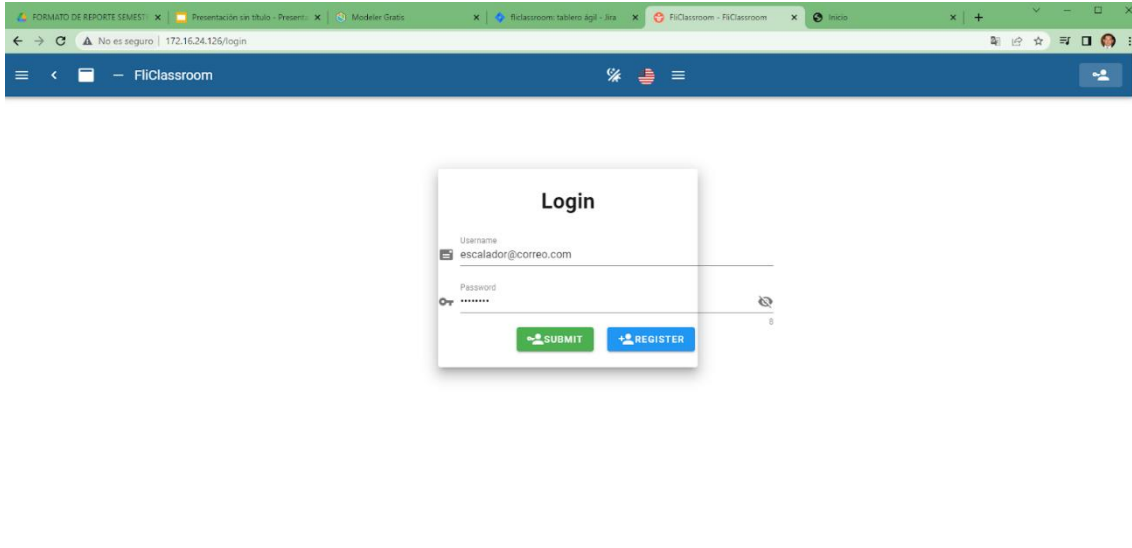
## PANTALLA INICIAL DE LA APLICACIÓN FRONT-END TECNOLOGÍA IMPLEMENTADA, VUEJS, VUETIFY

Introducción al sitio



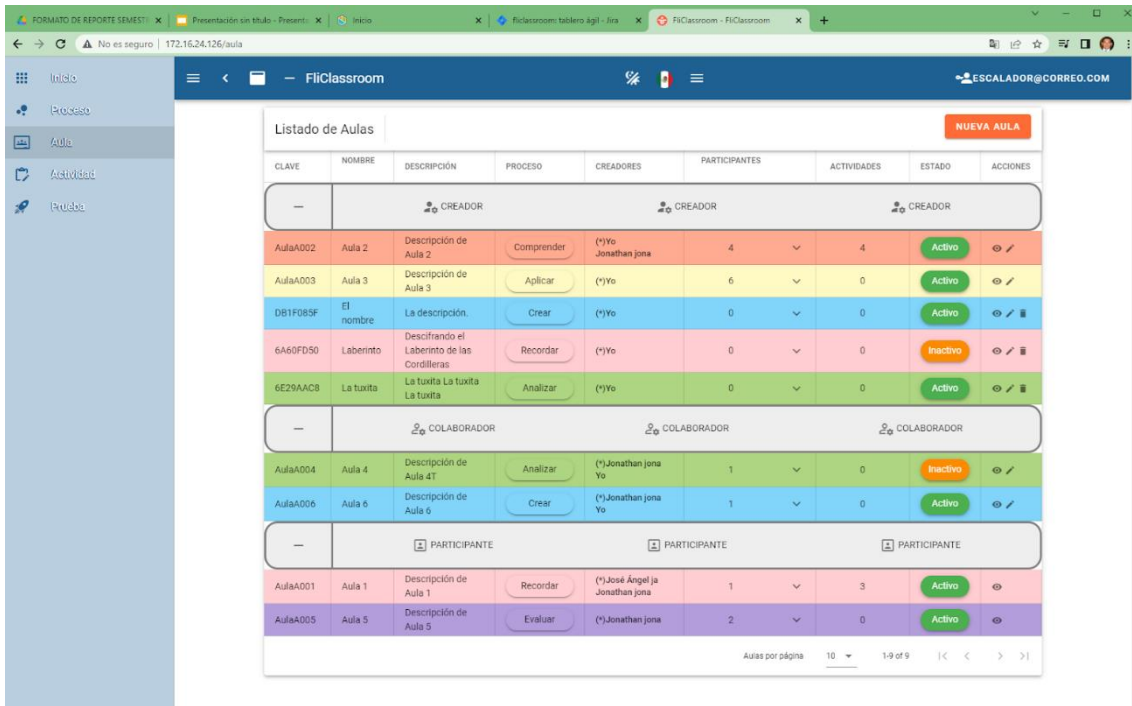
## PANTALLA LOGIN DE LA APLICACIÓN FRONT-END

### Login



## PANTALLA AULAS INVERTIDAS DE LA APLICACIÓN FRONT-END

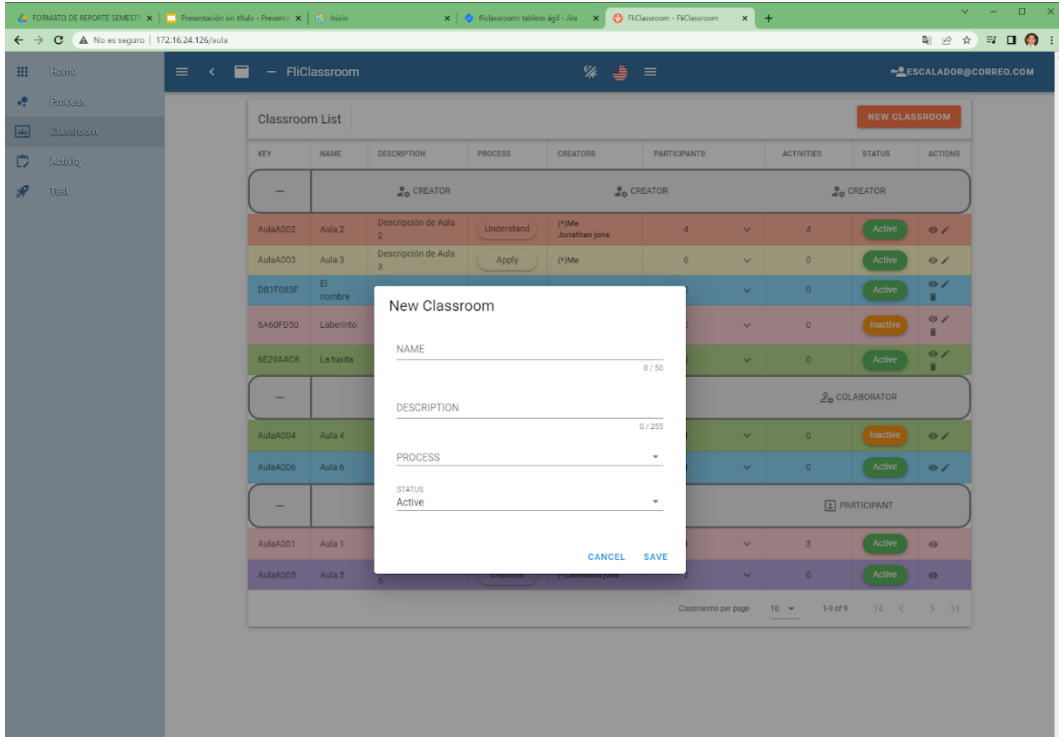
### Aulas invertidas



CLAVE	NOMBRE	DESCRIPCIÓN	PROCESO	CREADORES	PARTICIPANTES	ACTIVIDADES	ESTADO	ACCIONES
<div>CREADOR</div>								
AulaA002	Aula 2	Descripción de Aula 2	Comprender	(*)Yo Jonathan jona	4	4	Activo	
AulaA003	Aula 3	Descripción de Aula 3	Aplicar	(*)Yo	6	0	Activo	
D81F085F	El nombre	La descripción.	Crear	(*)Yo	0	0	Activo	
6AA0FD50	Laberinto	Descifrando el Laberinto de las Cordilleras	Recordar	(*)Yo	0	0	Inactivo	
6E29AAC8	La tuxita	La tuxita La tuxita	Analizar	(*)Yo	0	0	Activo	
<div>COLABORADOR</div>								
AulaA004	Aula 4	Descripción de Aula 4T	Analizar	(*)Jonathan jona Yo	1	0	Inactivo	
AulaA006	Aula 6	Descripción de Aula 6	Crear	(*)Jonathan jona Yo	1	0	Activo	
<div>PARTICIPANTE</div>								
AulaA001	Aula 1	Descripción de Aula 1	Recordar	(*)José Ángel ja Jonathan jona	1	3	Activo	
AulaA005	Aula 5	Descripción de Aula 5	Evaluar	(*)Jonathan jona	2	0	Activo	

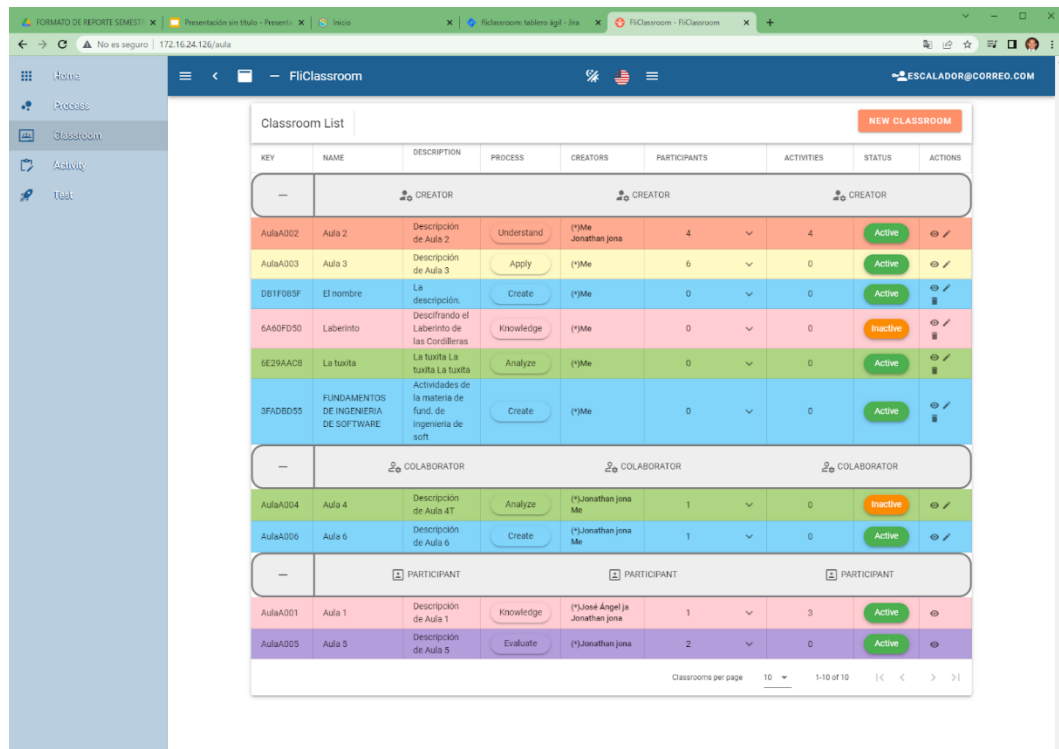
## PANTALLA CREAR AULAS INVERTIDAS DE LA APLICACIÓN FRONT-END

### Aulas invertidas



The screenshot shows the FliClassroom application interface. On the left is a sidebar with navigation links: Home, Process, Classroom, Activity, and Test. The main area displays a 'Classroom List' table. A 'NEW CLASSROOM' button is in the top right. A modal form titled 'New Classroom' is open in the center, allowing users to create a new classroom. The table lists classrooms with columns for KEY, NAME, DESCRIPTION, PROCESS, CREATORS, PARTICIPANTS, ACTIVITIES, STATUS, and ACTIONS. The status can be 'Active' or 'Inactive'.

KEY	NAME	DESCRIPTION	PROCESS	CREATORS	PARTICIPANTS	ACTIVITIES	STATUS	ACTIONS
CREATOR								
AulaA002	Aula 2	Descripción de Aula 2	Understand	(*)Me Jonathan jona	4	4	Active	
AulaA003	Aula 3	Descripción de Aula 3	Apply	(*)Me	6	0	Active	
DB1F085F	El nombre				0	0	Active	
6A60FD50	Laberinto				0	0	Inactive	
6E29AAC8	La tuxita				0	0	Active	
COLABORATOR								
AulaA004	Aula 4				0	0	Inactive	
AulaA006	Aula 6				0	0	Active	
PARTICIPANT								
AulaA001	Aula 1				3	0	Active	
AulaA005	Aula 5				0	0	Active	

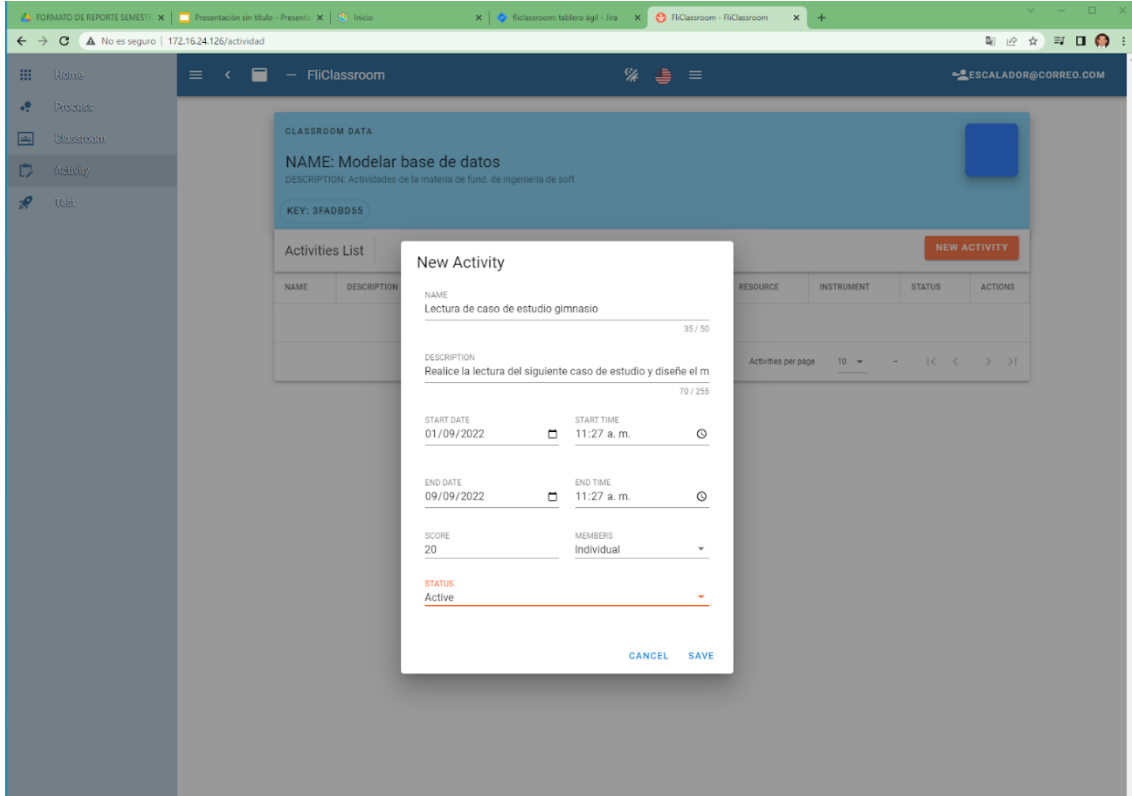


This screenshot shows the FliClassroom application with a more detailed view of the 'Classroom List'. The table includes additional information such as the creator's name, the number of participants, and the status of each classroom. The 'NEW CLASSROOM' button is still visible in the top right corner. The table is organized into sections for different roles: CREATOR, COLABORATOR, and PARTICIPANT.

KEY	NAME	DESCRIPTION	PROCESS	CREATORS	PARTICIPANTS	ACTIVITIES	STATUS	ACTIONS
CREATOR								
AulaA002	Aula 2	Descripción de Aula 2	Understand	(*)Me Jonathan jona	4	4	Active	
AulaA003	Aula 3	Descripción de Aula 3	Apply	(*)Me	6	0	Active	
DB1F085F	El nombre	La descripción.	Create	(*)Me	0	0	Active	
6A60FD50	Laberinto	Descifrando el Laberinto de las Cordilleras	Knowledge	(*)Me	0	0	Inactive	
6E29AAC8	La tuxita	La tuxita La tuxita	Analyze	(*)Me	0	0	Active	
3FADEB55	FUNDAMENTOS DE INGENIERIA DE SOFTWARE	Actividades de la materia de fund. de ingeniería de soft	Create	(*)Me	0	0	Active	
COLABORATOR								
AulaA004	Aula 4	Descripción de Aula 4T	Analyze	(*)Jonathan jona Me	1	0	Inactive	
AulaA006	Aula 6	Descripción de Aula 6	Create	(*)Jonathan jona Me	1	0	Active	
PARTICIPANT								
AulaA001	Aula 1	Descripción de Aula 1	Knowledge	(*)Jose Angel ja Jonathan jona	1	3	Active	
AulaA005	Aula 5	Descripción de Aula 5	Evaluate	(*)Jonathan jona	2	0	Active	

## PANTALLA CREAR ACTIVIDAD DE LA APLICACIÓN FRONT-END

### Crear Actividad

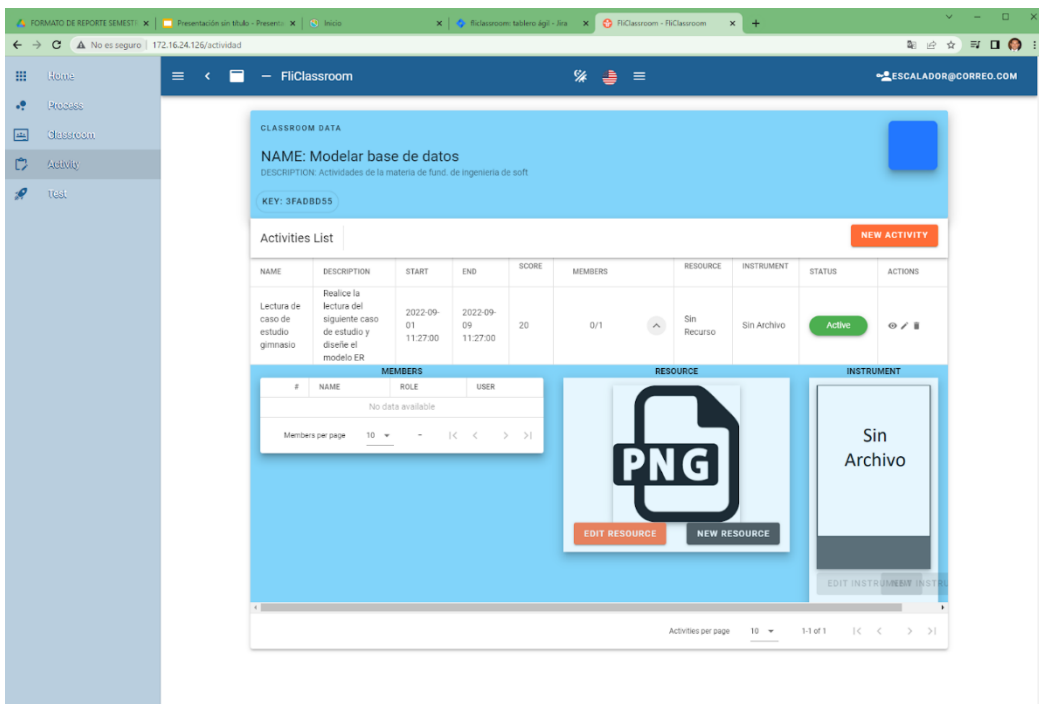


The screenshot shows the FliClassroom web application interface. A modal titled "New Activity" is open, allowing the user to create a new activity. The modal contains the following fields:

- NAME:** Lectura de caso de estudio gimnasio (35 / 50 characters)
- DESCRIPTION:** Realice la lectura del siguiente caso de estudio y diseñe el m (70 / 255 characters)
- START DATE:** 01/09/2022
- START TIME:** 11:27 a. m.
- END DATE:** 09/09/2022
- END TIME:** 11:27 a. m.
- SCORE:** 20
- MEMBERS:** Individual
- STATUS:** Active

At the bottom of the modal are "CANCEL" and "SAVE" buttons. The background shows the "CLASSROOM DATA" section with the activity name "Modelar base de datos" and a "NEW ACTIVITY" button.

### Subir recursos archivos a la actividad



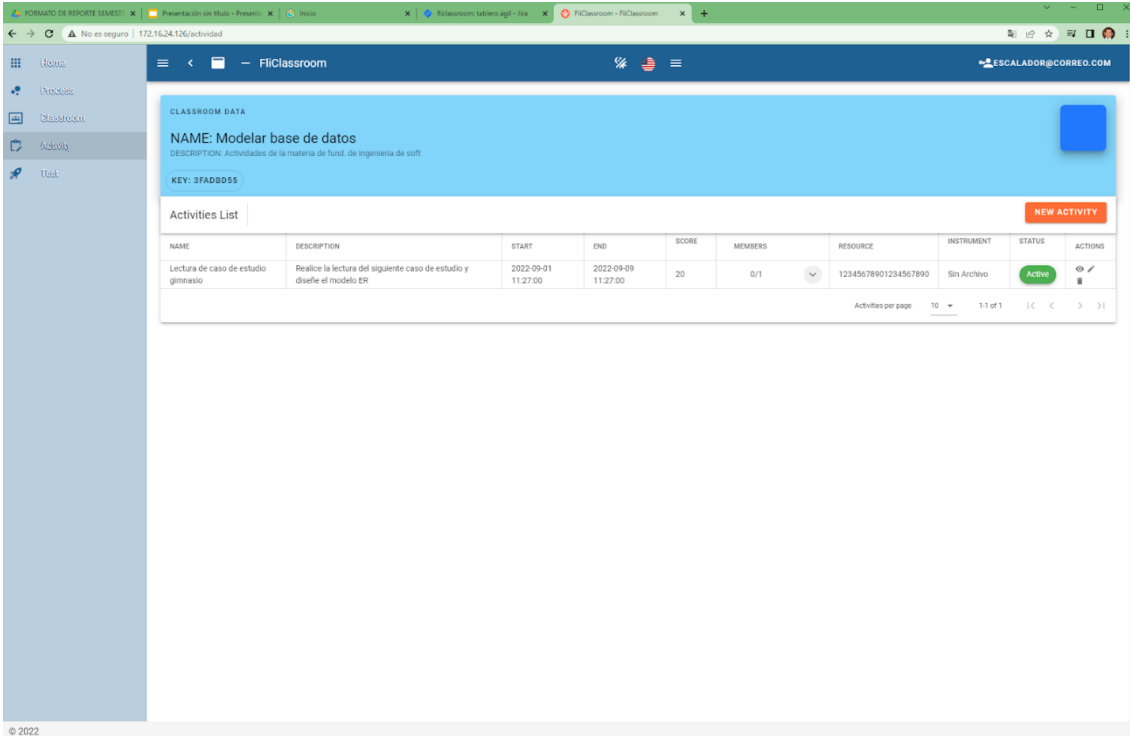
The screenshot shows the FliClassroom web application interface. The "Activities List" table displays the activity details, and the "RESOURCE" section shows the upload process.

NAME	DESCRIPTION	START	END	SCORE	MEMBERS	RESOURCE	INSTRUMENT	STATUS	ACTIONS
Lectura de caso de estudio gimnasio	Realice la lectura del siguiente caso de estudio y diseñe el modelo ER	2022-09-01 11:27:00	2022-09-09 11:27:00	20	0/1	Sin Recurso	Sin Archivo	Active	✎ ⚙

The "RESOURCE" section shows a large "PNG" icon with "EDIT RESOURCE" and "NEW RESOURCE" buttons. The "INSTRUMENT" section shows "Sin Archivo" with an "EDIT INSTRUMENT" button.

## PANTALLA LISTA DE ACTIVIDAD DE LA APLICACIÓN FRONT-END



Listado de actividades



The screenshot shows the FliClassroom web application interface. On the left is a sidebar with navigation links: Home, Modules, Classroom, Activity, and User. The main content area has a header with 'FliClassroom' and a user profile 'ESCALADOR@CORREO.COM'. Below the header, there's a 'CLASSROOM DATA' section with the following information:

- NAME:** Modelar base de datos
- DESCRIPTION:** Actividades de la materia de fund. de ingeniería de soft
- KEY:** 3FAD8D55

Below this is the 'Activities List' section, which contains a table with one activity. The table has columns: NAME, DESCRIPTION, START, END, SCORE, MEMBERS, RESOURCE, INSTRUMENT, STATUS, and ACTIONS.

NAME	DESCRIPTION	START	END	SCORE	MEMBERS	RESOURCE	INSTRUMENT	STATUS	ACTIONS
Lectura de caso de estudio gimnasio	Realice la lectura del siguiente caso de estudio y diseñe el modelo ER	2022-09-01 11:27:00	2022-09-09 11:27:00	20	0/1	12345678901234567890	Sin Archivo	Active	 

At the bottom right of the table, there is a pagination control: 'Activities per page 10 1-1 of 1'.

© 2022