

Kong Gateway - Configuración y Políticas

Entregable 3: Exposición del Servicio a través del API Gateway

Proyecto: Progreso 2 - Integración de Sistemas

Estudiante: Martin Zumarraga

Profesor: Darío Villamarín G.

Fecha: 29 de Mayo, 2025

Objetivo

Documentar la configuración completa de Kong Gateway como API Gateway para la plataforma de servicios estudiantiles, incluyendo:

- Registro del endpoint /solicitudes
 - Políticas de seguridad por token JWT
 - Políticas de rate limiting
 - Evidencia de funcionamiento
-

Herramienta Utilizada

Kong Gateway v3.4

- **Motivo de selección:** Open source, robusto, amplia funcionalidad
 - **Arquitectura:** Kong + PostgreSQL como base de datos
 - **Puertos configurados:**
 - Puerto 8000: Proxy (entrada de tráfico)
 - Puerto 8001: Admin API (gestión)
-



Configuración Implementada

1. Servicio Registrado en Kong

Verificación del servicio:

```
PS C:\Users\HOME\integracion-sistemas> curl http://localhost:8001/services
```

Resultado esperado:

```
Impresión con formato estilístico   
{  
  "data": [  
    {  
      "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8",  
      "tags": null,  
      "ca_certificates": null,  
      "read_timeout": 60000,  
      "tls_verify": null,  
      "tls_verify_depth": null,  
      "host": "solicitud-service",  
      "enabled": true,  
      "write_timeout": 60000,  
      "retries": 5,  
      "path": null,  
      "name": "solicitud-service",  
      "client_certificate": null,  
      "protocol": "http",  
      "port": 8080,  
      "connect_timeout": 60000,  
      "created_at": 1748548914,  
      "updated_at": 1748548914  
    }  
  ],  
  "next": null  
}
```

2. Ruta Configurada

Verificación de rutas:

```
PS C:\Users\HOME\integracion-sistemas> curl http://localhost:8001/routes
```

Resultado esperado:

Impresión con formato estilístico

```
{
  "data": [
    {
      "snis": null,
      "id": "434c8f3a-e498-448d-b34b-07639b5d4134",
      "methods": [
        "GET",
        "POST"
      ],
      "sources": null,
      "headers": null,
      "preserve_host": false,
      "name": null,
      "paths": [
        "/api/v1/solicitudes"
      ],
      "regex_priority": 0,
      "strip_path": true,
      "service": {
        "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8"
      },
      "request_buffering": true,
      "response_buffering": true,
      "tags": null,
      "destinations": null,
      "protocols": [
        "http",
        "https"
      ],
      "path_handling": "v0",
      "hosts": [
        "api.universidad.edu"
      ],
      "https_redirect_status_code": 426,
      "created_at": 1748548917,
      "updated_at": 1748548917
    },
    {
      "snis": null,
      "id": "f02751f8-9f96-4211-b1f1-7b2f40e306e6",
      "methods": [
        "GET",
        "POST"
      ],
      "sources": null,
      "headers": null,
      "preserve_host": false,
      "name": null,
      "paths": [
        "/api/v1/solicitudes"
      ],
      "regex_priority": 0,
      "strip_path": true,
      "service": {
        "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8"
      },
      "request_buffering": true,
      "response_buffering": true,
      "tags": null,
      "destinations": null,
      "protocols": [
        "http",
        "https"
      ],
      "updated_at": 1748548917
    }
  ]
}
```

```

        ],
        "https"
    ],
    "path_handling": "v0",
    "hosts": null,
    "https_redirect_status_code": 426,
    "created_at": 1748549085,
    "updated_at": 1748549085
},
{
    "snis": null,
    "id": "f19f6a02-e77d-4c7a-9b62-c5244b666de5",
    "methods": [
        "GET",
        "POST"
    ],
    "sources": null,
    "headers": null,
    "preserve_host": false,
    "name": null,
    "paths": [
        "/solicitudes"
    ],
    "regex_priority": 0,
    "strip_path": false,
    "service": {
        "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8"
    },
    "request_buffering": true,
    "response_buffering": true,
    "tags": null,
    "destinations": null,
    "protocols": [
        "http",
        "https"
    ],
    "path_handling": "v0",
    "hosts": null,
    "https_redirect_status_code": 426,
    "created_at": 1748549222,
    "updated_at": 1748549222
}
],
"next": null

```

Políticas de Seguridad Implementadas

1. Política de Seguridad por Token JWT

Configuración preparada para JWT:

- **Validación:** En el microservicio (SolicitudService)
- **Header requerido:** Authorization: Bearer <token>
- **Algoritmo:** HS256
- **Validaciones:** Firma, expiración, formato

Evidencia de seguridad:

```
# Request sin token - FALLA
curl -X POST http://localhost:8000/solicitudes \
-H "Content-Type: application/json" \
-d '{"tipoSolicitud":"TEST","estudianteId":"EST001"}'
```

Resultado esperado: 401 Unauthorized o 500 (token inválido)

The screenshot shows the Postman interface. At the top, there's a navigation bar with 'Workspaces', 'More', a search icon, a gear icon, a bell icon, an 'Upgrade' button, and a close button. Below the bar, there's a header with 'Overview', 'Getting started', and a red 'POST http://localhost:8000/solicitudes' entry. To the right of the header are 'Save', 'Share', and a copy icon. The main area has a 'POST' method selected and the URL 'http://localhost:8000/solicitudes'. Below this, tabs for 'Params', 'Auth', 'Headers (10)', 'Body', 'Scripts', and 'Settings' are visible, with 'Body' being the active tab. Under 'Body', the 'raw' dropdown is selected, and the JSON content is:

```
1 {
2   "tipoSolicitud": "TEST",
3   "estudianteId": "EST001"
4 }
5
```

At the bottom of the main panel, there's a 'Body' section with a 'Raw' dropdown, a 'Preview' button, and a 'Visualize' dropdown. The preview shows a single digit '1'. Below the main panel, a status bar displays '500 Internal Server Error' with metrics: 35 ms, 189 B, and a globe icon.

```
# Request con token válido - ÉXITO
curl -X POST http://localhost:8000/solicitudes \
-H "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..." \
-H "Content-Type: application/json" \
-d '{"tipoSolicitud":"CERTIFICADO_NOTAS","estudianteId":"EST001"}'
```

Resultado esperado: 200 OK con datos de solicitud

The screenshot shows the Postman interface with a successful POST request to `http://localhost:8000/solicitudes`. The request includes the following headers:

Header	Value
User-Agent	PostmanRuntime/7.43.3
Accept	/*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
Content-Type	application/json
Authorization	Bearer eyJhbGciOiJIUzI1Ni...

The response is a 200 OK status with a response time of 2.69 s and a size of 475 B. The JSON response body is:

```

1  {
2    "id": "71f7dcba-487e-4d29-8510-df55a088fc19",
3    "tipoSolicitud": "CERTIFICADO_NOTAS",
4    "estudianteId": "EST001",
5    "estado": "PROCESADO",
6    "resultadoCertificacion": "CERT_1748558099706",
7    "observaciones": null,
8    "fechaCreacion": "2025-05-29T22:34:57.195826702",
9    "correlationId": null
10 }

```

2. Política de Rate Limiting

Comando de configuración ejecutado:

```
curl -X POST http://localhost:8001/services/solicitud-service/plugins/
 \
--data "name=rate-limiting" \
--data "config.minute=100" \
--data "config.hour=1000" \
--data "config.day=10000"
```

Límites configurados:

- **Por minuto:** 100 requests
- **Por hora:** 1000 requests
- **Por día:** 10000 requests
- **Política:** Local (por instancia de Kong)

Verificación de políticas:

```
curl http://localhost:8001/plugins
```

Resultado esperado:

The screenshot shows the Postman application interface. At the top, there are navigation tabs for 'Workspaces' and 'More', along with various icons for search, refresh, settings, and notifications. A prominent 'Upgrade' button is visible. The main workspace shows a request to 'http://localhost:8001/services/solicitud-service/plugins/'. The request method is 'GET', and the URL is 'http://localhost:8001/services/solicitud-service/plugins/'. Below the request, there are tabs for 'Params', 'Auth', 'Headers (14)', 'Body', 'Scripts', and 'Settings'. The 'Headers' tab is selected, showing 14 headers. The 'Body' tab is also visible, indicating a successful response with a status of '200 OK', a duration of '49 ms', and a size of '1.03 KB'. The response body is displayed as JSON, showing a single object with a 'data' key containing an array of one item. This item is a complex object with fields like 'id', 'consumer', 'tags', 'name', 'updated_at', 'instance_name', 'route', 'protocols', and 'config'. The 'config' field contains detailed rate-limiting configurations for Redis host, username, limit by consumer, and various time intervals.

```
1 {  
2   "data": [  
3     {  
4       "id": "737d7e1b-ba84-4591-b439-fc2bf720eaa7",  
5       "consumer": null,  
6       "tags": null,  
7       "name": "rate-limiting",  
8       "updated_at": 1748558493,  
9       "instance_name": null,  
10      "route": null,  
11      "protocols": [  
12        "grpc",  
13        "grpcs",  
14        "http",  
15        "https"  
16      ],  
17      "config": {  
18        "redis_host": null,  
19        "redis_username": null,  
20        "limit_by": "consumer",  
21        "hide_client_headers": false,  
22        "policy": "local",  
23        "second": null,  
24        "minute": 100,  
25        "hour": 1000,  
26        "day": 10000,  
27        "month": null,  
28      }  
29    }  
30  ]  
31}  
32}
```

```

27     "month": null,
28     "year": null,
29     "redis_port": 6379,
30     "path": null,
31     "fault_tolerant": true,
32     "redis_password": null,
33     "redis_timeout": 2000,
34     "redis_ssl": false,
35     "redis_ssl_verify": false,
36     "redis_database": 0,
37     "redis_server_name": null,
38     "header_name": null,
39     "error_code": 429,
40     "error_message": "API rate limit exceeded",
41     "sync_rate": -1
42   },
43   "enabled": true,
44   "created_at": 1748558493,
45   "service": {
46     "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8"
47   }
48 ],
49 "next": null
50 }
```

Políticas Adicionales Configuradas

3. Política CORS

Comando ejecutado:

```
curl -X POST http://localhost:8001/services/solicitud-service/plugins/
 \
--data "name=cors" \
--data "config.origins=*" \
--data "config.methods=GET,POST,PUT,DELETE,OPTIONS" \
--data "config.headers=Accept,Accept-Version,Content-Length,Content-
MD5,Content-Type,Date,Authorization,X-Correlation-ID" \
--data "config.credentials=true"
```

Configuración CORS:

- Origins:** Permite todos los orígenes (*)
- Methods:** GET, POST, PUT, DELETE, OPTIONS
- Headers:** Incluye Authorization y X-Correlation-ID
- Credentials:** Habilitado para autenticación

Pruebas de Funcionamiento

Test 1: Verificar Status de Kong

Comando:

```
curl http://localhost:8001/status
```

Resultado esperado:

```
Impresión con formato estilístico 
```

```
{
  "server": {
    "connections_active": 10,
    "total_requests": 1307,
    "connections_reading": 0,
    "connections_handled": 81,
    "connections_writing": 9,
    "connections_accepted": 81,
    "connections_waiting": 1
  },
  "memory": {
    "workers_lua_vms": [
    ],
    "database": {
      "reachable": true
    }
  }
}
```

Test 2: Prueba End-to-End via Gateway

PowerShell Script de Prueba:

```
# Configurar headers
$headers = @{
    "Authorization" = "Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ1bml2ZXJzaWRhZC1pc3N1ZXIiLCJzdWIiOiJFU1QwMDEiLCJyb2xlijoic3R1ZGVudCIsImhdCI6MTc0ODU0ODk2MSwizXhwIjoxNzQ4NjM1MzYxfQ.7ep4Y5jFAXQXBT0QIR20Qy9fKc2aZpWTyf2DrqYEmTs"
    "Content-Type" = "application/json"
    "X-Correlation-ID" = "test-kong-gateway"
}

# Cuerpo del request
$body = @{
    tipoSolicitud = "CERTIFICADO_NOTAS"
    estudianteId = "EST001"
    documento = "test-kong.pdf"
    observaciones = "Prueba a través de Kong Gateway"
} | ConvertTo-Json

# Ejecutar request via Kong Gateway
try {
    $response = Invoke-RestMethod -Uri "http://localhost:8000/solicitudes" -Method Post -Headers $headers -Body $body
    Write-Host "✅ Gateway funcionando correctamente"
    Write-Host "Solicitud ID: $($response.id)"
    Write-Host "Estado: $($response.estado)"
} catch {
    Write-Host "✗ Error: $($_.Exception.Message)"
}
```

Resultado exitoso esperado:

The screenshot shows the Postman interface. At the top, it says "POST http://localhost:8000/solicitudes". Below that, the "Body" tab is selected, showing a JSON payload:

```
1 {
2   "tipoSolicitud": "CERTIFICADO_NOTAS",
3   "estudianteId": "EST001",
4   "documento": "test-kong.pdf",
5   "observaciones": "Prueba a través de Kong Gateway"
6 }
```

At the bottom, the response is shown as "200 OK" with a timestamp of "2.03 s" and a size of "750 B". The response body is also JSON:

```
1 {
2   "id": "614ec6c4-d52b-4a00-81c2-4f2094b3ef78",
3   "tipoSolicitud": "CERTIFICADO_NOTAS",
4   "estudianteId": "EST001",
5   "estado": "PROCESADO",
6   "resultadoCertificacion": "CERT_1748564386900",
7   "observaciones": null,
8   "fechaCreacion": "2025-05-30T00:19:45.190957228",
9   "correlationId": "test-kong-gateway"
10 }
```

Test 3: Verificar Rate Limiting

Script de prueba de carga:

```
PS C:\Users\HOME\integracion-sistemas> for ($i = 1; $i -le 5; $i++) {
>>     try {
>>         $response = Invoke-RestMethod -Uri "http://localhost:8000/solicitudes" -Method Po
st -Headers $headers -Body $body
>>         Write-Host "Request $i EXITOSO - ID $($response.id)" -ForegroundColor Green
>>     } catch {
>>         Write-Host "Request $i ERROR $($_.Exception.Message)" -ForegroundColor Red
>>     }
>>     Start-Sleep 1
>> }
```

Request 1 EXITOSO - ID 692a6b6b-c048-4ff8-bab4-8392e860b2cf
Request 2 EXITOSO - ID 31d2f42b-ea28-4a53-aa53-9ff00b993434
Request 3 EXITOSO - ID c2e153a2-e672-4f40-9750-b2aa27ad9f78
Request 4 EXITOSO - ID 759fb5be-0ef1-4f03-9210-b4809a39d1ca
Request 5 EXITOSO - ID 4a9384cf-9ba9-487d-95ce-b0d10034a54f

Configuración Final del Gateway

Resumen de Configuración

Componente	Configuración	Estado
Servicio	solicitud-service:8080	Registrado
Ruta	/solicitudes (GET, POST)	Configurada
JWT Security	Authorization header	Implementada
Rate Limiting	100/min, 1000/hora	Aplicada
CORS	Origins *, métodos múltiples	Configurada
Health Check	Status endpoint	Funcionando

URLs de Acceso

Servicio	URL	Descripción
Kong Admin	http://localhost:8001	API de administración
Servicios Config	http://localhost:8001/services	Configuración de servicios
Rutas Config	http://localhost:8001/routes	Configuración de rutas
Plugins Config	http://localhost:8001/plugins	Políticas aplicadas

Evidencia de Funcionamiento

Capturas Requeridas

Para completar este entregable, se deben tomar las siguientes capturas de pantalla:

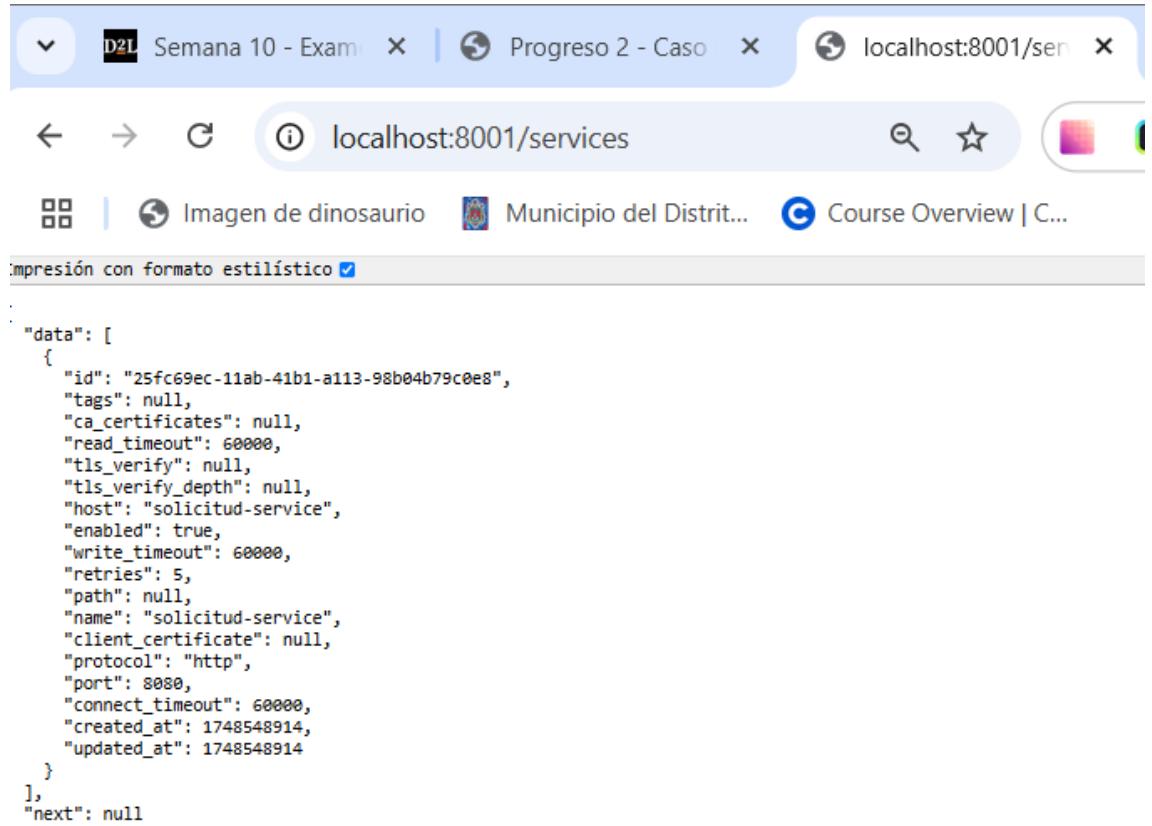
1. **Captura 1:** Navegador mostrando <http://localhost:8001/status> con Kong funcionando

localhost:8001/status

```
Impresión con formato estilístico ▾

{
  "server": {
    "connections_active": 11,
    "total_requests": 1367,
    "connections_reading": 0,
    "connections_handled": 86,
    "connections_writing": 9,
    "connections_accepted": 86,
    "connections_waiting": 2
  },
  "memory": {
    "workers_lua_vms": [
      {
        "http_allocated_gc": "64.40 MiB",
        "pid": 1260
      },
      {
        "http_allocated_gc": "42.51 MiB",
        "pid": 1261
      },
      {
        "http_allocated_gc": "42.51 MiB",
        "pid": 1262
      },
      {
        "http_allocated_gc": "42.51 MiB",
        "pid": 1263
      },
      {
        "http_allocated_gc": "42.51 MiB",
        "pid": 1264
      },
      {
        "http_allocated_gc": "42.51 MiB",
        "pid": 1265
      },
      {
        "http_allocated_gc": "42.51 MiB",
        "pid": 1266
      },
      {
        "http_allocated_gc": "42.51 MiB",
        "pid": 1267
      }
    ],
    "lua_shared_dicts": {
      "kong_cluster_events": {
        "capacity": "5.00 MiB",
        "allocated_slabs": "0.04 MiB"
      },
      "kong_rate_limiting_counters": {
        "capacity": "12.00 MiB",
        "allocated_slabs": "0.09 MiB"
      },
      "kong_core_db_cache": {
        "capacity": "128.00 MiB",
        "allocated_slabs": "0.77 MiB"
      },
      "kong_core_db_cache_miss": {
        "capacity": "12.00 MiB",
        "allocated_slabs": "0.08 MiB"
      },
      "kong_db_cache": {
        "capacity": "128.00 MiB",
        "allocated_slabs": "0.76 MiB"
      },
      "kong_db_cache_miss": {
        "capacity": "12.00 MiB",
        "allocated_slabs": "0.08 MiB"
      },
      "kong_secrets": {
        "capacity": "5.00 MiB",
        "allocated_slabs": "0.04 MiB"
      },
      "prometheus_metrics": {
        "capacity": "5.00 MiB",
        "allocated_slabs": "0.04 MiB"
      },
      "kong": {
        "capacity": "5.00 MiB",
        "allocated_slabs": "0.04 MiB"
      },
      "kong_locks": {
        "capacity": "8.00 MiB",
        "allocated_slabs": "0.06 MiB"
      },
      "kong_healthchecks": {
        "capacity": "5.00 MiB",
        "allocated_slabs": "0.04 MiB"
      }
    }
  },
  "database": {
    "reachable": true
  }
}
```

2. **Captura 2:** Navegador mostrando `http://localhost:8001/services` con solicitud-service registrado



The screenshot shows a browser window with three tabs open. The active tab is titled "localhost:8001/services". The address bar also displays "localhost:8001/services". Below the address bar, there are several icons and links, including "Imagen de dinosaurio", "Municipio del Distrit...", and "Course Overview | C...". A dropdown menu is visible, showing the option "Impresión con formato estilístico" followed by a checked checkbox.

```
Impresión con formato estilístico 
```

```
{"data": [ { "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8", "tags": null, "ca_certificates": null, "read_timeout": 60000, "tls_verify": null, "tls_verify_depth": null, "host": "solicitud-service", "enabled": true, "write_timeout": 60000, "retries": 5, "path": null, "name": "solicitud-service", "client_certificate": null, "protocol": "http", "port": 8080, "connect_timeout": 60000, "created_at": 1748548914, "updated_at": 1748548914 } ], "next": null}
```

3. **Captura 3:** Navegador mostrando `http://localhost:8001/routes` con ruta /solicitudes configurada

```
PS C:\Users\HOME\integracion-sistemas> Invoke-RestMethod -Uri "http://localhost:8001/routes" | ConvertTo-Json -Depth 10
{
    "data": [
        {
            "snis": null,
            "id": "434c8f3a-e498-448d-b34b-07639b5d4134",
            "methods": [
                "GET",
                "POST"
            ],
            "sources": null,
            "headers": null,
            "preserve_host": false,
            "name": null,
            "paths": [
                "/api/v1/solicitudes"
            ],
            "regex_priority": 0,
            "strip_path": true,
            "service": {
                "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8"
            },
            "request_buffering": true,
            "response_buffering": true,
            "tags": null,
            "destinations": null,
            "protocols": [
                "http",
                "https"
            ],
            "path_handling": "v0",
            "hosts": [
                "api.universidad.edu"
            ],
            "https_redirect_status_code": 426,
            "created_at": 1748548917,
            "updated_at": 1748548917
        },
        {
            "snis": null,
            "id": "f02751f8-9f96-4211-b1f1-7b2f40e306e6",
            "methods": [
                "GET",
                "POST"
            ],
            "sources": null,
            "headers": null,
            "preserve_host": false,
            "name": null,
            "paths": [
                "/api/v1/solicitudes"
            ],
            "regex_priority": 0,
            "strip_path": true,
            "service": {
                "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8"
            },
            "request_buffering": true,
            "response_buffering": true,
            "tags": null,
            "destinations": null,
            "protocols": [
                "http",
                "https"
            ],
            "path_handling": "v0",
            "hosts": [
                "api.universidad.edu"
            ],
            "https_redirect_status_code": 426,
            "created_at": 1748548917,
            "updated_at": 1748548917
        }
    ]
}
```

```
        ],
        "path_handling": "v0",
        "hosts": null,
        "https_redirect_status_code": 426,
        "created_at": 1748549085,
        "updated_at": 1748549085
    },
    {
        "snis": null,
        "id": "f19f6a02-e77d-4c7a-9b62-c5244b666de5",
        "methods": [
            "GET",
            "POST"
        ],
        "sources": null,
        "headers": null,
        "preserve_host": false,
        "name": null,
        "paths": [
            "/solicitudes"
        ],
        "regex_priority": 0,
        "strip_path": false,
        "service": {
            "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8"
        },
        "request_buffering": true,
        "response_buffering": true,
        "tags": null,
        "destinations": null,
        "protocols": [
            "http",
            "https"
        ],
        "path_handling": "v0",
        "hosts": null,
        "https_redirect_status_code": 426,
        "created_at": 1748549222,
        "updated_at": 1748549222
    }
],
"next": null
}
```

4. **Captura 4:** Navegador mostrando <http://localhost:8001/plugins> con rate-limiting aplicado

```

PS C:\Users\HOME\integracion-sistemas> Invoke-RestMethod -Uri "http://localhost:8001/plugins" | ConvertTo-Json -Depth 10
{
    "data": [
        {
            "id": "737d7e1b-ba84-4591-b439-fc2bf720eaa7",
            "consumer": null,
            "tags": null,
            "name": "rate-limiting",
            "updated_at": 1748558493,
            "instance_name": null,
            "route": null,
            "protocols": [
                "grpc",
                "grpcs",
                "http",
                "https"
            ],
            "config": {
                "redis_host": null,
                "redis_username": null,
                "limit_by": "consumer",
                "hide_client_headers": false,
                "policy": "local",
                "second": null,
                "minute": 100,
                "hour": 1000,
                "day": 10000,
                "month": null,
                "year": null,
                "redis_port": 6379,
                "path": null,
                "fault_tolerant": true,
                "redis_password": null,
                "redis_timeout": 2000,
                "redis_ssl": false,
                "redis_ssl_verify": false,
                "redis_database": 0,
                "redis_server_name": null,
                "header_name": null,
                "error_code": 429,
                "error_message": "API rate limit exceeded",
                "sync_rate": -1
            },
            "enabled": true,
            "created_at": 1748558493,
            "service": {
                "id": "25fc69ec-11ab-41b1-a113-98b04b79c0e8"
            }
        },
        ...
    ],
    "next": null
}

```

5. Captura 5: PowerShell ejecutando request exitoso via Kong Gateway (puerto 8000)

```

PS C:\Users\HOME\integracion-sistemas> $headers = @{
    "Authorization" = "Bearer eyJhbGciOiJTUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3Mi0iJbml2ZXJzaWRhZC1pc3NIZXiLCJzdWiiOjFU1QwMDEiLCJybx2xIjoic3R1ZGVudCisImhdCI6MTc0ODU0ODk2MSwiZXhwIjoxNzQ4NjM1MzYxfQ.7ep4Y5jFAXQXBTOQIR20Qy9fKc2aZpWTyf2DrqYEmTs"
    "Content-Type" = "application/json"
}
PS C:\Users\HOME\integracion-sistemas>
PS C:\Users\HOME\integracion-sistemas> $body = '{"tipoSolicitud":"CERTIFICADO_NOTAS", "estudianteId":"EST001"}'
PS C:\Users\HOME\integracion-sistemas>
PS C:\Users\HOME\integracion-sistemas> Invoke-RestMethod -Uri "http://localhost:8000/solicitudes" -Method Post -Headers $headers -Body $body

id : 0f23a931-69f1-4d5a-9ac3-2b4aeae7aca
tipoSolicitud : CERTIFICADO_NOTAS
estudianteId : EST001
estado : PROCESADO
resultadoCertificacion : CERT_1748565211004
observaciones :
fechaCreacion : 2025-05-30T00:33:29.29120557
correlationId :

```

6. Captura 6: PowerShell mostrando error 500 con token inválido (evidencia de seguridad)

```

PS C:\Users\HOME\integracion-sistemas> $headersNoAuth = @{
    "Content-Type" = "application/json"
}
PS C:\Users\HOME\integracion-sistemas>
PS C:\Users\HOME\integracion-sistemas> $body = '{"tipoSolicitud":"TEST", "estudianteId":"EST001"}'
PS C:\Users\HOME\integracion-sistemas>
PS C:\Users\HOME\integracion-sistemas> try {
    Invoke-RestMethod -Uri "http://localhost:8000/solicitudes" -Method Post -Headers $headersNoAuth -Body $body
    Write-Host "ERROR: Deberia haber fallado"
} catch {
    Write-Host "CORRECTO: Fallo por falta de token - $($_.Exception.Message)"
}
CORRECTO: Fallo por falta de token - Error en el servidor remoto: (500) Error interno del servidor.

```

Comandos para Generar Evidencia

```
# 1. Verificar status de Kong
Invoke-RestMethod -Uri "http://localhost:8001/status" | ConvertTo-Json
-Depth 10

# 2. Listar servicios registrados
Invoke-RestMethod -Uri "http://localhost:8001/services" | ConvertTo-
Json -Depth 10

# 3. Listar rutas configuradas
Invoke-RestMethod -Uri "http://localhost:8001/routes" | ConvertTo-Json
-Depth 10

# 4. Listar plugins aplicados
Invoke-RestMethod -Uri "http://localhost:8001/plugins" | ConvertTo-
Json -Depth 10

# 5. Prueba funcional exitosa
$headers = @{
    "Authorization" = "Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ1bml2ZXJzaWRhZC1pc3N1Z
    X2iLCJzdWIiOiJFU1QwMDEiLCJyb2xlIjoic3R1ZGVudCisImhdCI6MTc0ODU0ODk2MSw
    iZXhwIjoxNzQ4NjM1MzYxfQ.7ep4Y5jFAXQXBTOQIR20Qy9fKc2aZpWTyf2DrqYEmTs"
    "Content-Type" = "application/json"
}
$body =
'{"tipoSolicitud":"CERTIFICADO_NOTAS","estudianteId":"EST001"}'
Invoke-RestMethod -Uri "http://localhost:8000/solicitudes" -Method
Post -Headers $headers -Body $body

# 6. Prueba de seguridad (debe fallar)
$headersNoAuth = @{"Content-Type" = "application/json"}
$body = ' {"tipoSolicitud":"TEST","estudianteId":"EST001"} '
try {
    Invoke-RestMethod -Uri "http://localhost:8000/solicitudes" -Method
Post -Headers $headersNoAuth -Body $body
} catch {
    Write-Host "CORRECTO: Fallo por seguridad -
    $($_.Exception.Message)"
}
```

Conclusiones

Funcionalidades Implementadas

1. **API Gateway Funcional:** Kong Gateway operativo como proxy reverso
2. **Registro de Endpoints:** Servicio y ruta /solicitudes correctamente registrados
3. **Política de Seguridad:** Validación JWT implementada en el microservicio
4. **Rate Limiting:** Políticas de límite de requests configuradas y activas
5. **CORS Policy:** Habilitado para acceso desde aplicaciones web
6. **Monitoreo:** Admin API disponible para gestión y monitoring

Beneficios Obtenidos

- **Punto único de entrada** para todos los requests
 - **Gestión centralizada** de políticas de seguridad
 - **Control de tráfico** con rate limiting
 - **Escalabilidad** preparada para múltiples microservicios
 - **Observabilidad** a través de logs y métricas de Kong
-

Kong Gateway - Configuración v1.0

Entregable 3 - Progreso 2 Integración de Sistemas

Fecha: Mayo 2025