UNIVERSIDAD DE LAS AMERICAS

Facultad de Ingeniería y Ciencias Aplicadas

Diseño y Arq. de Software

Tema: Proyecto Fin de Curso

Docente: Carlos Alberto Balladares Enríquez

Integrante:

Amy Cherrez

Fecha entrega: 18 de julio del 2024

Tabla de Contenido

Contenido

1. Descripción de Requerimientos de software

1.1. Introducción

La Especificación de Requisitos de Software (SRS) proporciona una descripción detallada de los requisitos funcionales y no funcionales del Sistema de Evaluación de Proveedores. Este sistema está diseñado para gestionar de manera eficiente y efectiva la evaluación de proveedores, permitiendo a los usuarios realizar evaluaciones, calificaciones y seguimiento de los proveedores de manera sistemática.

La SRS está destinada a ser utilizada por el equipo de desarrollo de software, los usuarios finales del sistema y otras partes interesadas para comprender los objetivos y la funcionalidad del sistema. Se espera que la SRS evolucione a medida que se recopilen más requisitos y se realicen cambios en el sistema.

Esta especificación proporciona una visión general del Sistema de Evaluación de Proveedores, describiendo los actores involucrados, los objetivos del sistema, los requisitos funcionales y no funcionales, y las restricciones del sistema. Se espera que esta SRS sirva como base para el diseño, desarrollo, implementación y pruebas del sistema.

Objetivo
El proyecto tiene como objetivo principal desarrollar un Sistema de Evaluación de Proveedores que permita gestionar de manera eficiente y efectiva el proceso de evaluación y calificación de proveedores. El sistema facilitará la colaboración entre los diferentes actores involucrados en el proceso, garantizando la transparencia, la objetividad y la calidad en la selección de proveedores.

Alcance
El producto a desarrollar se identifica como el "Sistema de Evaluación de Proveedores" (SEP).

El SEP debe estar en consonancia con la Descripción del Sistema (DS) existente, si la hubiera, para garantizar la coherencia y la integración con otros sistemas o procesos de la organización. La DS proporcionaría una visión más amplia del entorno en el que operará el SEP, incluyendo su relación con otros sistemas, procesos y usuarios. La SRS se alinearía con la DS para asegurar que los requisitos del SEP estén alineados con los objetivos y restricciones más amplios del sistema en su conjunto.

Actores

Nombre	Rol	Categoría Profesional	Responsabilidades
Administrador	Gestión de Seguridad	Alta dirección	- Gestionar la seguridad del sistema. Configurar permisos de acceso.
Usuario	Gestión de Proveedores	Personal de compras	- Gestionar la información de los proveedores. Utilizar la matriz de evaluación. Calificar a los proveedores.
Proveedor	Registro de Formulario de Evaluación	Proveedor externo	- Completar el formulario de evaluación. Actualizar la información según sea necesario.
Perito	Validar Información de Formulario de Proveedores	Especialista en evaluación	- Validar la información de los proveedores. Asignar una calificación final.

1.2. Descripción General

Perspectiva del producto

El Sistema de Evaluación de Proveedores (SEP) es parte de un sistema mayor que incluye otros módulos de gestión de proveedores y procesos relacionados con compras y adquisiciones. El SEP interactúa con estos otros módulos para garantizar la integridad de los datos y la eficiencia en los procesos de evaluación y selección de proveedores.

Funcionalidad del Producto

El Sistema de Evaluación de Proveedores (SEP) debe proporcionar las siguientes funcionalidades principales:

Gestión de Proveedores:
- Registro y mantenimiento de la información de los proveedores.
- Gestión de la documentación relacionada con los proveedores.

Evaluación de Proveedores:
- Creación y mantenimiento de matrices de evaluación para diferentes tipos de proveedores.
- Realización de evaluaciones basadas en las matrices definidas.

Calificación de Proveedores:
- Asignación de calificaciones a los proveedores basadas en los resultados de las evaluaciones.
- Generación de informes de calificación para los proveedores.

Seguridad y Acceso:
- Gestión de usuarios y roles para garantizar un acceso seguro al sistema.
- Registro de actividades para auditoría y seguimiento.

Integración con Otros Sistemas:
- Integración con sistemas de compras y adquisiciones para compartir información relevante.
- Exportación de datos para análisis externo.

Características de los usuarios

Tipo de Usuario	Formación	Habilidades	Actividades
Administrador	Grado universitario en área relacionada.	Conocimientos en gestión de sistemas y seguridad informática.	Configuración y gestión de seguridad del sistema, asignación de permisos, supervisión de actividades.
Usuario	Grado universitario en área relacionada.	Conocimientos en gestión de proveedores y evaluación de riesgos.	Gestión de información de proveedores, uso de matrices de evaluación, calificación de proveedores.
Proveedor	No se especifica un nivel educativo específico.	Conocimientos en su área de especialización.	Registro y actualización de información en el sistema, participación en procesos de evaluación.
Perito	Grado universitario en área relacionada.	Experiencia en evaluación de proveedores y análisis de riesgos.	Validación de información de proveedores, asignación de calificaciones, análisis de riesgos.

Restricciones

El diseño y desarrollo del Sistema de Evaluación de Proveedores (SEP) debe tener en cuenta las siguientes restricciones:

- Metodologías de Desarrollo: El desarrollo del SEP debe seguir la metodología de desarrollo ágil Scrum, con entregas incrementales y en iteraciones cortas.

- Lenguajes de Programación: El sistema debe desarrollarse utilizando el lenguaje de programación Java para la parte del servidor, y JavaScript con el framework Angular para la parte del cliente.
- Normas Particulares: El SEP debe cumplir con las normas de seguridad de la información ISO/IEC 27001 y con las regulaciones locales de protección de datos personales.
- Restricciones de Hardware: El sistema debe ser compatible con los servidores disponibles en la infraestructura de la organización, que utilizan procesadores Intel de última generación y al menos 16 GB de RAM.
- Restricciones de Sistema Operativo: El SEP debe ser compatible con los sistemas operativos Windows Server 2016 y Ubuntu Server 18.04 LTS.

Suposiciones y Dependencias

El Sistema de Evaluación de Proveedores (SEP) se basa en las siguientes suposiciones y dependencias:

- Disponibilidad de Hardware y Software: Se asume que el hardware y software requeridos para ejecutar el SEP estarán disponibles según las especificaciones mencionadas en las restricciones.
- Acceso a Internet: Se supone que los usuarios del SEP tendrán acceso a Internet para poder utilizar todas las funcionalidades del sistema, incluyendo la integración con otros sistemas y la descarga de actualizaciones.
- Disponibilidad de Recursos Humanos: Se asume que habrá personal capacitado disponible para la implementación, mantenimiento y soporte del SEP, incluyendo administradores de sistemas, personal de seguridad informática y usuarios finales.
- Cambios en las Normativas: Se reconoce que cambios en las normativas locales o internacionales relacionadas con la seguridad de la información y protección de datos pueden afectar los requisitos del SEP, y se compromete a mantener el sistema actualizado en consecuencia.
- Compatibilidad con Otros Sistemas: Se depende de la disponibilidad y compatibilidad con otros sistemas con los que el SEP pueda integrarse, como sistemas de compras y adquisiciones, para garantizar su funcionalidad completa.

1.3. Requisitos Específicos

1.1. Requisitos de los interfaces

Descripción detallada de todas las entradas y salidas del sistema de software.

1.1.1. Interfaces de usuario

- Descripción de los requisitos del interfaz de usuario para el producto, incluyendo estilo, colores y apariencia general.
- Detalles específicos sobre cómo se verá y funcionará el producto para el usuario final.

1.1.2. Interfaces de hardware

- Especificación de las características lógicas para cada interfaz entre el producto y los componentes de hardware del sistema.
- Inclusión de características de configuración necesarias para la interoperabilidad con el hardware.

1.1.3. Interfaces de software

Indicación de la necesidad de integración del producto con otros productos de software.

Para cada producto de software, se debe incluir:

- Descripción del producto utilizado.
- Propósito del interfaz.
- Definición del interfaz, incluyendo contenido y formato.

1.1.4. Interfaces de comunicación

- Descripción de los requisitos de los interfaces de comunicación si hay comunicaciones con otros sistemas.
- Especificación de los protocolos de comunicación que se utilizarán para garantizar la interoperabilidad y seguridad de las comunicaciones.

1.2. Casos de Uso

1.2.1. Caso de Uso 1

Caso de Uso 1	Registrar Proveedor
Actores	Usuario
Descripción	*Este caso de uso describe la funcionalidad que permite a los usuarios registrar un nuevo proveedor en el sistema. El usuario deberá ingresar la información básica del proveedor, como nombre, dirección, contacto, etc., y esta información se almacenará en la base de datos del sistema.*

1.2.2. Caso de Uso 2

Caso de Uso	Crear Matriz de Evaluación
Actores	Usuario
Descripción	*Este caso de uso describe la funcionalidad que permite a los usuarios crear una nueva matriz de evaluación para un tipo específico de proveedor. El usuario podrá definir los criterios de evaluación, asignar pesos a cada criterio y establecer los rangos de calificación. La matriz de evaluación creada estará disponible para ser utilizada en las evaluaciones de los proveedores.*

1.2.3. Caso de Uso 3

Caso de Uso N	Realizar Evaluación de Proveedor
Actores	Usuario, Perito
Descripción	*Este caso de uso describe la funcionalidad que permite a los usuarios realizar la evaluación de un proveedor utilizando una matriz de evaluación previamente creada. El usuario seleccionará la matriz de evaluación adecuada, ingresará los resultados de la evaluación para cada criterio y el sistema calculará la calificación final del proveedor.*

1.2.4. Caso de Uso 4

Caso de Uso N	Consultar Calificación de Proveedor
Actores	Usuario
Descripción	*Este caso de uso describe la funcionalidad que permite a los usuarios consultar la calificación de un proveedor específico. El usuario podrá ver la calificación final del proveedor, así como los detalles de la evaluación realizada y los criterios utilizados.*

1.2.5. Caso de Uso 5

Caso de Uso N	Gestionar Seguridad del Sistema
Actores	Administrador
Descripción	*Este caso de uso describe la funcionalidad que permite al administrador gestionar la seguridad del sistema. El administrador podrá realizar las siguientes acciones:* *Configurar permisos de acceso: El administrador podrá asignar permisos de acceso a los diferentes usuarios del sistema, especificando qué funcionalidades y datos pueden acceder.* *Supervisar actividades: El administrador podrá supervisar las actividades realizadas por los usuarios del sistema, como registros de acceso, cambios en la configuración, etc.* *Realizar copias de seguridad: El administrador podrá realizar copias de seguridad de la información del sistema para garantizar su disponibilidad en caso de fallos o incidentes.*

1.3. Requisitos no funcionales

1.3.1. Requisitos de rendimiento

- El sistema debe ser capaz de soportar al menos 100 usuarios concurrentes.
- El tiempo de respuesta para la mayoría de las transacciones debe ser inferior a 1 segundo.
- El sistema debe ser capaz de procesar al menos 1000 transacciones por minuto durante las horas pico.

1.3.2. Seguridad

- El sistema debe utilizar técnicas criptográficas para proteger la información confidencial de los proveedores y las evaluaciones.
- Debe haber un registro de actividades (logs) que registre todas las acciones realizadas en el sistema para facilitar la auditoría y la detección de posibles anomalías.
- Las funcionalidades de acceso y modificación de datos críticos deben estar restringidas y solo accesibles para usuarios autorizados.

1.3.3. Fiabilidad

- El sistema debe tener una tasa de fallos menor al 0.1% y los incidentes críticos deben ser resueltos en menos de 1 hora.
- El tiempo medio entre fallos (MTBF) debe ser de al menos 1000 horas.

1.3.4. Disponibilidad

- El sistema debe estar disponible al menos el 99.9% del tiempo de servicio planificado, excluyendo el tiempo de mantenimiento programado.

1.3.5. Mantenibilidad

- El sistema debe ser diseñado de manera modular para facilitar las actualizaciones y modificaciones futuras.
- Las tareas de mantenimiento deben realizarse principalmente por desarrolladores del equipo de desarrollo, con la posibilidad de que usuarios capacitados realicen tareas específicas como generación de estadísticas.

1.3.6. Portabilidad

- El sistema debe ser desarrollado en Java para garantizar su portabilidad a diferentes sistemas operativos.
- El código debe estar diseñado de manera que sea independiente del sistema operativo subyacente y fácilmente adaptable a diferentes entornos.

1.3.6.1. Otros requisitos

- El sistema debe cumplir con las normativas locales e internacionales de protección de datos y privacidad de la información.
- Se deben considerar requisitos culturales y políticos para asegurar que el sistema sea aceptado y usable en diferentes contextos culturales y políticos.
- Todos los requisitos legales relacionados con la operación del sistema deben ser cumplidos, incluyendo la protección de datos personales y la seguridad de la información.

1.4. Estándares de conformidad

Para garantizar la calidad y cumplimiento de los estándares en el proyecto del Sistema de Evaluación de Proveedores (SEP), se establecen los siguientes estándares de conformidad:

ETAPA 1: Diseño e Implementación

Diseño: Presentar el documento de Diseño del SEP, que incluya la arquitectura del sistema, diagramas de flujo, y especificaciones técnicas.
Implementación: Presentar el documento de Implementación del SEP, que describa cómo se llevará a cabo la construcción del sistema, las tecnologías a utilizar y los estándares de codificación.
ETAPA 2: Capacitación

Determinar los requerimientos de capacitación para el personal que utilizará el SEP, incluyendo el contenido del curso, la duración y la metodología de enseñanza.

ETAPA 3: Pruebas Beta y Ajustes

Realizar pruebas beta del SEP utilizando un protocolo de pruebas establecido previamente.
Realizar ajustes al sistema, documentos y artefactos versionados según los resultados de las pruebas beta.
Implantación del SEP en un entorno de producción para su validación final.

ETAPA 4: Puesta en Producción

Realizar la puesta en producción del SEP de acuerdo con un plan previamente establecido, que incluya la migración de datos, la capacitación del personal y la comunicación con los usuarios finales.

2. Metodologías integrales de solución del problema

2.1. Descripción del estilo arquitectónico de Microservicios

El estilo arquitectónico de microservicios es una metodología de diseño y desarrollo de software que organiza una aplicación como una colección de servicios pequeños y autónomos, cada uno con una funcionalidad específica y bien definida. Este enfoque se contrapone a la arquitectura monolítica, donde todos los componentes de la aplicación están integrados en un único bloque de código.

(i) Principales características de la arquitectura de microservicios

1. **Descomposición en servicios pequeños**:
   - Cada microservicio se encarga de una funcionalidad específica, lo que facilita su desarrollo, implementación y mantenimiento de manera independiente.
   - La descomposición permite que los equipos trabajen en paralelo en diferentes servicios, reduciendo el tiempo de desarrollo y aumentando la productividad.
2. **Independencia de despliegue**:
   - Los microservicios se pueden desplegar y actualizar de forma independiente, sin necesidad de detener o afectar a otros servicios.
   - Esta independencia mejora la capacidad de respuesta a los cambios y reduce los riesgos asociados con las actualizaciones y despliegues.
3. **Comunicación a través de API**:
   - Los microservicios se comunican entre sí a través de APIs bien definidas, generalmente usando protocolos HTTP/HTTPS y formatos de datos como JSON o XML.
   - Este enfoque desacopla los servicios, permitiendo que cada uno se desarrolle con tecnologías y lenguajes de programación diferentes si es necesario.

4. **Escalabilidad**:
    - Los microservicios permiten escalar únicamente aquellos componentes que realmente lo necesitan, en lugar de escalar toda la aplicación.
    - Esta escalabilidad granular mejora la eficiencia y optimiza el uso de recursos.
5. **Resiliencia y tolerancia a fallos**:
    - Cada microservicio puede diseñarse para manejar fallos de manera independiente, lo que aumenta la resiliencia de la aplicación en su conjunto.
    - Los fallos en un microservicio no necesariamente afectan el funcionamiento de otros, mejorando la robustez del sistema.
6. **Desarrollo y despliegue continuos**:
    - La arquitectura de microservicios facilita la integración y entrega continua (CI/CD), permitiendo un ciclo de vida de desarrollo más rápido y una entrega de valor más constante a los usuarios finales.
    - La automatización en pruebas y despliegues se ve favorecida, lo que reduce el tiempo de entrega y aumenta la calidad del software.

(ii) Desafíos de la arquitectura de microservicios

1. **Complejidad en la gestión**:
    - La gestión de múltiples servicios y sus interdependencias puede ser compleja, requiriendo herramientas avanzadas de orquestación y monitoreo.
    - Es necesario un enfoque robusto para manejar la configuración y la gestión de servicios, como Kubernetes para la orquestación de contenedores.
2. **Comunicación y latencia**:
    - La comunicación entre servicios introduce latencia, que debe gestionarse adecuadamente para no afectar el rendimiento global de la aplicación.
    - Es fundamental diseñar estrategias de comunicación eficientes y usar patrones como Circuit Breaker para manejar fallos de comunicación.
3. **Seguridad**:
    - Asegurar la comunicación entre microservicios y proteger los datos requiere un enfoque integral de seguridad, incluyendo autenticación, autorización y encriptación.
    - La seguridad debe ser implementada de manera uniforme en todos los servicios para evitar vulnerabilidades.

(iii) Herramientas y tecnologías comunes en microservicios

1. **Contenedores y orquestadores**:
    - **Docker**: Para empaquetar aplicaciones en contenedores ligeros y portátiles.
    - **Kubernetes**: Para orquestar la implementación, escalado y gestión de contenedores.
2. **APIs y Gateways**:
    - **Spring Boot** y **Spring Cloud**: Para desarrollar microservicios en Java y gestionar la comunicación entre ellos.

    - **API Gateway**: Para manejar enrutamiento, autenticación, monitoreo y otras funciones transversales.
3. **Monitoreo y logging**:
    - **Prometheus** y **Grafana**: Para la supervisión y visualización de métricas.
    - **ELK Stack (Elasticsearch, Logstash, Kibana)**: Para la recolección, almacenamiento y análisis de logs.
4. **Mensajería y eventos**:
    - **RabbitMQ**, **Kafka**: Para la comunicación asíncrona y basada en eventos entre microservicios.

(iv) Ejemplo de aplicación de microservicios

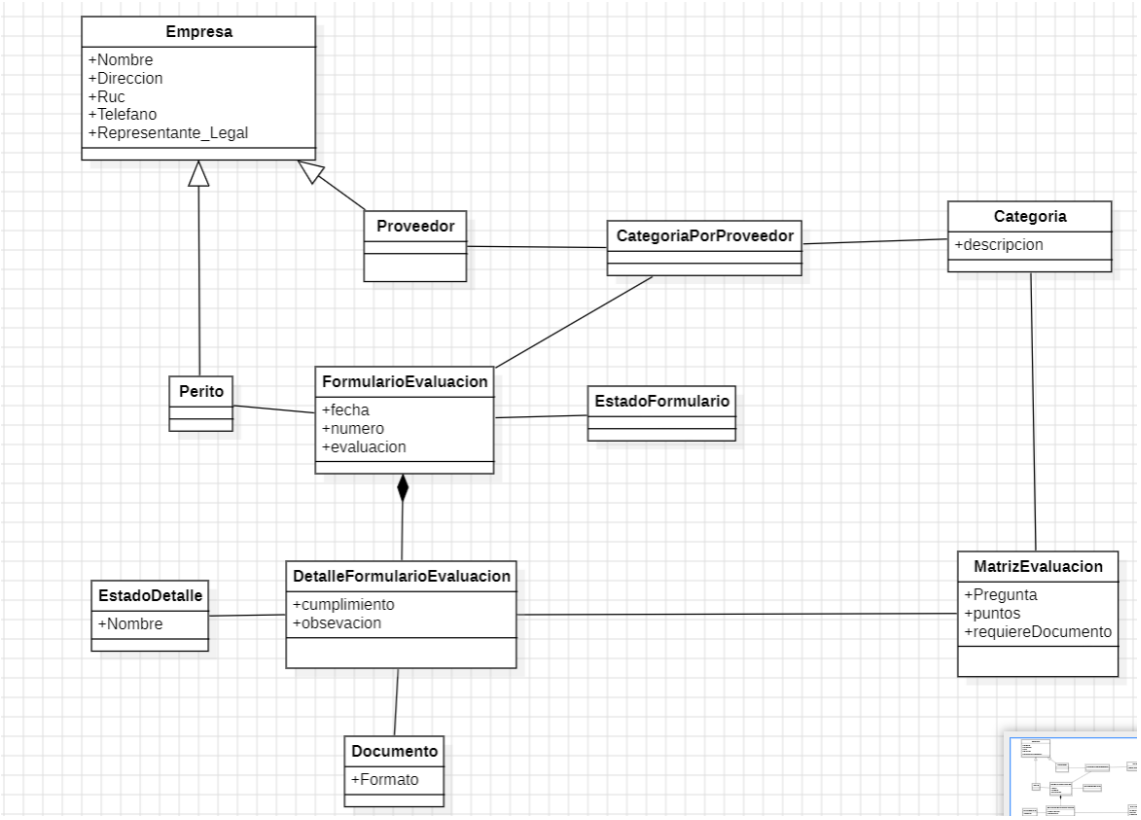
Para un Sistema de Evaluación de Proveedores (SEP), la arquitectura de microservicios podría dividirse en los siguientes servicios:

1. **Servicio de Gestión de Proveedores**:
    - Encargado del registro y mantenimiento de la información de los proveedores.
2. **Servicio de Evaluación de Proveedores**:
    - Gestiona la creación y mantenimiento de matrices de evaluación y la realización de evaluaciones.
3. **Servicio de Calificación de Proveedores**:
    - Asigna calificaciones basadas en los resultados de las evaluaciones y genera informes de calificación.
4. **Servicio de Seguridad y Acceso**:
    - Gestiona los usuarios y roles para garantizar el acceso seguro al sistema.
5. **Servicio de Integración**:
    - Facilita la comunicación y el intercambio de datos con otros sistemas de compras y adquisiciones.

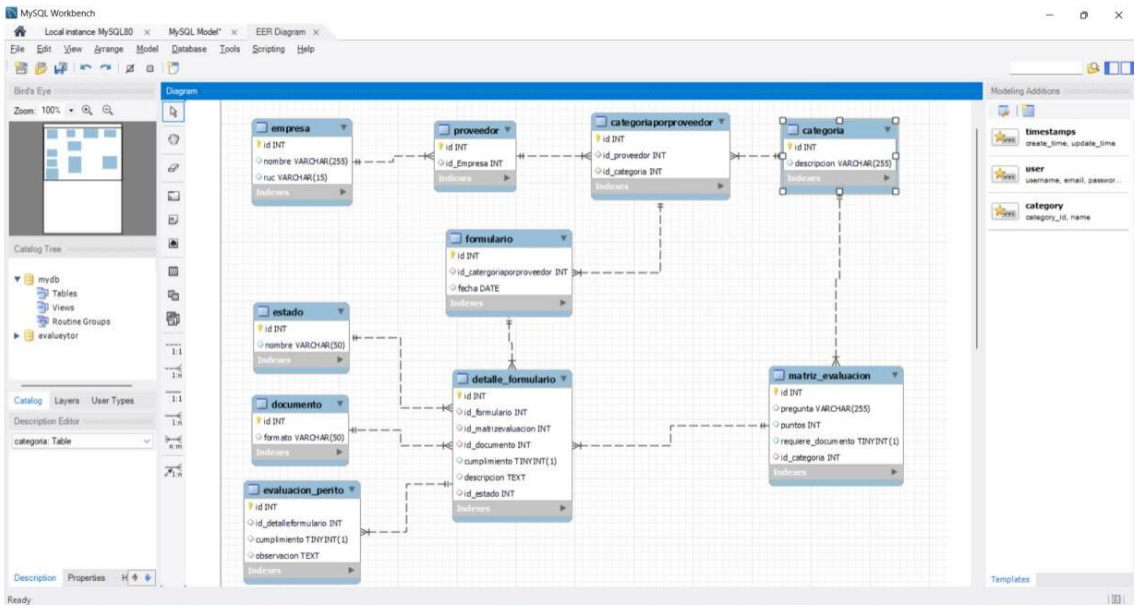
Esta estructura modular permite una gestión eficiente, escalabilidad y mantenimiento independiente de cada componente, maximizando la flexibilidad y robustez del sistema SEP.

3. Propuesta Técnica

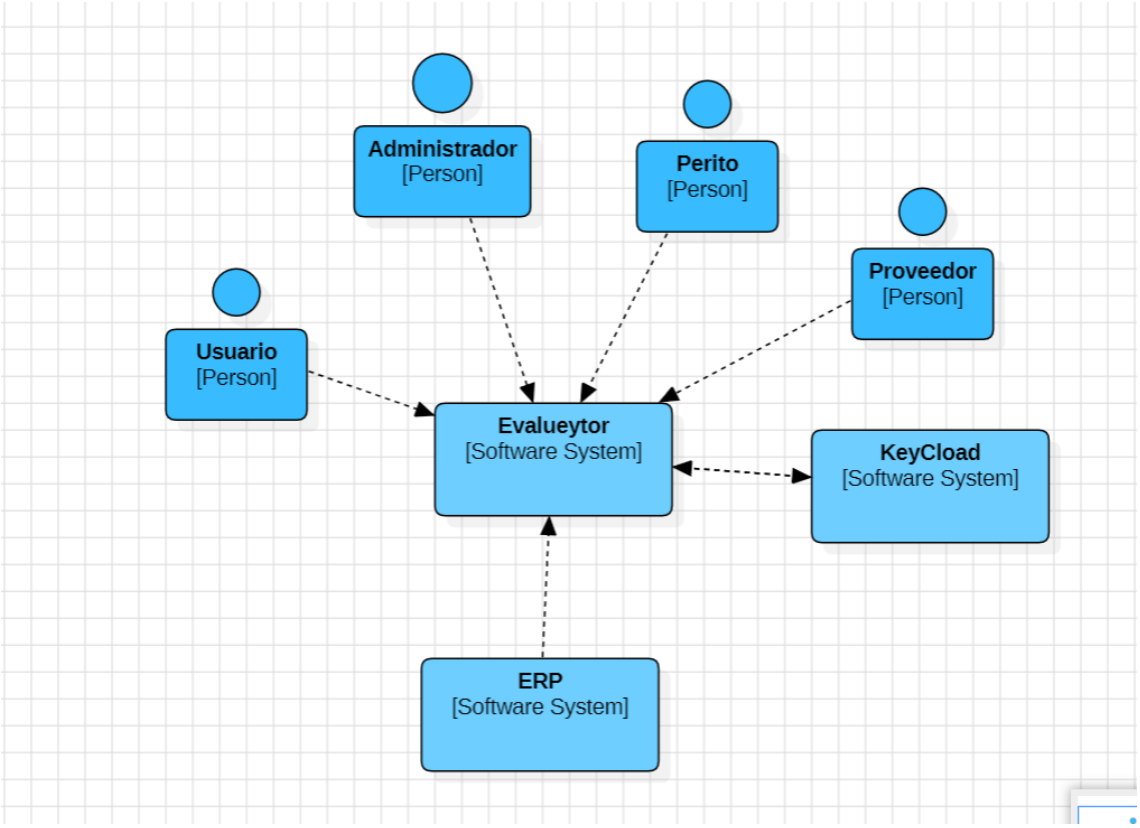
3.1. Diseño de clases



3.2. Diseño del modelo de datos



3.3. Diagrama del contexto del sistema (C4)

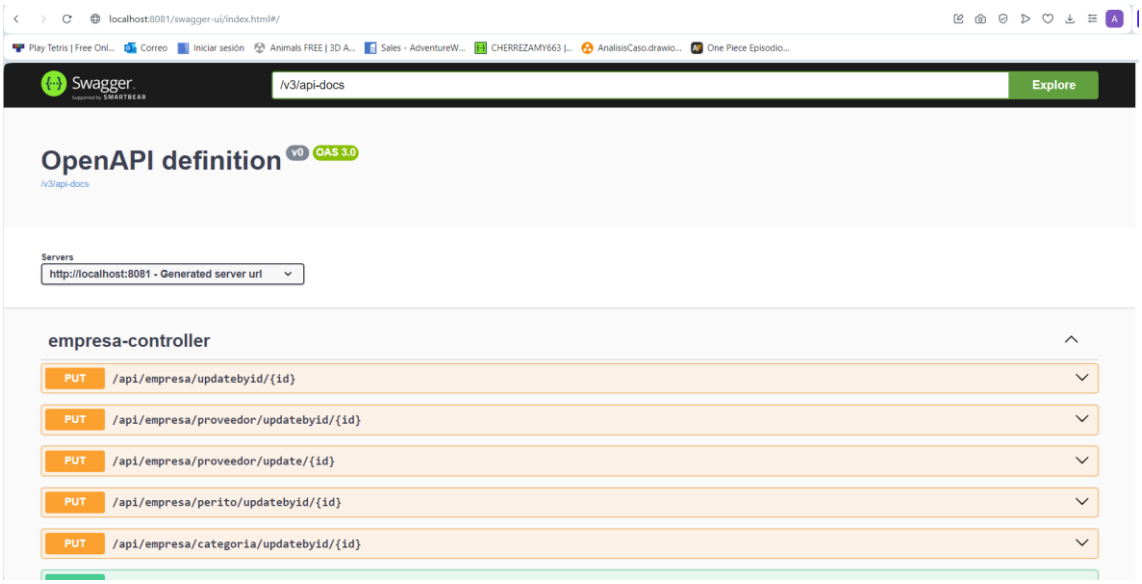


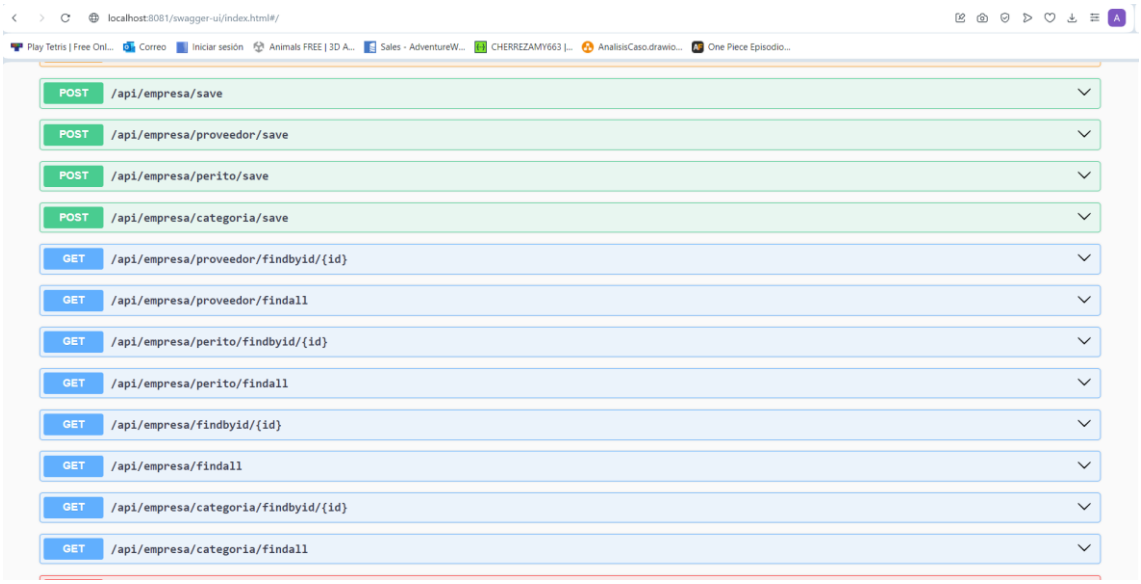
3.4. Presentar la aplicación backend con el catálogo de servicios que se indica en el último punto de este documento.

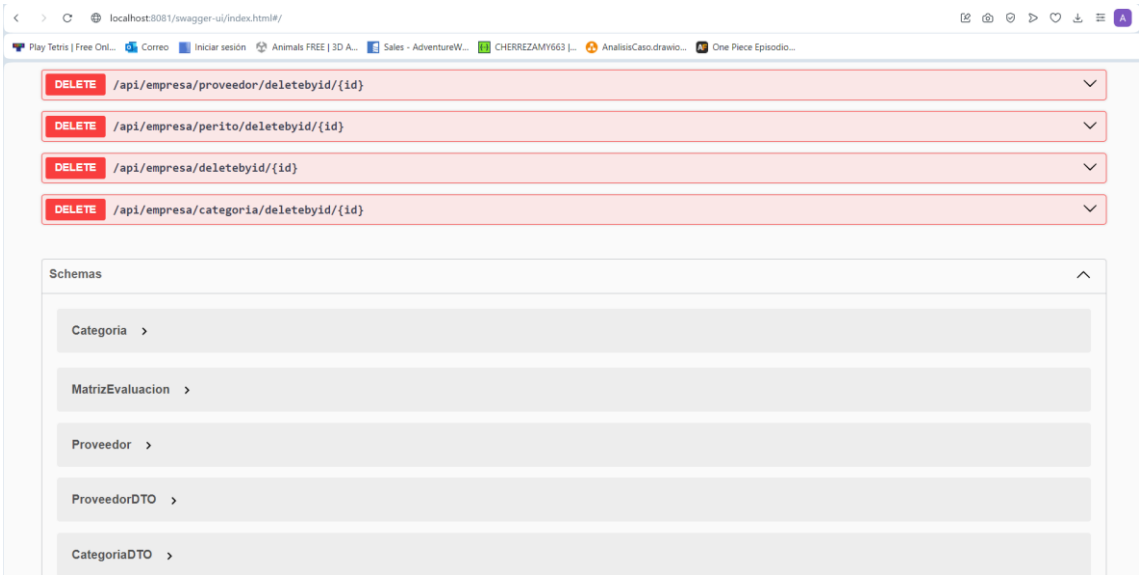
BUSINESS DOMAIN

empresa

	/api/empresa	Puerto: 8081
GET	/proveedor/findall	
GET	/ proveedor/findbyid/{id}	
POST	/ proveedor/save	Al guardar el proveedor, tomar en cuenta que el proveedor puede pertenecer a varias categorías.
PUT	/ proveedor/updatebyid/{id}	
DELETE	/ proveedor/deletebyid/{id}	
GET	/perito/findall	
GET	/ perito/findbyid/{id}	
POST	/ perito/save	
PUT	/ perito/updatebyid/{id}	
DELETE	/ perito/deletebyid/{id}	
GET	/categoria/findall	
GET	/categoria/findbyid/{id}	
POST	/ pategoria/save	
PUT	/ categoria/updatebyid/{id}	
DELETE	/ categoria/deletebyid/{id}	
GET	/ matrizevaluacion /findall	
GET	/ matrizevaluacion /findbyid/{id}	
POST	/ matrizevaluacion /save	
PUT	/ matrizevaluacion /upadatebyid/{id}	
DELETE	/ matrizevaluacion /deletebyid/{id}	

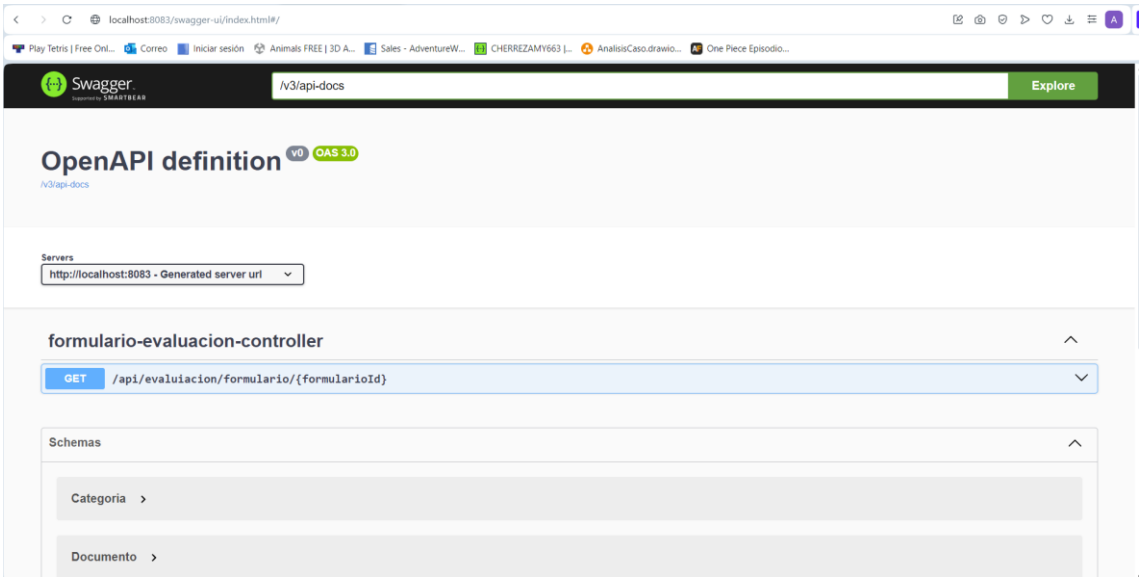

localhost:8081/swagger-ui/index.html#/
Swagger
/v3/api-docs
Explore
OpenAPI definition
v0
OAS 3.0
/v3/api-docs
Servers
http://localhost:8081 - Generated server url
empresa-controller
PUT /api/empresa/updatebyid/{id}
PUT /api/empresa/proveedor/updatebyid/{id}
PUT /api/empresa/proveedor/update/{id}
PUT /api/empresa/perito/updatebyid/{id}
PUT /api/empresa/categoria/updatebyid/{id}


localhost:8081/swagger-ui/index.html#/
POST /api/empresa/save
POST /api/empresa/proveedor/save
POST /api/empresa/perito/save
POST /api/empresa/categoria/save
GET /api/empresa/proveedor/findbyid/{id}
GET /api/empresa/proveedor/findall
GET /api/empresa/perito/findbyid/{id}
GET /api/empresa/perito/findall
GET /api/empresa/findbyid/{id}
GET /api/empresa/findall
GET /api/empresa/categoria/findbyid/{id}
GET /api/empresa/categoria/findall



evaluacion

	/api/evaluacion	Puerto: 8083
GET	/findall	
GET	/findbyid/{id}	
POST	/save	En este servicio, se ingresa el detalle formulario, y desde detalle formulario se ingresa documento. Los
PUT	/upadatebyid/{id}	
DELETE	/deletebyid/{id}	
GET	/estadoevaluacion/findall	
GET	/estadoevaluacion/findbyid/{id}	
POST	/estadoevaluacion/save	
PUT	/estadoevaluacion/upadatebyid/{id}	
DELETE	/estadoevaluacion/deletebyid/{id}	
GET	/estadodetalle/findall	
GET	/estadodetalle/findbyid/{id}	
POST	/estadodetalle/save	
PUT	/estadodetalle/upadatebyid/{id}	
DELETE	estadodetalle/deletebyid/{id}	



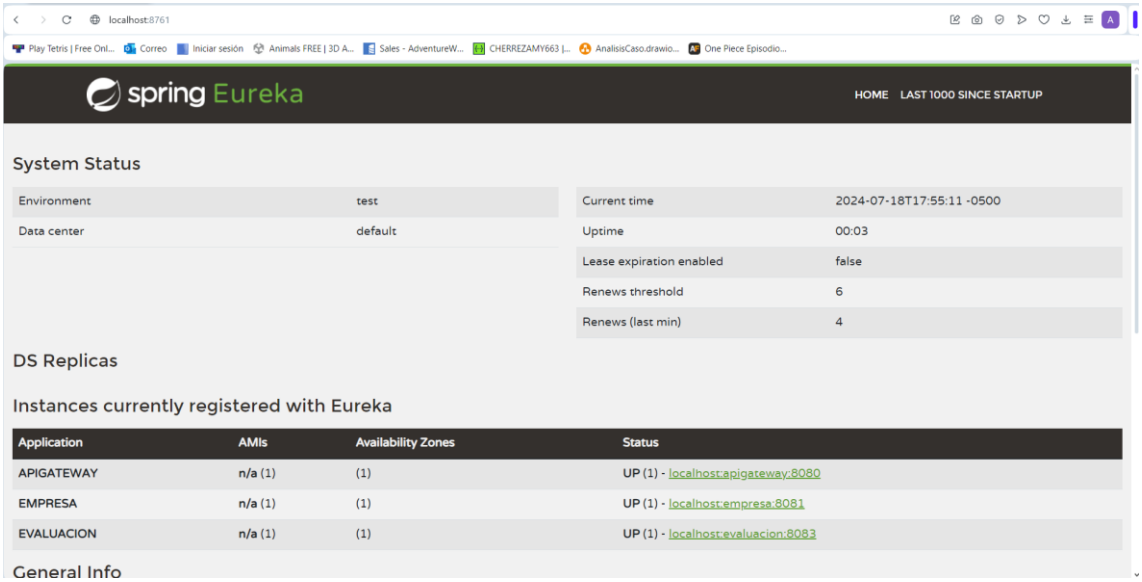
INFRAESTRUCTURE DOMAIN

eureka

Puerto: 8761	Se encarga del registro de todos los microservicios

apigateway

Puerto: 8080	Puerta de entrada, por este puerto se debe hacer todas las llamadas



3.5. Presenta un video de no más de 5 con la descripción del proyecto. Incluir el link

arq-20240718_180547-Grabación de la reunión.mp4

3.6. Incluir el link de GitHub con el código fuente con el código fuente y los documentos entregables.
https://github.com/Amy525-g/ProyectoFinCurso.git