UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

Plataforma para auditoria de cumplimiento de Sistema de Gestión de Seguridad de la Información

INFORME FINAL DE CC6909 PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

Gabriel Rojas Chamorro

MODALIDAD:
Práctica Extendida

PROFESOR GUÍA:
Eduardo Godoy Vega

SUPERVISOR:
Mauricio Castro García

SANTIAGO DE CHILE
2024

1. Solución inicial

El sistema que se está desarrollando tiene como objetivo principal facilitar la gestión del Sistema de Gestión de Seguridad de la Información (SGSI) de una empresa, siguiendo los estándares de la norma ISO 27001. Para lograr esto, se han definido varios módulos, cada uno con su conjunto de funcionalidades específicas. A continuación, se detallan las historias de usuario y modelación para cada uno de estos módulos: documentos, activos, riesgos y procesos.

Módulo de documentos

1.1.1. Historias de usuario

1. Como administrador, quiero poder crear categorías de controles para agrupar controles relacionados.
2. Como administrador, quiero poder crear controles para definir mi marco de SGSI.
3. Como administrador, quiero poder cargar todas las categorías y controles de ISO 27001 a partir de una plantilla, para tener una base al momento de implementar cada control.
4. Como administrador, quiero poder subir documentos a cada control, para definir mi implementación de dicho control.
5. Como administrador, quiero que los documentos queden versionados, para saber qué versiones han sido leídas por los usuarios y mantener un registro de modificaciones.
6. Como comité, quiero poder aprobar documentos, para validar su contenido.
7. Como usuario, quiero poder ver el listado de controles.
8. Como usuario, quiero poder ver el detalle de cada control.
9. Como usuario, quiero poder ver el detalle de cada documento.

1.1.2. Modelación de la base de datos

La modelación de la base de datos para el módulo de documentos involucra las siguientes entidades, listando sus atributos y la significancia de cada uno de estos:

1. **ControlCategory**:
   - `id`: el identificador único
   - `name`: el nombre de la categoría
2. **Control**:
   - `id`: el identificador único
   - `category`: la categoría
   - `title`: el título del control
   - `description`: la descripción del control
   - `document`: el documento asociado al control
3. **Document**:

- `id`: el identificador único
- `title`: el título del documento
- `description`: la descripción del documento

4. **DocumentVersion**:
   - `id`: el identificador único
   - `document`: el documento al cual pertenece la versión
   - `version`: el número de la versión
   - `file`: el archivo de la versión
   - `shasum`: el hash (sha256) del archivo de la versión
   - `is_approved`: booleano señalizando si la versión está aprobada
5. **DocumentVersionReadByUser**:
   - `id`: el identificador único
   - `document_version`: la versión leída por el usuario
   - `user`: el usuario que hizo la lectura
6. **Evidence**:
   - `id`: el identificador único
   - `document_version`: la versión del documento a la cual alude la evidencia
   - `process_activity`: la actividad a partir de la que se generó la evidencia
   - `file`: archivo de evidencia
   - `shasum`: el hash (sha256) del archivo de evidencia

Módulo de activos

1.2.1. Historias de usuario

1. Como administrador, quiero poder registrar activos de la empresa, para luego definir su riesgo asociado.
2. Como usuario, quiero poder ver el listado de activos.
3. Como usuario, quiero poder ver el detalle de cada activo.

1.2.2. Modelación de la base de datos

La modelación de la base de datos para el módulo de activos involucra las siguientes entidades, listando sus atributos y la significancia de cada uno de estos:

1. **Asset**:
   - `id`: el identificador único
   - `owner`: el dueño del activo
   - `name`: el nombre del activo
   - `description`: la descripción del activo
   - `asset_type`: el tipo de activo

- `criticality`: la criticidad del activo, está pudiendo ser `muy baja`, `baja`, `media`, `alta` o `muy alta`
- `classification`: la clasificación del activo, está pudiendo ser `publica`, `interna` o `privada`

Módulo de riesgos

1.3.1. Historias de usuario

1. Como administrador, quiero poder asignar un riesgo a cada uno de los activos.
2. Como administrador, quiero poder ver el listado de riesgos.
3. Como administrador, quiero poder ver el detalle de cada riesgo.

1.3.2. Modelación de la base de datos

La modelación de la base de datos para el módulo de riesgos involucra las siguientes entidades, listando sus atributos y la significancia de cada uno de estos:

1. **Risk**:
   - `id`: el identificador único
   - `asset`: el activo para el cual existe el riesgo
   - `control`: el control asociado al riesgo
   - `title`: el título del riesgo
   - `description`: la descripción del riesgo
   - `responsible`: el responsable del riesgo
   - `severity`: la gravedad de que se plasme el riesgo, está pudiendo ser `muy baja`, `baja`, `media`, `alta` o `muy alta`
   - `likelihood`: la probabilidad de que se plasme el riesgo, está pudiendo ser `muy baja`, `baja`, `media`, `alta` o `muy alta`
   - `treatment`: el tratamiento que se le dará al riesgo, esté pudiendo ser `mitigar`, `transferir`, `aceptar` o `eliminar`

Módulo de procesos

1.4.1. Historias de usuario

1. Como administrador, quiero poder definir procesos manuales, para generar evidencia de cierto control.
2. Como administrador, quiero poder definir procesos recurrentes, para generar evidencia de cierto control de manera periódica.
3. Como administrador, quiero poder asignar procesos a los usuarios directamente o indirectamente a través de grupos, para generar evidencia de los controles.

4. Como usuario, quiero ser notificado al tener un nuevo proceso asignado, para poder completarlo rápidamente.

1.4.2. Modelación de la base de datos

La modelación de la base de datos para el módulo de riesgos involucra las siguientes entidades, listando sus atributos y la significancia de cada uno de estos:

1. **ProcessDefinition**:
   - `id`: el identificador único
   - `name`: el nombre de la definición de un proceso
   - `control`: el control asociado a la definición de un proceso
   - `recurrency`: la recurrencia con la que se tiene que hacer el proceso
2. **ProcessActivityDefinition**:
   - `id`: el identificador único
   - `process_definition`: la definición de un proceso a la cual se le define la actividad
   - `order`: el orden respectivo a las otras definiciones de actividades de una definición de un proceso
   - `description`: la descripción de la actividad a realizar
   - `asignee`: el usuario encargado de realizar la actividad
   - `asignee_group`: el grupo de usuario a los cuales se les debe asignar la actividad
3. **Process**:
   - `id`: el identificador único
   - `process_definition`: la definición de un proceso con la cual se crea el proceso
   - `name`: el nombre del proceso
   - `control`: el control asociado al proceso
   - `completed`: booleano representando si el proceso fue completado
   - `completed_at`: fecha y hora en la cual fue completado el proceso
4. **ProcessActivity**:
   - `id`: el identificador único
   - `process`: el proceso al cual esta asociada la actividad
   - `activity_definition`: la definición de la actividad con la cual se crea la actividad
   - `order`: el orden respectivo a las otras actividades de un proceso
   - `description`: la descripción de la actividad a realizar
   - `asignee`: el usuario encargado de realizar la actividad
   - `asignee_group`: el grupo de usuario a los cuales se les asignó la actividad
   - `completed`: booleano representando si la actividad fue completada
   - `completed_at`: fecha y hora en la cual fue completada la actividad

Tecnologías escogidas

La solución se adapta de manera efectiva para abordar desafíos relacionados con la escalabilidad, el rendimiento y la seguridad del sistema, incorporando consideraciones específicas en su diseño y arquitectura.

En términos de escalabilidad, si bien el proyecto no está inicialmente diseñado para manejar un gran flujo de usuarios, la implementación en contenedores Docker permite una fácil replicación y despliegue detrás de un balanceador de carga. Esto facilita la escalabilidad horizontal, permitiendo la adición de nuevos contenedores según sea necesario. Para la gestión de datos, la escalabilidad vertical de la base de datos PostgreSQL y la opción de utilizar réplicas para lectura proporcionan una respuesta eficiente a posibles aumentos en la carga de datos.

En cuanto al rendimiento, la elección de tecnologías robustas y bien probadas, como Django, PostgreSQL y Typescript, proporciona una base sólida. La experiencia previa con sistemas similares garantiza que el escalamiento de la aplicación sea un proceso manejable, respaldado por las mejores prácticas y lecciones aprendidas de implementaciones anteriores.

La interoperabilidad entre las tecnologías utilizadas se ve respaldada por la compatibilidad inherente de Django con PostgreSQL y la elección de Typescript como lenguaje en el frontend. Además, se planea seguir estándares y prácticas documentadas para asegurar una integración fluida, aprovechando la documentación existente como guía.

La elección de Django junto con HTML y CSS se justifica por la naturaleza estática de los datos, donde los cambios no son frecuentes. En este contexto, una biblioteca de frontend como React no aportaría un beneficio significativo, ya que la actualización dinámica de la interfaz de usuario no es una prioridad, lo que hace que la simplicidad y la eficiencia de HTML y CSS sean suficientes para cumplir con los requisitos del proyecto.

Anexo

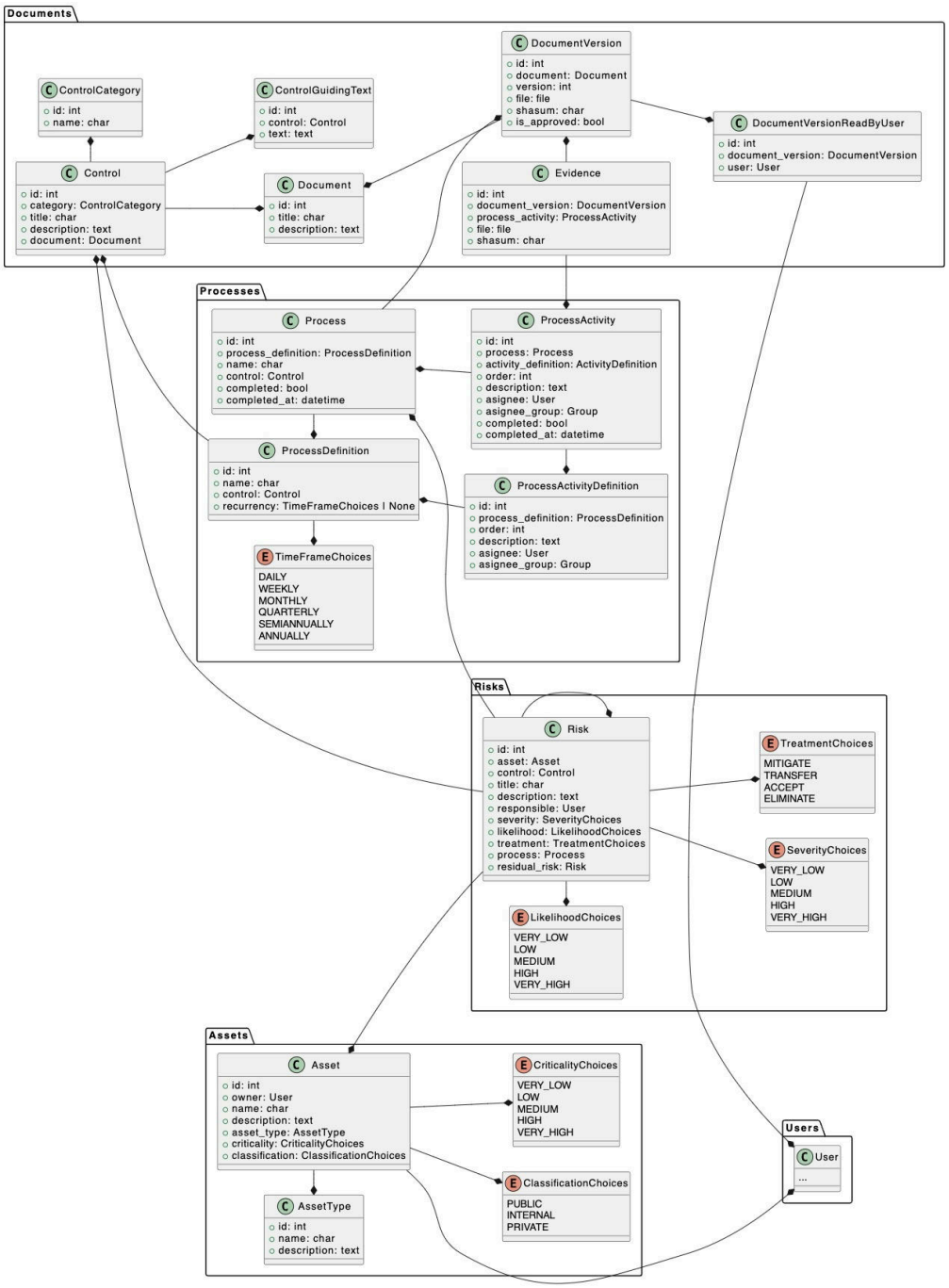

Figura 1: Modelo entidad-relación