Obligatorio Base de Datos

Entrega Final

Ingeniería en Informática

2020

Integrantes del grupo

Agustín Picos
Franco Gai
Micaela Olivera

Índice

Introducción

Se solicita realizar un sistema que permita gestionar usuarios y roles con sus determinadas responsabilidades, implementar un sistema de autenticación y autorización eficiente que posteriormente pueda ser escalable para integrar con aplicaciones de cualquier tipo de problemática.

El programa debe permitir a un usuario loguearse y según el rol que tenga en el mismo acceder a determinadas funcionalidades , métodos y/o menús equivalentes a su nivel de responsabilidad.

Deberá existir un usuario administrador el cual será el único con la potestad de autorizar la creación de usuarios y asignar roles a los mismos. Este usuario también será el habilitado para poder crear aplicaciones que usaran el módulo de seguridad, así como también crear modificar y/o eliminar los distintos menús que contendrá dicha aplicación y los métodos que poseerá el menú.

Así mismo el sistema deberá contar con un módulo de auditoría para poder identificar que usuarios realizan determinadas actividades en el mismo como lo son la creación, eliminación de usuarios y asignación de roles.

Se debe comenzar con una funcionalidad que permite crear, actualizar y modificar roles , usuarios , aplicaciones, métodos y menús, así como también las asignaciones correspondientes entre las entidades del sistema. Se deberá implementar medidas de seguridad en la creación de usuarios, como por ejemplo establecer pautas para la creación de contraseñas y posteriormente su encriptado así como también el control de los usuarios mediante el conteo de intentos fallidos (luego de tres intentos fallidos el usuario será bloqueado).

Además todas las solicitudes generadas durante la ejecución del sistema estarán sujetas a aprobación de un usuario administrador que podrá aprobar o no las mismas.

Análisis del Problema

Se identifican distintos requerimientos en el análisis del problema:

- En primer lugar se nota la necesidad de implementar una correcta gestión de usuarios (esto implica creación eliminación y modificación de los mismos).

- La misma necesidad del punto anterior se aplica a la gestión de los roles que deben existir en el sistema, indicando la responsabilidad que le corresponde a cada rol sobre el sistema.

- Se debe implementar la lógica correspondiente para poder autenticar a los usuarios, mediante un login, brindando la seguridad correspondiente sobre el usuario y contraseña.

- Los usuarios deberán contar con una interfaz en la cual puedan ingresar al negocio que le corresponda acceder según su rol.Un mismo sujeto podrá contar con varios usuarios.

- Se entiende el usuario y la persona como entidades diferentes, de esta manera se puede relacionar una persona a varios usuarios. Por lo tanto se necesita tener tanto el formulario de ingreso de persona, como el de creación de usuario.

- Asimismo el administrador del sistema deberá contar con una interfaz que le permita gestionar usuarios, roles, aplicaciones, métodos y menús.

- Se debe contar con un mecanismo de autorización, es decir el asignar a cada usuario su rol o roles correspondiente.

- Se detecta la necesidad de crear una aplicación escalable y adaptable a cualquier tipo de problemática.

- Auditoría: se deberá llevar un registro de todos los cambios que se hacen, tales como creación y eliminación de usuarios o asignación de permisos. Se debe elaborar una estructura que permita evaluar quién hizo los cambios, cuándo los realizó, y el detalle del cambio en sí mismo, es decir, qué variables fueron modificadas.

- Oposición de interés: existen tareas en las que es prioritario que sean gestionadas por más de una persona, es decir, que una sola persona no sea la única involucrada, sino que sea requerida la aprobación de otra, como por ejemplo la creación de un usuario. La finalidad de esto es generar cierto control sobre las acciones que se toman.

Desarrollo de la Solución

Para el desarrollo de la solución se deben tener en cuenta varios puntos y cómo los se van a resolver individualmente.

Gestión de personas y usuarios.

El sistema contará con dos formas diferentes de crear un usuario; la primera será por parte del administrador, el cual podrá ingresar personas al sistema y crearle un usuario. Posteriormente le asignará su/sus rol/roles correspondientes.

En la segunda opción, la persona ingresa a la aplicación por primera vez y decide crearse un usuario, para ello en la pantalla de login selecciona una opción para crear un usuario nuevo. En este caso, La persona tendrá un formulario a su disposición, el cual deberá llenar con sus datos personales junto con los datos de su primer usuario y lo deberá confirmar. Luego de realizado este proceso, el administrador (quien puede ver un lista de personas y usuarios con pendiente aprobación) podrá o no aprobar la creación de dicho perfil y asignarle su rol correspondiente.

Las aprobaciones o rechazos deben quedar registradas con sus respectivos comentarios, para garantizar la transparencia de la gestión.

Para la creación de un usuario se solicitará la siguientes información:

- Nombre y apellido.
- Nombre de usuario.
- Contraseña.
- Cédula de identidad.
- Correo electrónico.
- Fecha de nacimiento.
- Sexo.

Una vez aprobada la persona y logueada en el sistema, la misma podrá comenzar a trabajar en su aplicación asignada.

Gestión de Roles.

Para la creación de un nuevo rol se solicitarán los siguientes datos:

- Nombre del rol.
- Menús a los que puede acceder dicho rol.
- Métodos a los que puede acceder dicho rol.

Cabe destacar que solo el administrador podrá gestionar este módulo.

Autorización y autenticación.

En términos generales, es común que cuando una persona física utiliza un sistema, lo hace a través de un usuario dentro del sistema. Este usuario tiene un parámetro que lo identifica de forma única, como lo es el nombre de usuario. La persona además de utilizar ese nombre o identificador, debe tener algún mecanismo con el cual compruebe su identidad.

El concepto más básico es el de contraseña. En términos simples, la contraseña suele ser un atributo del usuario dentro del sistema. Para el usuario, es una cadena de caracteres. Esta contraseña deberá ser conocida únicamente por la persona física que utiliza ese usuario. Los usuarios y contraseñas deben ser individuales, es decir, un usuario es utilizado por una única persona física. Una de las medidas utilizadas en cuanto a la seguridad de las contraseñas es el uso de encriptado y sobre la seguridad de los usuarios es la implementación del bloqueo de usuarios luego de tres intentos fallidos (solo el administrador podría llegar desbloquear un usuario)

Es importante el concepto de confidencialidad de la contraseña. Ni siquiera los administradores del sistema deben conocer o tener acceso a esta. En estos casos, las contraseñas deben ser cifradas dentro del sistema para no ser guardadas en texto plano.

Una vez accedido al sistema, debe estar definido qué tipo de acceso tiene el usuario utilizado. Existen términos comunes como lo son el acceso de administrador, que generalmente puede acceder a todos los recursos del sistema y modificar todas la variables; o el usuario básico con permisos mínimos, que suelen ser de solo lectura, y sobre un conjunto determinado de datos. Luego existen permisos intermedios que poseen más privilegios que un usuario básico, pero sin llegar a ser administrador. Estos permisos intermedios se ajustan a la función del usuario dentro del sistema y están relacionados con los permisos, privilegios y funciones de la persona física que utiliza el usuario del sistema.

Es importante tener definido el conjunto de autorizaciones que posee cada uno de estos usuarios. A este conjunto de permisos se lo denomina comúnmente rol.

Otro factor importante a tener en cuenta es que un usuario, con un rol, no debería poseer permisos para auto asignarse permisos más elevados.

Pantalla Administrador.

Una vez que un administrador logue en el sistema tendrá a su disposición una pantalla en la cual podrá ver todos los menús del sistemas, así como también botones que lo lleven al módulo de usuarios, de roles, de métodos y de solicitudes de usuarios.

Desde esta pantalla principal el administrador podrá también salir del sistema.

Modulo de Usuarios.

Cuando un administrador ingresa al módulo de usuarios, podrá visualizar una lista de todos los usuarios creados, también podrá crear usuarios nuevos, editar y eliminarlos.

En este módulo también se llevará a cabo la asignación de roles para los usuarios creados, ya que un usuario sin rol no podrá loguearse en el sistema.

También cabe aclarar que el administrador no podrá ver en texto plano las contraseñas de los usuarios una vez creados los mismo.

Módulo de Roles.

Cuando se ingrese al módulo de roles, al igual que en el módulo de usuarios, se verán listados los roles existentes, así como también la posibilidad de crear nuevos roles y eliminarlos. En este módulo el administrador también será capaz de asignar y quitar los métodos que podrá o no manejar dicho rol.

Solicitudes de usuarios.

En esta pantalla, el administrador podrá ver la lista de usuarios creados que están a la espera de su aprobación. También visualizará la lista de solicitudes de usuarios que fueron bloqueados y esperan a ser desbloqueados y las solicitudes referentes a la asignación de roles a usuarios. Estos usuarios pendientes de aprobación, serán aquellos que fueron creados por personas sin rol de administrador. Una vez que el Administrador apruebe la creación de dicho usuario, deberá dirigirse al módulo de usuarios para asignarle su rol correspondiente.

Pantalla Usuario.

Una vez que el usuario logue en el sistema tendrá visibles todos los menu a los que tenga su acceso autorizado. Luego de seleccionado uno de estos menús, podrá visualizar los métodos disponibles según su nivel de responsabilidad para ese menú particular.

Desde su pantalla el usuario podrá navegar y salir del sistema.

Auditoría.

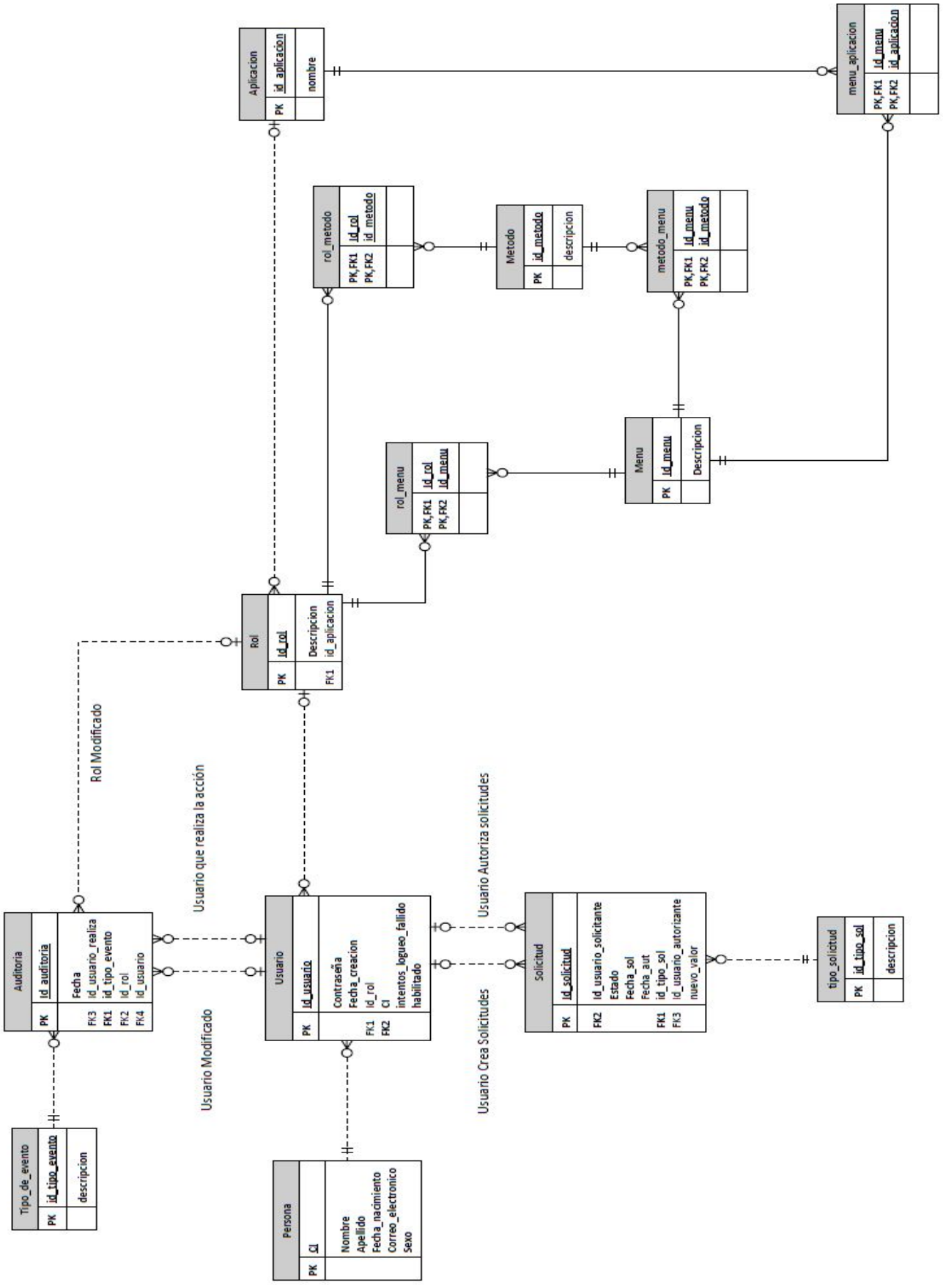
Es ideal que en los sistemas exista una figura que auspicie de contralor. El auditor debe ser una persona que no participe activamente en la sección auditada. El por qué es sencillo, no tiene sentido que un auditor investigue sus propias acciones o elementos con los que tenga relación.
El objetivo de la auditoría es controlar que todas las acciones realizadas dentro del sistema fueron correctas o investigar luego de ocurrido algún incidente, quienes fueron los involucrados, que hicieron y cómo.
En bases de datos, las operaciones de insertar, modificar y eliminar son las básicas a ser registradas para auditar, además de guardar la fecha y la información previa de los valores modificados y borrados.

Oposición de Interés.

Este es un mecanismo que tiene como objetivo brindar transparencia a las organizaciones.
A fin de evitar el ocultamiento o irregularidades, los procesos de una empresa no son abarcados por una única entidad, sino que se fragmentan y reparten en la organización. De esta forma se ejerce un control cruzado sobre las tareas.

A nivel del sistema planteado en este caso, el manejo de los usuarios no debe ser realizado por una única entidad sino que dicha tarea se debe repartir entre al menos otra persona que sea ajena a posibles intereses que hay dentro de un proceso dado. Un ejemplo de esto podría ser que se requiera la autorización de dos usuarios para la creación de un usuario o que quien lo crea no sea la misma persona que le asigna un rol.

Modelo Entidad Relación

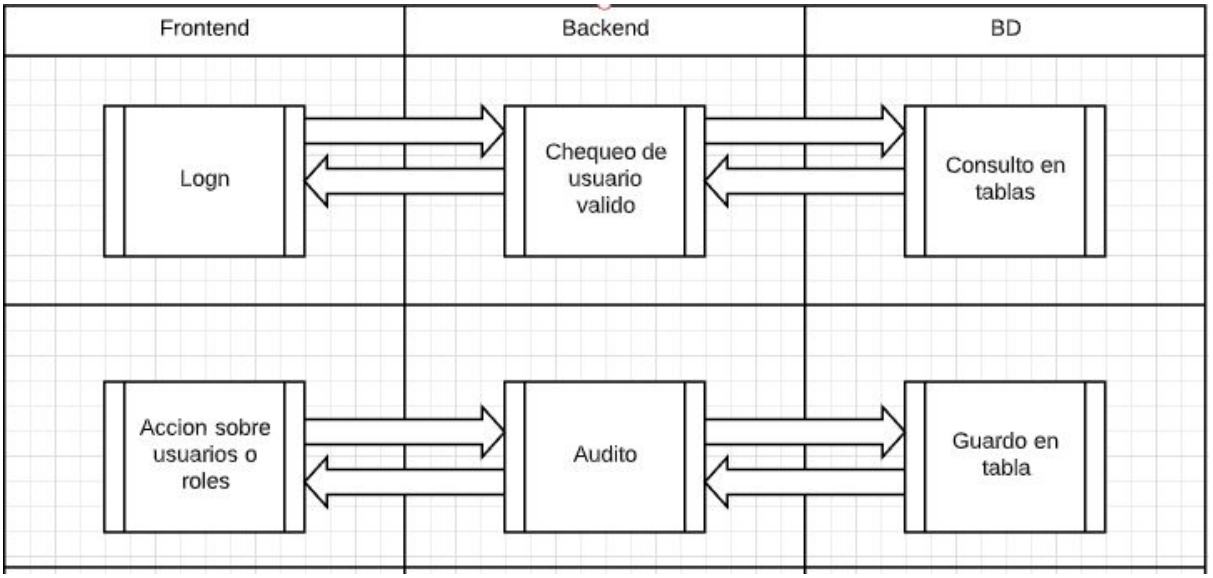


Estructuras creadas

Para cada una de las entidades principales (Persona, Usuario, Menú, Rol, Solicitud, Aplicación, Método, Auditoria, tipos de eventos y tipos de solicitud) fue creada una tabla. Luego, fueron necesarias tablas auxiliares (menu_aplicacion, metodo_menu, aplicacion_usuario, rol_menu, rol_metodo) para poder modelar relaciones de muchos a muchos entre tablas.
Se corroboró que todas las entidades fueran representadas de alguna forma (como tablas o atributos).

Diagrama genérico de colaboración de la aplicación

A continuación se adjunta un diagrama que muestra algunas de las interacciones del sistema. En el mismo se observa cómo se dividen las responsabilidades dentro del mismo y cómo interactúan las capas al momento de que se realizan las operaciones.



Ejemplo de colaboración entre capas dentro de la aplicación

Comprobación de satisfacción de condiciones

La tabla de ***Usuario*** permite la creación, eliminación y actualización de los usuarios. La tabla ***Persona***, a su vez, comprueba la existencia de la persona y permite diferenciar a las entidades y relacionarlas, a modo que una persona pueda tener varios usuarios. En este caso, el modelo no permite que un usuario sea utilizado por varias personas, ni que existan más de un usuario o persona con el mismo id. Una vez ingresados los datos, pueden ser modificados. La tabla ***Usuario***, cumple funciones de control para el logueo de la aplicación que son detalladas en el Anexo A.

La tabla de ***Rol*** tiene la función de especificar un conjunto de permisos (métodos) y menús a los que podrá acceder un usuario. En base a la tabla de ***Rol***, se puede especificar de

forma granular qué permisos y menús se poseen, pudiendo relacionar con gran flexibilidad estos valores. En este caso, un rol puede poseer muchos permisos y menús, pudiéndose dar la situación de que dos roles diferentes puedan acceder a varios componentes en común, pero teniendo diferencias en detalles como algunos permisos.

La tabla ***Auditoria*** permite registrar los eventos del sistema como inserción, actualización y eliminación de los datos de las tablas. Además retiene datos como el usuario involucrado, fechas, menú de origen y los valores previos y posteriores a dicho evento. La idea de esta entidad es que grabe situaciones generales a nivel de lógica de negocio y no por ejemplo cosas generales como, el script que se ejecutó cuantas filas afectó, y demás cosas que generalmente los motores de bases de datos son capaces de auditar de forma independiente.

La tabla ***Solicitud*** tiene como función dar una base al principio de oposición de interés, reteniendo datos como la persona que inició la solicitud, usuario autorizantes, la descripción, fechas de solicitud y autorización, además de indicar si el estado de estas se encuentra en pendiente, rechazado o aprobado. Mediante el uso de la aplicación se implementarán los conjuntos disjuntos entre los que se dará esta oposición de interés. Funcionalmente, este módulo recibe los datos de quien se loguea y muestra todas las solicitudes exceptuando aquellas que coinciden con la persona (mediante CI) que ingresó.
El tipo de solicitud es una tabla auxiliar en la que se desglosan los eventos, esto permite que sea más fácil el acceso para sacar métricas por ejemplo.

Se creó la tabla **Métodos** para poder distinguir qué elementos visuales se desplegarán en una determinada pantalla, teniendo en cuenta el rol del usuario y la aplicación a la que se ingresa. Esto permite un polimorfismo del formulario, lo cual significa que si el día de mañana una nueva aplicación quisiera usar nuestra aplicación. El cambio sería tan sencillo como ejecutar algunos scripts a nivel de base de datos sin tener que modificar el código fuente.

Se considera que el modelo tiene términos muy genéricos, pero completos a la vez, permitiendo la escalabilidad del sistema para poder ser utilizado en cualquier negocio.

Se agregaron reglas para imposibilitar la insercion de datos en casos que no corresponda, como por ejemplo en la tabla persona, el atributo sexo solo admite el valor M o F, cualquier otro valor sera rechazado. Lo mismo ocurre en el caso de Solicitud, cuyo estado solo admite pendiente, aprobado y rechazado.

Puede visualizar las consultas SQL utilizadas para armar la estructura de la base en el Anexo B

Introducción a la aplicación

Conexión a la base de datos:
Dentro del proyecto, ubicar el paquete Source Packages.accesosBD
Abrir la clase Configuración.java y editar las líneas como detalla a continuación:

private static final String USUARIO_DB = "**postgres**";//usuario
    private static final String PASSWORD_DB = "**passwd**";//contraseña
    private static final String URL_DB =
"jdbc:postgresql://**192.168.132.128:5432/proyectoFinal**";//nombre de la base de datos

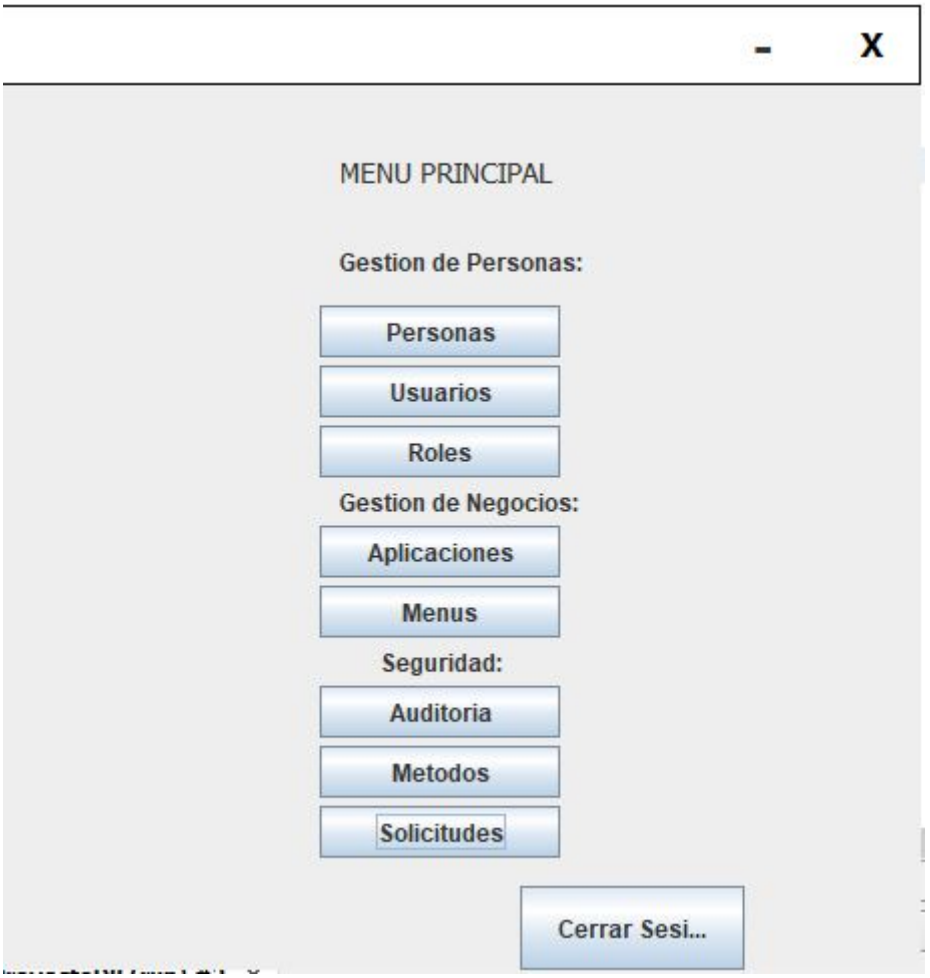
Esta aplicación está dividida en dos bloques:

1. El modulo de administracion
2. El módulo de usuarios

Para ambos módulos se accede mediante la misma pantalla de login. Es la aplicación la que lee los permisos de usuario y decide a qué módulo enviarlo.

Importante, la cantidad de logueos erróneos permitidos es de 3, a partir de este punto el usuario estará bloqueado y ya no podrá ingresar a ningún módulo. Una solicitud es enviada automáticamente a los administradores para evaluar su desbloqueo.

El módulo admin posee una serie de funcionalidades mostradas de la siguiente manera:



Desde aquí podrá administrar todo lo relacionado a:

- Gestión de personas
- Gestión de Negocios
- Seguridad

Gestión de personas

Todo lo relacionado a personas, usuarios y roles dentro de la aplicación

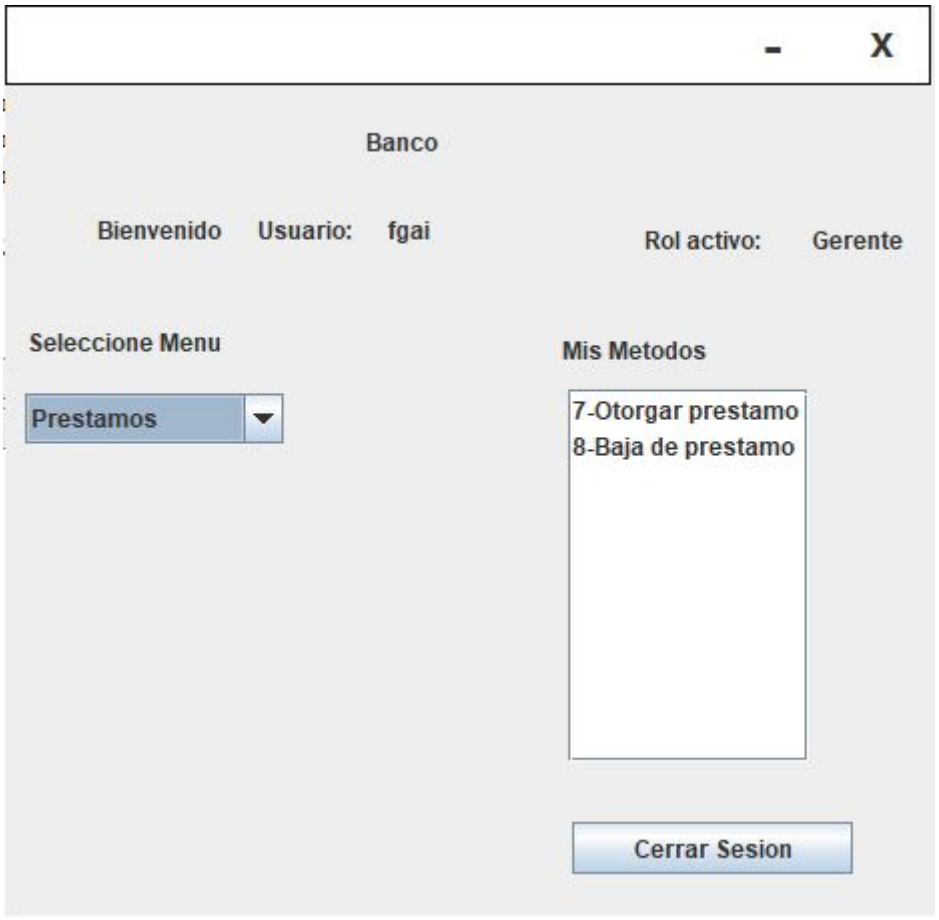
Gestión de Negocios

Todo lo relacionado con administrar los distintos tipos de aplicaciones, los menues a los que podrá acceder cada una de ellas

Seguridad

Auditorías, Gestión de solicitudes y Métodos por rol.

Continuando, veremos ahora la pantalla de los usuarios sin privilegios de administradores la cual se ve de la siguiente manera:



Si bien este menú como se explica en la letra del problema es a modo de muestra, es lo suficientemente escalable para identificar, usuario logueado, aplicación en la que se encuentra y los menús y métodos asociados.

Conclusiones

El proyecto cumplió con nuestras expectativas, ya que se logró aplicar los conocimientos vistos en clase y modelar una base de datos acorde al problema planteado.

Logramos realizar un programa capaz de simular un módulo de seguridad para un grupo de negocios así como también manejar datos de forma completa al poder modificarlos y asignar diferentes relaciones. También logramos reflejar la seguridad del módulo a través de la creación de las auditorías y de las solicitudes.

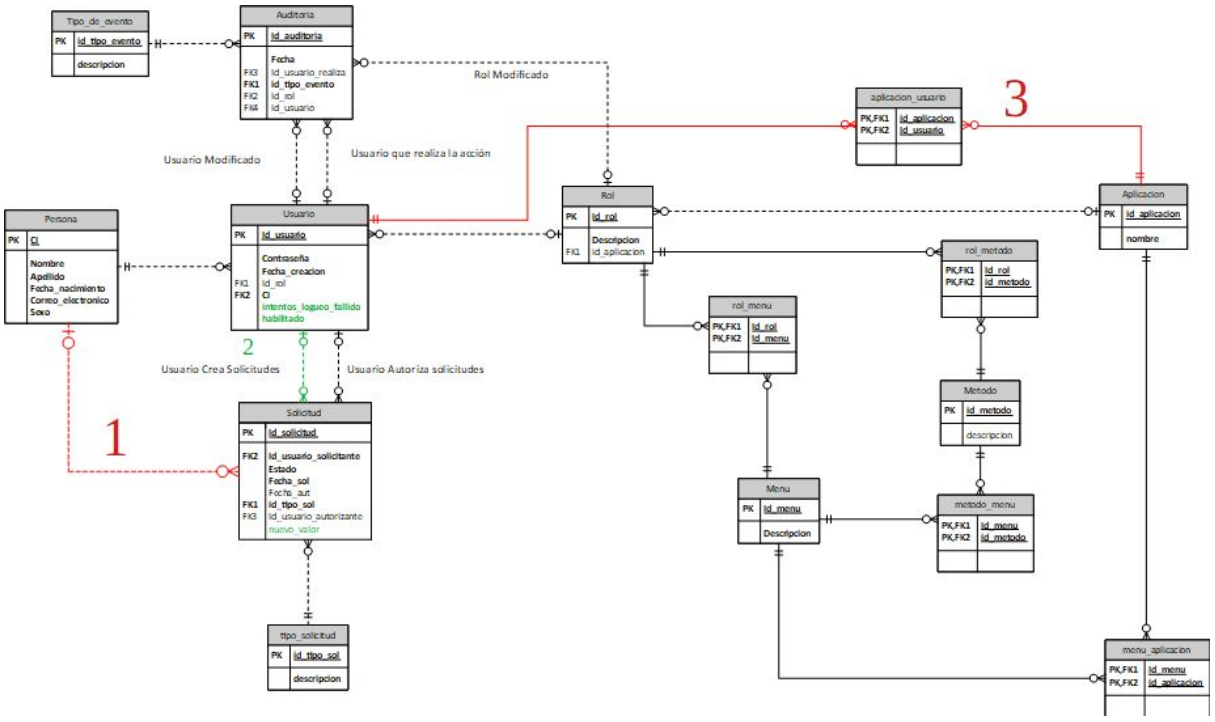
Es destacable también que logramos verificar la correcta manipulación de los datos y de la aplicación al poder ver reflejado en la aplicación de usuarios los diferentes negocios creados.

Dada la facilidad de acceso a la información y al trabajo en equipo, pudimos sortear las dificultades que se fueron presentando sin mayores inconvenientes.

Queremos concluir como estudiantes que, la experiencia al trabajar con la base de datos para la realización de este proyecto nos ayudó a mejorar nuestras habilidades manipulando las mismas. Es importante mencionar que si bien la situación sanitaria actual afecta de cierta medida la dinámica en clase y las instancias de los trabajos en grupos, podemos decir que logramos trabajar en equipo y pudimos aplicar y aprender más sobre las bases de datos y su funcionamiento en el ámbito práctico.

Anexo A

Modificaciones MER



En verde se muestran los elementos agregados respecto a la segunda entrega, y en rojo los quitados

Cambio 1: El objetivo de la relación entre persona y solicitud era evaluar qué usuario realiza la solicitud. En el caso de los usuarios, estos pasaron a ser creados directamente en la tabla de usuarios pero con un atributo de no-habilitado. Dado que presentar el id del usuario es más nutritivo que mostrar la cédula del solicitante, ya que un solicitante puede tener varias cédulas.
Cambio 2: Como consecuencia del cambio 1, se crea una relación entre usuario y solicitudes. En un caso, se guarda el usuario que realiza la solicitud y en el otro, es el usuario que autoriza la solicitud.
Cambio 3: Dado que los usuarios son exclusivos por aplicación, es decir, un usuario sólo accede a una aplicación, no es necesaria la tabla “aplicación usuario”.

Cambios Menores:

- Se le agregó el atributo nuevo_valor a la tabla solicitud. En este atributo, se guarda el valor (sin claves foráneas) de los valores nuevos a modificar. De esta forma, no quedan guardadas las claves foráneas y pueden ser guardadas las solicitudes, a la vez que se pueden borrar los datos asociados.
- En la tabla usuario, se agregó un atributo (“cantidad_de_logueos_fallidos”), que permite contar la cantidad de logueos fallidos, a modo que, cuando una persona intenta loguearse fallidamente muchas veces, su usuario queda bloqueado y se crea una solicitud para la nueva habilitación.

- En la tabla usuario, se agrega el atributo “habilitado”, que permite dejar sin funcionamiento, al ser seteado como “false”, para permitir el bloqueo de usuarios, o que un usuario sea creado, pero requiere una autorización para loguearse.

Anexo B

Parámetros de Creación de la Base de Datos

A continuación se detalla la estructura de la base de datos mediante las consultas SQL utilizadas

El detalle de las relaciones entre tablas puede verse en el Modelo Entidad Relación del Anexo A.

```
CREATE TABLE public.auditoria (
    id_auditoria integer NOT NULL,
    id_tipo_evento integer NOT NULL,
    id_usuario_realiza character varying,
    id_rol integer,
    id_usuario character varying,
    fecha date NOT NULL
);

CREATE TABLE public.tipo_de_evento (
    id_tipo_evento integer NOT NULL,
    descripcion character varying NOT NULL
);

CREATE TABLE public.menu (
    id_menu integer NOT NULL,
    descripcion character varying NOT NULL
);

CREATE TABLE public.menu_aplicacion (
    id_menu integer NOT NULL,
    id_aplicacion integer NOT NULL
);
CREATE TABLE public.metodo (
    id_metodo integer NOT NULL,
    descripcion character varying(20) NOT NULL
);
CREATE TABLE public.metodo_menu (
    id_metodo integer NOT NULL,
    id_menu integer NOT NULL
);
CREATE TABLE public.persona (
    ci numeric NOT NULL,
    nombre character varying NOT NULL,
    apellido character varying NOT NULL,
```

```
    fecha_nac character varying,
    correo character varying(30) NOT NULL,
    sexo character(1) NOT NULL
);
CREATE TABLE public.rol (
    id_rol integer NOT NULL,
    descripcion character varying NOT NULL,
    id_aplicacion integer
);

CREATE TABLE public.rol_menu (
    id_rol integer NOT NULL,
    id_menu integer NOT NULL
);

CREATE TABLE public.rol_metodo (
    id_rol integer NOT NULL,
    id_metodo integer NOT NULL
);

CREATE TABLE public.solicitud (
    id_solicitud integer NOT NULL,
    estado character varying NOT NULL,
    fecha_sol date NOT NULL,
    fecha date,
    id_usuario_solicitante character varying,
    id_usuario_autorizante character varying,
    id_tipo_sol integer NOT NULL,
    nuevo_valor character varying
);

CREATE TABLE public.tipo_solicitud (
    id_tipo_solicitud integer NOT NULL,
    descripcion character varying NOT NULL
);

CREATE TABLE public.usuario (
    id_usuario character varying(20) NOT NULL,
    contrasena character varying NOT NULL,
    fecha character varying NOT NULL,
    id_rol integer,
    ci numeric NOT NULL,
    intentos_logueo_fallido numeric NOT NULL,
    habilitado boolean NOT NULL
);
```

Anexo C

Vistas

A continuación se detallan las vistas elaboradas y la función que desempeñan

```
CREATE VIEW public.menus_en_aplicacion AS
 SELECT ma.id_menu,
    m.descripcion AS nombre_menu,
    ma.id_aplicacion,
    a.nombre AS nombre_aplicacion
   FROM ((public.menu_aplicacion ma
     JOIN public.menu m ON ((m.id_menu = ma.id_menu)))
     JOIN public.aplicacion a ON ((a.id_aplicacion = ma.id_aplicacion)));
```

Función: Al editar una aplicación, existe la posibilidad de agregarle o quitarle menús. Para esto, en la pantalla de aplicación-menú, se buscan todas las relaciones con la tabla menu_aplicacion y se cargan en pantalla. Para facilitar la visualización, se crea una vista que ofrezca los datos refiriéndose a los valores por su descripción o nombre, en vez de por un número de id.

```
CREATE VIEW public.rol_en_aplicacion AS
 SELECT r.id_rol,
    r.descripcion,
    a.id_aplicacion,
    a.nombre AS descripcionapp
   FROM (public.aplicacion a
     JOIN public.rol r ON ((r.id_aplicacion = a.id_aplicacion)));
```

Función: Al editar una aplicación, existe la posibilidad de agregarle o quitarle roles. Para esto, en la pantalla de aplicación-rol, se buscan todas las relaciones con la tabla rol_menu y se cargan en pantalla. Para facilitar la visualización, se crea una vista que ofrezca los datos refiriéndose a los valores por su descripción o nombre, en vez de por un número de id.

```
CREATE VIEW public.solicitudes_tipo_solicitudes_usuarios_con_roles_y_ci AS
 SELECT s.id_solicitud,
    s.estado,
    s.fecha_sol,
    s.fecha AS fecha_actualizacion,
    s.id_usuario_solicitante,
    s.id_usuario_autorizante,
    ts.id_tipo_solicitud,
    ts.descripcion AS descripcion_solicitud,
    r.descripcion AS descripcion_rol,
    a.id_aplicacion,
    a.nombre AS nombre_aplicacion,
    u.ci AS ci_solicitante,
    s.nuevo_valor
   FROM ((((public.solicitud s
     JOIN public.tipo_solicitud ts ON ((s.id_tipo_sol = ts.id_tipo_solicitud)))
     JOIN public.usuario u ON (((s.id_usuario_solicitante)::text = (u.id_usuario)::text)))
     LEFT JOIN public.rol r ON ((u.id_rol = r.id_rol)))
     LEFT JOIN public.aplicacion a ON ((r.id_aplicacion = a.id_aplicacion)));
```

Función: En las solicitudes de usuario, es necesario utilizar la ci de cada persona para comprobar que un usuario NO visualice las solicitudes creadas por sí mismo. Los roles son utilizados para evaluar la asignación de un nuevos rol. La tabla tipo_soliditud es utilizada para mostrar descripciones en vez de numero de id. Por último, la aplicación es cargada como referencia visual.

```
CREATE VIEW public.ususarios_con_roles_y_ci AS
 SELECT u.id_usuario,
    u.ci,
    u.fecha,
    r.descripcion AS descripcion_rol,
    a.nombre AS nombre_aplicacion,
    u.habilitado
   FROM ((public.usuario u
     LEFT JOIN public.rol r ON ((u.id_rol = r.id_rol)))
     LEFT JOIN public.aplicacion a ON ((r.id_aplicacion = a.id_aplicacion)));
```

Función: En el login, se utiliza esta vista para reunir el id del usuario, su cédula y su rol. El id del usuario se toma desde la ventana, luego se comprueba si es administrador o no, para poder enviarlo al menú de seguridad o de usuario.También se toma la cédula de identidad relacionada con el usuario para guardarla y utilizarla en la oposición de interés, que considera personas y no usuarios.

```
CREATE VIEW public.auditoria_y_eventos AS
 SELECT a.id_auditoria,
    a.fecha,
    a.id_usuario_realiza AS usuario_solicitante,
    te.descripcion,
    a.id_usuario AS usuario_afectado,
    a.id_rol AS rol_afectado
   FROM (public.auditoria a
     JOIN public.tipo_de_evento te ON ((a.id_tipo_evento = te.id_tipo_evento)));
```

Función: Esta vista nos trae los datos de auditoría junto con los tipos de eventos asociados a la misma ya que de manera aislada solamente contamos con el id del evento. Dentro de nuestro programa podemos aplicarle una condición where para buscar de forma mas específica entre las auditorias.

Bibliografía

sans.org. (2014). Password Security. [online] Available at: https://www.sans.org/reading-room/whitepapers/basics/password-security-thirty-five-years-35592

sans.org. (2003). The Use and Administration of Shared Accounts:. [online] Available at: https://www.sans.org/reading-room/whitepapers/basics/administration-shared-accounts-1271

sans.org. (2002). The Password Web Page. [online] Available at: https://www.sans.org/reading-room/whitepapers/basics/password-web-page-533

elauditor.info (2017) Control Por Oposición de Intereses.[online] Available at: https://elauditor.info/diccionario-del-control/control_a59c2f0d80041ac58e313c594

Database systems - The complete book – Ullman, Widom, Prentice-Hall, 2°Ed. 2002.

Fundamentos de Bases de Datos – Silberschatz, Korth, Sudarshan, MCGraw-Hill Inc. 4°Ed. 2002.

Introducción a los sistemas de Bases de Datos – Date, Prentice-Hall, 7°Ed. 2000.