Gestión de Finanzas Personales

Grado en Ingería Informática de Gestión y Sistemas de Información

TRABAJO FIN DE GRADO
Anexo 3 – Documentación del Código

Universidad del País Vasco Euskal Herriko Unibertsitatea

BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

Alumno: Cuesta Alario, David
Director: López Cuadrado, Javier

Curso: 2020 – 2021

Fecha: 2020-10-19

Índice de contenido

Índice de Ilustraciones

Diagrama de clases

La aplicación trabajará directamente contra la base de datos con la finalidad de que no sea necesario cargar todos los datos permanentes en objetos. De este modo conseguimos:

- Minimizar la cantidad de información temporal que debe ser almacenada.
- Minimizar la carga de trabajo durante el inicio de sesión.

Se almacenará en objetos aquellos cálculos que se realicen durante la ejecución de una sesión con la finalidad de minimizar el número de operaciones evitando volver a realizar las que ya se han ejecutado.

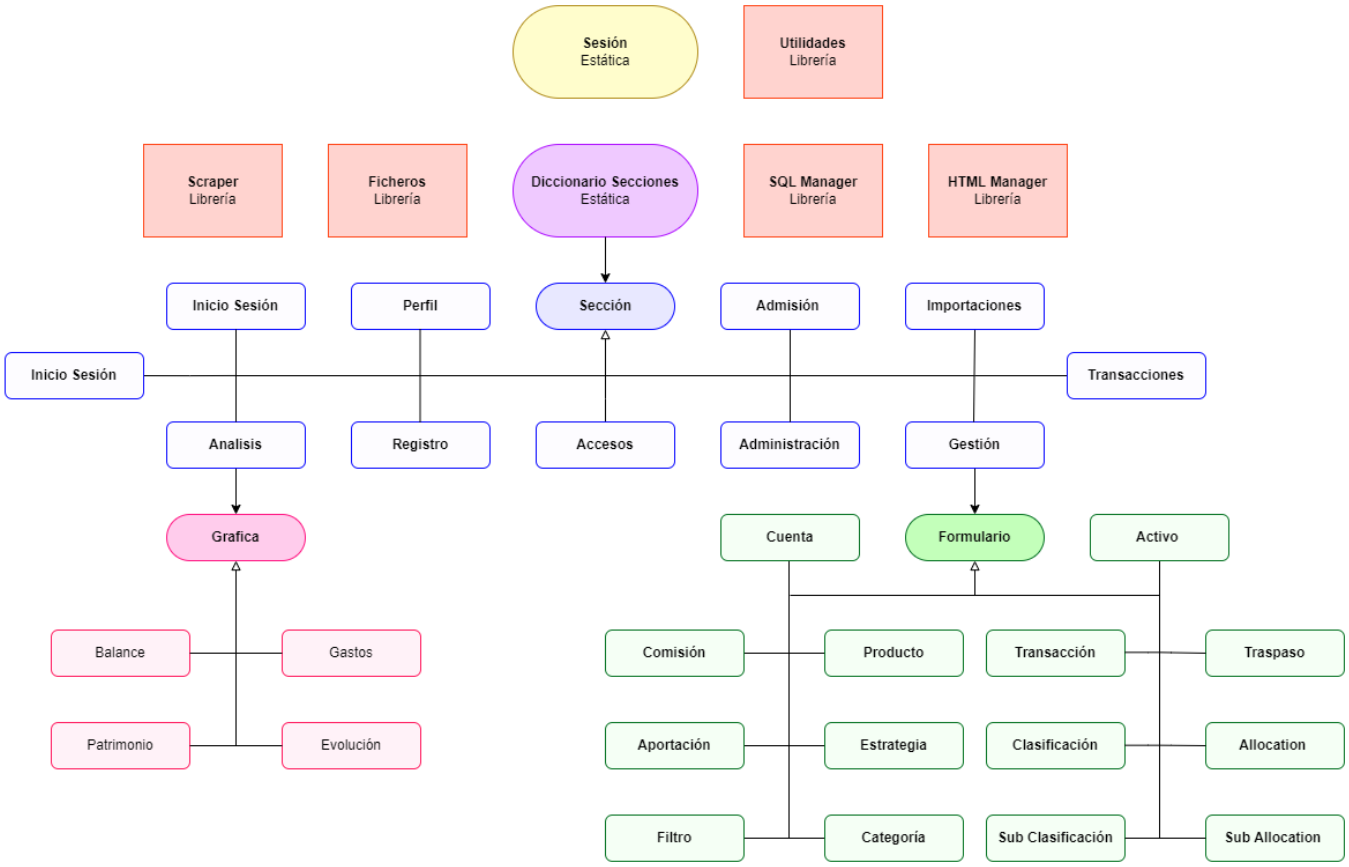

Ilustración 1 Diagrama de clases

A continuación, se explican con detalle los atributos y funciones de cada clase:

Sesión

Finalidad:

- Almacenar aquellos valores temporales que resultan imprescindibles para acceder a los datos permanentes del usuario que este autentificado.
- Gestionar la autentificación de los usuarios mediante el establecimiento y cierre de sesiones.
- Gestionar el acceso a la información permanente de los usuarios.

Características:

- **Estática**: Soló puede existir una única instancia por cada ejecución del programa.

Atributos:

- **Usuario**: Texto cuya finalidad es determinar cuál es el usuario que está identificado.
  - Si el usuario está identificado, su valor será el que aparece en la Base de datos como Usuario.
  - Si el usuario no está identificado, su valor será una cadena de texto vacía.
- **Permisos**: Texto cuya finalidad es determinar qué nivel de acceso tiene el usuario cuando está identificado.
  - Si el usuario está identificado, su valor será el que aparece en la Base de datos como Tipo Usuario.
  - Si el usuario no está identificado, su valor será una cadena de texto vacía.
- **Contraseña**: Texto cuya finalidad es almacenar temporalmente la contraseña para que el resto de la aplicación pueda acceder a la información encriptada.
  - Si el usuario está identificado, su valor será el que el usuario introdujo por teclado.
  - Si el usuario no está identificado, su valor será una cadena de texto vacía.
- **Tiempo Inicio**: Fecha cuya finalidad es almacenar temporalmente la fecha y hora en la que se produjo el inicio de sesión para determinar si el acceso a los datos permanentes ha expirado.
- **Tiempo Permitido:** Número entero cuya finalidad es almacenar la constante que representa el tiempo máximo permitido de inactividad antes de que se fuerce el cierre de la sesión.

Funciones:

- **Sesión:**
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos a sus valores por defecto.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen los valores por defecto.

- **Iniciar Sesión:**
  - **Descripción:** Permite a un usuario no identificado obtener la categoría de usuario identificado.
  - **Funcionamiento:**
    - Delega la comprobación de las condiciones a la función Comprobar Credenciales de la misma clase
    - Si se cumplen las condiciones:
      - Recupera los permisos del usuario.
      - Actualiza los atributos Usuario, Permisos y Contraseña con los valores del usuario.
      - Actualiza el tiempo de inicio con la fecha y hora actual del sistema.
      - Delega la redirección del usuario a la función Redireccionar de la clase Secciones. (Balance)
      - Delega la generación de un acceso a la función Generar Acceso de la librería SQL.
    - Si no se cumplen las condiciones:
      - Delega la generación de un mensaje error a la función Reportar Error la clase Secciones.
      - Delega la generación de un intento de acceso a la función Generar Acceso de la librería SQL.
  - **Entrada**: Valores de cada uno de los campos del formulario de inicio de sesión. (usuario y contraseña)
  - **Salida**: Devuelve un booleano que permite confirmar el inicio de sesión.
    - Booleano verdadero si se ha podido iniciar la sesión.
    - Booleano falso si no se ha podido iniciar la sesión.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**:
    - Si se cumplen las condiciones:
      - Actualiza los atributos Usuario, Permisos y Contraseña con los valores del usuario.
      - Actualiza el tiempo de inicio con la fecha y hora actual del sistema.
- **Cerrar Sesión:**
  - **Descripción:** Finaliza el acceso a los datos permanentes del usuario.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos a sus valores por defecto.
    - Delega la redirección del usuario a la función Redireccionar de la clase Secciones. (Inicio)
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: El usuario debe estar registrado y correctamente identificado.
  - **Postcondiciones**: Los atributos tienen los valores por defecto.
- **Añadir admisión**
  - **Descripción:** Permite añadir el usuario a la lista de admisiones.
  - **Funcionamiento:**
    - Delega la comprobación de las condiciones que se deben cumplir para añadir la admisión a la función comprobar admisión de la misma clase.
    - Calcula la sal y el hash de la contraseña.
    - Delega el proceso de añadir al usuario sin permisos a la función crear usuario de la librería SQL.
    - Delega el proceso de añadir la admisión a la función Añadir admisión de la librería SQL.
  - **Entrada**: Valores de cada uno de los campos del formulario de registro.
  - **Salida**:
    - Booleano Verdadero si el usuario se ha registrado.
    - Booleano Falso si el usuario no se ha registrado.
  - **Precondiciones**: Ninguna.

- **Postcondiciones**: Si el usuario se registra correctamente se le redirecciona a la sección de Inicio de sesión.

- **Añadir usuario**
  - **Descripción:** Permite añadir el usuario a la lista de usuarios.
  - **Funcionamiento:**
    - Obtiene los datos de la admisión que se va a convertir en usuario.
    - Delega el proceso de añadir al usuario a la función Añadir usuario de la librería SQL.
    - Delega el proceso crear la base de datos del usuario a la función Crear BD de la librería SQL.
    - Delega el proceso de agregar permisos al usuario a la función Asignar Permisos al Usuario de la librería SQL.
  - **Entrada**:
    - Nombre de usuario de la admisión que se ha aceptado.
    - Nivel de permisos que se le asignan al nuevo usuario.
  - **Salida**: Booleano Verdadero.
  - **Precondiciones**: La admisión aceptada existe.
  - **Postcondiciones**:
    - Se crea la Base de Datos del nuevo usuario.
    - Se asignan los permisos al usuario.
- **Añadir Aportación:**
  - **Descripción:** Permite al usuario identificado añadir una aportación.
  - **Funcionamiento:**
    - Delega el proceso de añadir la aportación a la función Añadir aportación de la librería SQL. (Inicio)
  - **Entrada**: Valores de cada uno de los campos de la tabla de aportaciones.
  - **Salida**: Ninguna.
  - **Precondiciones**: Se ha comprobado previamente que se cumplen los requisitos para añadir la aportación.
  - **Postcondiciones**: La aportación se añade a la base de datos.
- **Añadir Comisión:**
  - **Descripción:** Permite al usuario identificado añadir una comisión.
  - **Funcionamiento:**
    - Delega el proceso de añadir la Comisión a la función Añadir comisión de la librería SQL. (Inicio)
  - **Entrada**: Valores de cada uno de los campos de la tabla de comisiones.
  - **Salida**: Ninguna.
  - **Precondiciones**: Se ha comprobado previamente que se cumplen los requisitos para añadir la comisión.
  - **Postcondiciones**: La comisión se añade a la base de datos.
- **Añadir Transacción:**
  - **Descripción:** Permite al usuario identificado añadir una transacción.
  - **Funcionamiento:**
    - Delega el proceso de añadir la transacción a la función Añadir transacción de la librería SQL .(Inicio)
  - **Entrada**: Valores de cada uno de los campos de la tabla de transacciones.
  - **Salida**: Ninguna.
  - **Precondiciones**: Se ha comprobado previamente que se cumplen los requisitos para añadir la transacción.
  - **Postcondiciones**: La transacción se añade a la base de datos.

- **Añadir Traspaso:**
  - **Descripción:** Permite al usuario identificado añadir un traspaso.
  - **Funcionamiento:**
    - Delega el proceso de añadir el traspaso a la función Añadir traspaso de la librería SQL .(Inicio)
  - **Entrada**: Valores de cada uno de los campos de la tabla de traspasos.
  - **Salida**: Ninguna.
  - **Precondiciones**: Se ha comprobado previamente que se cumplen los requisitos para añadir el traspaso.
  - **Postcondiciones**: El traspaso se añade a la base de datos.
- **Eliminar Usuario:**
  - **Descripción:** Elimina toda la información permanente del usuario objetivo.
  - **Funcionamiento:**
    - Delega la eliminación de los datos permanentes a la función Eliminar Usuario de la librería SQL.
    - Delega la eliminación del usuario a la función Eliminar Usuario de la librería SQL.
    - Si el usuario objetivo el usuario identificado.
    - Delega el cierre de la sesión a la función Cerrar sesión de la misma clase.
  - **Entrada**:
    - Usuario objetivo o una cadena de texto vacía si el usuario objetivo es el usuario identificado**.**
  - **Salida**: Ninguna.
  - **Precondiciones**: El usuario objetivo existe.
  - **Postcondiciones**: El usuario y la información permanente asociada ya no existe.
- **Eliminar Admisión:**
  - **Descripción:** Elimina el registro de la admisión de un usuario.
  - **Funcionamiento:**
    - Delega la eliminación de la admisión a la librería SQL.
  - **Entrada**:
    - Usuario objetivo o una cadena de texto vacía si el usuario objetivo es el usuario identificado.
  - **Salida**: Ninguna.
  - **Precondiciones**: La admisión objetivo existe.
  - **Postcondiciones**: La admisión objetivo ya no existe.

- **Actualizar Credenciales:**
  - **Descripción:** Restablece las credenciales del usuario identificado por las recibidas por parámetro.
  - **Funcionamiento:**
    - Comprueba si las contraseñas coinciden.
    - Comprueba si la contraseña anterior coincide con la actual.
    - Si cumple todas las condiciones delega la actualización de la contraseña a la función Actualizar Contraseña de la misma clase.
  - **Entrada**: Valores de cada uno de los campos del formulario de inicio de cambio de contraseña. (contraseña anterior, contraseña nueva y contraseña repetida)
  - **Salida**:
    - Booleano verdadero si cumple todas las condiciones para cambiar la contraseña.
    - Booleano Falso si no cumple alguna de las condiciones para cambiar la contraseña.
  - **Precondiciones**: El usuario debe estar registrado y correctamente identificado.
  - **Postcondiciones**: Si no se cumple alguna condición modifica el mensaje de error indicando que condición es la que no se cumple.
- **Actualizar Contraseña:**
  - **Descripción:** Restablece las credenciales del usuario identificado por las recibidas por parámetro.
  - **Funcionamiento:**
    - Calcula una nueva sal para el usuario identificado.
    - Calcula el nuevo Hash para el usuario identificado.
    - Sustituye en la Base de Datos los nuevos valores de la Sal y el Hash.
  - **Entrada**: Texto que representa la nueva contraseña para el usuario.
  - **Salida**: Ninguna
  - **Precondiciones**: El parámetro de contraseña puede ser el mismo que la contraseña anterior.
  - **Postcondiciones**: Se actualizan las credenciales del usuario.

- **Actualizar Usuario:**
  - **Descripción:** Modifica los datos permanentes del usuario que está actualmente registrado con los datos que recibe por parámetro.
  - **Funcionamiento:** Delega el proceso de actualizar los datos permanentes del usuario a la función Actualizar Usuario de la librería SQL.
  - **Entrada**: Datos introducidos por teclado en el formulario Editar Perfil: nombre, primer apellido, segundo apellido, DNI, teléfono y email.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Se actualizan los datos permanentes del usuario.
- **Comprobar Permisos:**
  - **Descripción:** Permite conocer si los permisos del usuario son mayores que los recibidos como parámetro. Esta función se llamará cada vez que el usuario trate de acceder a una nueva sección. Su finalidad es determinar si se le permite o deniega el acceso.
  - **Funcionamiento:**
    - Comprueba si el usuario esta registrado.
    - Comprueba si los permisos recibidos como parámetro son válidos. (No Identificado, Usuario, Administrador, Creador)
    - Compara los permisos del usuario con los recibidos como parámetro
  - **Entrada**:
    - Texto que representa si:
      - El nivel de permisos puede ser Estrictamente Igual que el recibido como parámetro. (=)
      - El nivel de permisos puede ser Igual o Superior que el recibido como parámetro. (>=)
      - el nivel de permisos puede ser Igual o Inferior que el recibido como parámetro. (<=)
      - El nivel de permisos puede ser Superior que el recibido como parámetro. (>)
      - el nivel de permisos puede ser Menor que el recibido como parámetro. (<)
    - Texto que representa el nivel de permisos con el que se pretende comparar los permisos del usuario.
  - **Salida**:
    - Booleano Falso si no hay ningún usuario registrado.
    - Booleano Falso si los permisos recibidos como parámetro no son válidos.
    - Booleano Falso si los permisos del usuario no son acordes a los recibidos como parámetro.
    - Booleano Verdadero si los permisos del usuario son acordes a los recibidos como parámetro.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Comprobar Tiempo Sesión:**
  - **Descripción:** Permite conocer si se ha terminado el tiempo de sesión por inactividad**.**
  - **Características**: Se ejecutará cada vez que se intente un acceso a la información permanente.
  - **Funcionamiento:**
    - Obtiene la fecha y hora actual del sistema.
    - Suma el tiempo máximo permitido de inactividad a la fecha y hora registrada al iniciar sesión.
    - Si el tiempo sesión no supera el máximo permitido.
      - Actualiza la variable Tiempo de Inicio con la fecha y hora actual.
  - **Entrada**: Ninguna.
  - **Salida**:
    - Booleano Verdadero si el tiempo de sesión ha expirado.
    - Booleano Falso si el tiempo de sesión no ha expirado.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Si el tiempo de sesión no ha expirado actualiza el tiempo de sesión.
- **Comprobar Sesión Iniciada:**
  - **Descripción:** Permite conocer si la sesión está iniciada
  - **Funcionamiento:** Si el atributo usuario de la sesión está vacío, la sesión no está iniciada.
  - **Entrada**: Ninguna.
  - **Salida**:
    - Booleano Verdadero si la sesión está iniciada.
    - Booleano Falso si la sesión no está iniciada.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Si el tiempo de sesión no ha expirado, actualiza el tiempo de sesión.
- **Comprobar Contraseña:**
  - **Descripción:** Permite conocer si la contraseña de un usuario es correcta.
  - **Funcionamiento:**
    - Delega a la librería SQL la obtención de la Sal para ese usuario.
    - Calcula el Hash de la contraseña recibida como parámetro utilizando la Sal.
    - Delega a la librería SQL la comprobación de si el Hash calculado coincide con el almacenado por dicho usuario.
  - **Entrada**:
    - Usuario cuya contraseña se quiere verificar.
    - Contraseña que se pretende hacer pasar por la correcta.
  - **Salida**:
    - Booleano Verdadero si la contraseña es correcta.
    - Booleano Falso si la contraseña no es correcta.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Comprobar Credenciales:**
  - **Descripción:** Permite conocer si las credenciales de un usuario son correctas.
  - **Funcionamiento:**
    - Comprueba si existe algún usuario registrado con ese nombre de usuario.
    - Comprueba si existe alguna admisión con ese nombre de usuario.
    - Delega la comprobación de si la contraseña es correcta a la función Comprobar contraseña de la misma clase.
  - **Entrada**: Valores de cada uno de los campos del formulario de inicio de sesión. (usuario y contraseña)
  - **Salida**:
    - Booleano Verdadero si se cumplen todas las condiciones.
    - Booleano Falso si no se cumplen alguna condición.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Si no se cumple alguna condición modifica el mensaje de error, indicando que condición es la que no se cumple.
- **Comprobar admisión:**
  - **Descripción:** Permite determinar si el usuario puede ser añadido a la lista de admisiones.
  - **Funcionamiento:**
    - Comprobar si dicho usuario no existe en las tablas de usuarios y admisiones.
    - Comprobar que las contraseñas coincidan.
    - Comprobar que se halya marcada la casilla de Términos y condiciones.
  - **Entrada**: Valores de cada uno de los campos del formulario de registro.
  - **Salida**:
    - Booleano Verdadero si el usuario puede registrarse.
    - Booleano Falso si el usuario no puede registrarse.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Genera un mensaje de error enumerando las condiciones que no se cumplen.
- **Existe Cuenta:**
  - **Descripción:** Comprueba si el usuario identificado tiene registrada la cuenta cuyo identificador se recibe como parámetro.
  - **Funcionamiento:** Delega la comprobación a la función Existe Cuenta de la librería SQL.
  - **Entrada**: Identificador de la cuenta que se quiere comprobar si existe.
  - **Salida**:
    - Booleano verdadero si la cuenta existe.
    - Booleano falso si la cuenta no existe.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Existe Activo:**
  - **Descripción:** Comprueba si el usuario identificado tiene registrado el activo cuyo identificador se recibe como parámetro.
  - **Funcionamiento:** Delega la comprobación a la función Existe Activo de la librería SQL.
  - **Entrada**: Identificador del activo que se quiere comprobar si existe.
  - **Salida**:
    - Booleano verdadero si el activo existe.
    - Booleano falso si el activo no existe.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Existe Aportación:**
  - **Descripción:** Comprueba si el usuario identificado tiene registrada la aportación cuyos identificadores se reciben como parámetro.
  - **Funcionamiento:** Delega la comprobación a la función Existe Aportación de la librería SQL.
  - **Entrada**: Identificadores de la aportación que se quiere comprobar si existe.
  - **Salida**:
    - Booleano verdadero si el activo existe..
    - Booleano falso si el activo no existe.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Existe Comisión:**
  - **Descripción:** Comprueba si el usuario identificado tiene registrada la comisión cuyos identificadores se reciben como parámetro.
  - **Funcionamiento:** Delega la comprobación a la función Existe Comisión de la librería SQL.
  - **Entrada**: Identificadores de la comisión que se quiere comprobar si existe.
  - **Salida**:
    - Booleano verdadero si el activo existe.
    - Booleano falso si el activo no existe.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Existe Transacción:**
  - **Descripción:** Comprueba si el usuario identificado tiene registrada la transacción cuyos identificadores se reciben como parámetro.
  - **Funcionamiento:** Delega la comprobación a la función Existe Transacción de la librería SQL.
  - **Entrada**: Identificadores de la transacción que se quiere comprobar si existe.
  - **Salida**:
    - Booleano verdadero si el activo existe.
    - Booleano falso si el activo no existe.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Existe Traspaso:**
  - **Descripción:** Comprueba si el usuario identificado tiene registrado el traspaso cuyos identificadores se reciben como parámetro.
  - **Funcionamiento:** Delega la comprobación a la función Existe Traspaso de la librería SQL.
  - **Entrada**: Identificadores del traspaso que se quiere comprobar si existe.
  - **Salida**:
    - Booleano verdadero si el activo existe.
    - Booleano falso si el activo no existe.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Existe Clasificación:**
  - **Descripción:** Comprueba si el usuario identificado tiene registrada la clasificación cuyos identificadores se reciben como parámetro.
  - **Funcionamiento:** Delega la comprobación a la función Existe Clasificación de la librería SQL.
  - **Entrada**: Identificadores de la clasificación que se quiere comprobar si existe.
  - **Salida**:
    - Booleano verdadero si el activo existe.
    - Booleano falso si el activo no existe.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Get Usuario:**
  - **Descripción:** Obtiene de la base de datos todos los campos del usuario objetivo.
  - **Funcionamiento:** Delega la obtención de los datos a la librería SQL.
  - **Entrada**: Usuario objetivo o una cadena de texto vacía si el usuario objetivo es el usuario identificado.
  - **Salida**: Todos los datos del usuario objetivo.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Get Usuarios:**
  - **Descripción:** Obtiene de la base de datos todos los campos de todos los usuarios.
  - **Funcionamiento:** Delega la obtención de los datos a la librería SQL.
  - **Entrada**: Ninguna.
  - **Salida**: Lista de usuarios con todos los datos para cada usuario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Get Admisión:**
  - **Descripción:** Obtiene de la base de datos todos los campos de la admisión objetivo.
  - **Funcionamiento:** Delega la obtención de los datos a la librería SQL.
  - **Entrada**: Nombre de usuario de la Admisión objetivo.
  - **Salida**: Todos los datos de la admisión objetivo.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Get Admisiones:**
  - **Descripción:** Obtiene de la base de datos todos los campos de todas las admisiones.
  - **Funcionamiento:** Delega la obtención de los datos a la librería SQL.
  - **Entrada**: Ninguna.
  - **Salida**: Lista de admisiones con todos los datos para cada admisión.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Get Activos:**
  - **Descripción:** Obtiene de la base de datos todos los campos de todos los activos.
  - **Funcionamiento:** Delega la obtención de los datos a la librería SQL.
  - **Entrada**: Ninguna.
  - **Salida**: Lista de activos.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Secciones

Finalidad:

- Clasificar las funcionalidades de la aplicación conforme a una estructura que facilite:
  - El acceso a las características de cada funcionalidad.
  - La incorporación futura de nuevas funcionalidades
- Gestionar la navegación a través de las diferentes secciones de la aplicación.
- Podemos diferenciar las siguientes secciones:
  - **Principal**: Ofrece información general.
    - **Guía**: Sobre el funcionamiento de la aplicación
    - **Blog**: Guía de finanzas personales
  - **Sesión**: Gestiona la sesión.
    - **Iniciar Sesión**: Permite a un usuario no identificado iniciar la sesión.
    - **Cerrar Sesión**: Permite a un usuario identificado cerrar la sesión.
  - **Registro:** Permite a un usuario no identificado realizar una solicitud de admisión.
  - **Configuración:** Permite al usuario identificado personalizar la sesión.
    - **Perfil**: Permite modificar los datos registro.
  - **Admisiones:** Permite a los usuarios administradores:
    - **Admisión**: Gestionar las solicitudes de admisión.
    - **Administración**: Gestionar los permisos de los usuarios registrados.
    - **Accesos**: Comprobar los accesos de los usuarios registrados.
  - **Transacciones:** Permite al usuario identificado gestionar sus datos de manera individual.
  - **Importaciones:** Permite al usuario identificado gestionar sus datos financieros a granel.
    - **Importaciones**: Permite modificar los datos registro
    - **Exportaciones**: Permite modificar los datos registro

Características:

- **Estática**: Sólo puede existir una única instancia por cada ejecución del programa.

Atributos:

- **Secciones**: Diccionario que contiene cada una de las secciones que conforman la aplicación.
- **Sección Actual**: Texto que representa la sección que está visualizando actualmente el usuario.
- **Subsección Actual**: Texto que representa la subsección que se está visualizando actualmente.
- **Parámetros**: Texto que almacena los parámetros que necesiten las secciones que así los requieran.
- **Mensaje**: Texto que representa el siguiente mensaje de error que debe mostrarse al usuario:
  - Si no se ha producido ningún error que notificar al usuario será una cadena de texto vacío.
  - Si hay algún error que notificar al usuario será un texto con dicho mensaje.

Funciones:

- **Secciones:**
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Llama a la constructora de cada una de las secciones que conforman la aplicación.
    - Añade todas las secciones al diccionario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Todas las secciones de la aplicación han sido inicializadas y añadidas al diccionario.
- **Reportar Error:**
  - **Descripción**: Modifica el valor del atributo Mensaje con el parámetro recibido.
  - **Entrada**: Texto con el mensaje de error que debe mostrarse al usuario.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Se actualiza el valor del atributo Mensaje.
- **Redireccionar:**
  - **Descripción**: Permite indicar cuál es la sección y subsección que se debe visualizar.
  - **Funcionamiento:**
    - Modifica el valor del atributo Subsección actual.
    - En función de la subsección actual se calcula la sección actual.
  - **Entrada**: Texto con el identificador de la nueva subsección actual que debe mostrarse al usuario.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Se actualiza el valor de los atributos subsección actual y sección actual.

- **Generar Redirección:**
  - **Descripción**: Generar el código HTML que permite efectuar la redirección.
  - **Funcionamiento:**
    - Separa la dirección de la nueva sección de los parámetros.
    - Actualiza el valor de los parámetros.
    - Busca la nueva sección actual en el diccionario de secciones.
    - Si la sección no existe:
      - Delega la generación de un mensaje error a la función Reportar Error la misma clase.
      - Cambia a la sección actual a la principal mediante la función redireccionar de la misma clase.
    - Si la sección existe:
      - Delega la comprobación de los permisos de la nueva sección a la función Comprobar Acceso de la clase Sección de la nueva sección actual.
      - Si el usuario tiene acceso a la nueva sección:
        - Cambia a la sección actual a la indicada por parámetro mediante la función redireccionar de la misma clase.
      - Si el usuario no tiene acceso:
        - Delega la generación de un mensaje error a la función Reportar Error de la misma clase.
        - Cambia a la sección actual a la de inicio de sesión mediante la función Redireccionar de la misma clase.
    - Ejecuta la función redirect de la librería Flask utilizando como parámetro el identificador de la sección actual.
  - **Entrada**: texto con el identificador de la nueva dirección a la que se pretende acceder y los parámetros que requiera la subsección que ha sido llamada separados por el carácter '/'.
  - **Salida**: Texto con formato HTML que representa el código de redirección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**:
    - El contenido del atributo Redirección es una cadena de texto vacía.
    - El contenido del atributo Sección Actual se actualiza.
- **Generar Índice**:
  - **Descripción**: Genera dinámicamente el índice de navegación de la sección.
  - **Funcionamiento:** Genera un menú de navegación utilizando la función índice de la librería HTML basándose en:
    - El nivel de permisos de la sección actual y del usuario.
    - La sección y subsección actuales.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el índice de navegación de la sección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Generar Página:**
  - **Descripción**: Genera dinámicamente la página que se muestra al usuario en función de:
    - La sección actual.
    - El último mensaje de error que debe mostrarse al usuario
    - Si existe alguna redirección pendiente.
  - **Funcionamiento:**
    - Comprueba si existe alguna redirección pendiente.
      - Existe una redirección pendiente cuando la función Generar Contenido de la clase Sección devuelve una cadena vacía como contenido de la página actual.
    - Si existe delega alguna redirección pendiente delega el proceso de redirección a la función Redirect de la librería Flask.
    - Si no existe ninguna redirección pendiente:
      - Añade la cabecera de la página.
      - Delega la generación del índice de navegación de la sección actual a la función Generar Índice de la misma clase.
      - Delega la generación del mensaje de error a la función Generar Mensaje Error de la librería HTML.
      - Delega la generación del contenido de la sección actual a la función Generar Contenido de la clase Sección para la Sección actual.
      - Añade el pie de la página.
  - **Entrada**: Ninguna.
  - **Salida**:
    - Si existe alguna redirección pendiente: Texto con formato HTML que efectuara la redirección.
    - Si no existe ninguna redirección pendiente: Texto con formato HTML que concatena el índice de navegación, el mensaje de error y el contenido de la sección actual.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Sección

Finalidad:

- Generar el contenido para cada sección de manera universal.
- Evaluar la accesibilidad del usuario a cada sección.

Atributos:

- **Sección**: Texto que almacena una constante que representa distintas las secciones mediante un identificador único permitiendo diferenciarlas.
- **Subsección**: Texto que almacena una constante que representa distintas las subsecciones mediante un identificador único, permitiendo diferenciarlas.
- **Título**: Texto con el que se representa la sección.
- **Estricto**: Texto que representa la relación que debe existir entre los permisos del usuario y de la sección para que el usuario pueda acceder.
  - "=": Los permisos del usuario y de la sección deben coincidir.
  - ">": Los permisos del usuario deben ser mayores que los de la sección.
  - "<": Los permisos del usuario deben ser menores que los de la sección.
  - ">=": Los permisos del usuario deben ser mayores o iguales que los de la sección
  - "<=": Los permisos del usuario deben ser menores o iguales que los de la sección.
- **Permisos**: Texto que almacena la constante que representa el nivel de permisos necesario para que el usuario pueda acceder a esta sección. (No Identificado, Usuario, Administrador, Creador)

Funciones:

- **Sección**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos con sus valores por defecto.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen los valores por defecto.
- **Comprobar Acceso**:
  - **Descripción**: Permite determinar si el usuario puede acceder a esta sección.
  - **Funcionamiento**:
    - Delega la comparación de los permisos del usuario con los permisos necesarios de la sección a la función Comprobar Permisos de la clase Sesión.
    - Delega la comprobación de Si la sesión esta iniciada a la función Comprobar Sesión iniciada de la clase Sesión.
    - Si la sesión esta iniciada y los permisos de acceso son suficientes delega la comprobación de Si la sesión ha caducado a la función Comprobar Tiempo de Sesión de la clase Sesión.
  - **Entrada**: Ninguna.
  - **Salida**:
    - Booleano Verdadero si el usuario puede acceder a la sección.
    - Booleano Falso si el usuario no puede acceder a la sección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Generar Índice**:
  - **Descripción**: Genera dinámicamente el índice de la sección.
  - **Características**: Esta función debe ser reescrita por sus clases herederas.
  - **Funcionamiento:** Genera dinámicamente el índice de la sección.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el índice de la sección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Generar Contenido**:
  - **Descripción**:
    - Genera dinámicamente el contenido de la sección.
    - Está pendiente de los inputs introducidos por el usuario permitiendo:
      - Capturar los campos de los formularios mediante la función Request de la librería Flask
      - Interpretar cuando es necesario efectuar una redirección y notificarlo a la clase Secciones devolviendo una cadena vacía como contenido de la página.
  - **Características**: Esta función puede ser reescrita por sus clases herederas si necesitan realizar una funcionalidad más específica.
  - **Funcionamiento:**
    - Delega la generación del formulario a la función Generar Formulario de la misma clase.
    - Delega la preparación de la página para recibir los campos del formulario a la función Capturar campos de la misma clase.
  - **Entrada**: Ninguna.
  - **Salida**:
    - Texto con formato HTML que representa el contenido de la sección.
    - Una cadena vacía en caso de que sea necesario efectuar una redirección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Generar Formulario:**
  - **Descripción:** Permite generar un formulario.
  - **Características**: Esta función debe ser reescrita por sus clases herederas.
  - **Funcionamiento**: Delega su ejecución a las clases herederas.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario de la sección correspondiente.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Capturar Campos Global:**
  - **Descripción:** Permite capturar los campos que se envían en los formularios de las diferentes secciones.
  - **Funcionamiento:**
    - Captura el parámetro “Redirigir” encargado de detectar cuál es la funcionalidad que sea activado.
    - Delega la captura de campos específicos de cada sección a la función capturar campos.
    - Captura el campo “Ordenar” encargado de gestionar el orden de las filas en tablas.
  - **Entrada**: Ninguna.
  - **Salida**:
    - Booleano falso por defecto.
    - Booleano verdadero si es necesario refrescar la página para efectuar una redirección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción:** Permite capturar los campos que se envían en los formularios de las diferentes secciones.
  - **Características**: Esta función debe ser reescrita por sus clases herederas.
  - **Funcionamiento:** Delega su ejecución a las clases herederas.
  - **Entrada**:
    - Texto que permite detectar que funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso por defecto.
    - Booleano verdadero si es necesario refrescar la página para efectuar una redirección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Inicio Sesión

Finalidad:

- Generar el contenido relacionado con el inicio de sesión.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Sección.
- **Permisos Necesarios**: No Identificado.

Funciones:

- **Inicio Sesión**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos de su clase predecesora.
  - **Entrada**:
    - **Sección:** Texto que representa el nombre de la sección “Sesión”.
    - **Subsección:** Texto que representa el nombre de la subsección “Inicio de Sesión”.
    - **Título**: Texto con el que se representa la sección.
    - **Permisos**: Número entero que representa el nivel de permisos de la sección.
    - **Estricto**: Texto que representa la relación que debe existir entre los permisos del usuario y de la sección para que el usuario pueda acceder:
      - “=”: Los permisos del usuario y de la sección deben coincidir.
      - “>”: Los permisos del usuario deben ser mayores que los de la sección.
      - “<”: Los permisos del usuario deben ser menores que los de la sección.
      - “>=”: Los permisos del usuario deben ser mayores o iguales que los de la sección.
      - “<=”: Los permisos del usuario deben ser menores o iguales que los de la sección.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción:** Genera el formulario de Inicio de Sesión que permite a cualquier usuario no identificado introducir sus credenciales para identificarse.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario de inicio de sesión.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción:** Prepara a la página para recibir los campos del formulario.
  - **Funcionamiento:**
    - Si Redirigir es “Iniciar Sesión”:
      - Captura los campos del formulario de Inicio de Sesión.
      - Delega el proceso de inicio de sesión a la función Iniciar Sesión de la clase Sesión.
  - **Entrada**: Texto que permite detectar qué funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso si la sesión no se ha iniciado.
    - Booleano verdadero si la sesión se ha iniciado correctamente.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Cerrar Sesión

Finalidad:

- Generar el procedimiento que permite cerrar la sesión.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Sección.
- **Permisos Necesarios**: No Identificado.

Funciones:

- **Cerrar Sesión**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos de su clase predecesora.
  - **Entrada**:
    - **Sección:** Texto que representa el nombre de la sección “Sesión”.
    - **Subsección:** Texto que representa el nombre de la subsección “Cerrar Sesión”.
    - **Título**: Texto con el que se representa la sección.
    - **Permisos**: Número entero que representa el nivel de permisos de la sección.
    - **Estricto**: Texto que representa la relación que debe existir entre los permisos del usuario y de la sección para que el usuario pueda acceder.
      - “=”: Los permisos del usuario y de la sección deben coincidir.
      - “>”: Los permisos del usuario deben ser mayores que los de la sección.
      - “<”: Los permisos del usuario deben ser menores que los de la sección.
      - “>=”: Los permisos del usuario deben ser mayores o iguales que los de la sección.
      - “<=”: Los permisos del usuario deben ser menores o iguales que los de la sección.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Contenido:**
  - **Descripción:** Permite cerrar la sesión que estuviera iniciada.
  - **Funcionamiento:** Delega el proceso de cerrar sesión a la función cerrar sesión de la clase Sesión.
  - **Entrada**: Ninguna.
  - **Salida**: Una cadena de texto vacía si la sesión se ha cerrado correctamente.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Registro

Finalidad:

- Generar el contenido relacionado con el registro de nuevos usuarios
- Gestionar la inserción de nuevas admisiones utilizando los datos del formulario de registro.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Sección.
- **Permisos Necesarios**: No Identificado.

Atributos:

- **Nombre**: Texto que permite almacenar temporalmente el último valor introducido en el campo Nombre del formulario de registro.
- **Primer Apellido**: Texto permite almacenar temporalmente el último valor introducido en el campo Primer Apellido del formulario de registro.
- **Segundo Apellido**: Texto almacena temporalmente el último valor introducido en el campo Segundo Apellido del formulario de registro.
- **DNI**: Texto permite almacenar temporalmente el último valor introducido en el campo DNI del formulario de registro.
- **Teléfono**: Texto que permite almacenar temporalmente el último valor introducido en el campo Teléfono del formulario de registro.
- **Email**: Texto que permite almacenar temporalmente el último valor introducido en el campo Email del formulario de registro.

Funciones:

- **Registro:** Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos a sus valores por defecto.
    - Inicializa los valores de los atributos de su clase predecesora.
  - **Entrada**:
    - **Sección:** Texto que representa el nombre de la sección “Registro”.
    - **Subsección:** Texto que representa el nombre de la subsección “Registro”.
    - **Título**: Texto con el que se representa la sección.
    - **Permisos**: Número entero que representa el nivel de permisos de la sección.
    - **Estricto**: Texto que representa la relación que debe existir entre los permisos del usuario y de la sección para que el usuario pueda acceder.
      - “=”: Los permisos del usuario y de la sección deben coincidir.
      - “>”: Los permisos del usuario deben ser mayores que los de la sección.
      - “<”: Los permisos del usuario deben ser menores que los de la sección.
      - “>=”: Los permisos del usuario deben ser mayores o iguales que los de la sección.
      - “<=”: Los permisos del usuario deben ser menores o iguales que los de la sección.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Todos los atributos tienen los valores por defecto.

- **Generar Formulario:**
  - **Descripción**: Genera el formulario de registro que permite a cualquier usuario no identificado añadir su solicitud de admisión.
  - **Funcionamiento:**
    - Utiliza el valor de sus atributos para asignar el valor inicial a los inputs del formulario de Registro.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa la ventana del formulario de registro.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción**: Da comienzo al proceso de registro de un usuario.
  - **Funcionamiento:**
    - Si redirigir es "Añadir Admisión":
      - Captura los campos del formulario de Registro.
      - Modifica los valores de los atributos con los campos del formulario.
      - Delega la generación de la nueva admisión a la función Añadir Admisión de la clase Sesión.
  - **Entrada**:
    - Texto que permite detectar qué funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso si no se puede añadir el usuario a la lista de admisión.
    - Booleano verdadero si se puede añadir el usuario a la lista de admisión.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Si se cumplen las condiciones añade una nueva admisión.

Perfil

Finalidad:

- Generar el contenido relacionado con el registro de nuevos usuarios.
- Gestionar la edición de datos personales con los datos del formulario de Editar Perfil.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Sección.
- **Permisos Necesarios**: Usuario, Administrador o Creador.

Funciones:

- **Perfil**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos de su clase predecesora.
  - **Entrada**:
    - **Sección:** Texto que representa el nombre de la sección “Configuración”.
    - **Subsección:** Texto que representa el nombre de la subsección “Perfil”.
    - **Título**: Texto con el que se representa la sección.
    - **Permisos**: Número entero que representa el nivel de permisos de la sección.
    - **Estricto**: Texto que representa la relación que debe existir entre los permisos del usuario y de la sección para que el usuario pueda acceder.
      - “=”: Los permisos del usuario y de la sección deben coincidir.
      - “>”: Los permisos del usuario deben ser mayores que los de la sección.
      - “<”: Los permisos del usuario deben ser menores que los de la sección.
      - “>=”: Los permisos del usuario deben ser mayores o iguales que los de la sección.
      - “<=”: Los permisos del usuario deben ser menores o iguales que los de la sección.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción:** Genera los formularios de la sección Perfil que permitirán modificar la información permanente del usuario identificado.
  - **Funcionamiento:**
    - Delega generación del formulario para modificar los datos del perfil a la función Formulario Editar Perfil de la misma clase.
    - Delega generación del formulario para modificar la contraseña a la función Formulario Cambiar Contraseña de la misma clase.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa los formularios de la sección Perfil.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Formulario Editar Perfil:**
  - **Descripción:** Genera el formulario de Editar Perfil que permite modificar los datos personales del usuario identificado.
  - **Funcionamiento:**
    - Delega la obtención de los datos personales del usuario identificado a la función Desencriptar Usuario de la clase Sesión.
    - Utiliza el valor de los datos personales del usuario obtenido de la Base de Datos para asignar el valor inicial a los inputs del formulario de Editar Perfil.
    - Incluye un campo oculto con el identificador "Redirigir" que permite identificar este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario de Editar Perfil.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Formulario Cambiar Contraseña:**
  - **Descripción:** Genera el formulario de cambiar contraseña que permite al usuario identificado modificar su contraseña.
  - **Funcionamiento:**
    - Solicita al usuario su contraseña anterior y la contraseña que la sustituirá.
    - Incluye un campo oculto con el identificador "Redirigir" que permite identificar este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario de Editar Perfil.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción:** Da comienzo al proceso de Editar Perfil.
  - **Funcionamiento:**
    - Si redirigir es "Editar Perfil":
      - Captura los campos del formulario de Editar Perfil.
      - Delega el proceso de actualizar los datos del usuario a la función Actualizar Usuario de la clase Sesión.
    - Si redirigir es "Editar Contraseña":
      - Captura los campos del formulario de Editar Contraseña.
      - Delega la actualización la contraseña a la función Actualizar Credenciales de la clase Sesión.
  - **Entrada**:
    - Texto que permite detectar qué funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso si no se puede cambiar la contraseña.
    - Booleano verdadero si ha modificado la contraseña.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los datos del perfil se han actualizado.

Admisión

Finalidad:

- Generar el contenido relacionado con la gestión de las solicitudes de registro.
- Gestionar la inserción de nuevos usuarios.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Sección.
- **Permisos Necesarios**: Administrador o Creador.

Funciones:

- **Admisión**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos de su clase predecesora.
  - **Entrada**:
    - **Sección:** Texto que representa el nombre de la sección “Admisiones”.
    - **Subsección:** Texto que representa el nombre de la subsección “Admisión”.
    - **Título**: Texto con el que se representa la sección.
    - **Permisos**: Número entero que representa el nivel de permisos de la sección.
    - **Estricto**: Texto que representa la relación que debe existir entre los permisos del usuario y de la sección para que el usuario pueda acceder.
      - “=”: Los permisos del usuario y de la sección deben coincidir.
      - “>”: Los permisos del usuario deben ser mayores que los de la sección.
      - “<”: Los permisos del usuario deben ser menores que los de la sección.
      - “>=”: Los permisos del usuario deben ser mayores o iguales que los de la sección.
      - “<=”: Los permisos del usuario deben ser menores o iguales que los de la sección.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción:** Genera el menú que permite la gestión de las solicitudes de registro.
  - **Funcionamiento:**
    - Delega la obtención de los datos de admisiones a la función get Admisiones de la clase Sesión.
    - Utiliza los resultados obtenidos de la Base de Datos para generar una tabla en la que:
      - Se muestran los usuarios pendientes de aceptación.
      - Permite acceder a una funcionalidad para aceptar o rechazar las solicitudes.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el menú de gestión de usuarios registrados.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Formulario Añadir Usuario:**
  - **Descripción:** Genera la funcionalidad que permite aceptar una admisión.
  - **Funcionamiento:** Genera un formulario con:
    - Un campo oculto para identificar el botón que se ha pulsado.
    - Una lista desplegable que permite seleccionar los permisos que se le asignarán al nuevo usuario.
    - Un campo oculto para identificar al usuario que se está aceptando.
    - Un botón para aceptar la solicitud.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario para aceptar la solicitud de admisión.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Formulario Eliminar Admisión:**
  - **Descripción:** Genera la funcionalidad que permite eliminar una admisión.
  - **Funcionamiento:** Genera un formulario con:
    - Un campo oculto para identificar el botón que se ha pulsado.
    - Un campo oculto para identificar la admisión que se está rechazando.
    - Un botón para rechazar la solicitud.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario para rechazar la solicitud de admisión.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Formulario Editar Filtros:**
  - **Descripción:** Genera la funcionalidad que permite abrir el menú de edición de filtros.
  - **Funcionamiento:** Genera un formulario con:
    - Un campo oculto para identificar el botón que se ha pulsado.
    - Un botón para abrir el menú de edición de filtros.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario Abrir el menú de Edición de Filtros.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción:** Da comienzo al proceso de Aceptar o Rechazar una solicitud de admisión.
  - **Funcionamiento:**
    - Si redirigir es “Añadir Usuario”:
      - Captura los campos del formulario de Añadir Usuario.
      - Delega el proceso de añadir al usuario a la función Añadir Usuario de la clase Sesión.
      - Delega el proceso de eliminar la admisión a la función Eliminar Admisión de la clase Sesión.
    - Si redirigir es “Eliminar Admisión”:
      - Captura los campos del formulario de Eliminar Admisión.
      - Delega el proceso de eliminar la admisión a la función Eliminar Admisión de la clase Sesión.
  - **Entrada**: Texto que permite detectar qué funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso si se no se acepta o rechaza alguna admisión.
    - Booleano verdadero si se acepta o rechaza alguna admisión.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los datos del perfil se han actualizado.

Administración

Finalidad:

- Generar el contenido relacionado con la gestión de los usuarios registrados.
- Gestionar el nivel de permisos de los usuarios registrados.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Sección.
- **Permisos Necesarios**: Administrador o Creador.

Funciones:

- **Administración**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos de su clase predecesora.
  - **Entrada**:
    - **Sección:** Texto que representa el nombre de la sección “Admisiones”.
    - **Subsección:** Texto que representa el nombre de la subsección “Administración”.
    - **Título**: Texto con el que se representa la sección.
    - **Permisos**: Número entero que representa el nivel de permisos de la sección.
    - **Estricto**: Texto que representa la relación que debe existir entre los permisos del usuario y de la sección para que el usuario pueda acceder.
      - “=”: Los permisos del usuario y de la sección deben coincidir.
      - “>”: Los permisos del usuario deben ser mayores que los de la sección.
      - “<”: Los permisos del usuario deben ser menores que los de la sección.
      - “>=”: Los permisos del usuario deben ser mayores o iguales que los de la sección.
      - “<=”: Los permisos del usuario deben ser menores o iguales que los de la sección.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción:** Genera el menú que permite la gestión de usuarios registrados.
  - **Funcionamiento:**
    - Delega la obtención de los datos de usuarios a la función get Usuarios de la clase Sesión.
    - Utiliza los resultados obtenidos de la Base de Datos para generar una tabla en la que:
      - Se muestran los usuarios registrados y sus permisos actuales.
      - Permite acceder a una funcionalidad para modificar sus permisos.
      - Permite acceder a una funcionalidad para eliminarlos.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el menú de gestión de usuarios registrados.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Formulario Actualizar Permisos:**
  - **Descripción:** Genera la funcionalidad que permite modificar los permisos de un usuario registrado.
  - **Funcionamiento:** Genera un formulario con:
    - Un campo oculto para identificar el botón que se ha pulsado.
    - Una lista desplegable que permite seleccionar los nuevos permisos a los que se asignarán usuario.
    - Un campo oculto para identificar al usuario que se está modificando.
    - Un botón para confirmar la solicitud.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario modificar los permisos.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Formulario Eliminar Usuario:**
  - **Descripción:** Genera la funcionalidad que permite eliminar un usuario registrado.
  - **Funcionamiento:** Genera un formulario con:
    - Un campo oculto para identificar el botón que se ha pulsado.
    - Un campo oculto para identificar el usuario que se va a eliminar.
    - Un botón para confirmar la eliminación del usuario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario para eliminar al usuario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Formulario Editar Filtros:**
  - **Descripción:** Genera la funcionalidad que permite abrir el menú de Edición de filtros.
  - **Funcionamiento:** Genera un formulario con:
    - Un campo oculto para identificar el botón que se ha pulsado.
    - Un botón para abrir el menú de Edición de filtros.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario abrir el menú de Edición de Filtros
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción:** Da comienzo al proceso de Aceptar o Rechazar una solicitud de admisión.
  - **Funcionamiento:**
    - Captura el campo oculto “Redirigir” para identificar de qué formulario se trata.
    - Si redirigir es “Actualizar Permisos”:
      - Captura los campos del formulario de Actualizar Permisos.
      - Delega el proceso de modificar los permisos a la función Actualizar Permisos de la clase Sesión.
    - Si redirigir es “Eliminar Usuario”:
      - Captura los campos del formulario de Eliminar Usuario.
      - Delega el proceso de eliminar al usuario a la función Eliminar Usuario de la clase Sesión.
  - **Entrada**: Texto que permite detectar qué funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso si no se cambian permisos o se eliminan usuarios.
    - Booleano verdadero si se cambian permisos o se eliminan usuarios.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los datos del perfil se han actualizado.

Accesos

Finalidad:

- Generar el contenido relacionado con los intentos de acceso.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Sección.
- **Permisos Necesarios**: Creador.

Atributos:

- **Usuario**: Variable de tipo texto que representa el último valor seleccionado para el filtro de usuarios del menú de intentos de acceso. Su valor por defecto será una cadena vacía que indica que no se aplica ningún filtro.

Funciones:

- **Accesos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos de su clase predecesora.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Contenido:**
  - **Descripción:** Genera el contenido de la página Accesos.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el contenido de la página Accesos.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Generar Menú Accesos:**
  - **Descripción:** Genera el menú que permite visualización de los intentos de acceso.
  - **Funcionamiento:**
    - Delega la obtención de los datos de intentos de acceso a la función get Accesos de la librería SQL.
    - Utiliza los resultados obtenidos de la Base de Datos para generar una tabla en la que:
      - Se muestran los intentos de acceso de todos los usuarios ordenados cronológicamente.
      - Permite acceder a una funcionalidad que permite filtrar los intentos de acceso de un usuario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el menú de los intentos de acceso.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Filtros:**
  - **Descripción:** Actualiza los filtros del menú intentos de acceso.
  - **Funcionamiento:**
    - Captura los campos del formulario de Filtrar Accesos generado en el Menú de intentos de acceso.
    - Actualiza el valor del atributo Usuario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: El atributo usuario ha sido actualizado.

Gestión

Finalidad:

- Clasificar los distintos tipos de formularios que permiten a los usuarios introducir sus datos económicos.
- Gestionar la navegación a través de los diferentes formularios de la aplicación.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Sección.
- **Permisos Necesarios**: Usuario, Administrador o Creador.

Atributos:

- **Formularios**: Diccionario que contiene cada uno de los tipos de formulario que contempla la aplicación.
- **Formulario Actual**: Texto que representa el tipo de formulario que se está visualizando actualmente.
  - Su valor por defecto será “Activos” para acceder a la gestión de los activos.

Funciones:

- **Gestión**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos con sus valores por defecto.
    - Inicializa los valores de los atributos de su clase predecesora.
    - Llama a la constructora de cada uno de los tipos de formulario que contempla la aplicación.
    - Añade todos los tipos de formulario al diccionario.
  - **Entrada**:
    - **Sección:** Texto que representa el nombre de la sección “Transacciones”.
    - **Subsección:** Texto que representa el nombre de la subsección “Transacciones”.
    - **Título**: Texto con el que se representa la sección.
    - **Permisos**: Número entero que representa el nivel de permisos de la sección.
    - **Estricto**: Texto que representa la relación que debe existir entre los permisos del usuario y de la sección para que el usuario pueda acceder.
      - “=”: Los permisos del usuario y de la sección deben coincidir.
      - “>”: Los permisos del usuario deben ser mayores que los de la sección.
      - “<”: Los permisos del usuario deben ser menores que los de la sección.
      - “>=”: Los permisos del usuario deben ser mayores o iguales que los de la sección.
      - “<=”: Los permisos del usuario deben ser menores o iguales que los de la sección.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**:
    - Los atributos tienen los valores por defecto.
    - Todos los tipos de formulario han sido inicializados y añadidos al diccionario.

- **Generar Índice**:
  - **Descripción**: Genera dinámicamente el índice de la sección de transacciones.
  - **Características**: Esta función debe ser reescrita por sus clases herederas.
  - **Funcionamiento:** Genera dinámicamente el índice de la sección.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el índice de la sección de gestión.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Generar Contenido**:
  - **Descripción**:
    - Genera dinámicamente el contenido de la sección.
    - Está pendiente de los inputs introducidos por el usuario permitiendo:
      - Capturar los campos de los formularios mediante la función Request de la librería Flask.
      - Interpretar cuándo es necesario efectuar una redirección y notificarlo a la clase Secciones devolviendo una cadena vacía como contenido de la página.
  - **Funcionamiento:**
    - Delega la actualización del formulario Actual y el subformulario Actual a la función Capturar Redirección de la misma clase.
    - Delega la generación del formulario a la función Generar Formulario del formulario actual.
    - Delega la preparación de la página para recibir los campos del formulario a la función Capturar campos del formulario actual.
  - **Entrada**: Ninguna.
  - **Salida**:
    - Texto con formato HTML que representa el contenido de la sección.
    - Una cadena vacía en caso de que sea necesario efectuar una redirección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción:** Delega su funcionamiento a la función capturar campos del formulario actual.
  - **Funcionamiento:**
    - Delega su funcionamiento a la función capturar campos del formulario actual.
  - **Entrada**:
    - Texto que permite detectar qué funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso si no se cambian permisos o se eliminan usuarios.
    - Booleano verdadero si se cambian permisos o se eliminan usuarios.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los datos del perfil se han actualizado.

- **Capturar Redirección**:
  - **Descripción**: Permite generar el formulario de la sección.
  - **Funcionamiento:**
    - Obtiene los parámetros generados durante la redirección desde la clase secciones.
    - Comprueba si los parámetros se corresponden con el identificador de alguno de los formularios.
    - En caso afirmativo:
      - Modifica el valor del atributo subformulario actual con el valor de los parámetros.
      - En función del subformulario actual se calcula el formulario actual.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Actualiza los atributos Formulario Actual y subformulario Actual.

Análisis

Finalidad:

- Generar el contenido las ventanas de gráficos y resúmenes.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario, Administrador o Creador.

Atributos:

- **Formulario Actual**: Variable de tipo texto que almacena formulario que se representa actualmente.
- **Subformulario Actual**: Variable de tipo texto que almacena el subformulario que se representa actualmente.
- **Formularios**: Variable de tipo diccionario que almacena todos los formularios disponibles.

Funciones:

- **Análisis**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Llama a la constructora de cada una de las gráficas que conforman la aplicación.
    - Añade todos los formularios al diccionario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Todas las gráficas de la aplicación han sido inicializados y añadidos al diccionario.
- **Generar Índice**:
  - **Descripción**: Genera dinámicamente el índice de las gráficas.
  - **Funcionamiento:** Genera dinámicamente el índice de las gráficas.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el índice de la sección
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Generar Contenido:**
  - **Descripción:** Delega la generación del contenido a la clase Gráficas.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el contenido de la página.
  - **Precondiciones**: Ninguna
  - **Postcondiciones**: Ninguna
- **Capturar Campos:**
  - **Descripción:** Permite capturar los campos que se envían en las gráficas de las diferentes secciones.
  - **Funcionamiento:** Delega su ejecución a la clase Formulario.
  - **Entrada**: Texto que permite detectar que funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso por defecto.
    - Booleano verdadero si es necesario refrescar la página para efectuar una redirección.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Importaciones

Finalidad:

- Generar el contenido del formulario que permite importar los datos de un usuario mediante un fichero csv.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Importaciones**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos de su clase predecesora.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción:** Genera el menú que permite la importación de datos mediante ficheros csv.
  - **Funcionamiento:** Genera el Formulario que permite la importación de datos mediante ficheros csv.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el menú de importación de datos mediante ficheros csv.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción:** Da comienzo al proceso de Insertar los datos de los ficheros importados.
  - **Funcionamiento:**
    - Captura el campo oculto “Redirigir” para identificar de qué formulario se trata.
    - Si redirigir es “Importar CSV”:
      - Captura los campos del formulario de Importar CSV.
      - Delega el proceso de interpretar el fichero a la función Importar Datos CSV de la librería Ficheros.
  - **Entrada**: Ninguna.
  - **Salida**:
    - Booleano falso si no se ha introducido algún fichero.
    - Booleano verdadero si se ha introducido algún fichero.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Formulario

Finalidad:

- Generar de forma universal el contenido de cualquier tipo de formulario.

Atributos:

- **Formulario**: Constante de tipo texto que representa los distintos tipos de formularios mediante un identificador único que permite diferenciarlos.
- **Sub Formulario**: Constante de tipo texto que representa los distintos tipos de subformularios mediante un identificador único que permite diferenciarlos.
- **Título**: Constante de tipo texto que con el que se almacena el texto que se mostrará al usuario cuando abra el formulario.
- **Acción**: Variable de tipo texto que permite identificar la acción que ha seleccionado el usuario y permitirá redirigir a la ventana correspondiente a dicha acción.
  - **Tabla**: Valor por defecto. Redirige a la ventana en la que se muestra la información de aquellos campos que ya han sido registrados mediante este formulario.
  - **Añadir**: Redirige a la ventana en la que se muestra el formulario que permite añadir nuevos datos.
  - **Editar**: Redirige a la ventana en la que se muestra el formulario que permite modificar los datos del formulario desde el que se haya seleccionado dicha acción.
  - **Eliminar**: Redirige a la ventana en la que se muestra el formulario que permite eliminar los datos del formulario desde el que se haya seleccionado dicha acción.
- **Parámetros:** Variable de tipo texto que almacena aquella información que debe mantenerse entre los distintos formularios cuando se navega entre ellos.
- **Filtro**: Instancia de la clase formulario que permite generar un formulario de tipo filtro mediante el cual se pueden configurar ciertas características del formulario al que está asociado.

Funciones:

- **Formulario**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos con sus valores por defecto.
  - **Entrada**:
    - Texto que representa el identificador único del formulario que se quiere construir.
    - Texto que representa el identificador único del subformulario que se quiere construir.
    - Texto que representa el título que se mostrará al usuario cuando se abra el formulario
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen los valores por defecto.
- **Set Acción:**
  - **Descripción**: Modifica la variable acción del formulario.
  - **Entrada**: Texto que representa la nueva acción que se debe ejecutar.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Actualiza la variable acción.

- **Generar Formulario**:
  - **Descripción**: Genera el formulario para insertar datos en función del tipo de formulario seleccionado
  - **Funcionamiento:**
    - Comprueba el atributo acción
    - Delega la generación de la función a la función que está asociada a dicha acción
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario seleccionado.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Redirigir**:
  - **Descripción**: Implementa con formato HTML el formulario de un botón con la capacidad de efectuar una redirección entre formularios.
  - **Finalidad:** Se trata de una herramienta para simplificar el código de las clases herederas.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario del botón.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Formulario:**
  - **Descripción**:
    - Genera el formulario indicado por las constantes formulario y subformulario y la variable acción.
  - **Funcionamiento:** Se delega su ejecución a las clases herederas.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción:** Prepara a la página para recibir los campos del formulario.
  - **Funcionamiento:**
    - Si Redirigir es: “Añadir” modifica el valor de la acción a “Añadir” y recarga el formulario.
    - Si Redirigir es: “Editar” modifica el valor de la acción a “Editar” y recarga el formulario.
    - Si Redirigir es: “Eliminar” modifica el valor de la acción a “Eliminar” y recarga el formulario.
    - Si Redirigir es: “Tabla” modifica el valor de la acción a “Tabla” y recarga el formulario.
    - Delega a las clases herederas la posibilidad de añadir más redirecciones.
  - **Entrada**: Texto que permite detectar qué funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso no es necesario recargar el formulario.
    - Booleano verdadero si se debe recargar el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Activos

Finalidad:

- Generar el contenido del formulario de activos para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Activos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Cuentas

Finalidad:

- Generar el contenido del formulario de cuentas para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Cuentas**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.

- **Postcondiciones**: Ninguna

Estrategias

Finalidad:

- Generar el contenido del formulario de estrategias para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Activos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna
  - **Postcondiciones**: Ninguna

Categorías

Finalidad:

- Generar el contenido del formulario de categorías para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Cuentas**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.

- **Precondiciones**: Ninguna
- **Postcondiciones**: Ninguna

Productos

Finalidad:

- Generar el contenido del formulario de productos para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Activos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna

Categorización

Finalidad:

- Generar el contenido del formulario de categorización para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Cuentas**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.

- **Salida**: Texto con formato HTML que representa el formulario.
- **Precondiciones**: Ninguna.
- **Postcondiciones**: Ninguna.

Allocation

Finalidad:

- Generar el contenido del formulario de Allocation para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Activos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Sub-Allocation

Finalidad:

- Generar el contenido del formulario de Sub-Allocation para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Cuentas**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.

- **Entrada**: Ninguna.
- **Salida**: Texto con formato HTML que representa el formulario.
- **Precondiciones**: Ninguna.
- **Postcondiciones**: Ninguna.

Clasificación

Finalidad:

- Generar el contenido del formulario de Clasificación para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Activos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Sub-Clasificación

Finalidad:

- Generar el contenido del formulario de Sub-Clasificación para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Cuentas**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**

- **Descripción**: Genera el formulario indicado por la variable acción para este formulario
- **Entrada**: Ninguna
- **Salida**: Texto con formato HTML que representa el formulario
- **Precondiciones**: Ninguna
- **Postcondiciones**: Ninguna

Distribución

Finalidad:

- Generar el contenido del formulario de distribución para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario

Funciones:

- **Activos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Transacciones

Finalidad:

- Generar el contenido del formulario de transacciones para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Cuentas**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.

- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Aportaciones

Finalidad:

- Generar el contenido del formulario de aportaciones para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Activos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Comisiones

Finalidad:

- Generar el contenido del formulario de comisiones para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Cuentas**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.

- **Postcondiciones**: Los atributos tienen sus valores por defecto.

- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Traspasos

Finalidad:

- Generar el contenido del formulario de traspasos para sus distintas acciones posibles.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Formulario.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Activos**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:**
    - Inicializa los valores de los atributos de su clase predecesora.
    - Configura el filtro para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen sus valores por defecto.
- **Generar Formulario:**
  - **Descripción**: Genera el formulario indicado por la variable acción para este formulario.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Gráficas

Finalidad:

- Generar de forma universal el contenido de cualquier gráfica y tabla que resuma la información del usuario

Atributos:

- **Formulario**: Constante de tipo texto que representa los distintos tipos de gráficas mediante un identificador único que permite diferenciarlos.
- **Sub Formulario**: Constante de tipo texto que representa los distintos tipos de gráficas mediante un identificador único que permite diferenciarlos.
- **Título**: Constante de tipo texto con el que se almacena el texto que se mostrará al usuario cuando abra la ventana de resumen seleccionada.
- **Filtro**: Instancia de la clase formulario que permite generar un formulario de tipo filtro mediante el cual se pueden configurar ciertas características del formulario al que está asociado.

Funciones:

- **Gráficas**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos con sus valores por defecto.
  - **Entrada**:
    - Texto que representa el identificador único del formulario que se quiere construir.
    - Texto que representa el identificador único del subformulario que se quiere construir.
    - Texto que representa el título que se mostrará al usuario cuando se abra el formulario.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen los valores por defecto.
- **Generar Formulario**:
  - **Descripción**: Genera el formulario donde se representan los datos especificados.
  - **Funcionamiento:** Delega su ejecución a las clases herederas.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario seleccionado.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Balance

Finalidad:

- Generar las tablas y gráficas del formulario de Balance.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Gráficas
- **Permisos Necesarios**: Usuario.

Funciones:

- **Generar Formulario**:
  - **Descripción**: Genera el formulario donde se representan los datos especificados.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario seleccionado.

- **Precondiciones**: Ninguna.
- **Postcondiciones**: Ninguna.

Gastos

Finalidad:

- Generar las tablas y gráficas del formulario de Gastos.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Gráficas.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Generar Formulario**:
  - **Descripción**: Genera el formulario donde se representan los datos especificados.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario seleccionado.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Patrimonio

Finalidad:

- Generar las tablas y gráficas del formulario de Patrimonio.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Gráficas.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Generar Formulario**:
  - **Descripción**: Genera el formulario donde se representan los datos especificados.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario seleccionado.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Evolución

Finalidad:

- Generar las tablas y gráficas del formulario de Evolución.

Características:

- **Herencia**: Hereda todos los atributos y funciones de la clase Gráficas.
- **Permisos Necesarios**: Usuario.

Funciones:

- **Generar Formulario**:
  - **Descripción**: Genera el formulario donde se representan los datos especificados.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario seleccionado.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Filtro

Finalidad:

- Acotar los datos que se muestran en cualquier tabla en función de un conjunto de parámetros que puedan ser editados por el usuario.

Atributos:

- **Sección**: Constante de tipo texto que representa los distintos tipos de Filtro mediante un identificador único que permite diferenciarlos.
- **Sub-Sección**: Constante de tipo texto que representa los distintos tipos de Filtro mediante un identificador único que permite diferenciarlos.
- **Orden**: Constante de tipo texto que indica el campo en función del cual se ordenan los elementos del filtro.
- **Orden Tipo**: Constante de tipo texto que indica el criterio de ordenación de los elementos del filtro.
- **Campos**: Diccionario que contiene los campos a partir de los cuales se pueden filtrar.

Funciones:

- **Filtro**:
  - **Descripción**: Constructora de la clase.
  - **Funcionamiento:** Inicializa los valores de los atributos con sus valores por defecto.
  - **Entrada**: Texto que representa el nombre de la sección y la subsección a la que representa el filtro.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Los atributos tienen los valores por defecto.
- **Añadir Campo**:
  - **Descripción**: Permite añadir nuevos elementos al filtro.
  - **Funcionamiento:** Comprueba que dicho elemento no exista previamente
  - **Entrada**: Textos que representan el identificador, la etiqueta y el título del nuevo elemento del filtro.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna**.**
- **Generar Formulario**:
  - **Descripción**: Genera el formulario donde se representan los datos para cada campo del filtro.
  - **Entrada**: Ninguna.
  - **Salida**: Texto con formato HTML que representa el formulario del Filtro.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Capturar Campos:**
  - **Descripción**: Recarga la página aplicando los cambios establecidos en el filtro.
  - **Funcionamiento:**
    - Si redirigir es "Filtrar": Aplica los cambios del filtro seleccionado.
    - Si redirigir es "Restablecer": Elimina todos los filtros y aplica los cambios.
  - **Entrada**:
    - Texto que permite detectar que funcionalidad se ha activado.
  - **Salida**:
    - Booleano falso si no se debe recargar la pagina.
    - Booleano verdadero si se debe recargar la pagina.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Librería SQL

Finalidad:

- Actuará como intermediario entre la base de datos y la aplicación suministrando los datos necesarios.

Funciones:

- **Conectar**:
  - **Descripción**: Establece la conexión con el servidor.
  - **Entrada**:
    - **Usuario**: Texto que contiene el nombre de usuario.
    - **Contraseña**: Texto que contiene la contraseña del usuario.
    - **Base de Datos**: Texto que contiene el nombre de la base de datos con la que se establece conexión.
  - **Salida**:
    - **Conexión**: Objeto SQL que representa la conexión activa.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Consultar**:
  - **Descripción**: Ejecuta una sentencia SQL del tipo SELECT.
  - **Entrada**:
    - **Conexión**: Objeto SQL que representa la conexión activa.
    - **Consulta**: Texto que contiene las especificaciones de la consulta.
  - **Salida**:
    - **Cursor**: Objeto SQL que representa los resultados de una consulta.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Actualizar**:
  - **Descripción**: Ejecuta una sentencia SQL del tipo UPDATE.
  - **Entrada**:
    - **Conexión**: Objeto SQL que representa la conexión activa.
    - **Consulta**: Texto que contiene las especificaciones de la consulta.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Se actualizan filas correspondientes de la Base de Datos.
- **Existe**:
  - **Descripción**: Comprueba si existe un requerimiento especifico en una tabla.
  - **Procedimiento:**
    - Ejecuta una consulta mediante la función Consultar de la librería SQL.
    - Los datos consultados no existen cuando el cursor está vacío.
  - **Entrada**:
    - **Conexión**: Objeto SQL que representa la conexión activa.
    - **Consulta**: Texto que contiene las especificaciones de la consulta.
  - **Salida**:
    - Booleano Verdadero si los datos consultados existen.
    - Booleano Falso si los datos consultados no existen.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Copiar Tabla**:
  - **Descripción**: Dado el identificador de una tabla crea una copia exacta de la tabla de referencia que se utiliza como modelo.
  - **Entrada**:
    - **Cursor**: Objeto SQL que representa los resultados de una consulta.
    - **Usuario**: Texto que contiene el nombre de usuario.
    - **Tabla**: Texto que contiene el identificador de una tabla.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Se genera la tabla indicada para el usuario indicado.
- **Crear Base de Datos**:
  - **Descripción**: Crea una copia exacta de la Base de Datos de referencia para un nuevo usuario.
  - **Funcionamiento**: Ejecuta la función Copiar tabla para cada tabla de la Base de Datos de referencia pasando como parámetros el Cursor de la nueva Base de datos y el nombre del usuario Objetivo.
  - **Entrada**:
    - **Usuario Administrador**: Texto que contiene el nombre de usuario que genera la base de datos
    - **Contraseña**: Texto que contiene la contraseña del usuario que genera la base de datos.
    - **Usuario Objetivo**: Texto que contiene el nombre de usuario al que se le crea la nueva base de datos.
  - **Salida**: Ninguna.
  - **Precondiciones**: El usuario administrador tiene los permisos necesarios.
  - **Postcondiciones**: Se genera nueva Base de Datos al usuario objetivo.
- **Crear Usuario**:
  - **Descripción**: Crea un nuevo usuario y le asigna sus permisos.
  - **Entrada**:
    - **Usuario**: Texto que contiene el nombre de usuario.
    - **Contraseña**: Texto que contiene la contraseña del usuario.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Se genera una nueva instancia en la tabla de Usuarios con todos los permisos asignados
- **Asignar Permisos**:
  - **Descripción**: Modifica los permisos del usuario objetivo
  - **Entrada**:
    - **Usuario Administrador**: Texto que contiene el nombre de usuario que modifica los permisos
    - **Contraseña**: Texto que contiene la contraseña del usuario que modifica los permisos
    - **Usuario Objetivo**: Texto que contiene el nombre de usuario al que se le modifican los permisos
    - **Permisos**: Texto que representa el nuevo perfil de permisos del usuario objetivo
  - **Salida**: Ninguna
  - **Precondiciones**: El usuario administrador tiene los permisos necesarios
  - **Postcondiciones**: Se asignan al usuario objetivo los permisos indicados

- **Eliminar Consulta**:
  - **Descripción**: Ejecuta una sentencia SQL del tipo DELETE.
  - **Procedimiento:**
    - Ejecuta una consulta mediante la función Consultar de la librería SQL.
    - Los datos consultados no existen cuando el cursor este vacío.
  - **Entrada**:
    - **Usuario**: Texto que contiene el nombre de usuario.
    - **Contraseña**: Texto que contiene la contraseña del usuario.
    - **Consulta**: Texto que contiene las especificaciones de la consulta.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Se eliminan las filas especificadas de la base de datos del usuario.
- **Eliminar Usuario**:
  - **Descripción**: Ejecuta una sentencia SQL del tipo DROP USER.
  - **Especificaciones:** Si el usuario eliminado tiene permisos de DROP USER puede eliminarse a sí mismo.
  - **Procedimiento:**
    - Ejecuta una consulta mediante la función Consultar de la librería SQL.
    - Los datos consultados no existen cuando el cursor este vacío.
  - **Entrada**:
    - **Usuario Administrador**: Texto que contiene el nombre de usuario que elimina al usuario objetivo.
    - **Contraseña**: Texto que contiene la contraseña del usuario que elimina al usuario objetivo.
    - **Usuario Objetivo**: Texto que contiene el nombre del usuario que va a ser eliminado.
  - **Salida**: Ninguna.
  - **Precondiciones**: El usuario administrador tiene los permisos necesarios.
  - **Postcondiciones**: Se eliminan las filas especificadas de la base de datos del usuario.

Librería HTML

Finalidad:

- Actuará como intermediario entre interfaz gráfica y la aplicación configurando objetos HTML visibles.

Funciones:

- **Título**:
  - **Descripción**: Configura el título de una sección.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Clase**: Texto que contiene el identificador que se utilizará para asignar los estilos CSS.
    - **Nivel**: Número entero que representa el nivel del título.
    - **Titulo**: Texto que representa la información que se mostrará en el título.
  - **Salida**: Texto que representa al código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Párrafo**:
  - **Descripción**: Configura un párrafo.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Clase**: Texto que contiene el identificador que se utilizará para asignar los estilos CSS.
    - **Mensaje**: Texto que representa la información que se mostrará.
  - **Salida**: Texto que representa código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Representar Numero**:
  - **Descripción**: Configura un carácter numérico para que se represente correctamente.
  - **Especificaciones:** Permite configurar características como:
    - El color cuando el número es positivo o negativo.
    - El formato de los decimales.
    - El número de decimales mostrados.
  - **Entrada**:
    - **Número**: Número que se debe mostrar en formato int float o string indiferentemente.
    - **Tiene Color**: Booleano que indica si se debe asignar un color diferente en función del signo.
  - **Salida**: Texto que representa código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Mensaje Error**:
  - **Descripción**: Configura un mensaje de error.
  - **Especificaciones:** Utilizado para depurar ciertas características del código.
  - **Entrada**: Texto que representa el mensaje de error.
  - **Salida**: Texto que representa código HTML
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna
- **Working**:
  - **Descripción**: Muestra por pantalla la imagen de Trabajo en progreso.
  - **Especificaciones:** Utilizado para indicar al usuario que dicha página no ha sido implementada.
  - **Entrada**: Ninguna.
  - **Salida**: Texto que representa código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Refrescar**:
  - **Descripción**: Genera un enlace de hipertexto que redirecciona el flujo de ejecución de la página.
  - **Procedimiento:** Se trata de un formulario con submit a la dirección indicada.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Clase**: Texto que contiene el identificador que se utilizará para asignar los estilos CSS.
    - **Tipo**: tipo de redirección que se va a ejecutar.
    - **Función**: Nombre de la función que se ejecutará tras el Submit.
    - **Parámetros:** Lista con los parámetros que necesite la función.
    - **Título**: Texto que se mostrará en el enlace.
  - **Salida**: Texto que representa el código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Índice**:
  - **Descripción**: Genera una sección con un NAV que contiene un conjunto de enlaces de hipertexto dándoles un formato característico.
  - **Procedimiento:**
    - Utiliza la función de refrescar de la librería HTML.
    - Establece el parámetro función a “Redirigir”.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Clase**: Texto que contiene el identificador que se utilizará para asignar los estilos CSS.
    - **Lista**: Lista con el nombre de todos los elementos que formarán parte del índice.
  - **Salida**: Texto que representa el código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Cabecera Tabla**:
  - **Descripción**: Configura la cabecera de una tabla.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Clase**: Texto que contiene el identificador que se utilizará para asignar los estilos CSS.
    - **Título**: Texto que representa la información que se mostrará en el título.
  - **Salida**: Texto que representa el código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Cabecera de Formulario**:
  - **Descripción**: Genera la cabecera de un formulario.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Clase**: Texto que contiene el identificador que se utilizará para asignar los estilos CSS.
    - **Acción**: Nombre de la función a la que efectuará la redirección.
  - **Salida**: Texto que representa el código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Input de Formulario**:
  - **Descripción**: Genera un input para un formulario.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Clase**: Texto que contiene el identificador que se utilizará para asignar los estilos CSS.
    - **Etiqueta**: Texto que se muestra en la etiqueta.
    - **Tipo**: Tipo de dato que admite como input.
    - **Place Holder:** Texto que se muestra por defecto cuando el usuario no ha introducido ningún input.
  - **Salida**: Texto que representa código HTML.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Gráfico Simple**:
  - **Descripción**: Genera un canvas y configura el Script para que se genere un gráfico con una única serie.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Cabecera**: Lista con las cabeceras de cada serie de datos que se debe mostrar.
    - **Datos**: Lista de listas con los datos que se deben mostrar para cada serie de datos.
    - **Bordes:** Lista con los colores para los bordes de cada serie de datos.
    - **Fondos:** Lista con los colores para los fondos de cada serie de datos.
    - **Tipo**: Tipo de representación de los datos: Tarta, Lineal, Barras, Donut, etc.
  - **Salida**: Texto que representa código HTML y JavaScript.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Gráfico Agrupado**:
  - **Descripción**: Genera un canvas y configura el Script para que se genere un gráfico compuesto por más de una serie de datos.
  - **Entrada**:
    - **ID**: Texto que contiene el identificador único de la etiqueta.
    - **Cabecera**: Lista con las cabeceras de cada serie de datos que se debe mostrar.
    - **Datos**: Lista de listas con los datos que se deben mostrar para cada serie de datos.
    - **Bordes:** Lista con los colores para los bordes de cada serie de datos.
    - **Fondos:** Lista con los colores para los fondos de cada serie de datos.
    - **Tipo**: Tipo de representación de los datos: Tarta, Lineal, Barras, Donut, etc.
  - **Salida**: Texto que representa código HTML y JavaScript.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Librería Ficheros

Finalidad:

- Permite leer archivos CSV, validar el formato de los datos e insertar la información en la base de datos.

Funciones:

- **Importar Datos CSV**:
  - **Descripción**: Permite importar datos a la Base de Datos de un usuario directamente desde un fichero CSV.
  - **Funcionamiento**
    - Carga el fichero en una ruta conocida.
    - Identifica el tipo de fichero. (Aportaciones, Comisiones, Gastos, Traspasos)
    - Lee los datos del fichero.
    - Comprueba que todos los datos cumplen las restricciones necesarias.
    - Carga los datos en la Base de Datos del usuario.
  - **Entrada**:
    - **Fichero**: Archivo con formato CSV.
    - **Nombre Fichero**: texto que representa el nombre del Fichero.
  - **Salida**: Ninguna.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**:
    - Si los datos cumplen todas las restricciones se cargan en la Base de Datos
    - Se genera un mensaje de error con todas aquellas filas que no cumplan alguna restricción indicando cuál es el requerimiento que no se cumplió.

Librería Scraper

Finalidad:

- Permite obtener información específica sobre los activos mediante la técnica de Web Scraping.

Atributos:

- Se ha implementado un diccionario que permite almacenar las cotizaciones distinguiéndolas por producto y fecha con la finalidad de saltarse el paso de consultar en la Web si dicho dato ya ha sido obtenido para la misma fecha y el mismo producto. Este atributo se restablece cada cierto periodo de tiempo.
  El coste computacional del Web Scraping es el más alto en toda la aplicación con bastante diferencia.

Funciones:

- **Cotización Actual**:
  - **Descripción**: Obtiene el dato de la cotización actual de un producto consultando en yahoo Finance
  - **Funcionamiento:** "https://es.finance.yahoo.com/quote/" + pProducto + "?p=" + pProducto
  - **Entrada**:
    - **Producto**: Texto que representa el nombre del producto.
  - **Salida**: Número decimal que representa última cotización registrada en yahoo Finance.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Serie Cotizaciones**:
  - **Descripción**: Obtiene una lista con todas las cotizaciones históricas registradas entre dos fechas en yahoo Finance.
  - **Funcionamiento:** Utiliza la función Request sobre la Api Alphavantage
  - **Entrada**:
    - **Producto**: Texto que representa el nombre del producto.
    - **Fecha Inicio:** Fecha del primer dato que se recupera.
    - **Fecha Fin:** Fecha del último dato que se recupera.
  - **Salida**: Lista de tuplas fecha, cotización registrada en yahoo Finance
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Cotización Fecha**:
  - **Descripción**: Obtiene una lista con la cotización histórica registradas para una fecha concreta en yahoo Finance.
  - **Funcionamiento:** Se delega el procedimiento a la función Serie Cotizaciones de la Librería Scraper utilizando como fecha de fin el día siguiente al de la fecha de inicio requerida.
  - **Entrada**:
    - **Producto**: Texto que representa el nombre del producto.
    - **Fecha Inicio:** Fecha del primer dato que se recupera
  - **Salida**: Tuplas fecha, cotización registrada en yahoo Finance
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Librería Utilidades

Funciones:

- **Color Aleatorio**:
  - **Descripción**: Genera aleatoriamente un color.
  - **Entrada**: Ninguna.
  - **Salida**: Tupla con tres números enteros comprendidos entre cero y doscientos cincuenta y cinco.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Degradado**:
  - **Descripción**: Genera el siguiente color en la escala RGB.
  - **Funcionamiento**: Degradar un color consiste en incrementar el valor de R G o B en tantas unidades como indica la velocidad, si se sobrepasa el límite de 255 de una propiedad se vuelve cero y se itera la siguiente.
  - **Entrada**:
    - **Color**: Color en formato RGB del que se parte.
    - **Velocidad**: Número entero que representa la cantidad que se degrada.
  - **Salida**: Color en formato RGB aplicando el degradado.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Color De Hexadecimal A RGB**:
  - **Descripción**: Convierte el formato del color de Hexadecimal a RGB.
  - **Entrada**: Color en formato Hexadecimal.
  - **Salida**: Color en formato RGB.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Color De RGB A Hexadecimal**:
  - **Descripción**: Convierte el formato del color de RGB a Hexadecimal.
  - **Entrada**: Color en formato RGB.
  - **Salida**: Color en formato Hexadecimal.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

- **Comprobar Fecha**:
  - **Descripción**: Comprueba si una fecha es válida
  - **Entrada**: Fecha en formato día mes año.
  - **Salida**:
    - Booleano Verdadero si la fecha es válida
    - Booleano Falso si la fecha no es válida
  - **Precondiciones**: Ninguna
  - **Postcondiciones**: Ninguna
- **Ajustar Fecha**:
  - **Descripción**: Comprueba el formato de una fecha y lo convierte al formato de la librería DATETIME.
  - **Entrada**: Fecha en cualquier formato. (incluso texto)
  - **Salida**: Fecha en formato DATETIME.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Depurar**:
  - **Descripción**: Dada una cadena de texto elimina todos los caracteres sospechosos.
  - **Finalidad:** Evitar SQL Inyección en los inputs de los formularios.
  - **Entrada**: Cadena de texto que se pretende validar.
  - **Salida**: Cadena de texto depurada.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.
- **Generar Sal**:
  - **Descripción**: Genera una cadena de texto tipo Hash en función del DATETIME del sistema.
  - **Entrada**: Ninguna.
  - **Salida**: Cadena de texto Hash.
  - **Precondiciones**: Ninguna.
  - **Postcondiciones**: Ninguna.

Diagramas de Flujo

Este apartado de la documentación está destinado a facilitar el entendimiento del resto del documento. De este modo se incluirán las definiciones de los términos clave para su correcto entendimiento, así como las referencias de las que se ha obtenido la información para su elaboración.

Gestión de usuarios

Registrarse

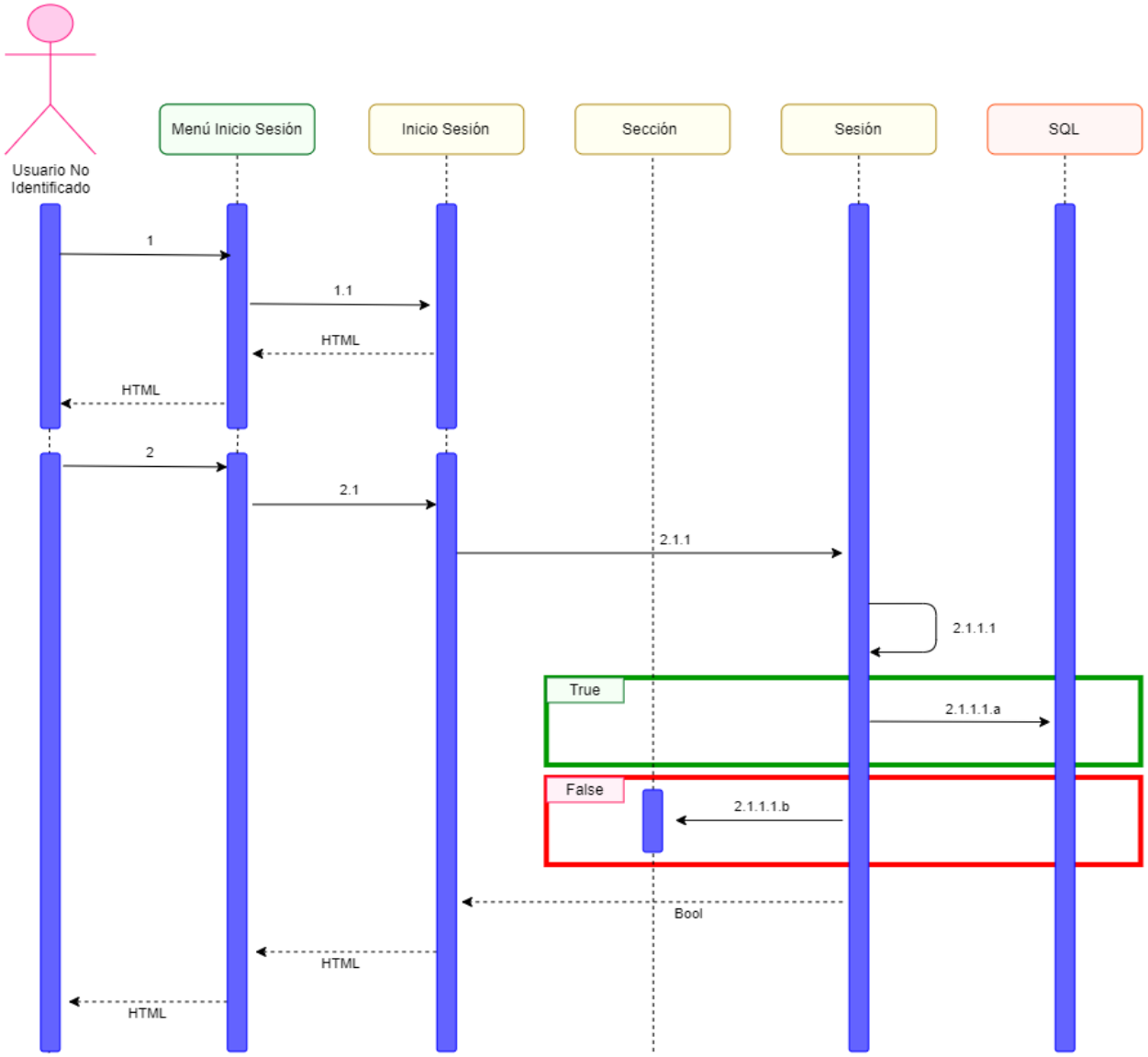

Ilustración 2 Diagrama de flujo – Registrarse

1 El Usuario No Identificado accede al formulario de registro.

1.1. Se ejecuta la función Generar Contenido de la clase Sección para la subclase Registro.

2 El Usuario No Identificado rellena el formulario de registro y pulsa el botón de aceptar.

2.1. Se ejecuta la función Generar Contenido de la clase Sección para la subclase Registro.

2.1.1. Se ejecuta la función Añadir Admisión de la clase Sesión.

2.1.1.1. Se ejecuta la función Comprobar admisión de la clase Sesión.

2.1.1.1.a En caso verdadero ejecuta la función Añadir Admisión de la librería SQL.

2.1.1.1.b En caso negativo se ejecuta la función Reportar Error de la clase Sesión.

Identificarse

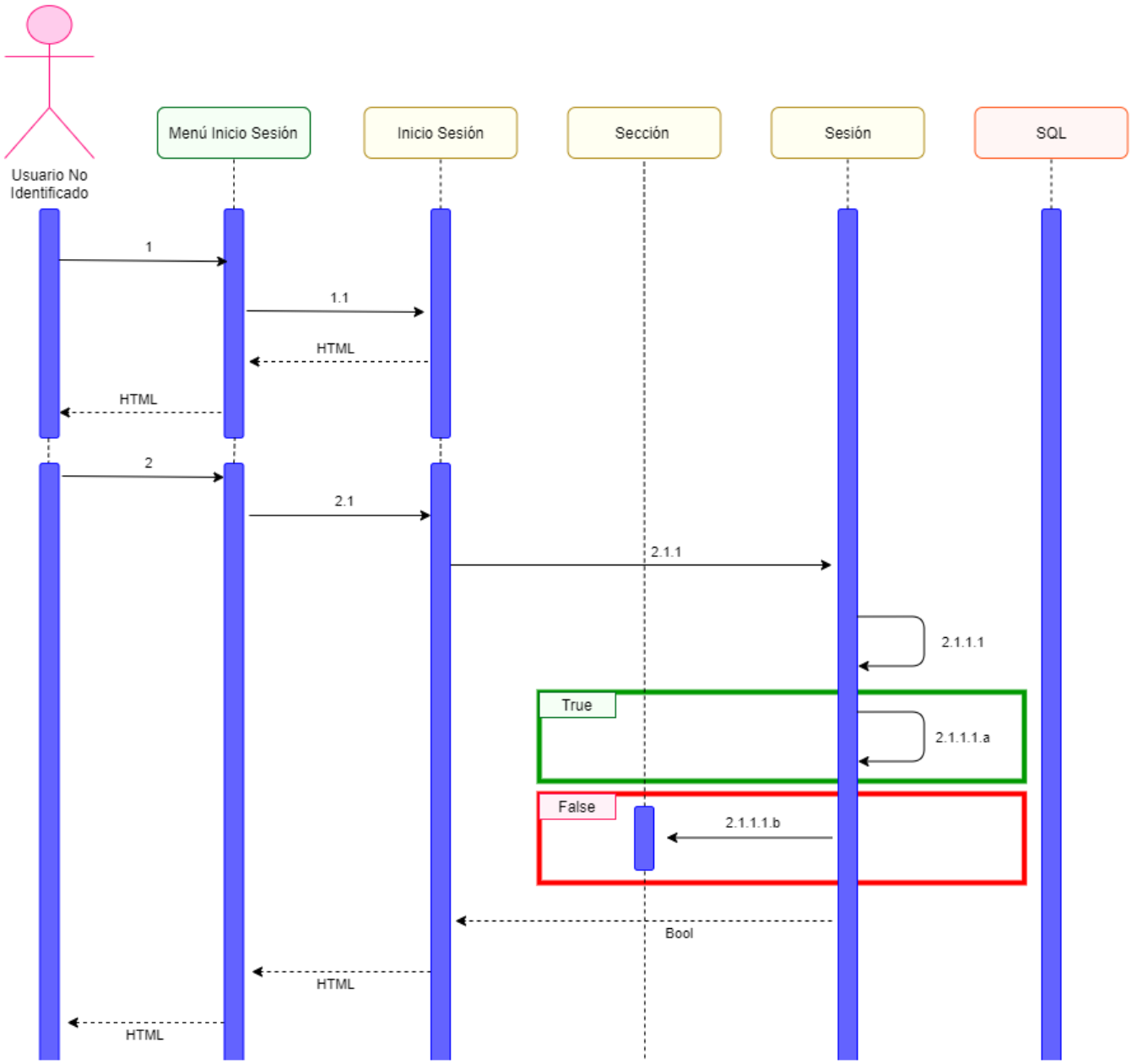

Ilustración 3 Diagrama de flujo – Identificarse

1 El Usuario No Identificado accede al formulario de registro.

1.1. Se ejecuta la función Generar Contenido de la clase Sección para la subclase Inicio Sesión.

2 El Usuario No Identificado rellena el formulario de registro y pulsa el botón de aceptar.

2.1. Se ejecuta la función Generar Contenido de la clase Sección para la subclase Inicio Sesión.

2.1.1. Se ejecuta la función Iniciar Sesión de la clase Sesión.

2.1.1.1. Se ejecuta la función Comprobar Credenciales de Sesión de la clase Sesión.

2.1.1.1.a En caso verdadero se actualizan los atributos Usuario, Permisos y Contraseña.

2.1.1.1.b En caso negativo se ejecuta la función Reportar Error de la clase Sección.

Funciones de apoyo

Generar Contenido

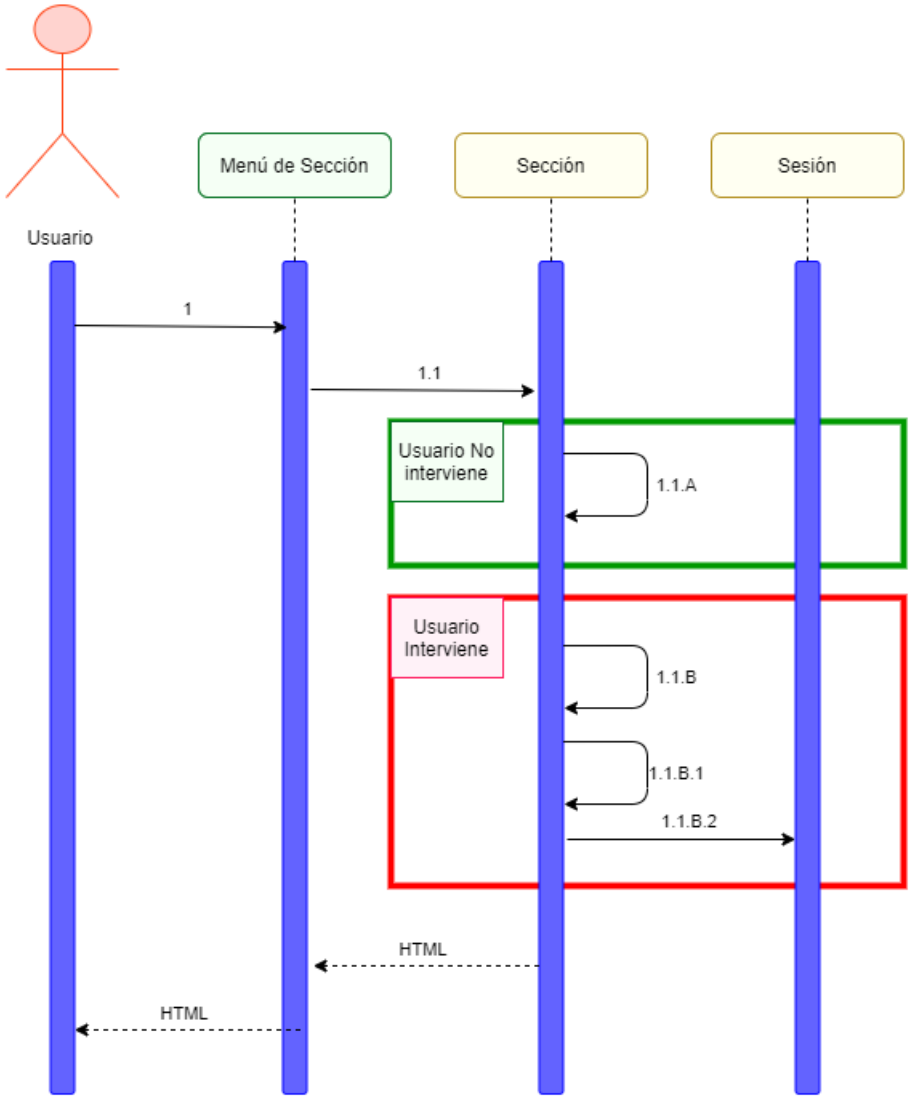

Ilustración 4 Diagrama de flujo – Generar Contenido

3 El Usuario No Identificado accede al formulario deseado.

3.1. Se ejecuta la función Generar Contenido de la clase sección.

1.1.A Si el usuario no ha completado el formulario de la sección Seleccionada.

1.1.A.1 Se ejecuta la función Generar Formulario de la subclase sección Seleccionada.

1.1.B Si el usuario ha completado el formulario de la sección Seleccionada.

1.1.B.1 Se ejecuta la función Capturar Campos de la subclase sección Seleccionada.

1.1.B.1.1 Se obtienen los datos de los campos del formulario.

1.1.B.1.2 Se ejecuta la función de la clase Sesión Asociada a la sección.