Texto

Descripción generada automáticamente

Errores en pruebas 25/11:

* Cuando haces 2 veces click en la misma cuenta en el menú de cliente tira un error 500
* A veces cuando haces algún préstamo se bugean los datos y aparecen todas las cuentas con el numero 5200.
* En el menú de aprobación de prestamos cuando entro con mi usuario que tiene 2 cuentas, pido un préstamo con cada uno, y cuando quiero aprobar o rechazar algún préstamo de la primer cuenta, me tira error.
* A veces hay que hacer muchos clicks para que te tome las elecciones(desconozco el porque)
* Al administrador se le generan cuentas( el admin no debe tener cuentas)y además puede pedir cuentas(lo cual tampoco debería pasar).
* Al transferir entre las cuentas de la misma persona no se acutaliza los montos disponibles para las transferencias
* El admin no debería tener prestamos cargados

1) Se debe participar en el foro del TP integrador. Los alumnos que no participen en el foro serán considerados que no participaron de la tarea y por ende desaprobaran la misma. También se evaluará el contenido que aporto cada alumno y a partir de esto se definirá la nota que obtendrá cada uno. En cada aporte al foro deberán aclarar que modificación están realizando.

2) Debe haber un acceso al sistema (LOGIN).

3) En cada pantalla se podrá observar el usuario que accedió al sistema. (USUARIO LOGUEADO).

4) Los listados deben estar paginados. **Verificar**

* **Falta Paginado en el listado de las cuotas de los prestamos**
* **Falta Paginado en el listado de los movimientos**
* **A revisar en alta prestamos y mejorar botón de aprobar y rechazar**
* **No sale la leyenda de préstamo rechazado ni tampoco les cambia el estado le cambia el estado HAY QUE CAMBIAR LA LEYENDA**
* **Arreglar el diseño del detalle de los prestamos**
* **Cuando pagamos la cuota de un préstamo, no se actualiza el monto de la caja de ahorro, pero si en movimientos**
* **Mejorar la parte visual cuando se genera un nuevo cliente/usuario**
* **Arreglar para cuando se crea un nuevo cliente, el cbu debe tener 22/23 digitos**
* **No funcionan las transferencias DONE**

5) Utilizar búsquedas y filtros en los listados. Las búsquedas hacerlas generales utilizando el comparador LIKE %contenido% para traer todos los textos que cumplen con la condición, no solo el texto que sea idéntico. También trabajar con filtros, los filtros no son iguales a las búsquedas, los filtros permiten seleccionar opciones como, por ejemplo, en el listado de movimientos que te permita seleccionar los movimientos cuyo importe total este en un rango entre 1000 a 2000, o =100, etc

6) Mensajes aclaratorios “Se agrego correctamente en la base de datos”

7) Trabajar con programación en tres capas, utilizar interfaces y manejar las excepciones.

8) Es necesario que el código se encuentre divididos en paquetes. Las excepciones estarán en un paquete separado y deberán crear al menos dos excepciones propias.

9) Acerca de las entidades, las clases deben estar correctamente diagramadas. Ejemplo: la clase movimiento debe tener un objeto de tipo movimiento en su interior, no un int ID\_movimiento ni tampoco string descripción\_movimiento**.**

10) No tiene que haber desde los JSP, llamados al DAO ni al NEGOCIO desde los JSP se tiene que llamar al SERVLET y el SERVLET llama al negocio. Por último, el negocio llama al DAO. El SERVLET es el “controlador” de nuestras aplicaciones web.

11) En los descolgables deben aparecer también las descripciones y no solo sus identificadores.

12) Manejar bajas lógicas y no físicas.

13) Trabajar con VALORES CON DECIMALES. Revisar todos los montón en decimales, solo deben tener 2 decimales

14) No guardar campos vacíos en la base de datos. Para eso usar validaciones. Ejemplo: validación de números, de textos, correo electrónico, etc. validar que si se hace clic sobre el botón y no hay nada cargado entonces no se guarde ese registro en la base de datos.

15) Al modificar, se podrán modificar todos los campos de la tabla, obviamente no se modificará el campo clave. Se deben cargar automáticamente todos los campos que se le va a permitir modificar al usuario, luego este cambia solo el dato que necesita. Al modificar un solo campo de la base de datos, no se deben borrar los demás campos, solo se deberá actualizar el campo que el usuario esté cambiando.

16) Utilizar mensajes de confirmación. Ejemplo: ¿Está seguro que desea eliminar el registro?

* **Falta en dar de baja una cuenta DONE**

17) Al dar de alta un usuario pedir la clave dos veces y usar asteriscos para no mostrarla.

18) Tener cuidado con los repetidos en la base de datos, por ejemplo: no podría haber dos personas con el mismo DNI en la base de datos.

19) Luego de realizar un cambio sobre un registro por ejemplo al agregar, limpiar todos los controles. Es decir, se deberá borrar la información que acaba de ingresar el usuario.

20) Incluir reportes/estadísticas con parámetros. NO ES LO MISMO UN LISTADO QUE UN REPORTE, UN REPORTE ES INFORMACIÓN QUE SE OBTIENE A PARTIR DE LOS DATOS QUE SE ENCUENTRAN EN LA BASE DE DATOS. Ejemplo: Seleccionar fecha de inicio y fecha de fin, y obtener el monto total de egreso e ingreso en la empresa. Los parámetros en este caso sería la selección de fechas. **VERIFICAR ALTA DE PRESTAMOS**

21) Utilizar descolgables en los casos que sea necesario. Ejemplo: para provincia, localidades, géneros, etc. Si se modifica dentro de una grilla y tienen campos que aplican para ser descolgables, entonces utilizar dicho control.

22) En el caso de dar de alta un usuario, un usuario común NO PUEDE DAR DE ALTA UN ADMINISTRADOR.

23) Cambiar el nombre de las columnas de las grillas y dejar uno apropiado. Usar para ello la instrucción AS del lenguaje SQL. Ejemplo: “Select p\_nombre AS nombre from Persona.

24) Antes de rendir, se deberá cargar la base de datos con varios registros con información válida para la prueba. Mínimamente 15 registros por tabla para poder probar búsquedas y filtros. **CREAR NUEVOS REGISTROS**