25/10/24

Llevo varias semanas trabajando en crear una base de datos de partida para comenzar con el desarrollo de mi aplicación.

La base de datos “lascuentasclaras” está estructurada para soportar una aplicación financiera personal y familiar con funcionalidades avanzadas de gestión de usuarios, transacciones y educación financiera. La organización en múltiples tablas permite un control detallado de roles, permisos, y agrupación de usuarios en familias y grupos. A continuación, se describe cada componente en profundidad:

1. Usuarios, Roles y Permisos:

- Tabla “usuarios”: Registra cada usuario con detalles como nombre, alias, correo, nivel de usuario (“superadmin”, “admin”, “usuario”, “registro”), y estado (“activo” o “inactivo”). También se permite definir si el usuario es premium mediante una contraseña adicional.

- Tabla “roles”: Define los roles posibles en la aplicación.

- Tabla “roles\_permisos”: Asocia permisos a cada rol para acciones específicas en módulos como “gastos”, “ingresos”, “familias” y “grupos”. Los permisos (“leer”, “escribir”, “eliminar”) regulan el acceso a las funcionalidades según el rol del usuario.

2. Familias y Grupos:

- Tabla “familias”: Define grupos familiares con un “nombre\_familia”, una contraseña para acceso restringido y un “estado” (“activo”, “inactivo”).

- Tabla “grupos”: Organiza grupos de usuarios similares a “familias”, con sus propias contraseñas y características.

- Tablas de relación “usuarios\_familias” y “usuarios\_grupos”: Permiten que los usuarios se asocien a múltiples familias o grupos.

- Tablas “administradores\_familias” y “administradores\_grupos”: Asignan administradores específicos a cada familia o grupo, permitiendo la supervisión de operaciones y la gestión interna.

3. Movimientos Financieros:

- Tabla “ingresos”: Almacena los ingresos de cada usuario, incluyendo “importe”, “origen” (banco, efectivo, tarjeta, etc.), “concepto”, y la “fecha”. Cada ingreso puede asociarse opcionalmente a una “familia” o “grupo”.

- Tabla “gastos”: Similar a “ingresos”, registra todos los gastos con detalles de importe, categoría, origen, concepto y fecha. También admite la asociación a una “familia” o “grupo” para facilitar la gestión grupal.

4. Resumen Financiero y Notificaciones:

- Tabla “situacion”: Calcula y almacena resúmenes de saldo, ingresos y gastos en períodos de tiempo definidos (mensual, semanal, anual o por intervalo). Este resumen ayuda a los usuarios a analizar su situación financiera de manera continua.

- Tabla “notificaciones”: Permite enviar mensajes y recordatorios a los usuarios sobre eventos importantes o cambios en su situación financiera. Cada notificación tiene una fecha y un estado de lectura.

5. Información Financiera:

- Tabla “definiciones\_financieras”: Ofrece términos financieros comunes con definiciones detalladas, almacenados en “texto\_concepto\_eco”. Estos términos son utilizados para la educación financiera de los usuarios.

- Tabla “news\_letter\_envios”: Gestiona los envíos de newsletters que contienen definiciones financieras y el resumen financiero de cada usuario. Cada registro incluye “saldo\_total”, “gastos\_totales” e “ingresos\_totales” del usuario, ayudando a la creación de boletines informativos personalizados.

6. Categorías de Transacciones:

- Tabla “categorias”: Permite definir categorías para “gastos” e “ingresos”, con atributos como “nombreCategoria”, “estado\_categoria” (“activo”, “inactivo”) y “tipo\_categoria” (“gasto” o “ingreso”). Esta tabla asegura una organización clara de las transacciones según el tipo y el estado de la categoría.

7. Integridad y Restricciones:

- La base de datos implementa restricciones de integridad referencial, lo que asegura que la eliminación de usuarios, familias o grupos actualice o elimine en cascada los datos relacionados. Esto mantiene la consistencia de la base de datos al eliminar registros y evita datos huérfanos en tablas relacionadas como “usuarios\_familias”, “ingresos” y “gastos”.

S 26/10/24

Tras crear la base de datos he comenzado con los siguientes archivos.

1. **Archivo Config.php**:  
   Se ha desarrollado el archivo Config.php, el cual centraliza los parámetros esenciales de conexión a la base de datos, como el host, nombre de la base de datos, usuario y contraseña. También incluye un sistema de manejo de errores que registra las excepciones en un archivo de log y proporciona información útil para depuración cuando está activado el modo debug.
2. **Archivo GastosModelo (classModelo.php)**:  
   Se ha creado el archivo classModelo.php, ubicado en la carpeta modelo. Este archivo actúa como el modelo principal para interactuar con la base de datos. Contiene un constructor que establece la conexión utilizando los valores definidos en Config.php. Además, cuenta con métodos como pruebaConexion() para verificar que la conexión a la base de datos esté activa, ofreciendo una capa adicional de control.
3. **Archivo de prueba (test\_conexion.php)**:  
   Se ha implementado el archivo test\_conexion.php, ubicado en la raíz del proyecto. Este archivo permite comprobar de forma rápida y directa si la conexión a la base de datos se ha realizado correctamente, mostrando un mensaje de éxito o de error en función de la respuesta del sistema.
4. **Bibliotecas Adicionales**:
   * **bGeneral.php**: Contiene funciones generales que se pueden reutilizar en distintas partes del proyecto.
   * **bSeguridad.php**: Proporciona funciones relacionadas con la seguridad, incluyendo validación de usuarios y control de acceso, ayudando a garantizar la integridad y protección del sistema.

S26-D27/10

Desde la última implementación, he avanzado en la estructura y funcionalidades principales de la aplicación. A continuación, detallo los aspectos que he ido completando:

**1. Estructura General**

* **Archivo index.php**: Este archivo es el punto de entrada de la aplicación. He configurado un sistema de enrutamiento con Router.php para que sea más fácil gestionar las rutas y asegurar que solo los usuarios autenticados accedan a ciertas páginas.

**2. Librerías y Componentes de Configuración**

* **Router.php**: Este archivo gestiona todas las rutas, permitiéndome asignar cada una a un controlador específico y verificando que los usuarios solo accedan a las páginas adecuadas según su nivel de acceso.

**3. Controlador de Autenticación**

He implementado el controlador de autenticación AuthController.php en app/controlador, que gestiona toda la funcionalidad de registro e inicio de sesión:

* **Métodos**:
  + iniciarSesion(): Procesa el inicio de sesión de los usuarios.
  + registro(): Permite que los usuarios nuevos se registren.
  + home(), inicio(), y salir(): Facilitan la navegación y el cierre de sesión.
  + Además, he añadido el método render() para cargar dinámicamente las vistas, como el formulario de inicio de sesión (formIniciarSesion.php) y el de registro (formRegistro.php).

**4. Vistas y Plantillas**

En la carpeta web/templates, he configurado las plantillas principales:

* **layout.php**: Es la estructura base de la aplicación, donde he definido el head, header, y footer, además de agregar Bootstrap y un CSS personalizado para un diseño responsivo y juvenil.
* **Formularios (formIniciarSesion.php y formRegistro.php)**: He creado formularios seguros para el inicio de sesión y el registro, con validaciones de entrada y protección contra ataques CSRF.

**5. Estilo de la Aplicación**

En web/css, he creado:

* **estilo.css**: Un archivo de estilo que da un aire moderno y atractivo a la aplicación, con colores azul y blanco para mantener la sensación de confianza y claridad. Este estilo se complementa con Bootstrap 4.5.2 para mejorar el diseño y hacer la aplicación responsiva.

L28-M29/10/24

**Implementación de Gestión de Usuarios**

1. **Listado de Usuarios**: Se ha creado una vista listarUsuarios.php en la carpeta de plantillas (web/templates) para mostrar un listado detallado de los usuarios registrados en el sistema. Esta vista presenta datos esenciales, como el nombre, alias, correo electrónico, nivel de usuario, así como la familia o grupo al que pertenecen. Dependiendo del rol del usuario en sesión (superadmin o admin), se muestran opciones adicionales para la creación, edición y eliminación de usuarios. La implementación incluye:
   * Botones dinámicos de creación de usuario para roles con permisos elevados.
   * Acciones de edición y eliminación con restricciones de acceso según el rol.
   * Clasificación del usuario como “Individual”, “Familiar”, “En Grupo”, o ambos, en función de su asociación con una familia o grupo.
2. **Métodos de classModelo.php para Obtener Información de Usuarios**: Se han añadido varios métodos nuevos en el archivo classModelo.php para gestionar la obtención de datos de los usuarios:
   * obtenerUsuariosGestionados: Recupera los usuarios que son gestionados específicamente por un administrador. Este método permite a los administradores ver solo los usuarios que les corresponden, mientras que el superadmin tiene acceso a todos.
   * obtenerUsuarios: Proporciona un listado general de usuarios junto con la familia y grupo al que están asociados, usando consultas optimizadas y manejo de errores para mejorar el rendimiento y la estabilidad.
   * obtenerTodosLosUsuarios: Obtiene un listado de datos básicos de todos los usuarios, siendo una versión más ligera y rápida para mostrar solo la información esencial.
3. **Optimización y Buenas Prácticas**: Para mantener un rendimiento adecuado y una estructura clara en el código, se han realizado las siguientes prácticas:
   * Inclusión de registros de errores (error\_log) en los métodos, lo que permite una fácil depuración en caso de problemas.
   * Uso de COALESCE en las consultas SQL para mostrar “Sin Familia” o “Sin Grupo” cuando el usuario no tiene asignaciones, mejorando la claridad en la visualización.
   * Implementación de controles condicionales en la vista para restringir el acceso a acciones según el rol del usuario.

Del X30/10-D10/11/24

**Esto es resumido lo que he implementado:**

1. **Gestión de Roles y Menú Dinámico en el Layout**:
   * Se ha actualizado el layout para mostrar menús personalizados según el rol del usuario (Superadmin, Admin, Usuario Regular).
   * El Superadmin ve todas las opciones, mientras que el Admin y Usuario Regular solo ven las opciones aplicables a sus permisos.
   * El Admin tiene acceso a gestionar usuarios, familias, y grupos de los que es administrador, con restricciones según el número de familias y grupos gestionados.
   * El Usuario Regular ve solo opciones de consulta y gestión de sus propias finanzas.
2. **Implementación de Métodos de Acceso para Admin en classModelo.php**:
   * **obtenerFamiliasAdmin** y **obtenerGruposAdmin**: Métodos que recuperan las familias y grupos administrados por un usuario de rol Admin. Estos métodos ayudan a limitar la visualización en el menú de Admin según las asociaciones permitidas.
   * **Control de Permisos por Usuario**: Añadido para garantizar que los Admin solo gestionen los usuarios, familias y grupos asignados.
3. **Mejoras en la Funcionalidad de Edición de Categorías**:
   * Implementación de vistas y lógica para que los roles con permiso puedan editar categorías de gastos e ingresos. Se ha implementado control de acceso y validación para evitar modificaciones no autorizadas.
4. **Optimización de Vistas y Consultas SQL**:
   * Consultas SQL optimizadas en métodos que obtienen usuarios y categorías, con uso de COALESCE para mostrar descripciones predeterminadas en casos de valores nulos.
   * Implementación de controles condicionales en las vistas, limitando las acciones según el rol y reduciendo la carga visual y lógica en usuarios con permisos restringidos.
5. **Mejoras de Seguridad en el Registro de Acciones**:
   * Añadido error\_log en métodos críticos para facilitar la depuración y control de eventos, especialmente en acciones de creación, edición y eliminación realizadas por usuarios con rol Superadmin y Admin.

Este es el árbol hasta el momento:

C:\XAMPP2\HTDOCS\PROY\LASCUENTASCLARAS

| index.php

| test\_conexion.php

|

+---app

| +---controlador

| | AuthController.php

| | CategoriaController.php

| | FamiliaGrupoController.php

| | UsuarioController.php

| |

| +---libs

| | bGeneral.php

| | bSeguridad.php

| | Config.php

| | Router.php

| |

| +---log

| | php\_errors.log

| |

| +---modelo

| | classModelo.php

| |

| \---sql

| lascuentasclaras(2).sql

| lascuentasclaras.sql

|

+---doc

| | Fechas Proyecto.JPG

| | Instrucciones para el desarrollo de proyectos (DAW\_ doc alumnos)\_2023-24.pdf

| | Memoria\_LasCuentasClaras.docx

| | Plantilla\_propuesta\_DAW\_Proyecto\_Rafa\_Muro000.pdf

| | Plantilla\_propuesta\_DAW\_Proyecto\_Rafa\_Muro\_G\_signed.pdf

| |

| \---ÍNDICE

| +---propuesta1

| | Propuesta borrador ÍNDICE proyecto DAW - Rafa Muro -Las Cuentas Claras.docx

| |

| \---propuesta2

| Propuesta2 borrador ÍNDICE proyecto DAW - Rafa Muro -Las Cuentas Claras - sin decrip\_1pag.docx

| Propuesta2 borrador ÍNDICE proyecto DAW - Rafa Muro -Las Cuentas Claras - sin descrip.docx

| Propuesta2 borrador ÍNDICE proyecto DAW - Rafa Muro -Las Cuentas Claras.docx

|

\---web

+---css

| estilo.css

|

\---templates

error.php

footer.php

formAsignarUsuario.php

formCrearFamilia.php

formCrearGrupo.php

formCrearUsuario copy.php.txt

formCrearUsuario.php

formEditarCategoriaGasto.php

formEditarCategoriaIngreso.php

formEditarFamilia.php

formEditarGrupo.php

formEditarUsuario.php

formIniciarSesion.php

formRegistro.php

home.php

inicio.php

layout.php

listarFamilias.php

listarGrupos.php

listarUsuarios.php

verCategoriasGastos.php

verCategoriasIngresos.php