DESCRIPCIÓN DE MÓDULOS - SISTEMA DE EXPEDIENTE CLÍNICO DIGITAL

1. MÓDULO DE LOGIN (Inicio de Sesión)

Archivo: index.html, js/login.js
Propósito: Controlar el acceso al sistema mediante autenticación de usuarios
Funcionalidades:

- Validación de credenciales de usuario
- Creación de sesión de usuario
- Almacenamiento de datos del usuario en sesión
- Redirección al Dashboard después del login exitoso
- Validación de campos obligatorios

Tecnologías:

- HTML5 para la estructura
- JavaScript para la lógica de validación
- LocalStorage para el manejo de sesiones

Interacción con otros módulos:

- Todos los módulos verifican si existe una sesión activa antes de permitir el acceso

2. MÓDULO DE DASHBOARD (Panel Principal)

Archivo: dashboard.html, js/dashboard.js
Propósito: Mostrar una vista general del estado del sistema con estadísticas clave
Funcionalidades:

- Visualización de total de pacientes registrados
- Contador de citas programadas para el día actual
- Visualización de médicos activos
- Contador de pagos pendientes
- Lista de citas recientes (últimas 5)
- Navegación rápida a todos los módulos del sistema

Datos mostrados:

- Tarjetas estadísticas con números relevantes
- Tabla resumen de citas recientes con: paciente, médico, fecha y estado
- Información del usuario en sesión
- Botón de cerrar sesión

Tecnologías:

- HTML5 con estructura de cards responsivos
- JavaScript para carga dinámica de datos
- CSS Grid para layout de tarjetas

Fuente de datos:

- LocalStorage (para versión sin BD)
- Base de datos MySQL (para módulos integrados)

3. MÓDULO DE PACIENTES (Control de Pacientes)

Archivo: pacientes.html, js/pacientes.js
Propósito: Gestión completa del registro de pacientes
Funcionalidades:

- CRUD completo:
  - Crear: Registro de nuevos pacientes con todos sus datos
  - Leer: Visualización de lista completa de pacientes
  - Actualizar: Edición de información de pacientes existentes
  - Eliminar: Baja de pacientes del sistema
- Búsqueda en tiempo real por nombre, CURP o teléfono
- Filtrado de pacientes activos/inactivos

Campos del formulario:

- Nombre completo (obligatorio)
- CURP (18 caracteres, obligatorio)
- Fecha de nacimiento (obligatorio)
- Sexo (M/F, obligatorio)
- Teléfono (obligatorio)
- Correo electrónico (obligatorio)
- Dirección
- Contacto de emergencia y teléfono
- Alergias
- Antecedentes médicos
- Estado (Activo/Inactivo)

Validaciones:

- Campos obligatorios no pueden estar vacíos
- CURP debe tener exactamente 18 caracteres
- Correo debe tener formato válido
- Teléfono máximo 20 caracteres

Almacenamiento:

- LocalStorage (versión actual)
- Tabla pacientes en base de datos (preparado para implementación futura)

4. MÓDULO DE AGENDA (Control de Citas)

Archivo: agenda.html, js/agenda.js
Propósito: Gestión del calendario de citas médicas
Funcionalidades:

- Creación de citas médicas
- Asignación de paciente y médico a cada cita
- Programación de fecha y hora
- Registro del motivo de consulta
- Gestión de estados de citas (Programada, Cancelada, Atendida)
- Búsqueda de citas por paciente, médico o motivo
- Visualización de citas ordenadas por fecha

Campos del formulario:

- Paciente (selección de lista, obligatorio)
- Médico (selección de lista, obligatorio)
- Fecha y hora (obligatorio)
- Motivo de consulta (obligatorio)
- Estado (Programada/Cancelada/Atendida)
- Observaciones

Relaciones:

- Cada cita está relacionada con un paciente (idPaciente)
- Cada cita está relacionada con un médico (idMedico)
- Una cita puede generar un expediente clínico
- Una cita puede estar asociada a un pago

Estados de la cita:

- Programada: Cita agendada pero no atendida
- Cancelada: Cita que se canceló
- Atendida: Cita que ya fue realizada

Almacenamiento:

- LocalStorage (versión actual)
- Tabla citas en base de datos (preparado para implementación futura)

5. MÓDULO DE MÉDICOS (Control de Médicos)

Archivo: medicos.html, js/medicos.js, php/medicos_api.php
Propósito: Gestión del catálogo de médicos del sistema
Funcionalidades:

- CRUD completo con base de datos:
  - Crear: Registro de nuevos médicos
  - Leer: Consulta de médicos con su especialidad
  - Actualizar: Edición de datos de médicos
  - Eliminar: Baja de médicos (con validación de citas asociadas)
- Búsqueda por nombre, cédula o especialidad
- Asignación de especialidad médica
- Control de estado activo/inactivo

Campos del formulario:

- Nombre completo (obligatorio)
- Cédula profesional (50 caracteres, obligatorio)
- Especialidad (selección de catálogo, obligatorio)
- Teléfono (obligatorio)
- Correo electrónico (obligatorio)
- Horario de atención
- Estado (Activo/Inactivo)

Validaciones en PHP:

- No permite eliminar médicos con citas asociadas
- Valida que la especialidad exista
- Campos obligatorios

Base de datos:

- Tabla: medicos
- Relación: especialidadId → especialidades.idEspecialidad
- API REST: php/medicos_api.php

Operaciones API:

- GET ?accion=listar - Obtiene todos los médicos con JOIN a especialidades
- GET ?accion=obtener&id=X - Obtiene un médico específico
- POST ?accion=crear - Crea un nuevo médico
- POST ?accion=actualizar - Actualiza un médico existente
- GET ?accion=eliminar&id=X - Elimina un médico

6. MÓDULO DE REPORTES

Archivo: reportes.html, js/reportes.js
Propósito: Generación de reportes del sistema
Funcionalidades:

- Generación de diferentes tipos de reportes
- Filtrado por paciente y médico
- Registro de reportes generados
- Consulta de historial de reportes

Tipos de reportes:

- Reporte de Médicos: Información de médicos y sus consultas
- Reporte Financiero: Resumen de ingresos y pagos
- Reporte de Citas: Estadísticas de citas programadas/atendidas
- Reporte de Pacientes: Lista de pacientes y sus consultas

Campos del formulario:

- Tipo de reporte (obligatorio)
- Paciente (opcional)
- Médico (opcional)

Datos registrados:

- ID del reporte
- Tipo de reporte
- Fecha de generación
- Usuario que lo generó
- Ruta del archivo (simulada)
- Descripción

Almacenamiento:

- LocalStorage (versión actual)
- Tabla reportes en base de datos (preparado para implementación futura)

7. MÓDULO DE PAGOS (Gestor de Pagos)

Archivo: pagos.html, js/pagos.js, php/pagos_api.php
Propósito: Control financiero de pagos de consultas
Funcionalidades:

- CRUD completo con base de datos:
  - Crear: Registro de nuevos pagos
  - Leer: Consulta de pagos con información del paciente
  - Actualizar: Modificación de datos de pago
  - Eliminar: Eliminación de registros de pago
- Búsqueda por paciente, método o estado
- Asociación de pagos con citas y pacientes
- Control de métodos de pago

- Gestión de estados de pago

Campos del formulario:

- Cita asociada (obligatorio)
- Paciente (obligatorio)
- Monto (obligatorio, decimal)
- Método de pago (Efectivo/Tarjeta/Transferencia, obligatorio)
- Referencia (opcional)
- Estado del pago (Pagado/Pendiente/Cancelado)

Métodos de pago:

- Efectivo
- Tarjeta
- Transferencia

Estados del pago:

- Pagado: Pago completado
- Pendiente: Pago por realizar
- Cancelado: Pago cancelado

Base de datos:

- Tabla: pagos
- Relaciones:
  - idCita → citas.idCita
  - idPaciente → pacientes.idPaciente
- API REST: php/pagos_api.php

Operaciones API:

- GET ?accion=listar - Obtiene todos los pagos con JOIN a pacientes
- GET ?accion=obtener&id=X - Obtiene un pago específico
- POST ?accion=crear - Registra un nuevo pago
- POST ?accion=actualizar - Actualiza un pago existente
- GET ?accion=eliminar&id=X - Elimina un pago

8. MÓDULO DE TARIFAS (Gestor de Tarifas/Servicios)

Archivo: tarifas.html, js/tarifas.js, php/tarifas_api.php
Propósito: Administración del catálogo de servicios y sus precios
Funcionalidades:

- CRUD completo con base de datos:
  - Crear: Registro de nuevas tarifas
  - Leer: Consulta de tarifas con especialidad asociada
  - Actualizar: Modificación de precios y descripciones
  - Eliminar: Eliminación de tarifas
- Búsqueda por descripción o especialidad
- Asociación opcional con especialidades
- Control de estado activo/inactivo
- Gestión de costos base

Campos del formulario:

- Descripción del servicio (150 caracteres, obligatorio)
- Costo base (decimal, obligatorio)
- Especialidad asociada (opcional)
- Estado (Activo/Inactivo)

Validaciones:

- Costo debe ser mayor a 0
- Descripción no puede estar vacía

Base de datos:

- Tabla: tarifas
- Relación: especialidadId → especialidades.idEspecialidad (opcional)
- API REST: php/tarifas_api.php

Operaciones API:

- GET ?accion=listar - Obtiene todas las tarifas con JOIN a especialidades
- GET ?accion=obtener&id=X - Obtiene una tarifa específica
- POST ?accion=crear - Crea una nueva tarifa
- POST ?accion=actualizar - Actualiza una tarifa existente

- GET ?accion=eliminar&id=X - Elimina una tarifa

9. MÓDULO DE BITÁCORAS (Bitácoras de Usuarios)

Archivo: bitacoras.html, js/bitacoras.js
Propósito: Auditoría y registro de acciones del sistema
Funcionalidades:

- Registro automático de todas las acciones
- Consulta de historial de acciones
- Búsqueda por usuario, acción o módulo
- Limpieza de bitácoras (requiere confirmación)
- Visualización ordenada por fecha (más recientes primero)

Datos registrados:

- ID de la bitácora
- ID del usuario que realizó la acción
- Nombre del usuario
- Fecha y hora de la acción
- Acción realizada (descripción)
- Módulo donde se realizó

Ejemplos de acciones registradas:

- "Consulta de bitácoras"
- "Limpieza de bitácoras"
- "Inicio de sesión"
- "Creación de paciente"
- "Edición de médico"

Funcionalidad especial:

- La función registrarAccion() puede ser utilizada en cualquier módulo para registrar actividades

Almacenamiento:

- LocalStorage (versión actual)
- Tabla bitacoras en base de datos (preparado para implementación futura)

10. MÓDULO DE ESPECIALIDADES (Especialidades Médicas)

Archivo: especialidades.html, js/especialidades.js, php/especialidades_api.php
Propósito: Gestión del catálogo de especialidades médicas
Funcionalidades:

- CRUD completo con base de datos:
  - Crear: Registro de nuevas especialidades
  - Leer: Consulta de todas las especialidades
  - Actualizar: Modificación de especialidades existentes
  - Eliminar: Baja de especialidades (con validación de médicos asociados)
- Búsqueda por nombre o descripción
- Catálogo pre-cargado con especialidades comunes

Campos del formulario:

- Nombre de la especialidad (100 caracteres, obligatorio)
- Descripción (250 caracteres, opcional)

Validaciones en PHP:

- No permite eliminar especialidades con médicos asociados
- Nombre no puede estar vacío

Especialidades pre-cargadas:

1. Medicina General
2. Pediatría
3. Cardiología
4. Dermatología
5. Ginecología

Base de datos:

- Tabla: especialidades
- API REST: php/especialidades_api.php

Operaciones API:

- GET ?accion=listar - Obtiene todas las especialidades
- GET ?accion=obtener&id=X - Obtiene una especialidad específica
- POST ?accion=crear - Crea una nueva especialidad
- POST ?accion=actualizar - Actualiza una especialidad existente
- GET ?accion=eliminar&id=X - Elimina una especialidad

Uso en otros módulos:

- Módulo de Médicos: Asignación de especialidad a cada médico
- Módulo de Tarifas: Asociación opcional de servicios con especialidades

11. COMPONENTES COMPARTIDOS

Barra de Navegación Lateral

Presente en: Todos los módulos excepto Login
Funcionalidades:

- Menú con enlaces a todos los módulos
- Resaltado del módulo activo
- Logo del sistema
- Diseño responsive (se adapta a móviles)

Encabezado

Presente en: Todos los módulos excepto Login
Componentes:

- Título del módulo actual
- Nombre del usuario en sesión
- Botón de cerrar sesión

Barra de Búsqueda

Presente en: Pacientes, Médicos, Agenda, Pagos, Tarifas, Bitácoras, Especialidades
Funcionalidades:

- Búsqueda en tiempo real
- Filtrado de resultados mientras se escribe
- Sin necesidad de presionar botón de búsqueda

Modales

Presente en: Todos los módulos con formularios
Funcionalidades:

- Ventana emergente para crear/editar registros
- Botón de cerrar (X)
- Fondo oscuro transparente
- Formulario con validaciones
- Botón de guardar

Tablas de Datos

Presente en: Todos los módulos con listados
Características:

- Diseño responsive
- Scroll horizontal en pantallas pequeñas
- Botones de acción en cada fila (Editar/Eliminar)
- Hover effect en filas
- Estados con colores (badges)

12. SISTEMA DE ESTILOS

Paleta de Colores

- Color primario: Negro (#1a1a1a) - Fondo principal
- Color secundario: Gris oscuro (#2a2a2a) - Tarjetas y secciones
- Color terciario: Gris medio (#3a3a3a) - Inputs y elementos
- Color de acento: Morado (#8b5cf6) - Botones y enlaces
- Color de texto: Gris claro (#e5e5e5)
- Color de éxito: Verde (#10b981)

- Color de peligro: Rojo (#ef4444)
- Color de advertencia: Naranja (#f59e0b)

Características del Diseño

- Diseño oscuro (dark mode)
- Bordes redondeados
- Sombras sutiles
- Transiciones suaves
- Diseño responsive (se adapta a todos los dispositivos)
- Fuente: Segoe UI