Orden de Trabajo: Cierre de Fase 1 (Versión Definitiva)

Proyecto: Plataforma de Consultoría de Ciberseguridad

Estado: Bloqueante para entrega.

Objetivo: Implementar la lógica de roles (quién ve qué), arreglar el catálogo y habilitar el perfil de usuario.

TAREA CONJUNTA: Lógica de Roles y Mapa de Vistas (RBAC)

Prioridad: MÁXIMA (Define la seguridad de la app).

Responsables: Backend (Define permisos) + Frontend (Oculta menús).

1. Mapa de Vistas en el FRONTEND (Clientes)

El cliente no es único. Hay que distinguir entre el Jefe y el Empleado.

- **Rol cliente_admin (El Jefe):**
    - **DEBE VER:** Dashboard de Empresa (Gráficas riesgo, Proyectos ISO), Botón "Gestionar Usuarios" (Altas de empleados), Menú "Facturación".
- **Rol cliente_user (El Empleado):**
    - **NO DEBE VER:** Nada de la empresa (ni facturas, ni estado de auditoría).
    - **SOLO VE:** "Mis Cursos", "Mis Diplomas" y botón "Reportar Incidente". Su Dashboard es personal.

2. Mapa de Vistas en el BACKEND (Personal Interno)

Configurar la visibilidad del menú lateral (left.php) según el rol:

- **admin:** Ve TODO. Acceso total.
- **manager:** Ve "Proyectos" (Visión Global) y "Rentabilidad". Asigna tareas.
- **comercial:** Ve "Catálogo" y "CRM/Clientes Potenciales". **Ocultar:** Auditorías y Tickets técnicos.
- **consultor / auditor:** Ven "Mis Proyectos Asignados". **Ocultar:** Facturación y Configuración.
- **analista_soc:** Ve "Tickets" y "Monitorización". **Ocultar:** Auditorías de GRC.

Instrucción Técnica (Cómo programarlo)

Para aplicar esta "Barrera Visual", utilizad la función can() de Yii2 en los archivos de vista del menú (views/layouts/left.php o main.php).

Ejemplo de código a copiar/adaptar:

PHP

```
// Ejemplo en el array de items del menú
$menuItems = [
   ['label' => 'Inicio', 'url' => ['/site/index']],
];

// Solo el JEFE DEL CLIENTE ve la facturación
if (Yii::$app->user->can('cliente_admin')) {
   $menuItems[] = ['label' => 'Facturación', 'url' => ['/factura/index']];
   $menuItems[] = ['label' => 'Gestionar Usuarios', 'url' => ['/usuario/equipo']];
}

// Solo el EMPLEADO ve sus cursos
if (Yii::$app->user->can('cliente_user')) {
   $menuItems[] = ['label' => 'Mis Cursos', 'url' => ['/formacion/mis-cursos']];
}

// EN EL BACKEND: Solo COMERCIAL o MANAGER ven Ventas
if (Yii::$app->user->can('comercial') || Yii::$app->user->can('manager')) {
   $menuItems[] = ['label' => 'CRM / Ventas', 'url' => ['/cliente/potenciales']];
}
```

TAREAS ESPECÍFICAS POR EQUIPO

Tarea F1 (Frontend): Vista Detallada de Servicios

Problema: No hay botón para contratar

Contenido Obligatorio:

1. **Añadir Botón "Contratar":**
    - En la nueva vista view.php, añadir un botón verde grande.

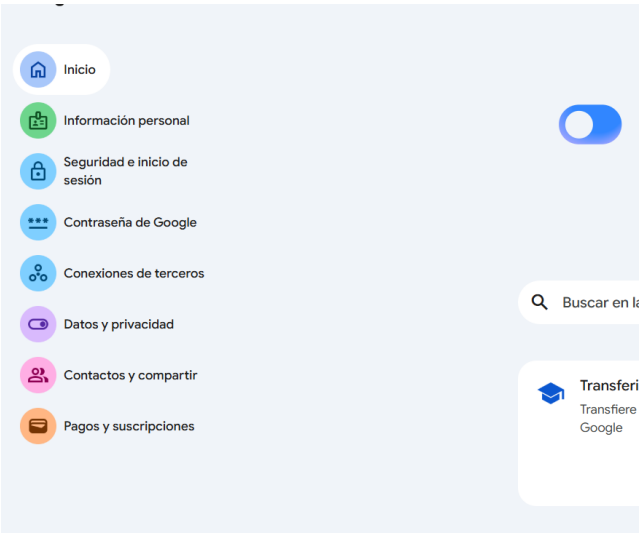
- **Lógica:** Debe llevar al formulario de creación de proyecto, pasando el ID del servicio para que se pre-seleccione.
- *Código:* <?= Html::a('Contratar Servicio', ['/proyecto/create', 'servicio_id' => $model->id], ['class' => 'btn btn-success btn-lg']) ?>

Tarea F2 (Frontend): Diseño del Panel "Mi Cuenta" (Estilo Google)

Objetivo: Implementar una interfaz limpia con menú lateral a la izquierda y el formulario a la derecha.

Estructura de la Vista (views/settings/security.php y profile.php): El equipo de diseño debe usar una rejilla (Grid) de Bootstrap para esta estructura:

- **Columna Izquierda (Menú de Navegación):**
  - Debe ser una lista de enlaces (list-group).
  - Debe marcarse como "Activo" (active) la sección donde esté el usuario.
  - Items:
    1. **Información Personal** (Lleva a actionProfile)
    2. **Seguridad e Inicio de Sesión** (Lleva a actionSecurity)
- **Columna Derecha (Área de Trabajo):**
  - Aquí se renderiza el formulario (ActiveForm).
  - Debe estar dentro de una "Card" o panel con sombra suave (shadow-sm) y bordes redondeados.



Tarea B1 (Backend): Lógica de Configuración de Usuario

Objetivo: Crear un controlador dedicado para gestionar la cuenta del usuario, separando la lógica de "Datos Personales" de la de "Seguridad".

Instrucciones Técnicas:

1. **Nuevo Controlador (SettingsController):**
    - No usar el SiteController. Crear uno nuevo en frontend/controllers/SettingsController.php.
    - **Action actionProfile():** Carga el modelo del usuario actual para editar nombre/email.
    - **Action actionSecurity():** Instancia el modelo de cambio de contraseña (ver punto 2).
2. **Modelo de Validación (ChangePasswordForm):**
    - Crear un modelo que **no** sea ActiveRecord (en common/models/ChangePasswordForm.php).
    - **Campos obligatorios:**
        - password_actual: Validar que coincide con la BBDD (validatePassword).
        - password_nueva: Mínimo 8 caracteres.
        - password_repetir: Debe ser idéntica a password_nueva (compare).

Tarea B2 (Backend): Limpieza Final

Acción:

- Eliminar botones de "Login" duplicados dentro del panel.
- Eliminar enlaces redundantes del menú superior.
- Asegurar que el botón "Logout" solo sale si estás logueado.