



# PROYECTO DE CARNET DIGITAL

---

## Programación V

Colegio Universitario de Cartago

Prof. Ing. Gabriel González Solano

# Carnet Digital del CUC

## Objetivo general proyecto

Realizar la experiencia de la elaboración de un proyecto que se espera salga a producción, pasando por todas las etapas del desarrollo de software.

## Objetivos específicos

- Analizar requisitos de desarrollo de distintos componentes de software.
- Diseñar la propuesta de solución para los distintos componentes de software.
- Realizar la programación de los componentes de software que formarán parte de la solución.
- Probar los distintos componentes de software que formarán parte de la solución.
- Implementar los componentes de software que formarán parte de la solución.
- Trabajar en equipo para alcanzar la consecución de las metas del proyecto.
- Utilizar software que facilite la integración de código fuente y la gestión del ciclo de vida del software.

## Contexto del proyecto

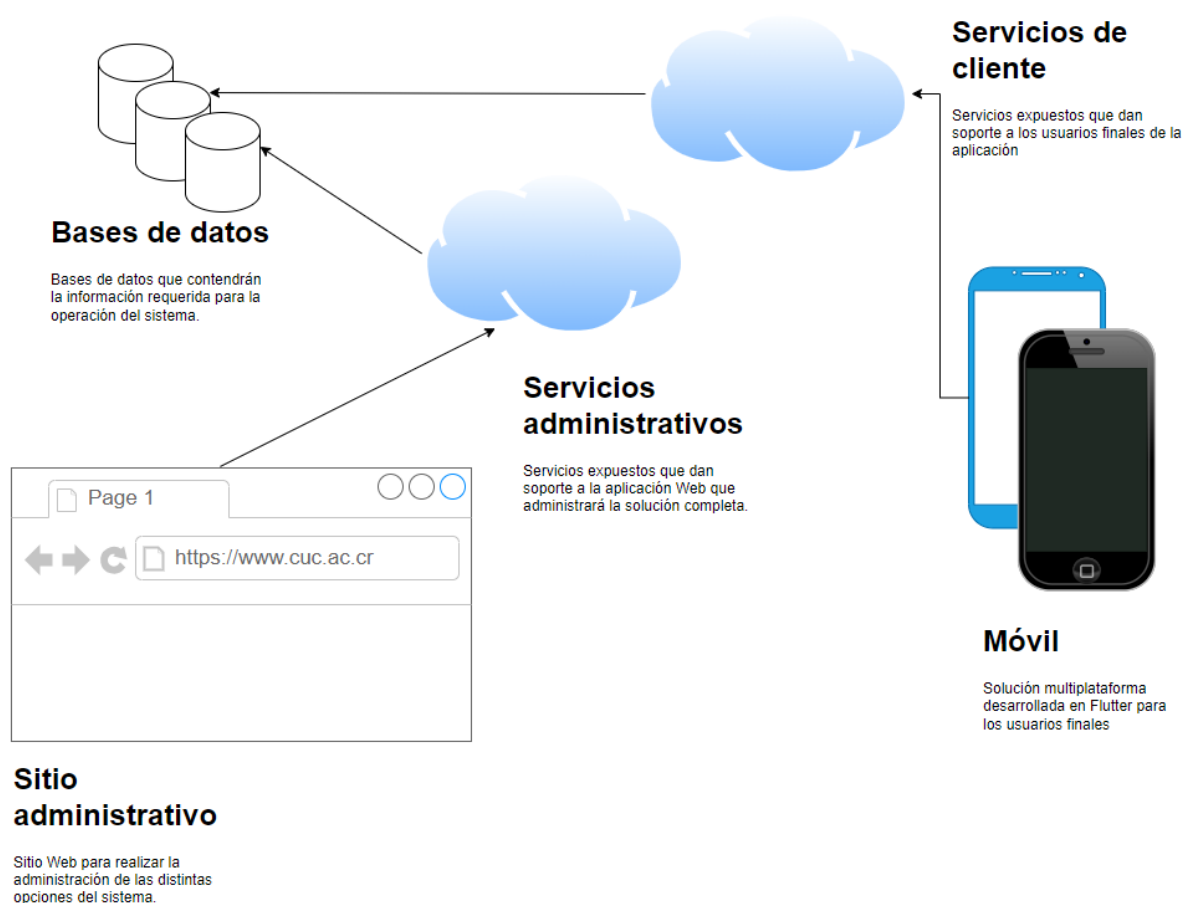
Recientemente y por directriz de la decanatura, el CUC ha fortalecido las medidas de seguridad en el ingreso a la institución, por medio del uso del carnet institucional, tanto para funcionarios como para estudiantes. Como toda medida de seguridad que se aplica, conlleva la aparición de algunos inconvenientes para las personas en las que se aplica esta.

El principal inconveniente que se ha presentado es cuando una persona olvida su carnet físico pues debe realizar un trámite que retrasa su ingreso a la institución y que provoca que el personal de seguridad deba atender la situación.

Como parte de las iniciativas que surgen a lo interno de nuestra institución, se ha planteado la posibilidad de realizar un proyecto que cubra los temas del curso de programación V y que permita al final de su ejecución poseer la mayor parte de la solución del Carnet Digital del CUC, el cual se espera que permita a las personas portar su identificación institucional en un dispositivo móvil y al personal de seguridad poder corroborarlo de una manera sencilla.

## Arquitectura general del sistema

A continuación, se presenta un vistazo arquitectónico de alto nivel de cada uno de los componentes que formarán parte de la solución a implementar:



*Ilustración 1 Arquitectura general del sistema*

## Organización del trabajo y de los equipos

Para trabajar el proyecto se seguirá el modelo de trabajo propuesto en Scrum. Dada la importante cantidad de requerimientos a atender, cada uno de los subgrupos formados realizará la atención de diferentes componentes.

El grupo completo se subdividirá en 3 grupos que lo conoceremos como unidad. Cada unidad se asignará un nombre de grupo por medio del cual será reconocido y lo indicará el primer día de clases. Cada grupo de trabajo se distribuirá las historias de usuario que se le asignen entre sus integrantes. Si bien es cierto al final la responsabilidad es individual, cada grupo es responsable de las historias que se le asignen para lograr un sprint exitoso.

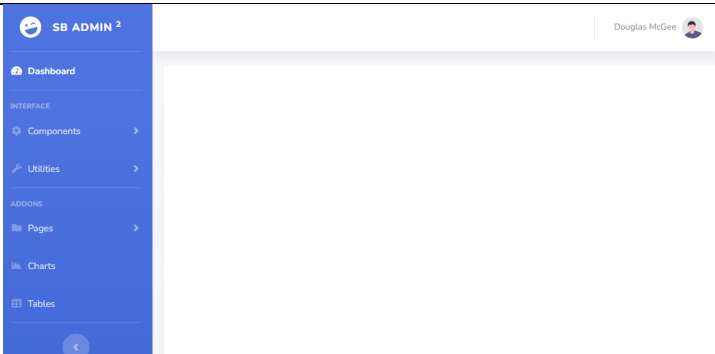
## Historias de usuario

Para la atención de los requisitos a desarrollar para el sistema, se crearán una serie de historias de usuario donde se detallará qué espera el usuario del sistema. A continuación, se realiza el detalle de las historias de usuario y sus criterios de aceptación.

*Segundo alcance – Consumo de Servicios Rest a través de pantallas Web*

ID	Historia de usuario	Criterios de aceptación
Web1	Yo como usuario del sistema quiero una pantalla de login para ingresar al sistema	<ul style="list-style-type: none"> <li>• Se requiere que el sistema tenga una pantalla de ingreso a este, para poder autenticar el usuario que ingresa.</li> <li>• Esta pantalla debe solicitar usuario y contraseña.</li> <li>• Debe poseer un botón “Aceptar”, que tendrá el siguiente comportamiento:               <ul style="list-style-type: none"> <li>○ Si no se indican el usuario y contraseña o se indican incorrectos, debe mostrar el mensaje “Usuario y/o contraseña incorrectos.”.</li> <li>○ Si las credenciales son correctas, entonces debe ingresar al sistema a la página de bienvenida (HU Web2). Esto además debe iniciar la sesión del</li> </ul> </li> </ul>

		<p>usuario y habilitar el ingreso al resto de opciones del sistema.</p> <ul style="list-style-type: none"> <li>○ Si se presentan 3 login fallidos el usuario debe bloquearse permanentemente.</li> <li>• Debe incluir un logo alusivo al sistema (pueden proponer un diseño).</li> <li>• Si se trata de ingresar a una pantalla sin haber iniciado sesión, se debe redireccionar a esta pantalla e indicar el mensaje: “Por favor inicie sesión para utilizar el sistema”.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web2	Yo como usuario del sistema quiero una pantalla para dar la bienvenida al usuario	<ul style="list-style-type: none"> <li>• Debe crearse una página de bienvenida que se muestre al usuario posterior al proceso exitoso de login. Esta página debe poseer los siguientes elementos:           <ul style="list-style-type: none"> <li>○ Indicar “Bienvenido, &lt;NOMBRE COMPLETO DEL USUARIO&gt;”.</li> <li>○ Debe incluir el logo de la empresa.</li> </ul> </li> <li>• Debe utilizar la plantilla del sitio indicada en la HU Web3.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web3	Yo como usuario del sistema quiero definir una plantilla para definir el comportamiento y	<ul style="list-style-type: none"> <li>• Debe definir una plantilla global del sitio que será utilizada en todas las páginas. Debe ser similar a la siguiente:</li> </ul>

	<p>visualización global del sitio</p>	 <ul style="list-style-type: none"> <li>○ Debe poseer en la parte superior a la derecha el nombre del usuario, junto con el avatar.</li> <li>○ A la izquierda debe poseer el nombre del sistema (el equipo de trabajo lo define) y un icono representativo.</li> <li>● A la izquierda debe poseer las opciones de menú del sistema.</li> <li>● Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web4	<p>Yo como usuario del sistema una pantalla para administración de usuarios.</p>	<ul style="list-style-type: none"> <li>● Debe crearse una pantalla que permita la administración de usuarios:           <ul style="list-style-type: none"> <li>○ Crear un usuario.</li> <li>○ Modificar un usuario.</li> <li>○ Eliminar un usuario.</li> <li>○ Obtener los datos de todos los usuarios.</li> <li>○ Obtener los datos de un usuario por su llave primaria.</li> <li>○ Obtener datos de los usuarios filtrados por: identificación, nombre, tipo.</li> </ul> </li> <li>● Los datos que se requieren para un usuario son:</li> </ul>



		<ul style="list-style-type: none"><li>○ Email (será la identificación única del usuario).</li><li>○ Tipo de identificación. Debe permitirse que eventualmente existan más tipos de identificación.</li><li>○ Identificación.</li><li>○ Nombre completo.</li><li>○ Contraseña (debe almacenarse encriptada).</li><li>○ Tipo de usuario (funcionario, estudiante o administrador). Debe permitirse que eventualmente existan más tipos de usuario.</li><li>○ Carreras asociadas (puede ser más de una).</li><li>○ Áreas asociadas (puede ser más de una).</li><li>○ Teléfono(s) de contacto.</li></ul> <ul style="list-style-type: none"><li>● Los teléfonos no son obligatorios.</li><li>● Si el usuario es estudiante, tendrá carreras asociadas. Si el usuario es funcionario tendrá áreas asociadas.</li><li>● Debe validarse el formato del email.</li><li>● Los emails deben ser de los dominios cuc.cr o cuc.ac.cr.</li><li>● Si el email es del dominio cuc.cr, el tipo de usuario debe ser estudiante.</li><li>● Si el dominio es cuc.ac.cr, el tipo de usuario debe ser funcionario o administrador.</li><li>● El nombre completo no puede ser vacío ni espacios en blanco.</li></ul>
--	--	---

		<ul style="list-style-type: none"> <li>• Todas las operaciones requieren ser utilizadas solo por usuarios autenticados.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web5	Yo como usuario del sistema quiero una pantalla para la administración de carreras de la institución	<ul style="list-style-type: none"> <li>• Debe crearse una pantalla que permita la administración de carreras:               <ul style="list-style-type: none"> <li>○ Crear una carrera.</li> <li>○ Modificar una carrera.</li> <li>○ Eliminar una carrera.</li> <li>○ Obtener los datos de todas las carreras</li> <li>○ Obtener los datos de una carrera por su llave primaria.</li> </ul> </li> <li>• Los datos requeridos para una carrera son:               <ul style="list-style-type: none"> <li>○ Identificador de la carrera.</li> <li>○ Nombre de la carrera.</li> <li>○ Director de la carrera.</li> <li>○ Email.</li> <li>○ Teléfono.</li> </ul> </li> <li>• Todos los datos son requeridos y no pueden ser vacíos ni espacios en blanco.</li> <li>• Debe validarse el formato del email.</li> <li>• El campo teléfono solo permite valores numéricos.</li> <li>• Todas las operaciones requieren ser utilizadas solo por usuarios autenticados.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web6	Yo como usuario del sistema quiero una pantalla para	<ul style="list-style-type: none"> <li>• Debe crearse una pantalla que permita la administración de tipos de usuario:               <ul style="list-style-type: none"> <li>○ Crear un tipo de usuario.</li> </ul> </li> </ul>



	la administración de tipos de usuario.	<ul style="list-style-type: none"> <li>○ Modificar un tipo de usuario.</li> <li>○ Eliminar tipos de usuario.</li> <li>○ Obtener los datos de todos los tipos de usuario.</li> <li>○ Obtener los datos de los tipos de usuario por su llave primaria.</li> </ul> <ul style="list-style-type: none"> <li>• Los datos requeridos para los tipos de usuario son:               <ul style="list-style-type: none"> <li>○ Identificador del tipo de usuario.</li> <li>○ Nombre del tipo de usuario.</li> </ul> </li> <li>• Todos los datos son requeridos y no pueden ser vacíos ni espacios en blanco.</li> <li>• Todas las operaciones requieren ser utilizadas solo por usuarios autenticados.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web7	Yo como usuario del sistema quiero una pantalla para la administración de tipos de identificación.	<ul style="list-style-type: none"> <li>• Debe crearse una pantalla que permita la administración de tipos de identificación:               <ul style="list-style-type: none"> <li>○ Crear un tipo de identificación.</li> <li>○ Modificar un tipo de identificación.</li> <li>○ Eliminar tipos de identificación.</li> <li>○ Obtener los datos de todos los tipos de identificación.</li> <li>○ Obtener los datos de los tipos de identificación por su llave primaria.</li> </ul> </li> <li>• Los datos requeridos para los tipos de identificación son:               <ul style="list-style-type: none"> <li>○ Identificador del tipo de identificación.</li> <li>○ Nombre del tipo de identificación.</li> </ul> </li> <li>• Todos los datos son requeridos y no pueden ser vacíos ni espacios en blanco.</li> </ul>

		<ul style="list-style-type: none"> <li>• Todas las operaciones requieren ser utilizadas solo por usuarios autenticados.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web8	Yo como usuario del sistema quiero una pantalla para la administración de la fotografía del usuario.	<ul style="list-style-type: none"> <li>• Debe crearse una pantalla que permita la administración de la fotografía:               <ul style="list-style-type: none"> <li>○ Actualizar fotografía: le agrega o actualiza la fotografía al usuario, la recibe en formato Base 64. Idealmente las imágenes a utilizar no deben ser superior a 1 MB de tamaño y deben estar en formato 4:3. Recibe como parámetros el identificador del usuario y la fotografía.</li> <li>○ Eliminar fotografía: elimina el dato de la fotografía del usuario. Recibe como parámetro el identificador del usuario.</li> <li>○ Obtener fotografía: obtiene la fotografía almacenada para el usuario en formato Base 64. Recibe como parámetro el identificador del usuario.</li> </ul> </li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> <li>• Todas las operaciones requieren ser utilizadas solo por usuarios autenticados</li> </ul>
Web9	Yo como usuario del sistema quiero una pantalla para la generación de un código QR con	<ul style="list-style-type: none"> <li>• Debe crearse una pantalla que permita obtener un código QR con la información del usuario indicado como parámetro.</li> <li>• Recibe como parámetro la identificación del usuario.</li> </ul>

	la información del carnet.	<ul style="list-style-type: none"> <li>• Todas las operaciones requieren ser utilizadas solo por usuarios autenticados.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web10	Yo como usuario del sistema quiero una pantalla para la administración de áreas de trabajo de funcionarios.	<ul style="list-style-type: none"> <li>• Debe crearse una pantalla que permita la administración de áreas de trabajo:               <ul style="list-style-type: none"> <li>○ Crear un área de trabajo.</li> <li>○ Modificar un área de trabajo.</li> <li>○ Eliminar área de trabajo.</li> <li>○ Obtener los datos de las áreas de trabajo.</li> <li>○ Obtener los datos de las áreas de trabajo por su llave primaria.</li> </ul> </li> <li>• Los datos requeridos para las áreas de trabajo son:               <ul style="list-style-type: none"> <li>○ Identificador del área de trabajo.</li> <li>○ Nombre del área de trabajo.</li> </ul> </li> <li>• Todos los datos son requeridos y no pueden ser vacíos ni espacios en blanco.</li> <li>• Todas las operaciones requieren ser utilizadas solo por usuarios autenticados.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
Web11	Yo como usuario del sistema quiero una pantalla que me permita cambiar el estado de un usuario	<ul style="list-style-type: none"> <li>• Debe crearse una pantalla que permita la modificación del estado de un usuario.</li> <li>• Los estados básicos son activo e inactivo, pero debe permitirse que se puedan agregar nuevos estados.</li> <li>• Para cambiar el estado se requiere del identificador del usuario y estado que se desea asignar.</li> </ul>

		<ul style="list-style-type: none"> <li>• Todos los datos son requeridos y no pueden ser vacíos ni espacios en blanco.</li> <li>• Todas las operaciones requieren ser utilizadas solo por usuarios autenticados.</li> <li>• Todas las operaciones deben realizarse a través del endpoint respectivo.</li> </ul>
--	--	--

## Manejo de bitácoras

Cada acción importante (operación de CRUD) que realice una pantalla debe ser registrada mediante una bitácora. Además, se deben registrar los errores técnicos que sucedan en el uso de las pantallas. Los datos de las bitácoras son los siguientes:

- Fecha de la bitácora.
- Usuario que ejecuta la acción.
- Descripción de la acción (Acción realizada + JSON).

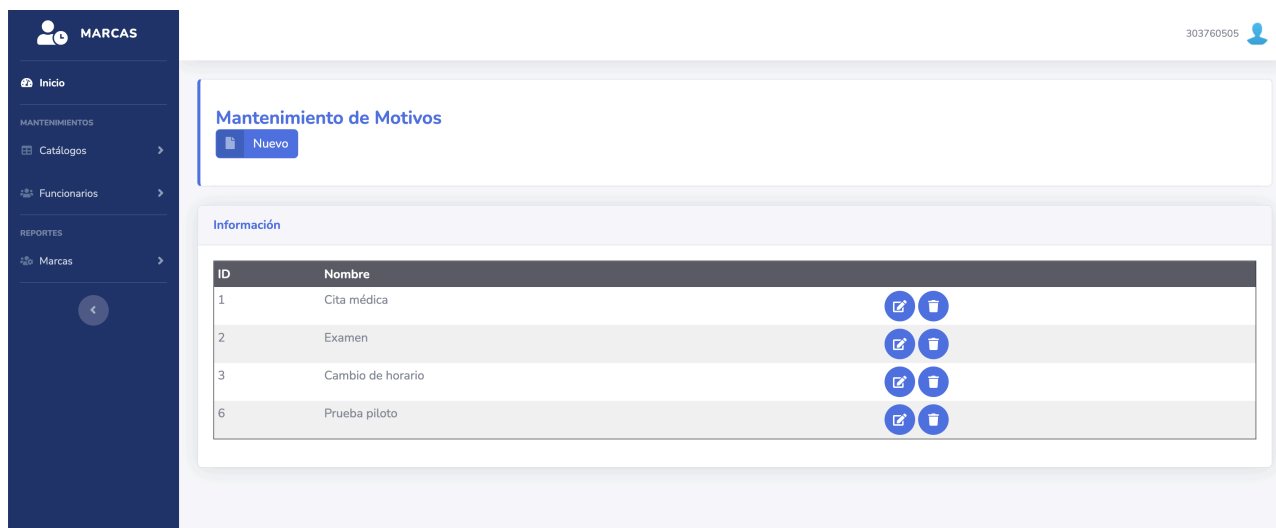
La descripción de la acción debe tener el siguiente detalle:

- En el caso de registros nuevos debe indicar el formato json el detalle del nuevo registro realizado.
- En el caso de actualización de registros, se debe indicar el formato json de la información del registro anterior y del registro actual.
- En el caso de eliminación de registros se debe indicar el formato json de la información eliminada.
- En el caso de consultas, únicamente se debe indicar “El usuario consulta <elemento>”.

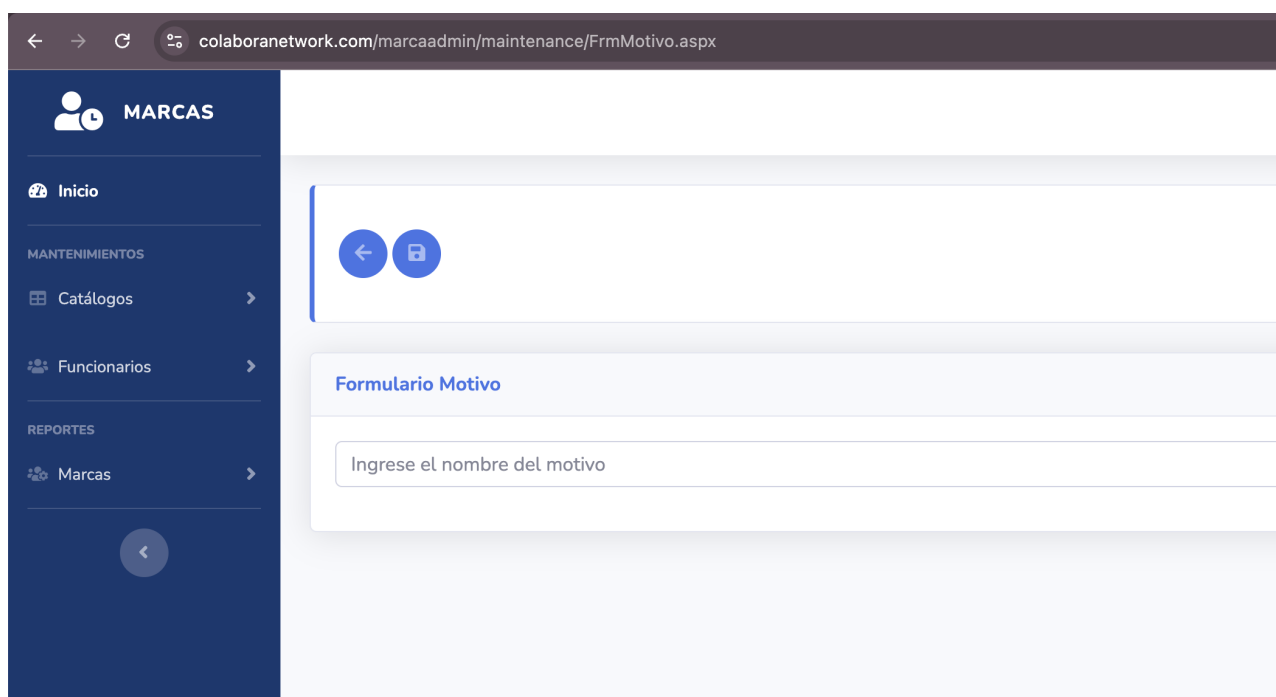
## Flujo de pantallas

Con respecto al flujo que deben seguir las pantallas de mantenimientos deben seguir una dinámica similar a la siguiente:

- Se debe mostrar un listado de elementos correspondientes al mantenimiento, esta será la pantalla principal del mantenimiento, a la cual se ingresa justo al seleccionar la opción de menú:

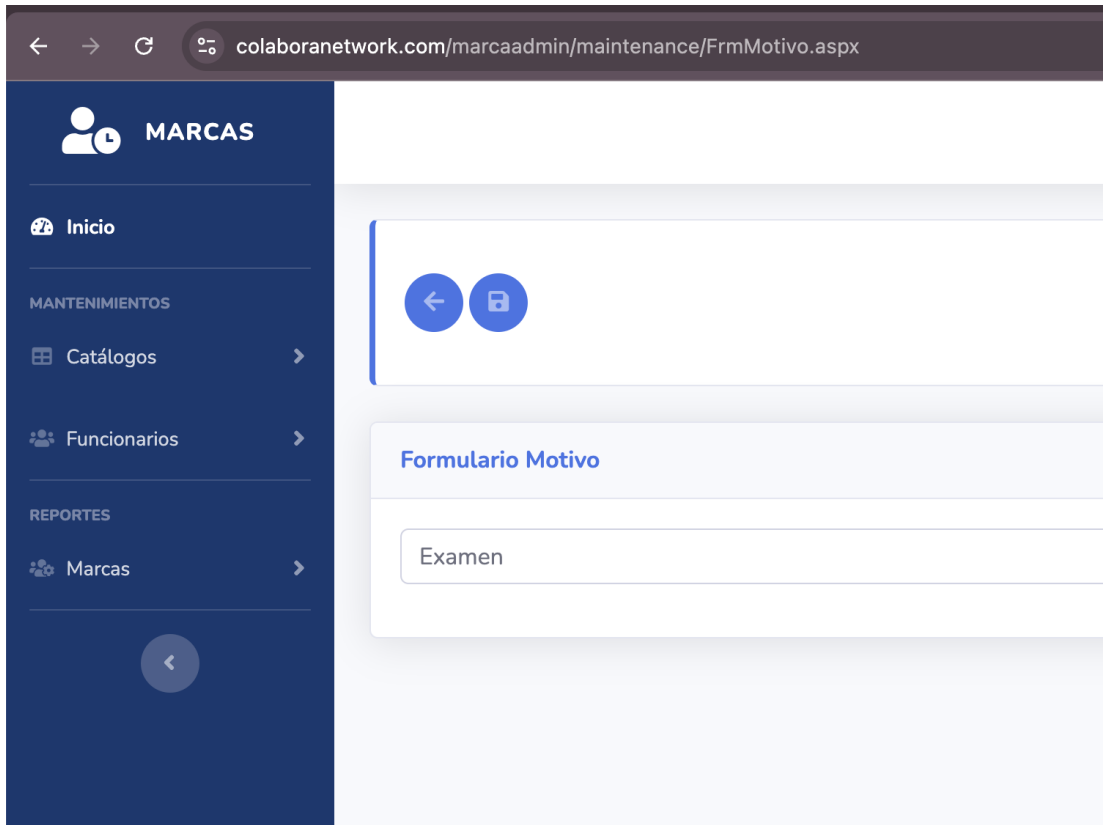


- El listado se debe mostrar siempre paginado, desplegando un máximo de 10 elementos por página.
- Para crear un nuevo elemento, debe existir un botón “Nuevo”, que, al presionarlo, lleva al formulario que permite ingresar los datos del nuevo elemento:



Este formulario además tendrá un botón para almacenar el nuevo elemento y uno para regresar al listado. Al usar el botón de guardar, si la operación es exitosa, debe regresar al listado y mostrar un mensaje de éxito, además, debe aparecer el nuevo elemento en el listado.

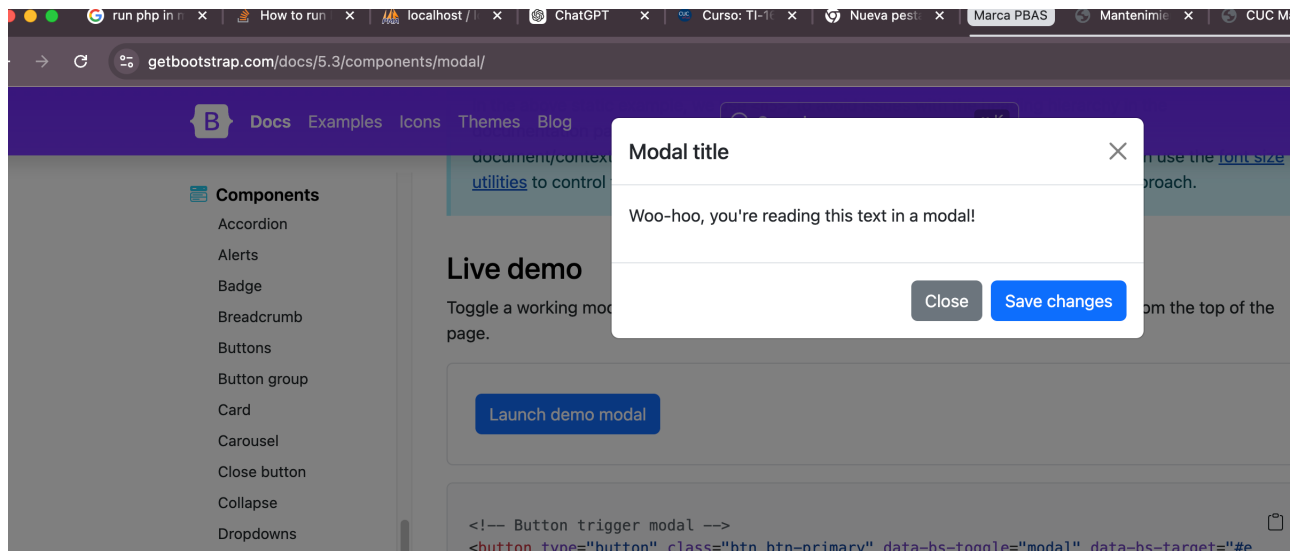
- Para editar cada fila del listado, debe tener un botón que ejecute esta acción, al presionarlo, debe redirigir a una pantalla similar a la de crear un nuevo elemento, pero con la información del elemento que se desea editar precargada, la cual podrá ser modificada y actualizada si se presiona el botón de salvar. De igual manera debe brindar la opción de regresar al listado principal.



- Finalmente, para eliminar, debe tener un botón que ejecute esta acción, al presionarlo, debe mostrar un diálogo de confirmación que indique: “¿Realmente desea eliminar el elemento seleccionado?” que indique dos opciones Si y No. Si selecciona “Si”, se debe eliminar el elemento.

Respecto a los mensajes al usuario y cuadros de diálogo se deben mostrar al usuario similar a como se muestra el diálogo a continuación:



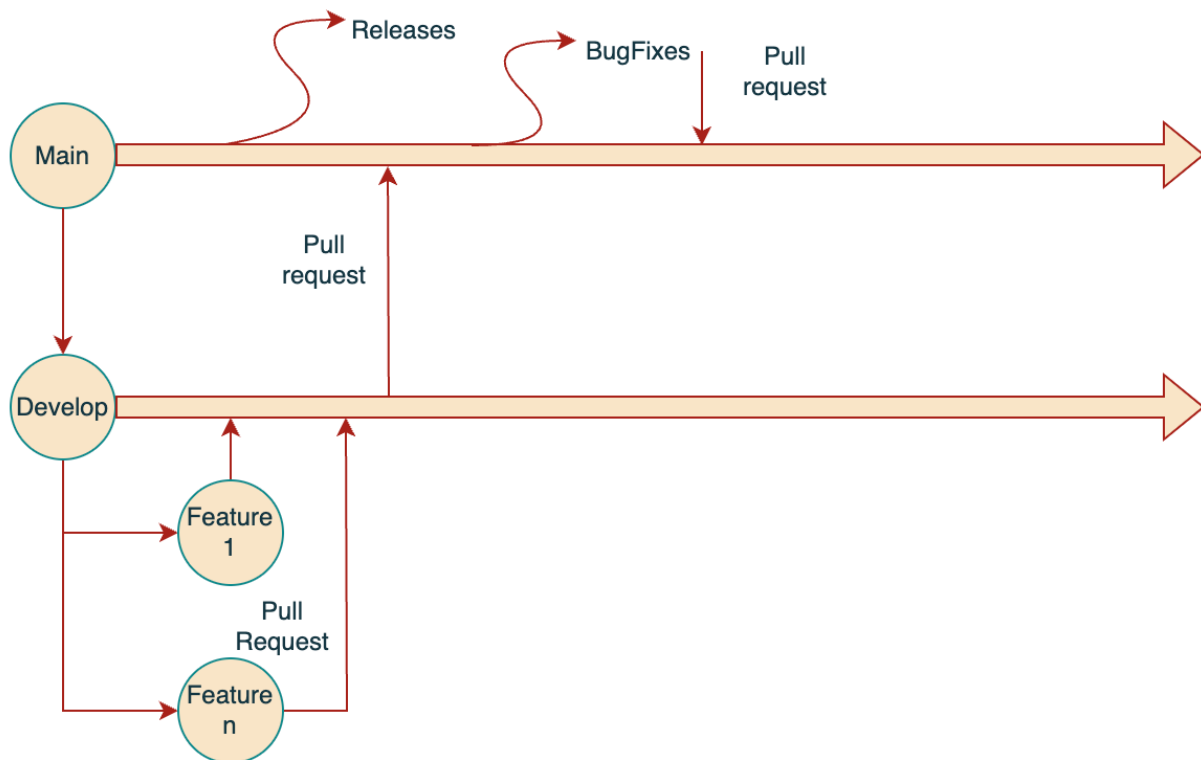


## Administración de las Historias de Usuario

Las historias de usuario se cargarán en proyectos de alguna herramienta de administración de historias de usuario y seguimiento como Azure DevOps o Gitlab.

## Control del código fuente

El código fuente se administrará en una herramienta de repositorio de control de código fuente. En este caso se propone usar repositorio GIT en la herramienta que se acuerde, durante el desarrollo del curso se valorará la mejor herramienta. Se tendrá la siguiente estructura de ramas:



## Creación y administración de cambios de la base de datos

La base de datos se tomará el tiempo de definirla en clases, para esto en la segunda lección los distintos equipos formados deberán traer todo el problema analizado y presentar su propuesta de la base de datos. Las mejores ideas de solución se tomarán para la base de datos que se usará en el proyecto.

Si posterior a la definición se necesita de un cambio justificado, debe gestionarse en el grupo de teams del curso, de ser aprobado el que propone creará un script para el cambio y solicitará al profesor aplicar el cambio. Todos los equipos que se vea afectados por este cambio deben ajustar su código fuente para que maneje apropiadamente el cambio.

## Tecnologías por utilizar

Para el segundo alcance las páginas se desarrollarán en tecnologías Web a elección del equipo de trabajo. Todos los miembros deben usar la misma tecnología y deben presentar un solo proyecto integrado.

## Entregables del proyecto

1. Documentación de análisis y diseño (enfocada en las HU que debe atender el equipo).
  - a. Portada.
  - b. Introducción (Resumen del problema a resolver).
  - c. Diagrama de base de datos (completo).
  - d. Diagramas de casos de uso.
  - e. Diagramas de clases.
  - f. Pruebas técnicas.
  - g. Conclusiones y recomendaciones.
  - h. Bibliografía.
2. Script de base de datos.
3. Código fuente de los servicios y pantallas desarrollados por el equipo (tomado de la rama de la HU atendida por el equipo de trabajo).
4. Código fuente de los servicios Rest completos (tomado de la rama integrada Develop con los cambios de todos los equipos de trabajo).
5. Historias de usuario actualizadas.
6. Pull Request hacia la rama Main (o rama en GIT donde se encuentra la versión final de su código).

Con respecto a la documentación de pruebas técnicas, en esta actividad debe ejecutar pruebas sobre el servicio de tal forma que se cumpla con las funcionalidades y criterios de aceptación establecidos. Cada criterio de aceptación debe generar una prueba y debe documentarse de la siguiente manera:

# HU	Criterio de aceptación	Evidencia
ID de la HU respectiva	Descripción del criterio que se probó	Pantallazo con la ejecución exitosa.

## Aspectos administrativos

1. El proyecto se elaborará en los equipos de trabajo definidos. La calificación será individual.
2. La fecha de entrega del segundo alcance será el jueves 10 de julio de 2025.
3. Valor del proyecto 15%.