

|  |
| --- |
| **Departamento: Tecnología** |
| **Responsables: Desarrollo Digital** |
| **Central Salamanca; España** |
| **Fecha: 13/09/2022** |

**Manual de Pruebas**

**– Directus 9 –**

**Components Module**

**ÍNDICE**

[1. – INTRODUCCIÓN 1](#_Toc113973154)

[2. – OBJETIVO 1](#_Toc113973155)

[3. – PRIMEROS PASOS 2](#_Toc113973156)

[3.1 – Instalación de Directus 2](#_Toc113973157)

[3.2 – Instalación de extensiones 2](#_Toc113973158)

[4. – CASOS DE USO 3](#_Toc113973159)

[5. – ADMINITRACIÓN DEL CONTENIDO 3](#_Toc113973160)

[5.1 – [CU-01] – Inserción del contenido 3](#_Toc113973161)

[5.2 – [CU-02] – Custom Module 3](#_Toc113973162)

[5.3 – [CU-03] – Estilización del módulo 4](#_Toc113973163)

# – INTRODUCCIÓN

Directus permite agregar nuestros propios módulos dentro de la aplicación. Los módulos personalizados ofrecen un lienzo en blanco para crear funcionalidades totalmente nuevas o diferentes, como conjuntos de datos compuestos o integraciones de terceros.

# – OBJETIVO

Intentando desarrollar una interfaz de usuario lo más amigable posible para el gestor de contenidos[[1]](#footnote-1), se propone la implementación de un nuevo módulo, para que pueda visualizar los componentes disponibles.

El objetivo del proyecto será que el GC pueda previsualizar los componentes antes de añadirlos a la aplicación. De esta manera se evitarán errores de inserción de componentes y se dinamizará el proceso de inclusión de contenido.

# – PRIMEROS PASOS

## – Instalación de Directus

* Crear el directorio del proyecto

mkdir <nombre del proyecto>

cd <nombre del proyecto>

* Instalar Directus 9

npm init directus-project <nombre del directorio de la api>

Durante el proceso de instalación hay que seleccionar la base de datos e introducir los datos de administración de la herramienta.

* Lanzar Directus 9

cd <ruta del directorio de la api>

npx directus start

Servidor lanzado en <http://localhost:8055>

## – Instalación de extensiones

Directus es modular, lo que permite la extensión con JavaScript personalizado. Usaremos la extensión de tipo ‘modules’ para crear nuestra propia página dentro de la aplicación. Aprovechando la interfaz de Directus extenderemos su funcionalidad.

* Crear la extensión

npm init directus-extension

Se solicitará el tipo de extensión que desearemos crear. Para nuestro proyecto usaremos una extensión de tipo hook.

? Choose the extensión type **modules**

? Choose a name for the extensión **preview-components**

? Choose the language to use **javascript**

Una vez finalizada la instalación se nos habrá creado una carpeta llamada **preview-components** en nuestro directorio raíz como proyecto independiente.

* Para llevar nuestro módulo a producción y ejecutar los cambios:

cd preview-components

npm run watch

# – CASOS DE USO

1. [CU-01] Inserción de las imágenes de los componentes del proyecto.
2. [CU-02] Creación de un nuevo módulo en Directus que permita la previsualización de los ficheros añadidos.
3. [CU-03] Estilizar el nuevo módulo creado.

# – ADMINITRACIÓN DEL CONTENIDO

En base a los componentes creados por el equipo de UX/UI, se crearán las imágenes asociadas a cada uno de ellos. Las imágenes se extraerán en formato .PNG directamente del editor de gráficos vectorial.

## – [CU-01] – Inserción del contenido

El proceso para la inserción de las imágenes será el siguiente:

* Recopilación de las imágenes a insertar.
* Desde la interfaz de Directus se creará una nueva carpeta para alojarlas:

**File Library > Create Folder**

* Posteriormente en dicha carpeta se irán agregando cada una de las imágenes:

**File Library > Create Item**

* Una vez añadidas, a cada una de ellas se le dará un título representativo del componente al que representa.

## – [CU-02] – Custom Module

Una vez creada la extensión, tendremos que desarrollar nuestro propio módulo en base a los requisitos descritos anteriormente.

* Haremos una llamada a la API de Directus para obtener las imágenes insertadas en la carpeta de componentes.

const response = await this.api.get(“/files?filter[folder][\_eq]=…&fields[]=id&fields[]=title”);

this.items = response.datsa.data;

* Pintaremos esas imágenes para que el GC pueda previsualizarlas a través de dicho módulo.

<v-list-item v-for=”item in items” :key=”item.id”>

<v-image :src”’http://localhost:8055/assets/’ + item.id” :alt=”item.title” />

</v-list-item>

## – [CU-03] – Estilización del módulo

Para que el módulo se integre correctamente y no rompa con el diseño de la interfaz de Directus, se han añadido algunos estilos a la extensión.

<template>

<private-view title="Componentes">

<div :style="styleMain">

<div :style="styleHeader">

<v-icon name="box" left :style="styleIcon" />

<h2 :style="styleH2">{{subtitulo}}</h2>

</div>

<v-list :style="styleGrid">

<v-list-item v-for="item in items" :key="item.id" :style="styleItem">

<div :style="styleCard">

<v-image :src="'http://localhost:8055/assets/' + item.id" :alt="item.title" :style="styleImage" />

</div>

<div :style="styleDetails">

<p :style="styleText">{{item.title}}</p>

<p :style="styleTextLight">Disponible</p>

</div>

</v-list-item>

</v-list>

</div>

</private-view>

</template>

1. **A partir de ahora al gestor de contenidos se le nombrará como GC** [↑](#footnote-ref-1)