# **App Adopción**

Crear una nueva aplicación npx create-react-app app-adopcion

Dejar solo el archivo index.js

En la carpeta public dejar solo el archivo html y borrar las importaciones

Para este ejercicio utilizaremos chakra css

<https://chakra-ui.com/docs/getting-started>

Instalar: npm i @chakra-ui/react @emotion/react@^11 @emotion/styled@^11 framer-motion@^4

Crear una carpeta ruutes y un archivo en su interior con el nombre de app

Interfaz de usuario gráfica, Aplicación, Sitio web

Descripción generada automáticamente

Para trabajar con chakra necesitamos realizar la configuración de un Provider, esta configuración la realizamos en el archivo app de routes

Copiar y pegar el contenido de la página oficial a app.js

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Organizar el archivo para poder usarlo

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamenteAgregar un componente de chakra, para este vamos a usar un botón



Agregar esta extensión a visual studio, permite generar rutas relativas mucho más fácil

Texto

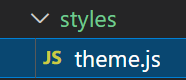
Descripción generada automáticamente

Chakra nos permite crear nuestros propios temas

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Crear una carpeta para los estilos



Ir a la url y extraer la información del tema personalizado

<https://github.com/Garcia091/CursoFrontend_2021/blob/main/Gits/theme.js>

Pegar en el archivo theme.js

Importar el archivo theme en App de routes

Este se va a enviar por parámetros al componente de chakra, como se ve en la imagen. Todo lo que estamos haciendo está en la documentación oficial

Texto

Descripción generada automáticamente

Si queremos agregar fuentes de Google fonts debemos instalar el siguiente comando

<https://chakra-ui.com/guides/using-fonts>

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

npm install @fontsource/nunito

una vez instalado, importar las fuentes

import "@fontsource/nunito/800.css";

import "@fontsource/nunito/700.css";

import "@fontsource/nunito/600.css";

import "@fontsource/nunito/400.css";

# **Agregar enrutamiento**

Instalar: npm i react-router-dom

# **Agregar Carpetas de trabajo**

Components = componentes de la página

Containers = También llamado Pages

* En la carpeta container crear un archivo AdopcionHome.jsx
* En AdopcionHome importar a chakra
* Crear una etiqueta center, box y tex
* En box vamos a enviar props con estilos:

<https://chakra-ui.com/docs/features/style-props>

margin-top= 400

Font-size= 32px

Padding = 4

Color = Blanca

Texto

Descripción generada automáticamente

En el archivo App de routes, importar el container de AdopcionHome

Texto

Descripción generada automáticamente

# **Elementos de Enrutamiento En App**



Cuando usamos Switch el nos va a validar que la url no muestre información de componentes que no corresponde

Texto

Descripción generada automáticamente

Texto, Patrón de fondo

Descripción generada automáticamente

Crear la interfaz de la aplicación de la página principal

Texto

Descripción generada automáticamente

Usamos un Flex para dar dirección de nuestros elementos en columnas, con margen top de 5, alineamos los elementos al centro.

Tenemos una cabecera para el titulo (heading) con margin top y margin bottom

El texto le vamos a dar una margin top y un estilo de encabezado de tipo h3.

## **Organizar los elementos de una pila: Stack**

<https://chakra-ui.com/docs/layout/stack#notes-on-stack-vs-flex>

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Utilizar el VStack:

Texto

Descripción generada automáticamente

Al interior de la pila, ponemos dos cabeceras para simular la información que llega de dos componentes diferentes.

Texto

Descripción generada automáticamente

Resultado:

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

La línea blanca entre los elementos del VStack es el divider

* Crear los componentes [CategorieItem.jsx](https://github.com/JohnFlorez25/adoption-app/blob/master/src/components/CategorieItem.jsx) y [CategorieList.jsx](https://github.com/JohnFlorez25/adoption-app/blob/master/src/components/CategorieList.jsx) para mostrar su respectiva información.
* Importar Chakra
* Importar las etiquetas necesarias.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Importar los componentes y usarlo en AdopcionHome

Se debe mostrar la misma información

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

**WRAP:**

Wrap es un componente de diseño que agrega un espacio definido entre sus hijos. Envuelve a sus hijos automáticamente si no hay suficiente espacio para caber más en la misma fila.

Piense en él como un más inteligente flex-wrapcon spacingsoporte. Funciona muy bien con cosas como botones de diálogo, etiquetas y chips.

Interfaz de usuario gráfica, Sitio web

Descripción generada automáticamente

El Wrap la vamos a usar para mostrar las categorías.

Para estas categorías vamos a usar una estructura ya consolidada

Categorías: <https://github.com/Garcia091/CursoFrontend_2021/blob/main/Gits/categorias.js>

Lista: <https://github.com/Garcia091/CursoFrontend_2021/blob/main/Gits/CategorieList.js>

Componente Categorías Items

Texto

Descripción generada automáticamente

Componente Categorías Lista

Texto

Descripción generada automáticamente

Crear la Api Rest Falsa para consumir la información de las mascotas.