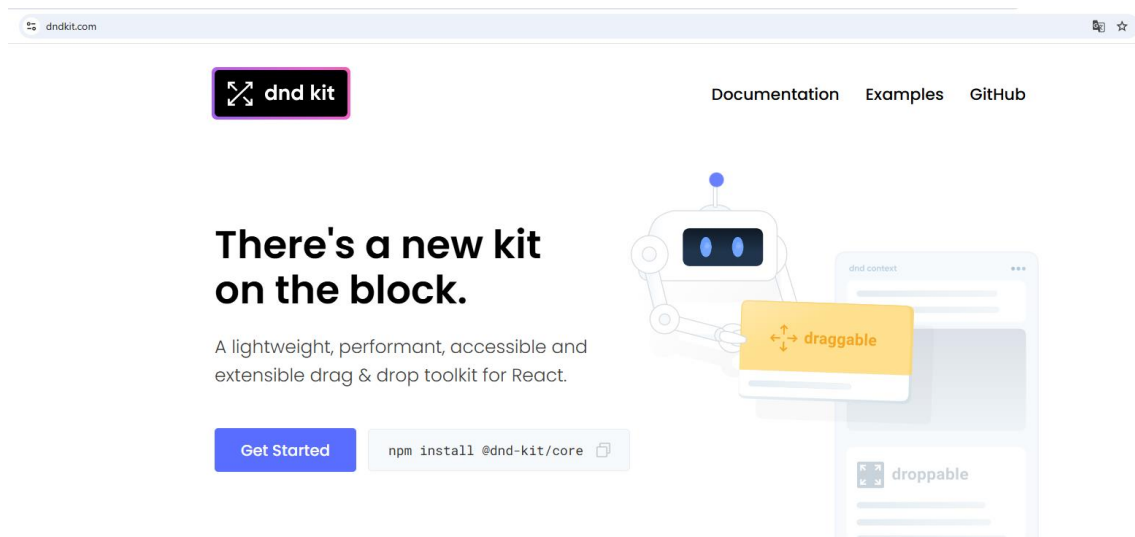


Añadiendo Orden a tus Enlaces

Vamos a ver cómo preparar todo para incorporar la librería de Drag and Drop. Estaremos usando “dnd kit”.

dndkit.com



Pero antes, para que no almacene en nuestra bd enlaces no válidos, hacemos este último ajuste quitando ese queryClient.

```

LinkTreeView.tsx | AppLayout.tsx | DevTree.tsx | DevTreeLink.tsx | DevTreeInput.tsx | TS DevTreeAPI.ts | TS index.ts
frontend > src > views > LinkTreeView.tsx | LinkTreeView | handleEnableLink | updatedLinks | devTreeLinks.map() callback
10 export default function LinkTreeView() {
28   useEffect(() => {
33     const updatedData = devTreeLinks.map(item => {
37       if (userlink) {
38         return { ...item, url: userlink.url, enabled: userlink.enabled }
39       }
40       return item
41     })
42     setDevTreeLinks(updatedData)
43   }, [])
44
45   const handleUrlChange = (e : React.ChangeEvent<HTMLInputElement>) => {
46
47     // Mapea cada elemento del estado actual
48     // Si el nombre del enlace coincide con el del input que generó el evento,
49     // actualiza su propiedad 'url' con el nuevo valor ingresado
50     const updatedLinks = devTreeLinks.map(link => link.name === e.target.name ? { ...link, url: e.target.value } : link)
51
52     //console.log(updatedLinks)
53
54     // Actualiza el estado con la nueva lista de enlaces
55     setDevTreeLinks(updatedLinks)
56
57     queryClient.setQueryData(['user'], (prevData: User) => {
58       return {
59         ...prevData,
60         links: JSON.stringify(updatedLinks) //estos links son un arreglo
61       }
62     })
63   }
64 }
65
66 const handleEnableLink = (socialNetwork: string) => {
67   //console.log(socialNetwork)
68   //const updatedLinks = devTreeLinks.map(link => link.name === socialNetwork ? { ...link, enabled: link.enabled ? !link

```

Bien, ahora para que funcione el Drag and drop, las redes deben tener un ID único.

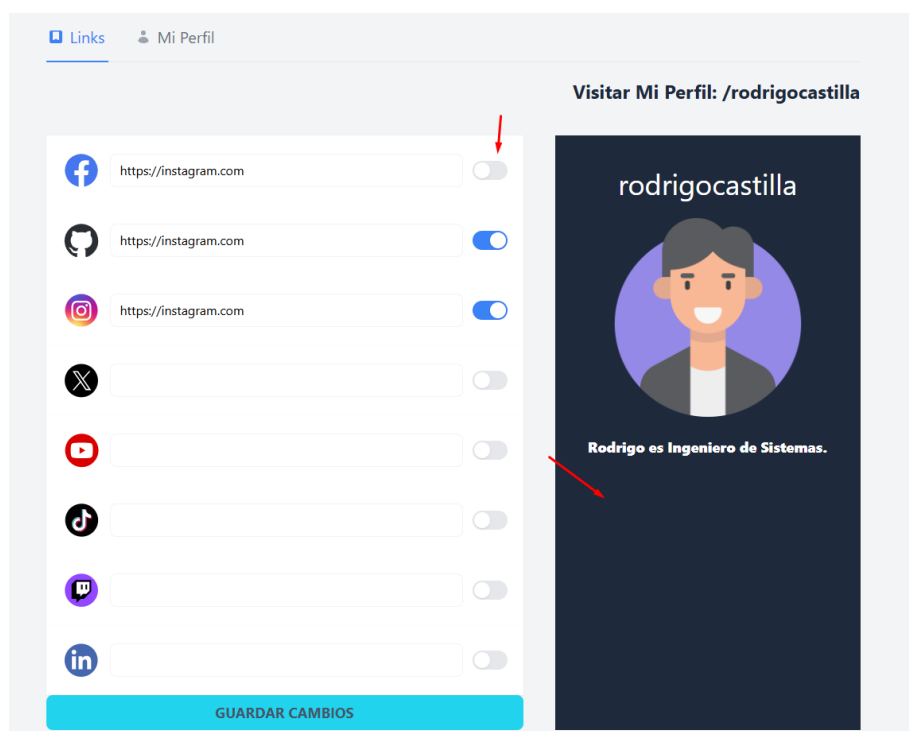
Añadiendo un Orden a cada enlace

```

LinkTreeView.tsx • AppLayout.tsx • DevTree.tsx • DevTreeLink.tsx • DevTreeInput.tsx • TS DevTreeAPI.ts • TS index.ts
frontend > src > views > LinkTreeView.tsx > LinkTreeView
10 export default function LinkTreeView(){
60   const links : SocialNetwork[] = JSON.parse(user.links)
61
62   const handleEnableLink = (socialNetwork: string) => {
63     //console.log(socialNetwork)
64     //const updatedLinks = devTreeLinks.map(link => link.name === socialNetwork ? {...link, enabled: !link.enabled} : link)
65
66     const updatedLinks = devTreeLinks.map(link => {
67       if(link.name === socialNetwork){
68         if(isValidUrl(link.url)){
69           return {...link, enabled: !link.enabled}
70         }else{
71           toast.error('URL no válida')
72         }
73       }
74       return link
75     })
76
77     //console.log(updatedLinks)
78     setDevTreeLinks(updatedLinks)
79
80     let updatedItems: SocialNetwork[] = []
81
82     const selectedSocialNetwork = updatedLinks.find(link => link.name === socialNetwork)
83     if(selectedSocialNetwork?.enabled){
84
85       const newItem = {
86         ...selectedSocialNetwork,
87         id: links.length + 1
88       }
89       updatedItems = [...links, newItem]
90     }else{
91       console.log('Deshabilitando')
92     }
93
94
95     queryClient.setQueryData(['user'], (prevData: User)=>{
96       return {
97         ...prevData,
98         links: JSON.stringify(updatedItems) //estos links son un arreglo
99       }
100     })

```

Probamos y al parecer todo está bien, pero si deshabilito una red social como facebook, se desaparece todo de la sección del perfil.



Esto pasa porque cada vez que se ejecuta un enlace `updatedItems` queda como un arreglo vacío y se almacena en la BD, hay que trabajar un poco con la parte de deshabilitar.

Actualizando el orden al habilitar y deshabilitar redes sociales

```

10  export default function LinkTreeView(){
62  const handleEnableLink = (socialNetwork: string) => {
77      //console.log(updatedLinks)
78      setDevTreeLinks(updatedLinks)
79
80      let updatedItems: SocialNetwork[] = []
81
82      const selectedSocialNetwork = updatedLinks.find(link => link.name === socialNetwork)
83      if(selectedSocialNetwork?.enabled){
84
85          const newItem = {
86              ...selectedSocialNetwork,
87              id: links.length + 1
88          }
89          updatedItems = [...links, newItem]
90      }else{
91          updatedItems = links.filter(link => link.name !== socialNetwork)
92      }
93
94
95      queryClient.setQueryData(['user'], (prevData: User)=>{
96          return {
97              ...prevData,
98              links: JSON.stringify(updatedItems) //estos links son un arreglo
99          }
100      })
101  }
  
```

Pero, ahora tenemos un problema con los ID's (de orden) de las redes sociales.

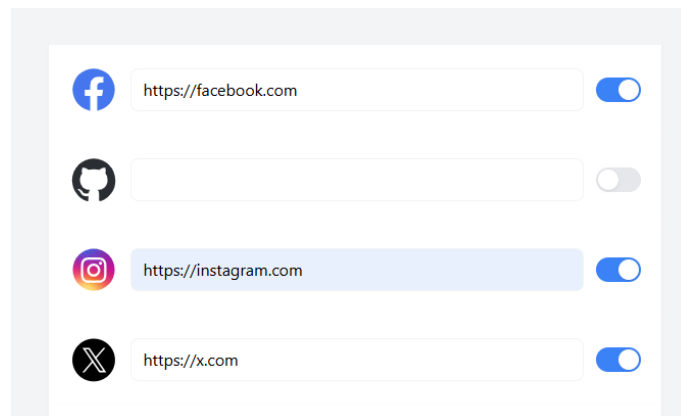
Evitar ID's duplicados.

```

LinkTreeView.tsx x  AppLayout.tsx  DevTree.tsx  DevTreeLink.tsx  DevTreeInput.tsx  TS DevTreeAPI.ts  TS index.ts
frontend > src > views > LinkTreeView.tsx > LinkTreeView > handleEnableLink > links.map() callback
10  export default function LinkTreeView(){
62    const handleEnableLink = (socialNetwork: string) => {
66      const updatedLinks = devTreeLinks.map(link => {
73        }
74        return link
75      })
76
77      //console.log(updatedLinks)
78      setDevTreeLinks(updatedLinks)
79
80      let updatedItems: SocialNetwork[] = []
81
82      const selectedSocialNetwork = updatedLinks.find(link => link.name === socialNetwork)
83      if(selectedSocialNetwork?.enabled){
84
85        const newItem = {
86          ...selectedSocialNetwork,
87          id: links.length + 1
88        }
89        updatedItems = [...links, newItem]
90      }else{
91        const indexToUpdate = links.findIndex(link => link.name === socialNetwork)
92        updatedItems = links.map(link =>{
93          if(link.name === socialNetwork){
94            return {
95              ...link,
96              id: 0,
97              enabled: false
98            }
99          }else if(link.id > indexToUpdate){
100            return {
101              ...link,
102              id: link.id - 1
103            }
104          } else{
105            return link
106          }
107        })
108      }
109    }

```

En consola, ya tenemos cada red social con su ID.



```

LinkTreeView.tsx:
(3) [{...}, {...}, {...}]
  ▶ 0: {name: 'facebook', url: 'https://facebook.com', enabled: true, id: 1}
  ▶ 1: {name: 'instagram', url: 'https://instagram.com', enabled: true, id: 2}
  ▶ 2: {name: 'x', url: 'https://x.com', enabled: true, id: 3}
  length: 3

```

Si retiro Instagram:

```

[object Object]                                AppLayout.tsx:23
▼ (3) [{...}, {...}, {...}] ⓘ                  LinkTreeView.tsx:16
  ▼ 0:
    enabled: true
    id: 1
    name: "facebook"
    url: "https://facebook.com"
    ▶ [[Prototype]]: Object
  ▼ 1:
    enabled: false
    id: 0
    name: "instagram"
    url: "https://instagram.com"
    ▶ [[Prototype]]: Object
  ▼ 2:
    enabled: true
    id: 2
    name: "x"
    url: "https://x.com"
    ▶ [[Prototype]]: Object
    length: 3
    ▶ [[Prototype]]: Array(0)
[object Object],[object Object],[object Object] LinkTreeView.tsx:16

```

El ID de Instagram pasa a ser 0 (y false), el ID de Facebook pasa a ser 1, y el ID de X pasa a ser 2.

Hasta ahí, todo bien. Sin embargo, si quitamos y agregamos una red social, se va sumando su ID como si fuera un bucle:

```

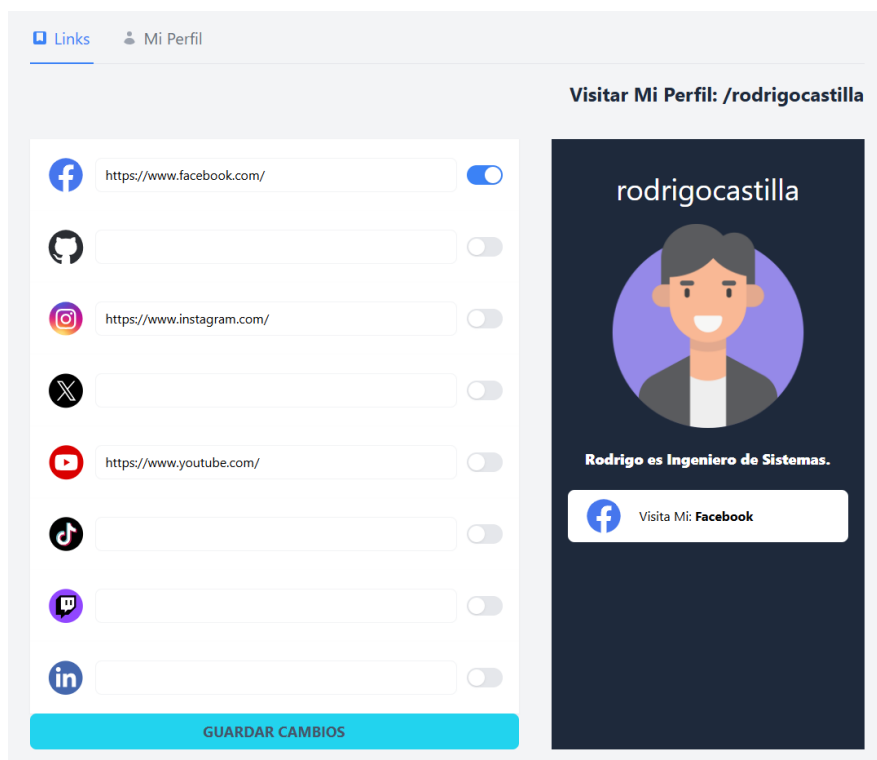
[object Object]                                AppLayout.tsx:23
▶ (3) [{-}, {-}, {-}] ←                        LinkTreeView.tsx:16
[object Object],[object Object],[object Object] LinkTreeView.tsx:16
{ id: '6865c0557bd9e2eb63e5e304', handle: 'rodrigocastilla', name: 'Rodrigo Castil
▶ la', email: 'rodrigocastilla@gmail.com', description: 'Rodrigo es Ingeniero de Sis
temas.', ...}
[object Object]                                AppLayout.tsx:23
▶ (4) [{-}, {-}, {-}, {-}] ←                  LinkTreeView.tsx:16
[object Object],[object Object],[object Object],[object Object] LinkTreeView.tsx:16
{ id: '6865c0557bd9e2eb63e5e304', handle: 'rodrigocastilla', name: 'Rodrigo Castil
▶ la', email: 'rodrigocastilla@gmail.com', description: 'Rodrigo es Ingeniero de Sis
temas.', ...}
[object Object]                                AppLayout.tsx:23
▶ (4) [{-}, {-}, {-}, {-}] ←                  LinkTreeView.tsx:16
[object Object],[object Object],[object Object],[object Object] LinkTreeView.tsx:16
{ id: '6865c0557bd9e2eb63e5e304', handle: 'rodrigocastilla', name: 'Rodrigo Castil
▶ la', email: 'rodrigocastilla@gmail.com', description: 'Rodrigo es Ingeniero de Sis
temas.', ...}
[object Object]                                AppLayout.tsx:23
▶ (5) [{-}, {-}, {-}, {-}, {-}] ←             LinkTreeView.tsx:16
[object Object],[object Object],[object Object],[object Object],[object Object] LinkTreeView.tsx:16
AppLayout.tsx:23

```

```

LinkTreeView.tsx | Applayout.tsx | DevTree.tsx | DevTreeLink.tsx | DevTreeInput.tsx | TS DevTreeAPI.ts | TS index.ts
frontend > src > views > LinkTreeView.tsx > LinkTreeView > handleEnableLink
10  export default function LinkTreeView(){
62    const handleEnableLink = (socialNetwork: string) => {
66      const updatedLinks = devTreeLinks.map(link => {
75      })
76
77      //console.log(updatedLinks)
78      setDevTreeLinks(updatedLinks)
79
80      let updatedItems: SocialNetwork[] = []
81
82      const selectedSocialNetwork = updatedLinks.find(link => link.name === socialNetwork)
83      if(selectedSocialNetwork?.enabled){
84
85        const id = links.filter(link => link.id).length + 1
86
87        if(links.some(link => link.name === socialNetwork)){
88          updatedItems = links.map(link=>{
89            if(link.name===socialNetwork){
90              return {
91                ...link,
92                enabled: true,
93                id
94              }
95            }else{
96              return link
97            }
98          })
99        }else{
100          const newItem = {
101            ...selectedSocialNetwork,
102            id
103          }
104          updatedItems = [...links, newItem]
105        }
106      }
107    }
108    const indexToUpdate = links.findIndex(link => link.name === socialNetwork)

```



Con ello, ya podemos guardar los cambios y persistirán cuando recarguemos la página. Además, en consola, podemos validar que los ID's se van asignando correctamente si habilito o deshabilito.