

Proyecto Full stack

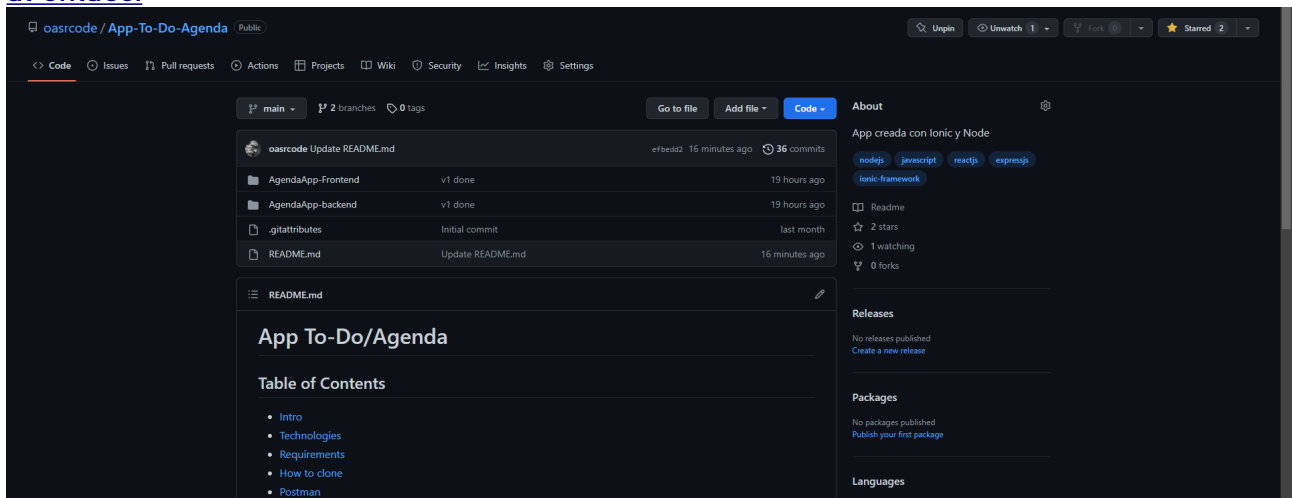
App To Do/ Agenda con Ionic

Índice

1.Capturas Github.....	3
2.Capturas Postman.....	3
3.CRUD ToDos.....	4
4.CRUD Photos.....	6
5.CRUD User.....	8
6.CRUD desde IONIC.....	9
7.Componentes de Ionic.....	12

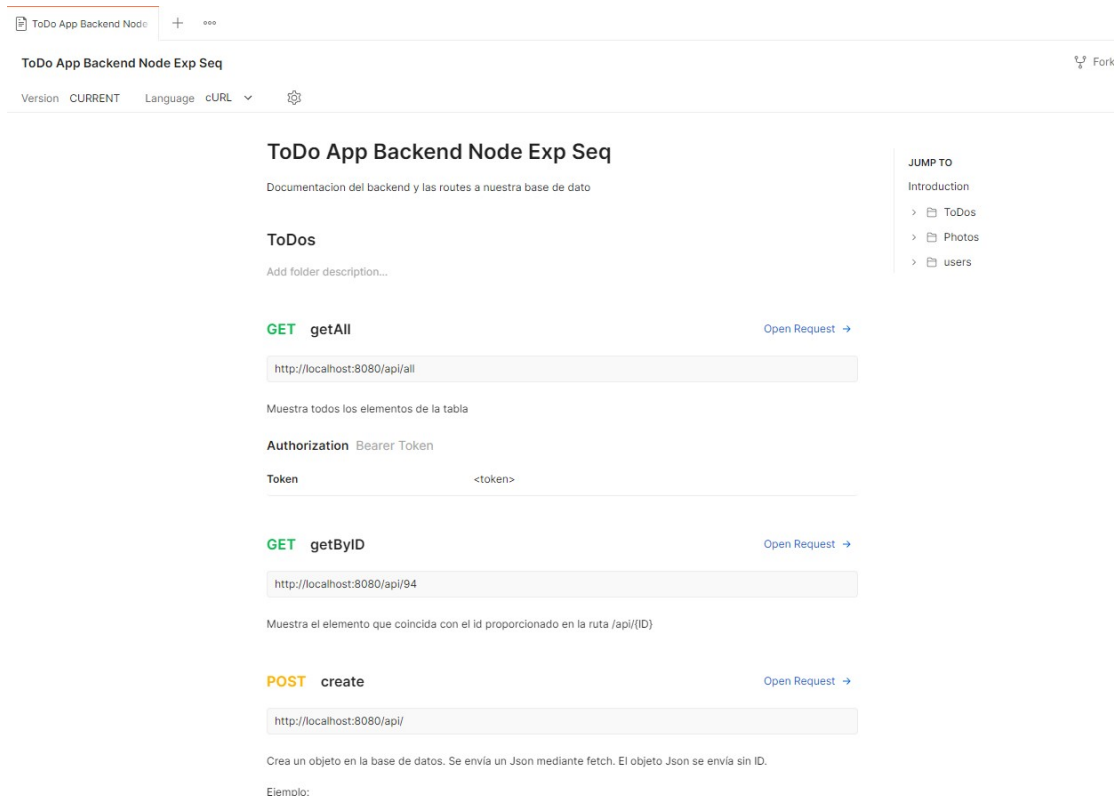
1.Capturas Github

Capturas del repositorio github, donde se el proyectos subido con su readme. [Clic para ir al enlace.](#)

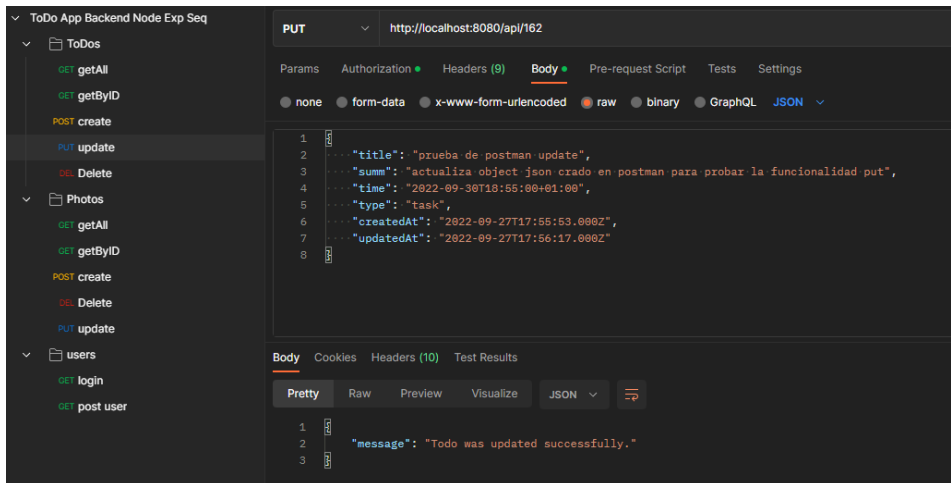


2.Capturas Postman

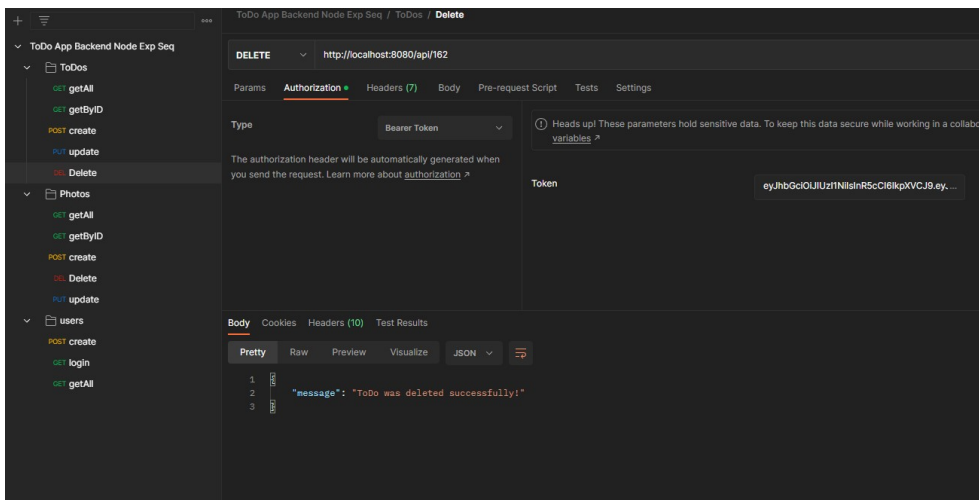
[Enlace de postman donde están todas la request al backend.](#)



Put



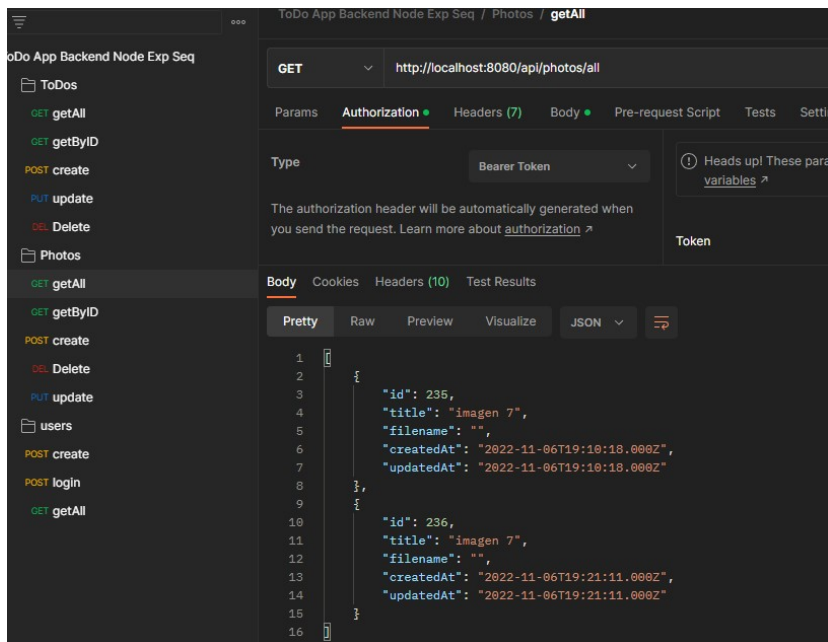
Delete



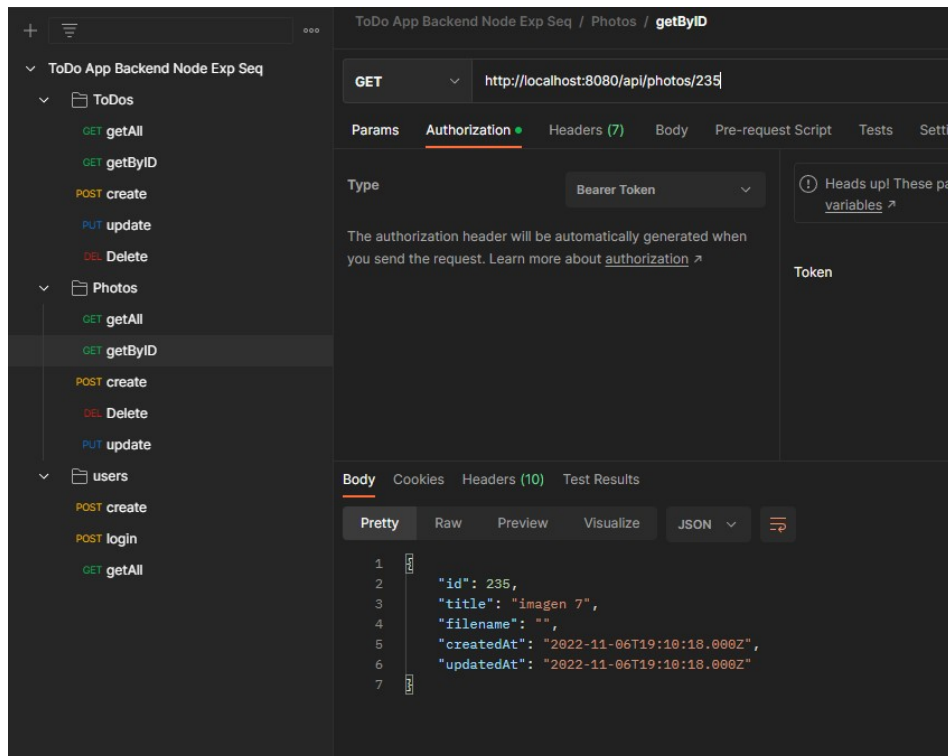
4.CRUD Photos

Photos usa bearer tokens para poder realizar el crud.

Get All



Get ID



Post

The screenshot shows the Postman interface for a POST request. The left sidebar lists the API collection 'ToDo App Backend Node Exp Seq' with folders 'Todos', 'Photos', and 'users'. The 'Photos' folder is expanded, showing a 'POST create' request. The main panel displays the request details for 'POST' to 'http://localhost:8080/api/photos/'. The 'Body' tab is selected, showing 'form-data' as the content type. The body contains two key-value pairs: 'title' with value 'Imagen 7' and 'file' with value 'Sea-wave-Wallpaper-300x585.jpg'. The 'Test Results' tab shows a successful response with a 200 status code and a JSON body:

```
{  "id": 236,  "title": "Imagen 7",  "filename": "",  "updatedAt": "2022-11-06T19:21:11.486Z",  "createdAt": "2022-11-06T19:21:11.486Z"}
```

Put

The screenshot shows the Postman interface for a PUT request. The left sidebar lists the API collection 'ToDo App Backend Node Exp Seq' with folders 'Todos', 'Photos', and 'users'. The 'Photos' folder is expanded, showing a 'PUT update' request. The main panel displays the request details for 'PUT' to 'http://localhost:8080/api/photos/235'. The 'Body' tab is selected, showing 'form-data' as the content type. The body contains two key-value pairs: 'title' with value '22323' and 'file' with value 'www.mobilesmask.net_nature_4460.jpg'. The 'Test Results' tab shows a successful response with a 200 status code and a JSON body:

```
{  "message": "Photo was updated successfully."}
```

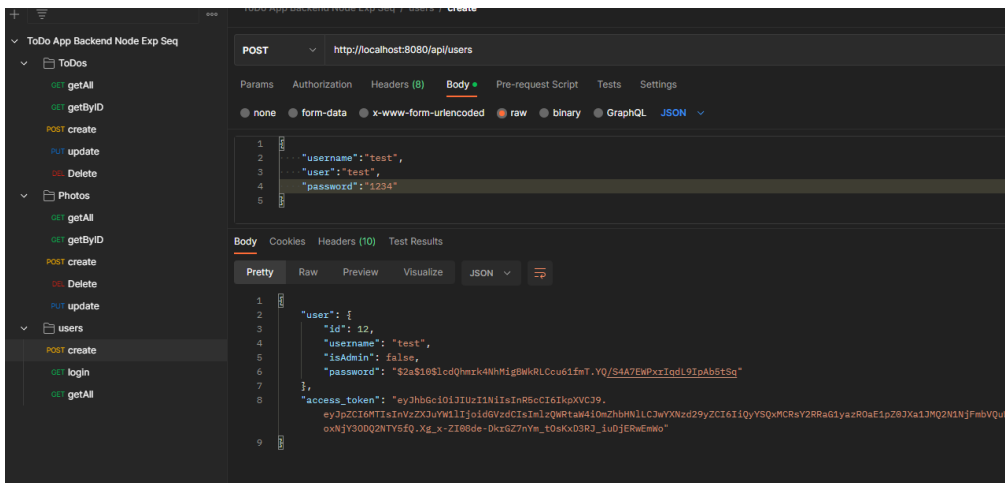
Delete

The screenshot shows the Postman interface for a DELETE request. The left sidebar lists the API collection 'ToDo App Backend Node Exp Seq' with folders 'Todos', 'Photos', and 'users'. The 'Photos' folder is expanded, showing a 'DELETE Delete' request. The main panel displays the request details for 'DELETE' to 'http://localhost:8080/api/photos/235'. The 'Authorization' tab is selected, showing 'Bearer Token' as the authorization type. The 'Test Results' tab shows a successful response with a 200 status code and a JSON body:

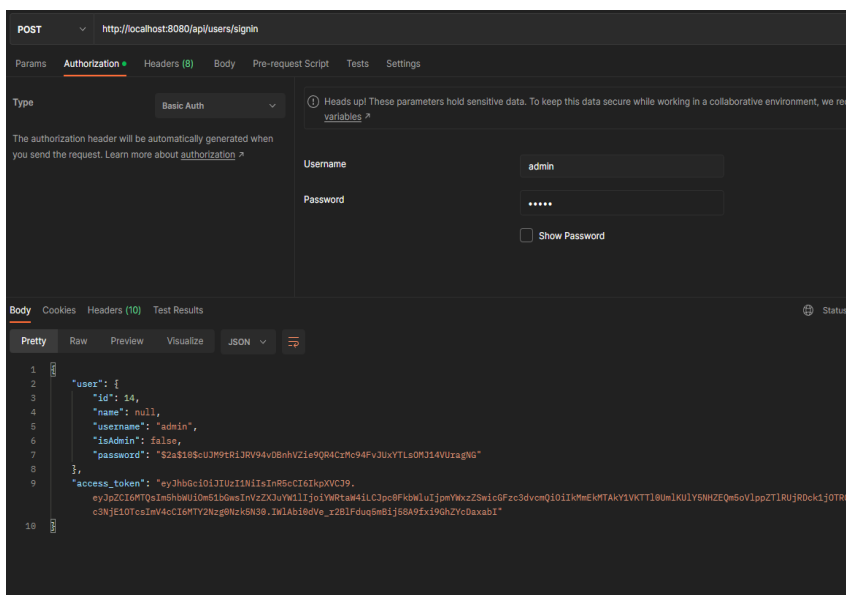
```
{  "message": "Photo was deleted successfully!"}
```

5.CRUD User

Post



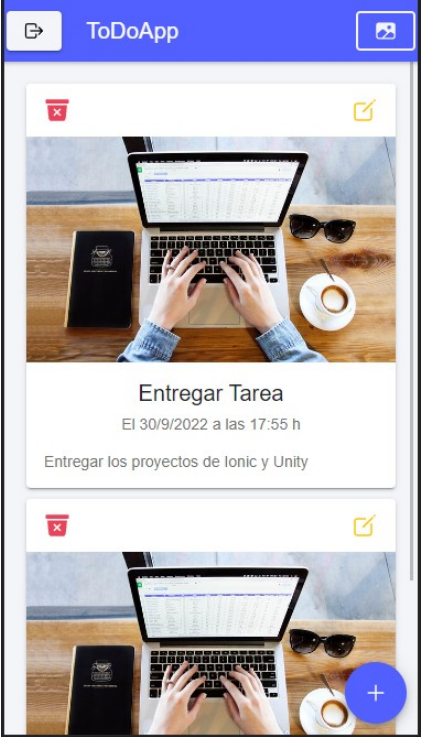
SignIn



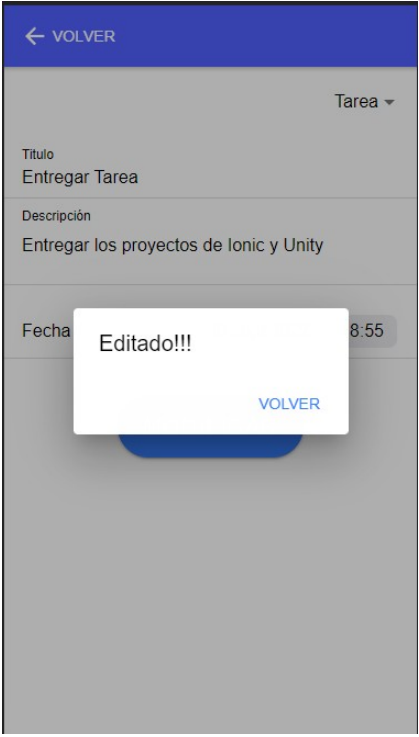
6.CRUD desde IONIC

ToDoS

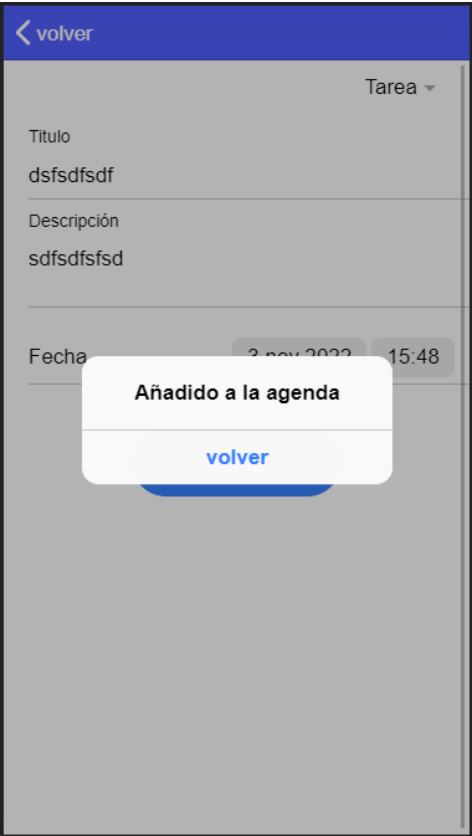
Get All



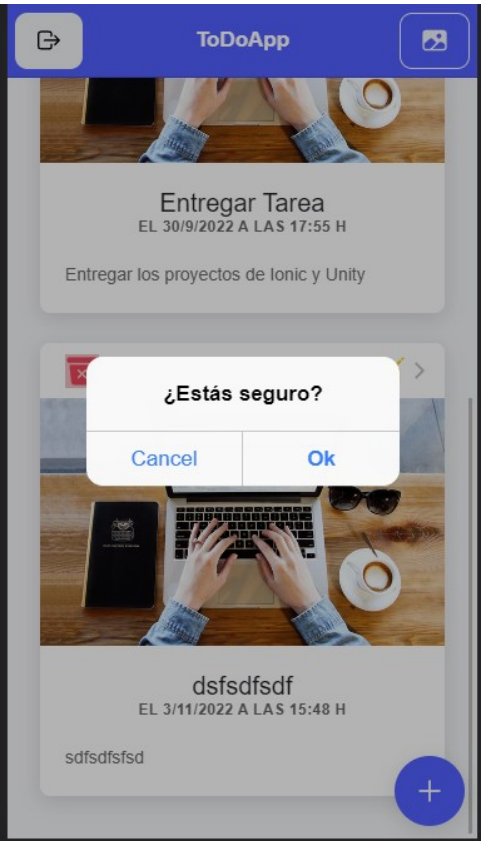
Edit



Create

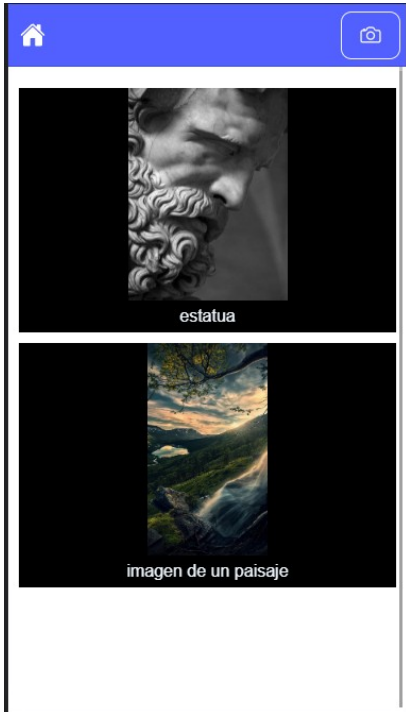


Delete

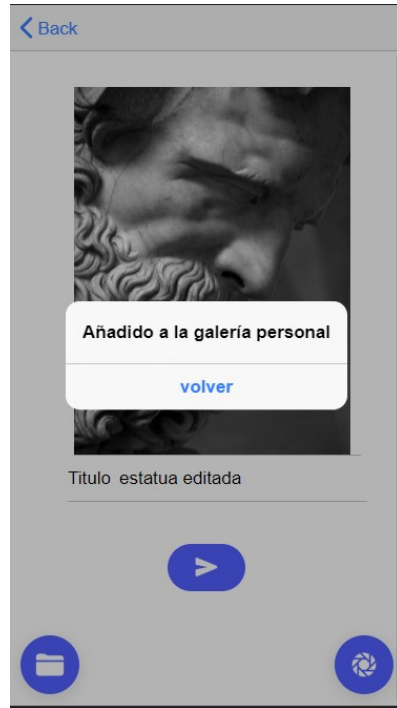


Photos

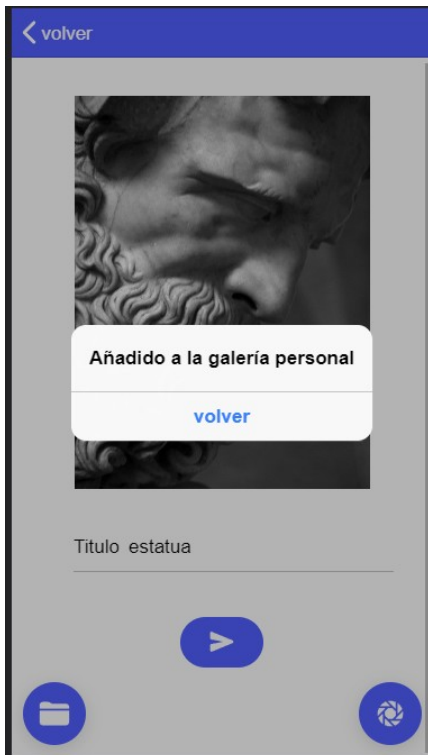
Get All



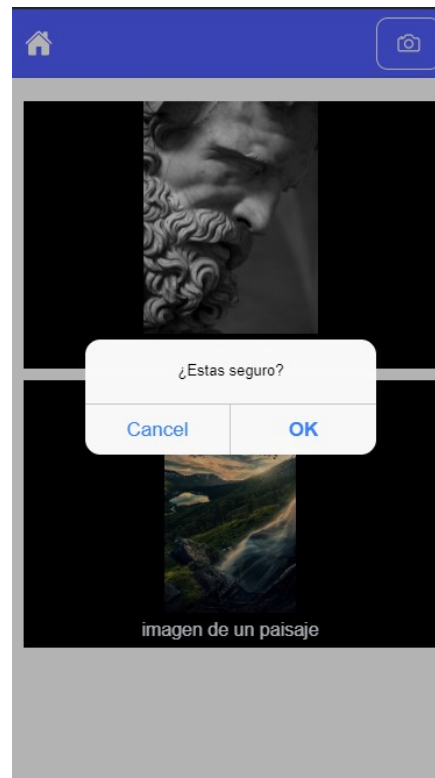
Edit: clic sobre la foto para ver el menú



Create

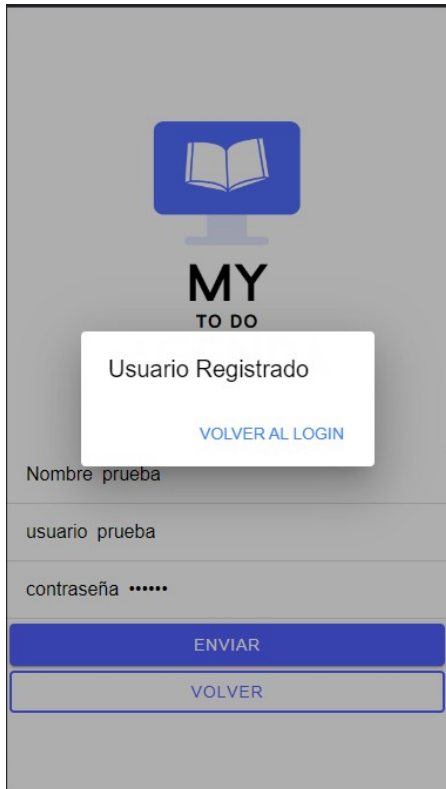


Delete:clic sobre la foto para ver el menú



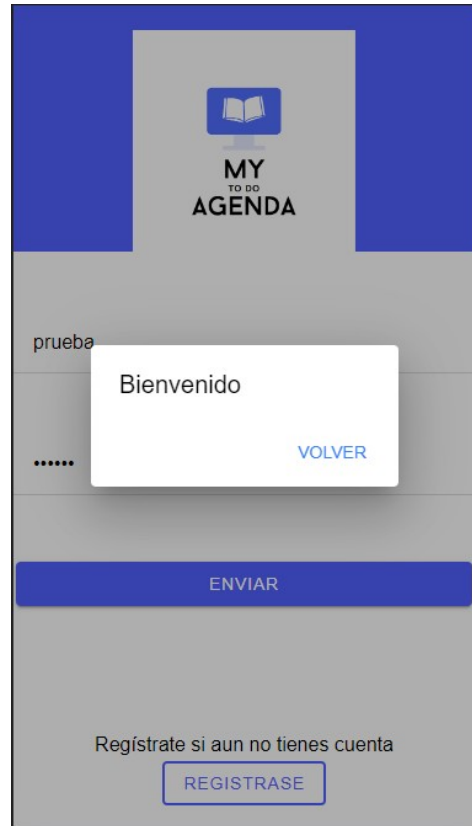
User

Registrar usuario



The registration form features a logo at the top consisting of a blue square with a white open book icon, with the text "MY TO DO" below it. A white modal box is centered over the form, displaying "Usuario Registrado" and a blue link "VOLVER AL LOGIN". The form itself has three input fields: "Nombre prueba", "usuario prueba", and "contraseña *****". Below these fields are two buttons: a blue "ENVIAR" button and a grey "VOLVER" button.

Login Usuario



The login form features a logo at the top consisting of a blue square with a white open book icon, with the text "MY TO DO AGENDA" below it. A white modal box is centered over the form, displaying "Bienvenido" and a blue link "VOLVER". The form has two input fields: "prueba" and ".....". Below these fields is a blue "ENVIAR" button. At the bottom of the form, there is a link "Regístrate si aun no tienes cuenta" and a blue "REGISTRASE" button.

7.Componentes de Ionic

IonHeader IonToolbar IonTitle

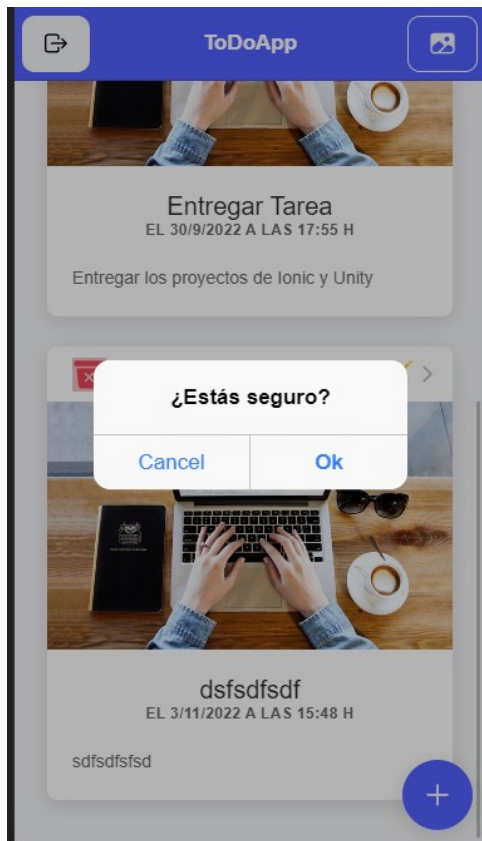


**IonButton
IonIcon**

**IonCard
IonCardHeader
IonCardTitle
IonCardSubtitle
IonImage
IonContentCard
IonGrid**

IonFab

IonAlert



IonActionShet



← VOLVER

Elige el tipo ▾

Titulo

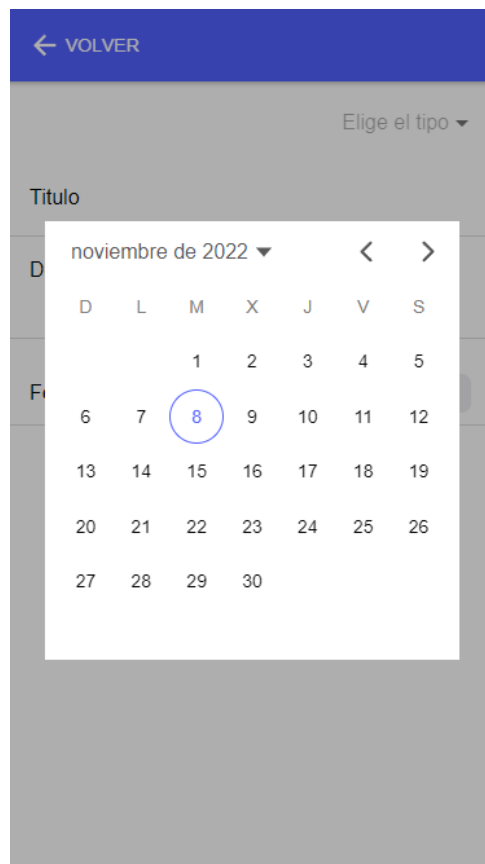
Descripción

Fecha

8 nov 2022

19:36

CREAR



IonItem
IonSelect
IonSelectOption
IonDatetimeButton
IonImage
IonContentCard

7. Autenticación

Nos creamos un usuario en la página de registrarse, hacemos login escribiendo el usuario y la contraseña. Si es correcto, haciendo uso de Storage, este la guardará en la base de datos indexada en el navegador. Yo parseo a string el json que me devuelve la api, usuario y token, y lo almaceno. La idea es que, en un futuro, filtrar la agenda y las fotos por id de usuario utilizando el json parseado en la base de datos.

