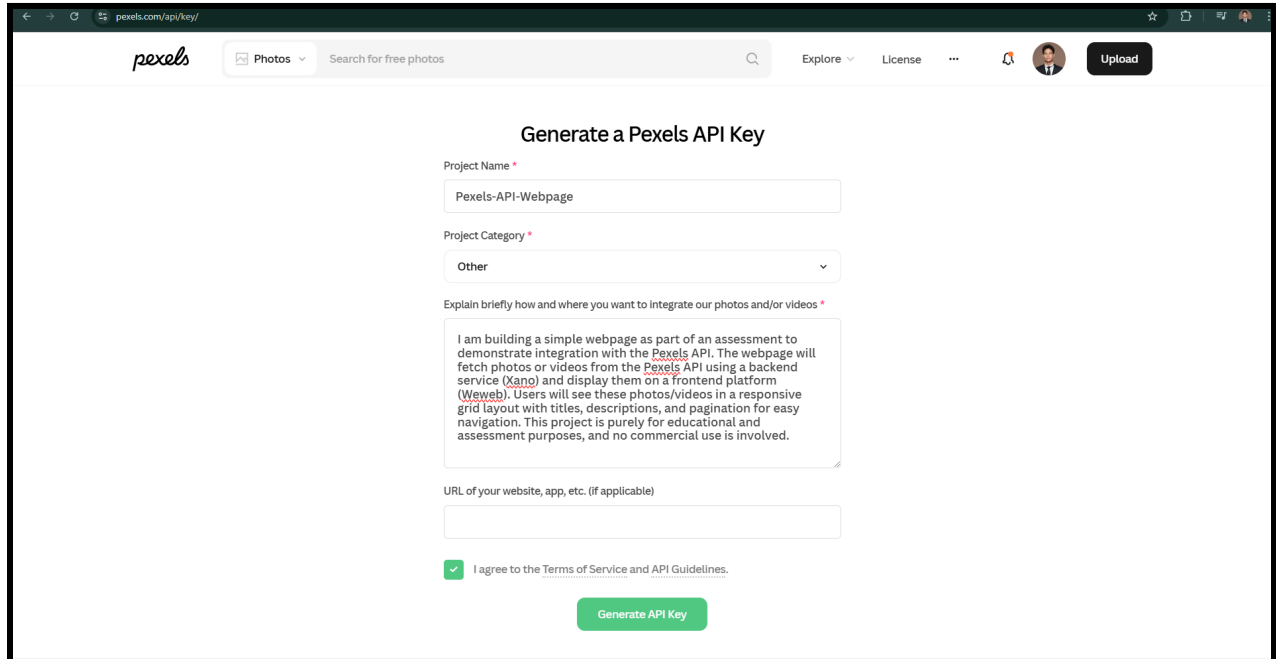


Documentation

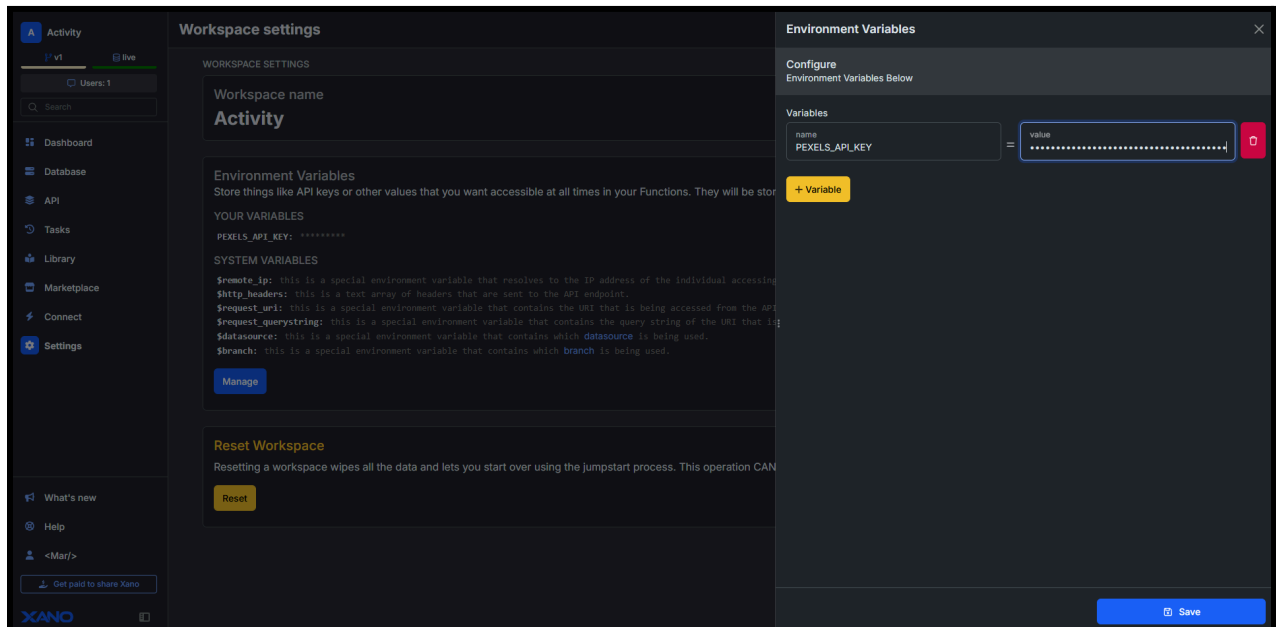
1. Created accounts in Xano, WeWeb, and Pexels.
2. In Pexels, I obtained the API Key to be passed into Xano.



The screenshot shows the 'Generate a Pexels API Key' page on the Pexels website. The page has a white background with a dark header. The header includes the Pexels logo, a search bar, and navigation links like 'Explore', 'License', and 'Upload'. The main content area is titled 'Generate a Pexels API Key' and contains the following fields:

- Project Name ***: A text input field containing 'Pexels-API-Webpage'.
- Project Category ***: A dropdown menu with 'Other' selected.
- Explain briefly how and where you want to integrate our photos and/or videos ***: A text area containing a paragraph about building a simple webpage for an assessment, integrating the Pexels API with a backend service (Xano) and displaying photos/videos on a frontend platform (WeWeb).
- URL of your website, app, etc. (if applicable)**: An empty text input field.
- Agreement**: A checked checkbox next to the text 'I agree to the Terms of Service and API Guidelines.'
- Generate API Key**: A green button at the bottom.

3. Saved the API Key in our environment variable for security purposes.



The screenshot shows the 'Workspace settings' panel in the Xano application. The panel is divided into two main sections: 'Workspace settings' and 'Environment Variables'.

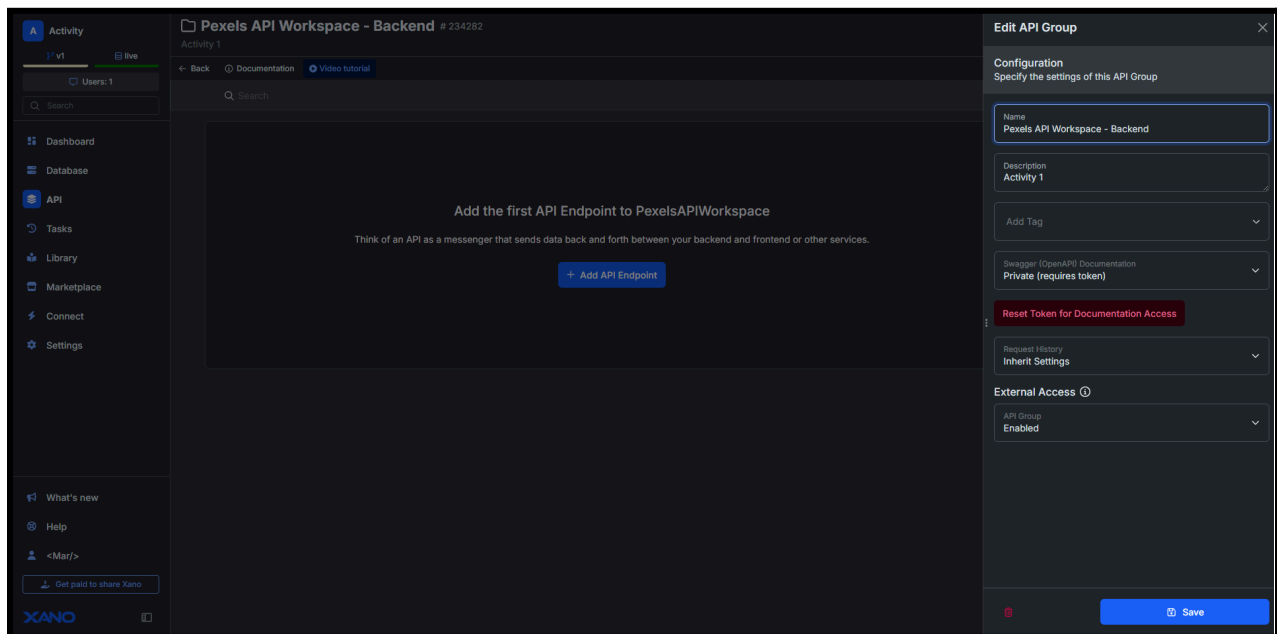
Workspace settings:

- Workspace name:** 'Activity'.
- Environment Variables:** A section explaining that environment variables store things like API keys. It lists 'YOUR VARIABLES' (PEXELS_API_KEY) and 'SYSTEM VARIABLES' (\$remote_ip, \$http_headers, \$request_url, \$request_querystring, \$datasource, \$branch).
- Reset Workspace:** A section explaining that resetting wipes all data. It includes a 'Reset' button.

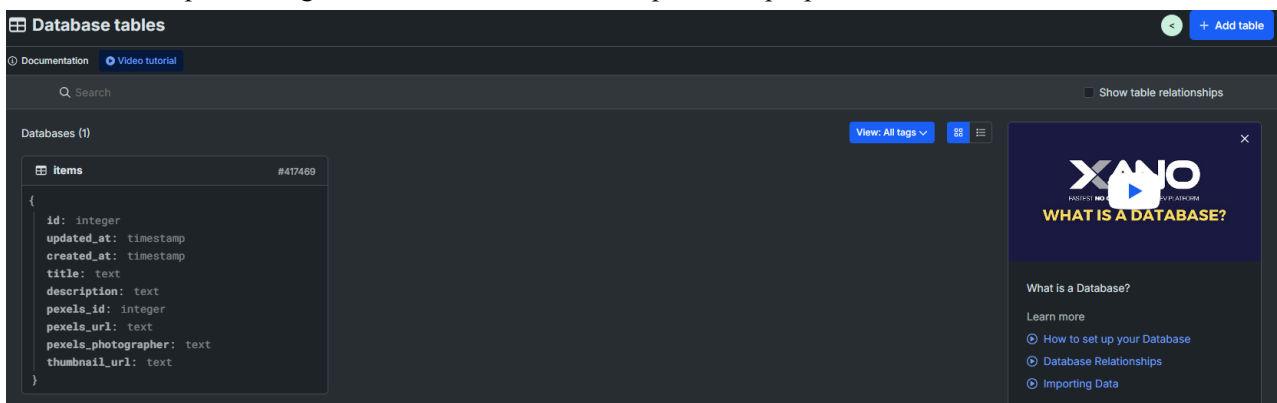
Environment Variables:

- Configure Environment Variables Below:** A section for adding new variables.
- Variables:** A table with two columns: 'name' and 'value'. The first row shows 'PEXELS_API_KEY' as the name and a masked value (represented by dots) as the value.
- + Variable:** A button to add a new variable.
- Save:** A blue button at the bottom right.

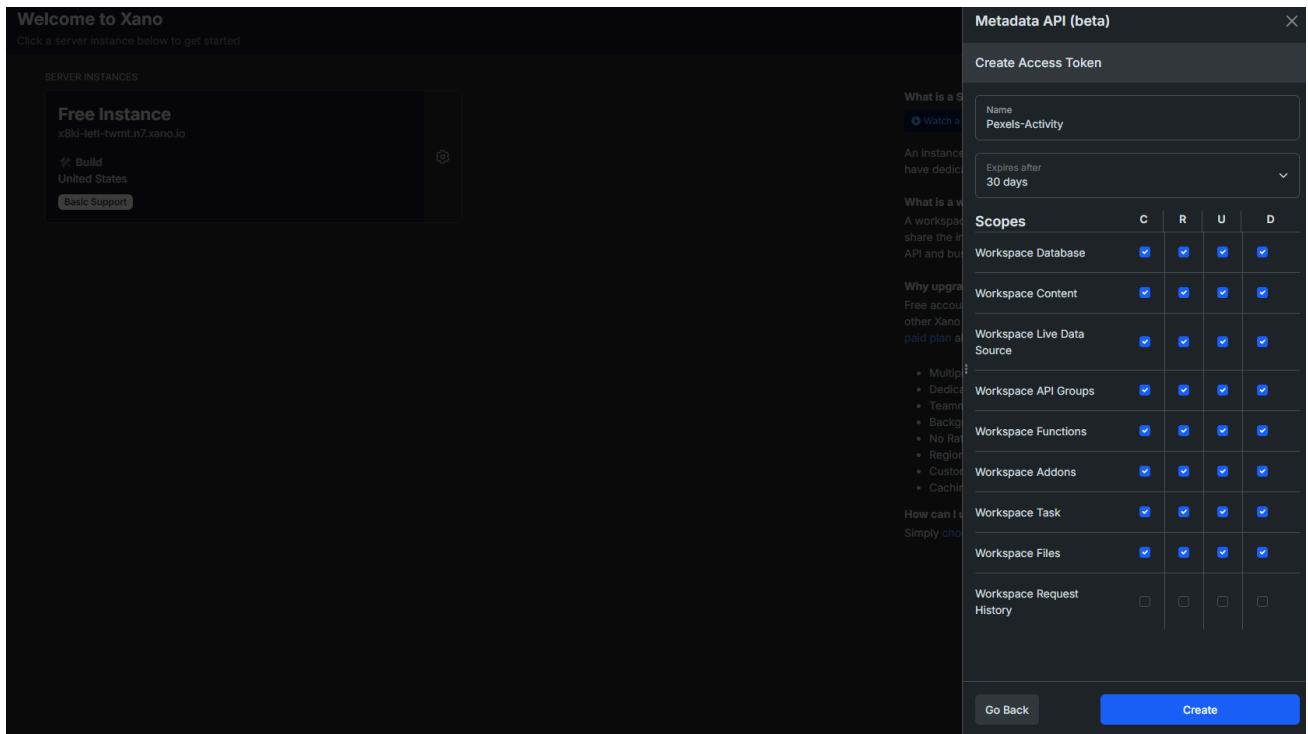
4. While waiting, I set up my workspace for backend development.



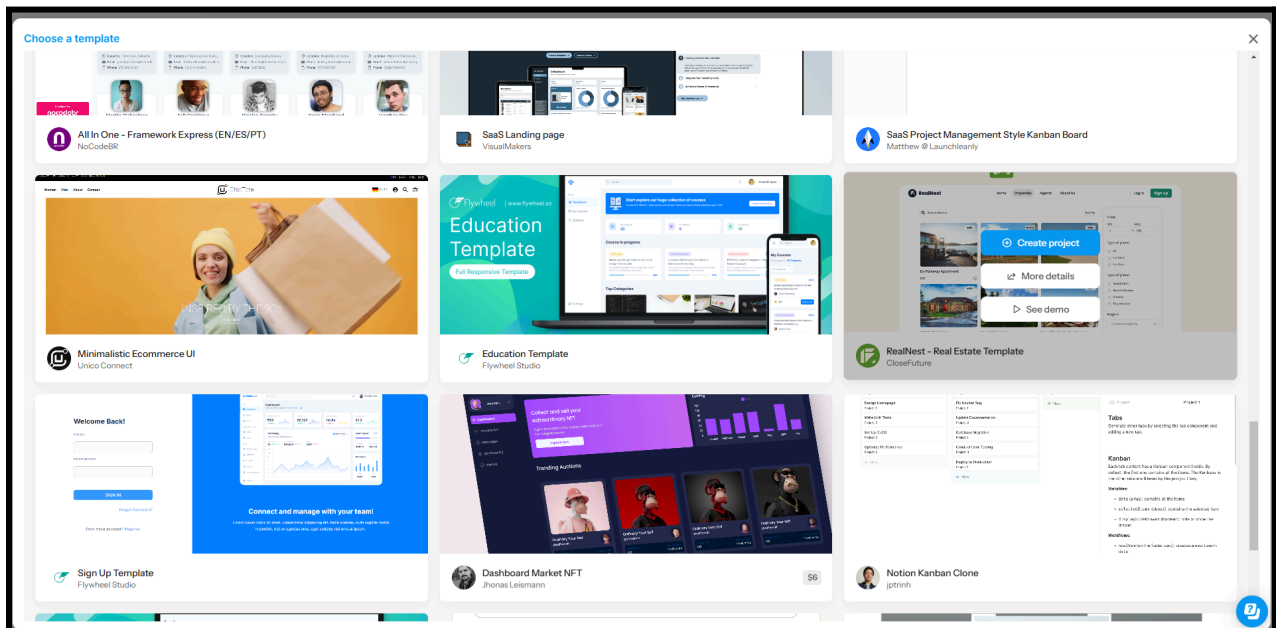
5. Due to the limitation of the Pexels API (read-only), we cannot perform POST, PUT, or DELETE operations. To address this, I will implement a separate backend for CRUD operations on items. Before proceeding, a database needs to be set up for this purpose.



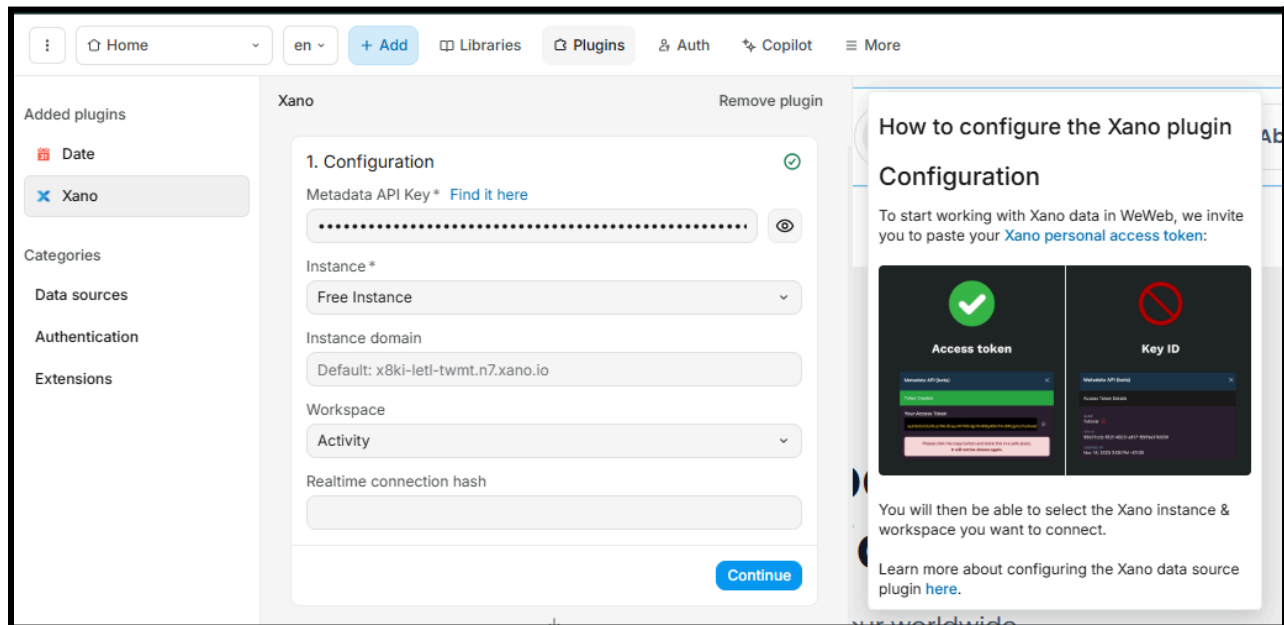
6. Generated the access token in Xano's instances and added it to our frontend.



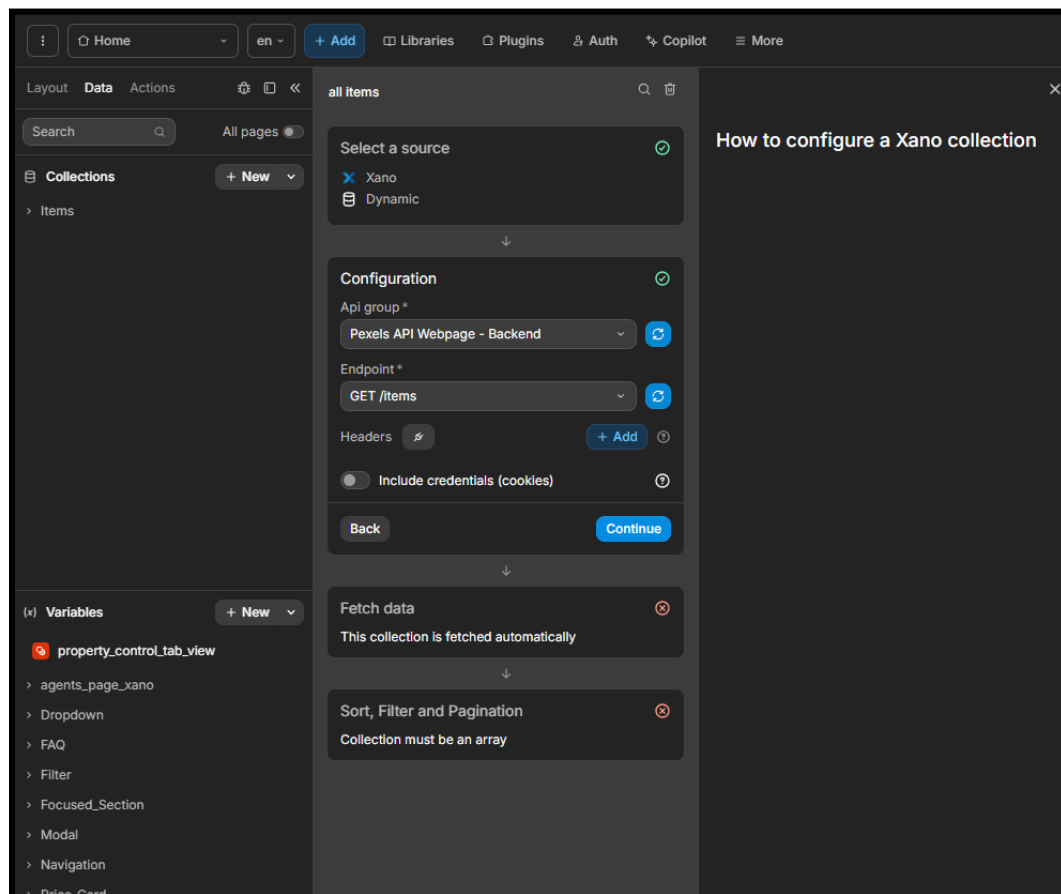
7. Selected a template in WeWeb that suits the needs of this activity and modified it to serve as a good starting point.



8. Connected Xano to our frontend in WeWeb. Although we can later adjust the API for Pexels images, the immediate goal was to establish the connection and create a basic working interface.

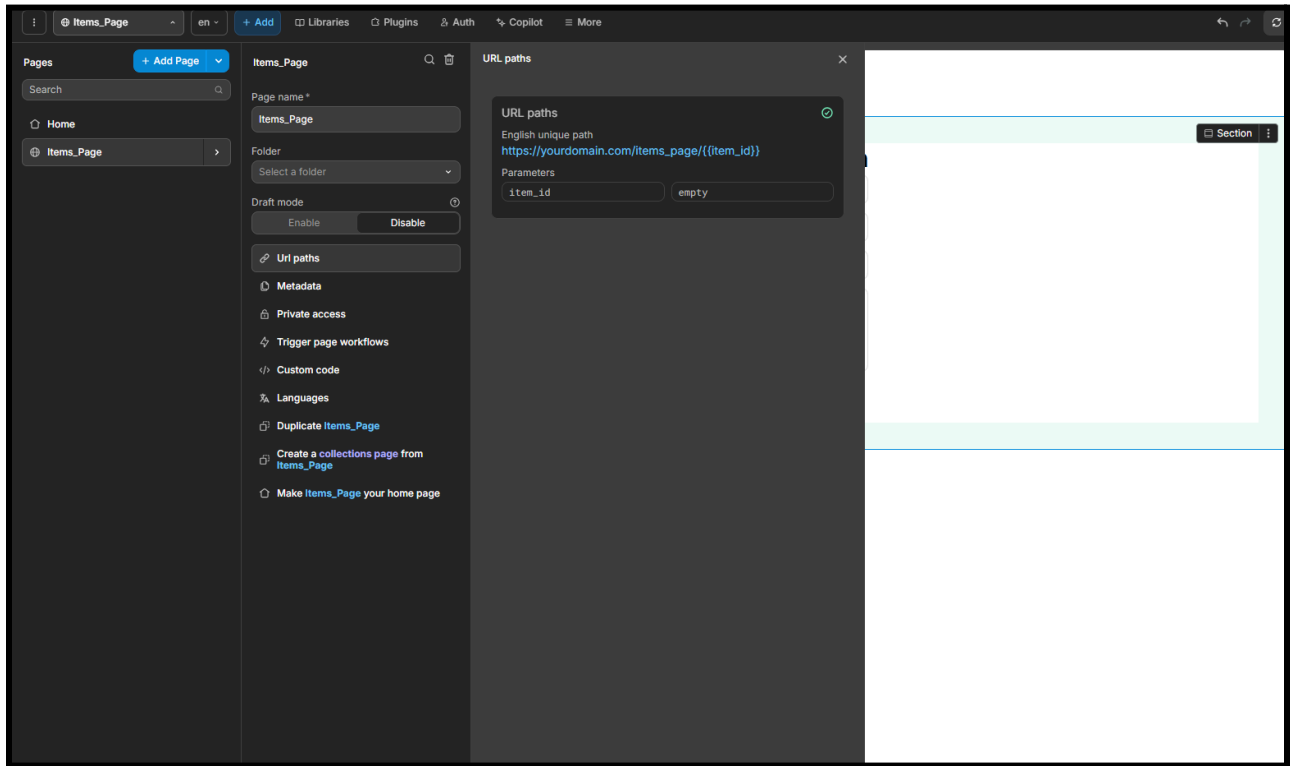


9. Created the collection of queries that will be used in the frontend.

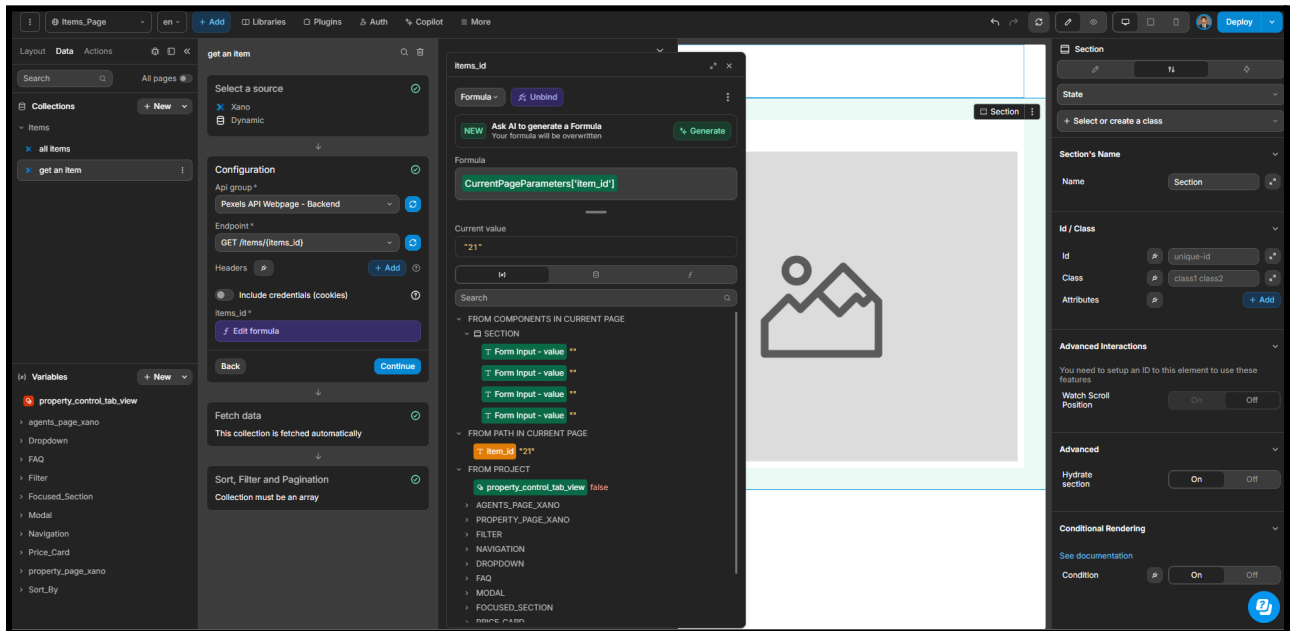


The screenshot displays the React Native development environment. On the left, a sidebar contains navigation options: Home, Add, Libraries, Plugins, Auth, Copilot, and More. Below this is a 'Layout' section with a search bar and a list of components: New bar, Container, Modal, Section, Columns, Card, Flexbox, Text, and Paginator. The main area shows a mobile app preview with a 'Welcome To My Activity 1' header and a grid of items, each with a placeholder image and a text description. The right sidebar shows the 'Expo DevTools' interface, including a 'Text' component inspector and a 'Formula' section with a 'Generate' button. The bottom of the screen shows a navigation bar with icons for Home, Add, Libraries, Plugins, Auth, Copilot, and More.

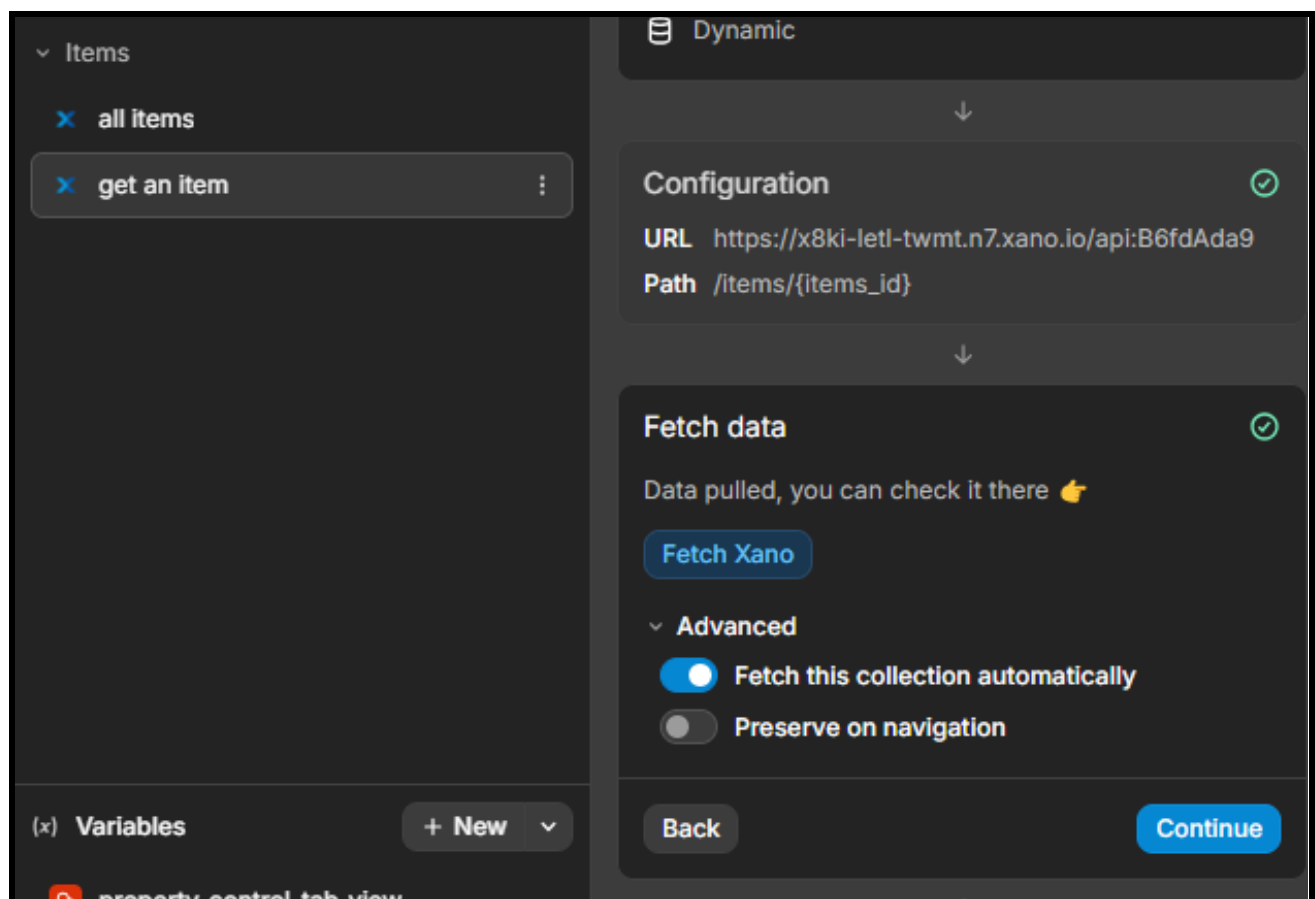
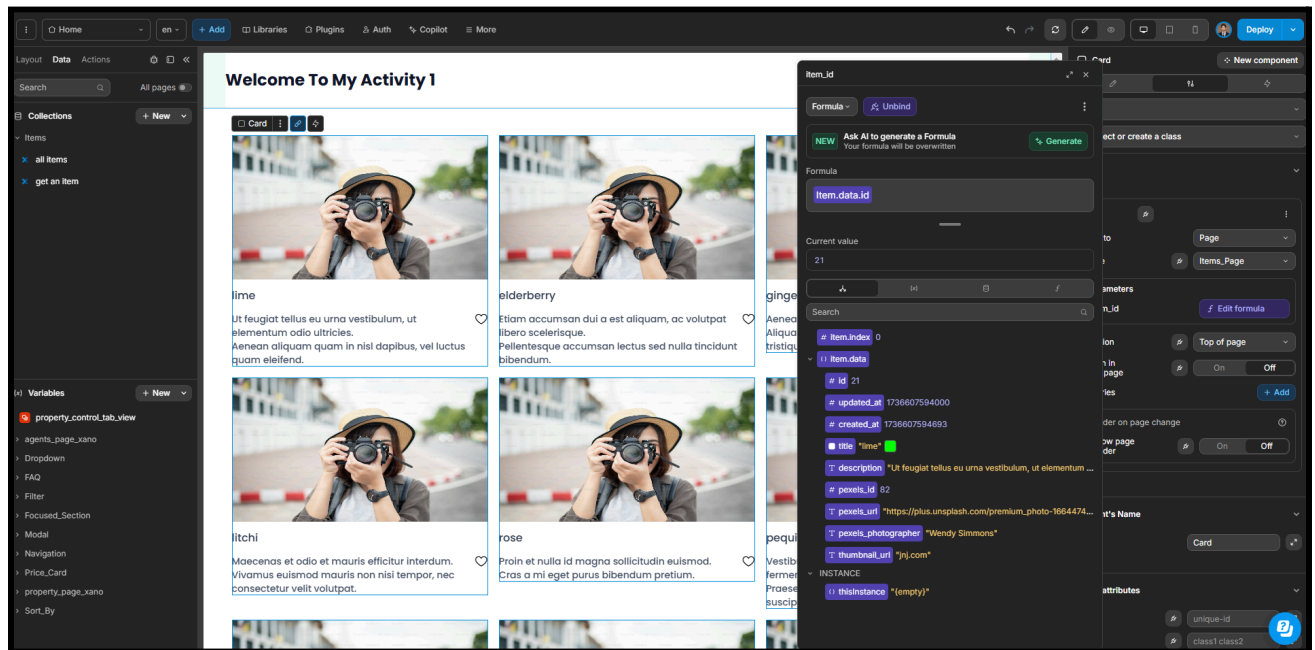
12. Created a new page for editing, which can also be reused for adding items. The edit page uses a dynamic route that takes the item ID.



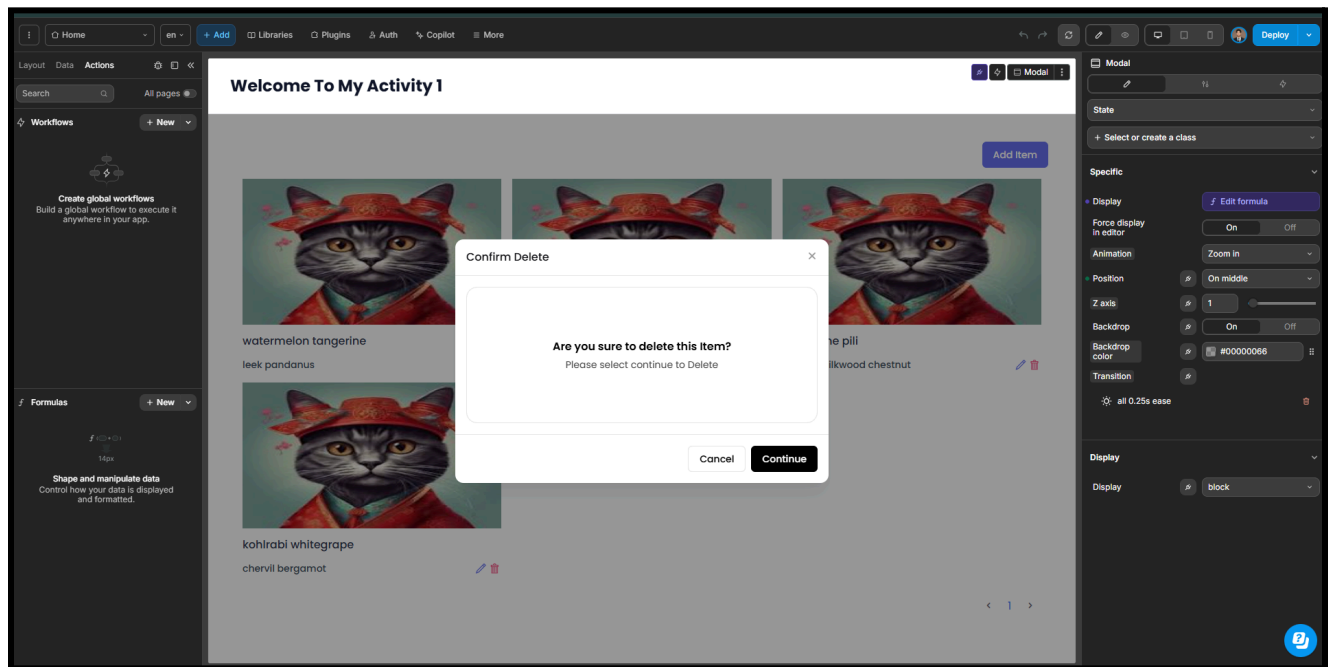
13. Updated the GET request for specific items in the collection to ensure it correctly fetches the relevant item.



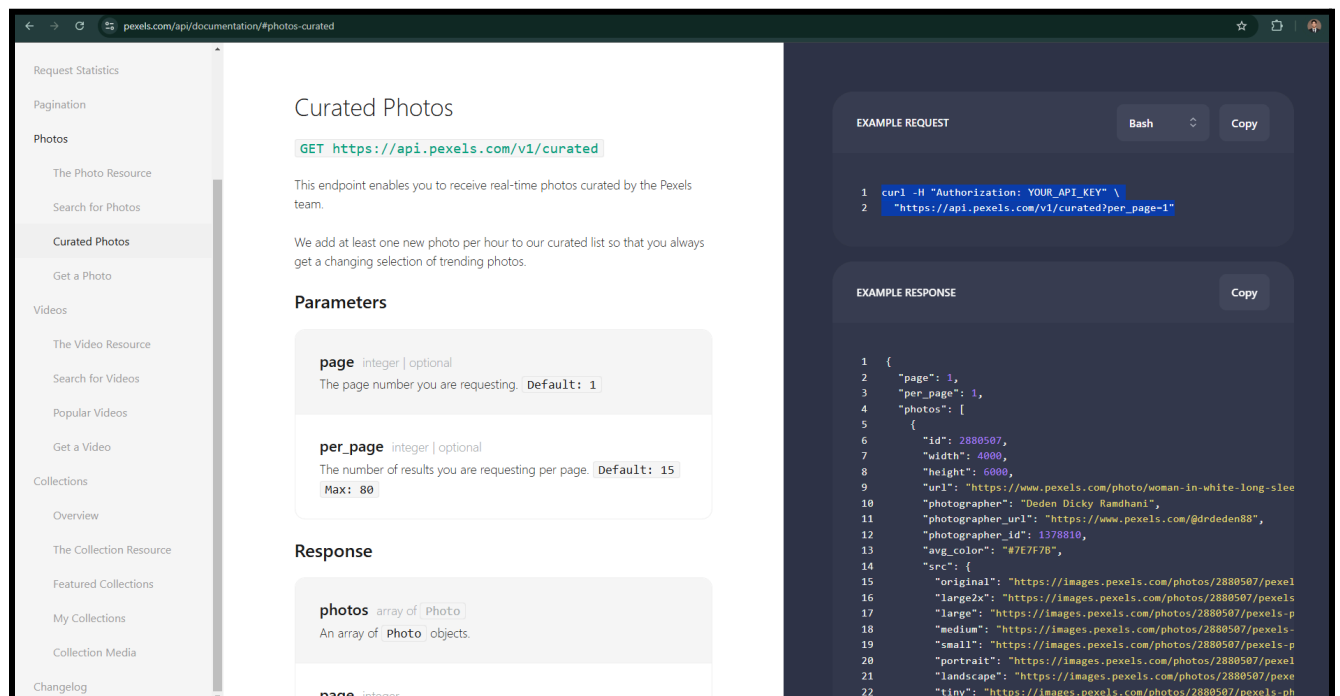
14. Bound the item ID of the selected item as a parameter to the item's page while disabling the preservation of fetched data during navigation to avoid bugs in data fetching.

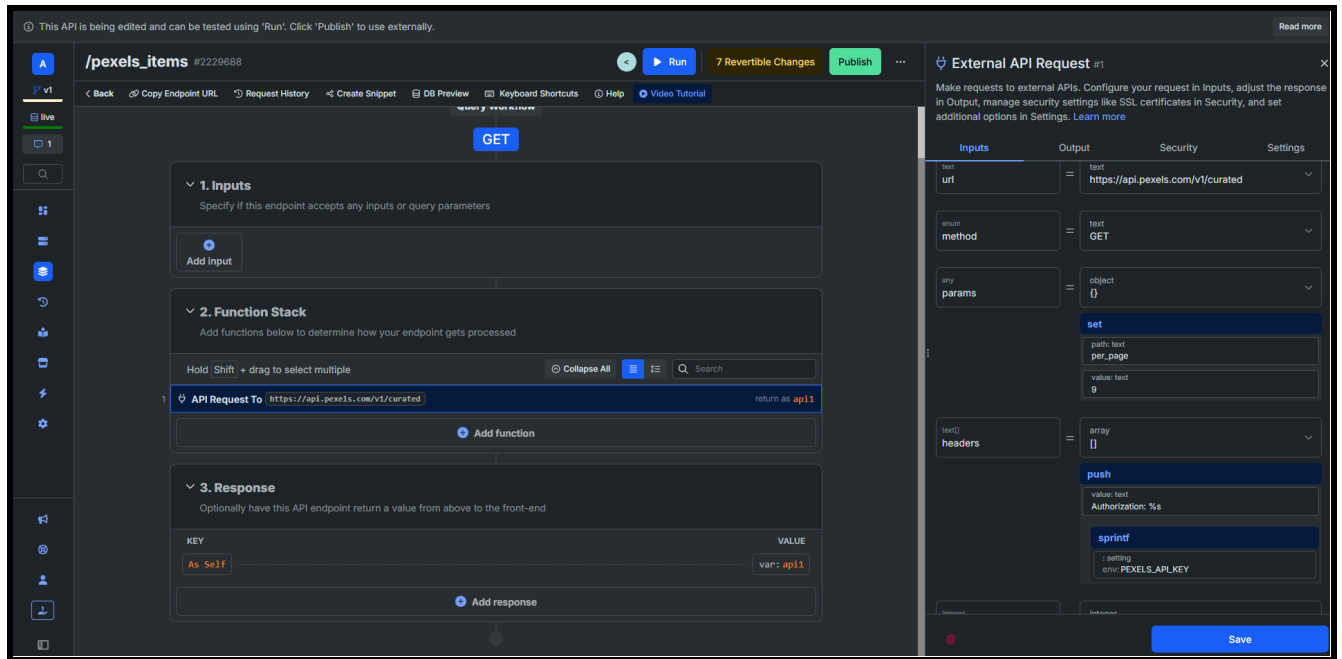


15. Created forms for CRUD operations, focusing primarily on Add and Edit functionalities. For Delete operations, I encountered an issue where I needed to create a new page to allow the home page enough time to reload the new data.

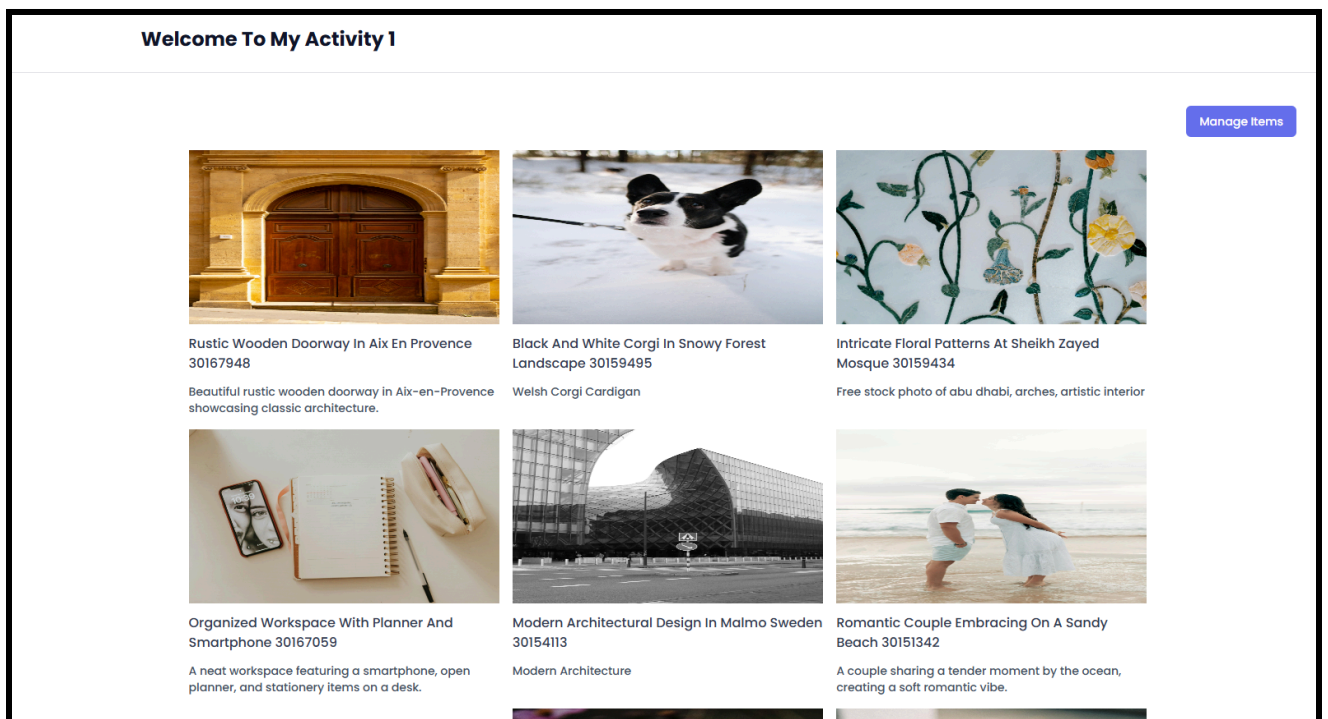


16. Created separate API endpoints for Pexels to fetch images. I used the PEXELS API KEY from environment variables and passed it to the function in Xano. To save time and simplify the creation of the function, I used cURL import to customize the returned items when accessing the endpoint.





17. Integrated the Pexels API into the frontend.



18. Added customizations to the forms for better user experience and functionality.

Note: In addition, due to the complexity of matching the returned data from the Pexels API to the table in Xano, I opted to create a separate endpoint for the Pexels API that directly integrates with the backend. Although aligning the data could be done, I decided to take this approach to avoid the extensive testing required to match the values. I'm not yet an expert in this technology, but I'm moving forward with the most efficient solution for the current needs.