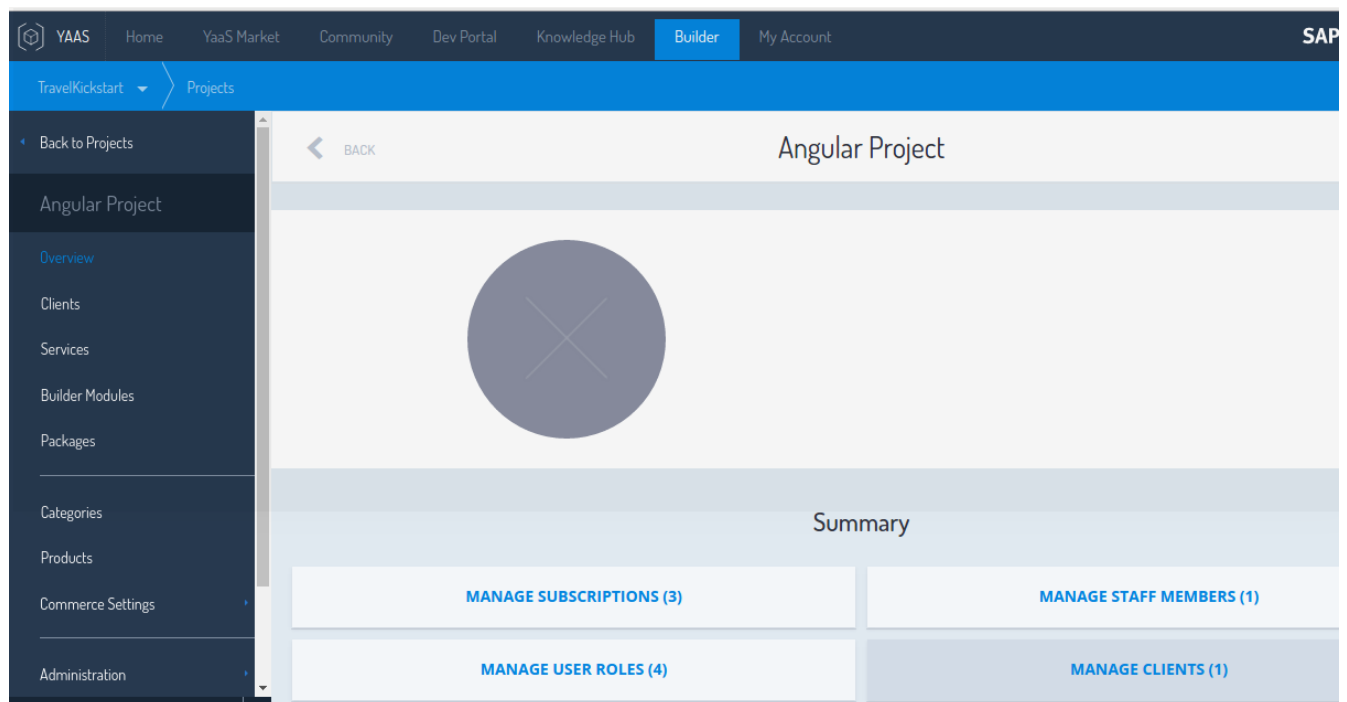
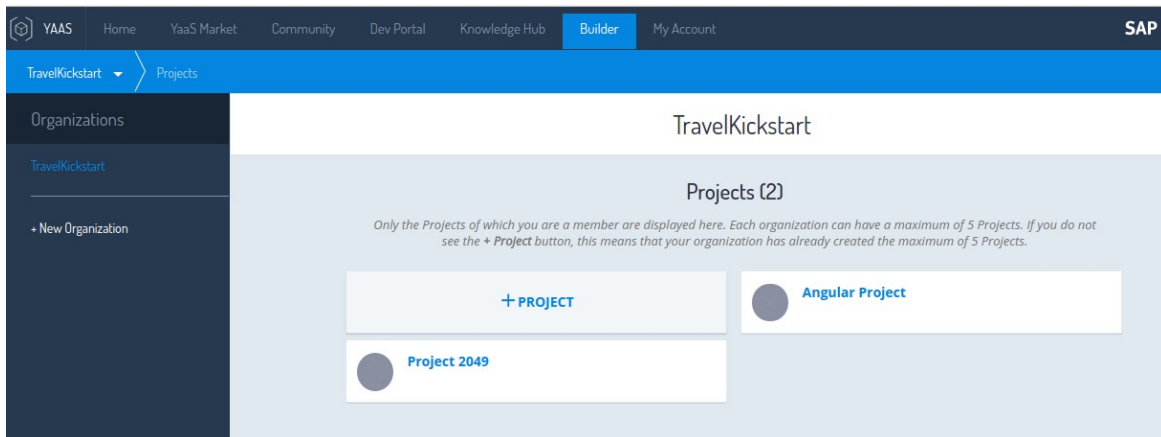


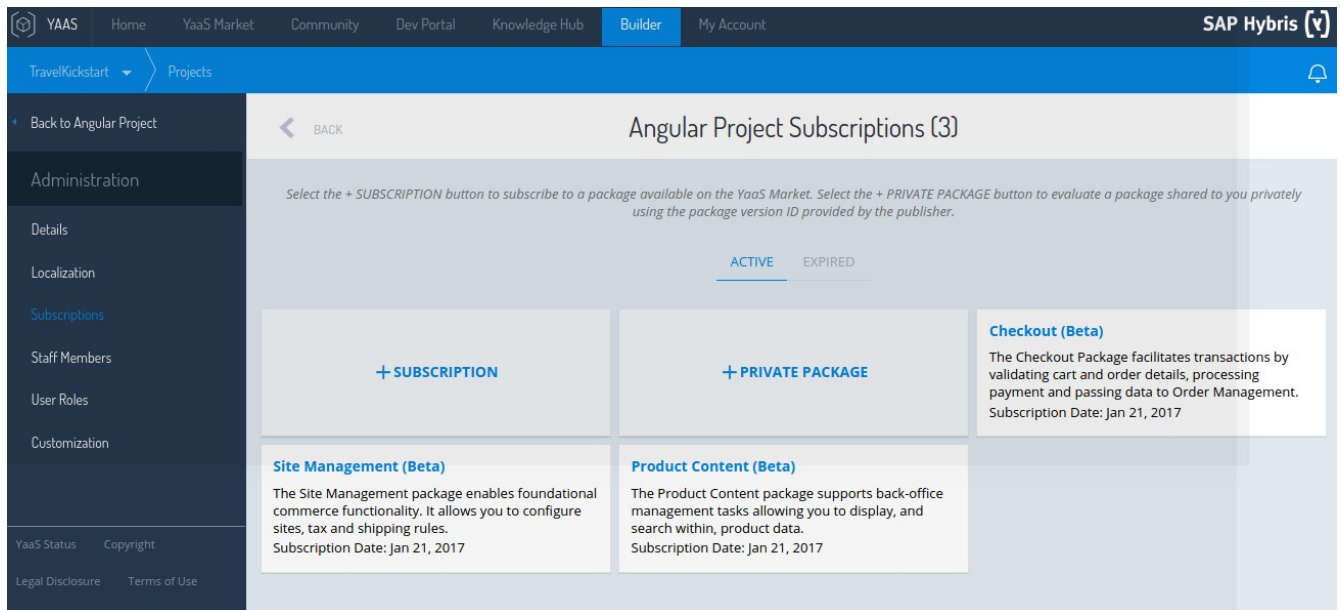
## YaaS Builder

<https://builder.yaas.io>

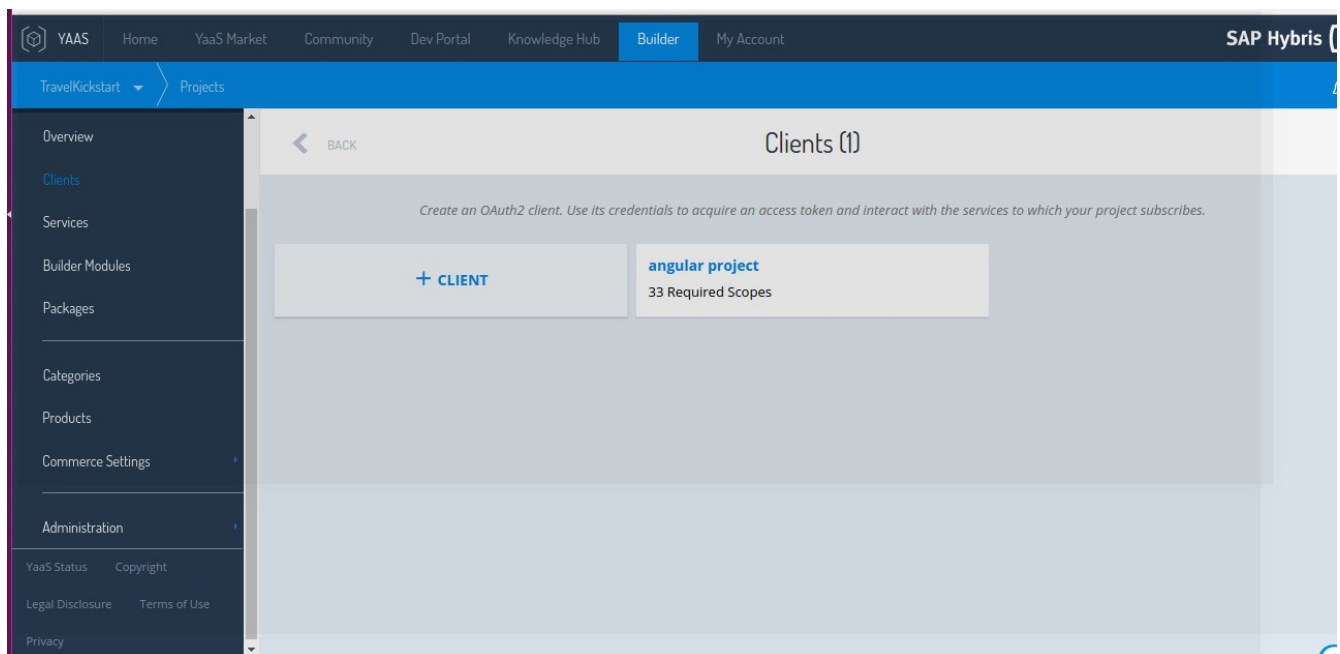
enter credentials ([lenmorld@gmail.com](mailto:lenmorld@gmail.com))



add subscriptions based on what 'packages' your app needs  
here we add basic Product and Checkout management  
and other required packages to make it work  
but a lot of other subscriptions are available  
(adding a sub brings you to the YaaS market)



also need a 'Client' which will be our app  
our client also needs certain 'scopes' that defines  
the features that we can use and the data that we pass  
and receive to/from the REST API during a request/response



creating a client is as simple as putting the name so we'll skip that part

scopes also depend on subscriptions  
so after adding/removing sub, scopes must be updated

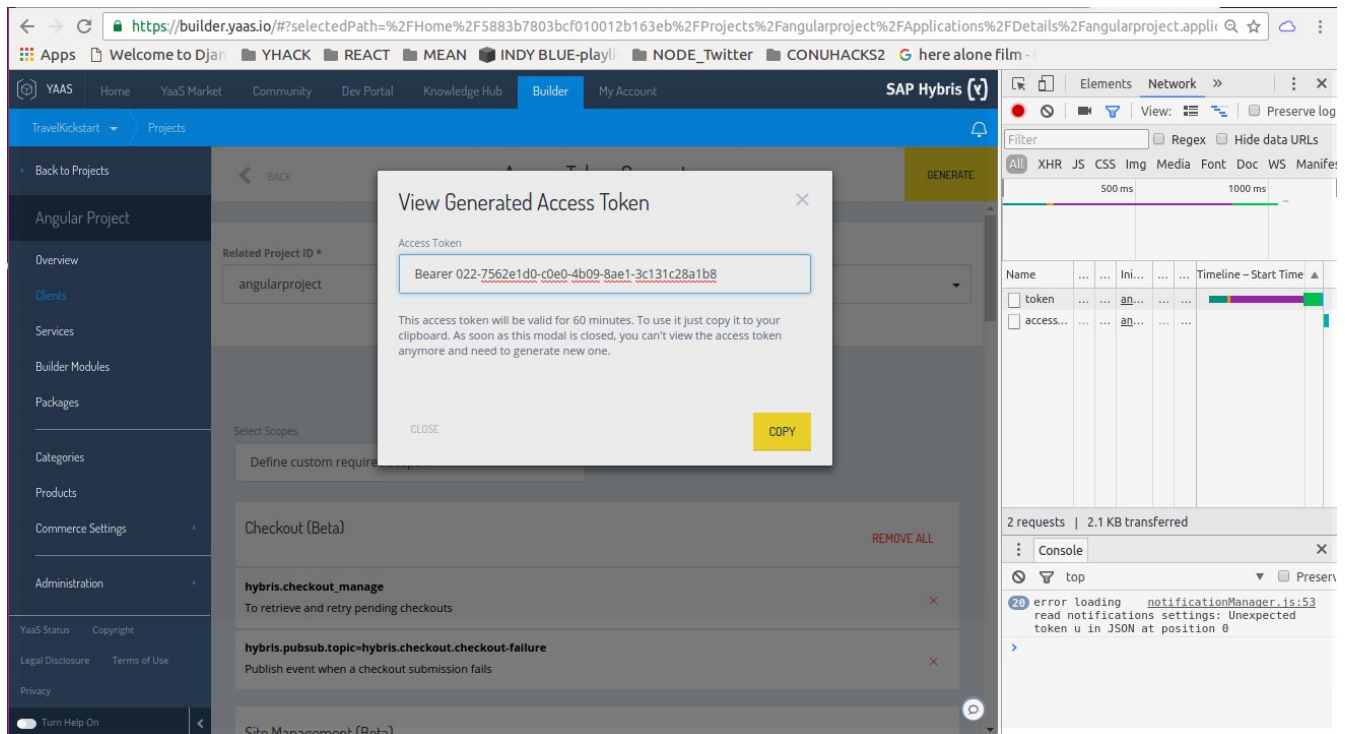
inside scope...

The screenshot shows the SAP Hybris Builder interface for an 'angular project'. The top navigation bar includes links for YAAS, Home, YaaS Market, Community, Dev Portal, Knowledge Hub, Builder (active), and My Account. The left sidebar contains a navigation menu with options like Back to Projects, Angular Project, Overview, Clients, Services, Builder Modules, Packages, Categories, Products, Commerce Settings, and Administration. The main content area is titled 'angular project' and includes a 'BACK' button. It features a 'Display Name' field with the value 'angular project', a 'Description' field, an 'Identifier' field with the value 'angularproject.application', and a 'Redirect URIs' field with the value 'http://'. Below these fields is a 'DELETE' button. The interface is divided into two main sections: 'Client Authorization' and 'Required Scopes (33)'. The 'Client Authorization' section has a 'GENERATE CREDENTIALS' button and displays 'Client ID' and 'Client Secret' as masked strings (\*\*\*\*\*). A 'SHOW' button is located below the 'Client ID' field, and a 'GENERATE ACCESS TOKEN' button is at the bottom right of this section. The 'Required Scopes (33)' section has a 'MANAGE REQUIRED SCOPES' button and lists two scopes: 'hybris.checkout\_manage' and 'hybris.pubsub.topic=hybris.checkout.checkout-failure', each with a red 'X' icon. A blue circular icon with a white 'D' is located at the bottom right of the 'Required Scopes' section.

most important part here for us is the credentials part  
that allows us to have our Client ID and Client Secret  
and generate

The screenshot shows the 'Access Token Generator' page in the SAP Hybris Admin Console. The page has a blue header with navigation links: Home, SAP Market, Community, Dev Portal, Knowledge Hub, Builder, and My Account. Below the header, there's a blue bar with 'TravelKickstart' and 'Projects'. A dark blue sidebar on the left contains a menu with 'Back to Projects', 'Angular Project', 'Overview', 'Clients', 'Services', 'Builder Modules', 'Packages', 'Categories', 'Products', 'Commerce Settings', and 'Administration'. The main content area has a light blue background. At the top, there's a 'BACK' button and a 'GENERATE' button. Below these, there are two input fields: 'Related Project ID \*' with the value 'angularproject' and 'Related Region \*' with a dropdown arrow. Further down, there's a section titled 'Requested Scopes' with a 'Select Scopes' dropdown menu.

Click “Generate Access Token”  
set Region to US  
and select the **Highlighted Scopes** in the dropdown  
(its a good idead)  
click Generate



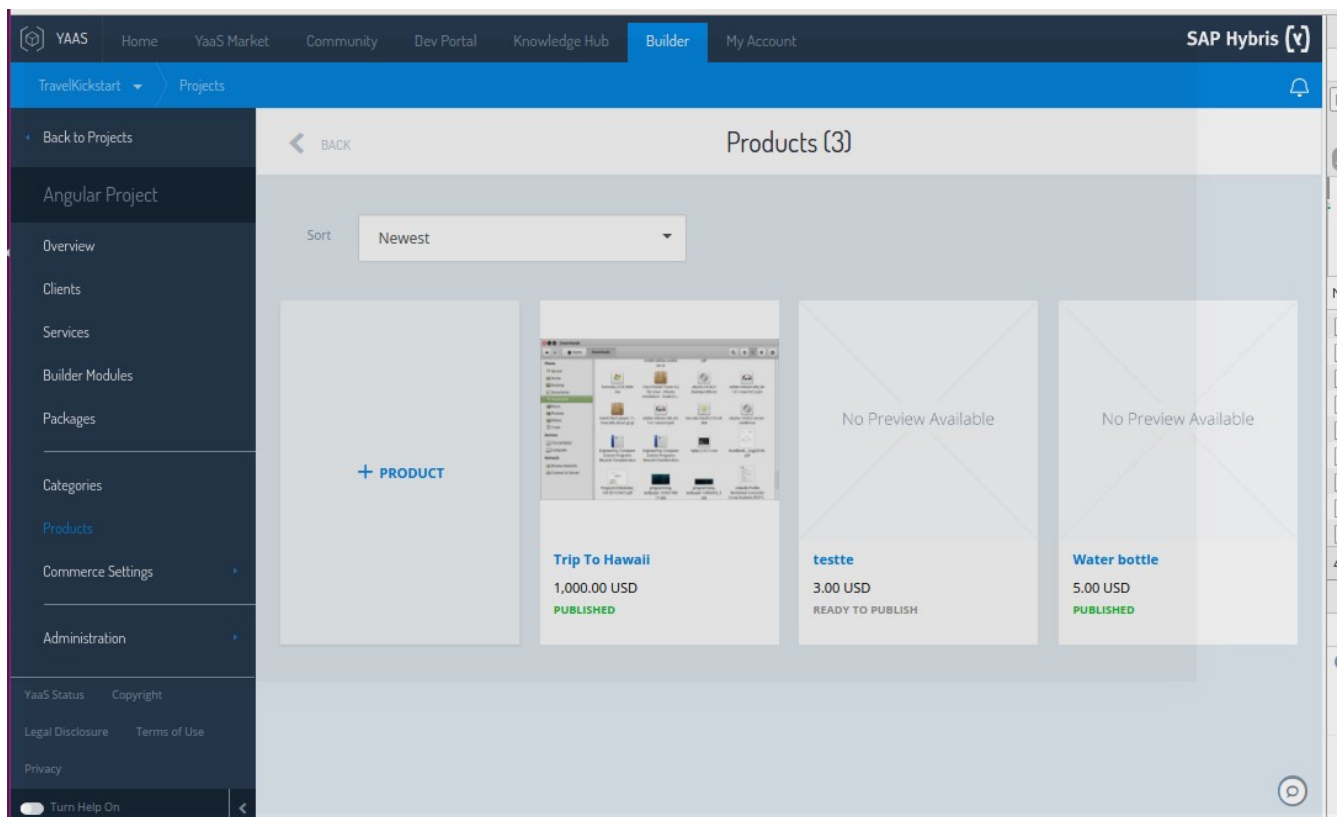
take a note of the Generated Access Token, this will be used in the Auth. header

**Bearer 022-7562e1d0-c0e0-4b09-8ae1-3c131c28a1b8**

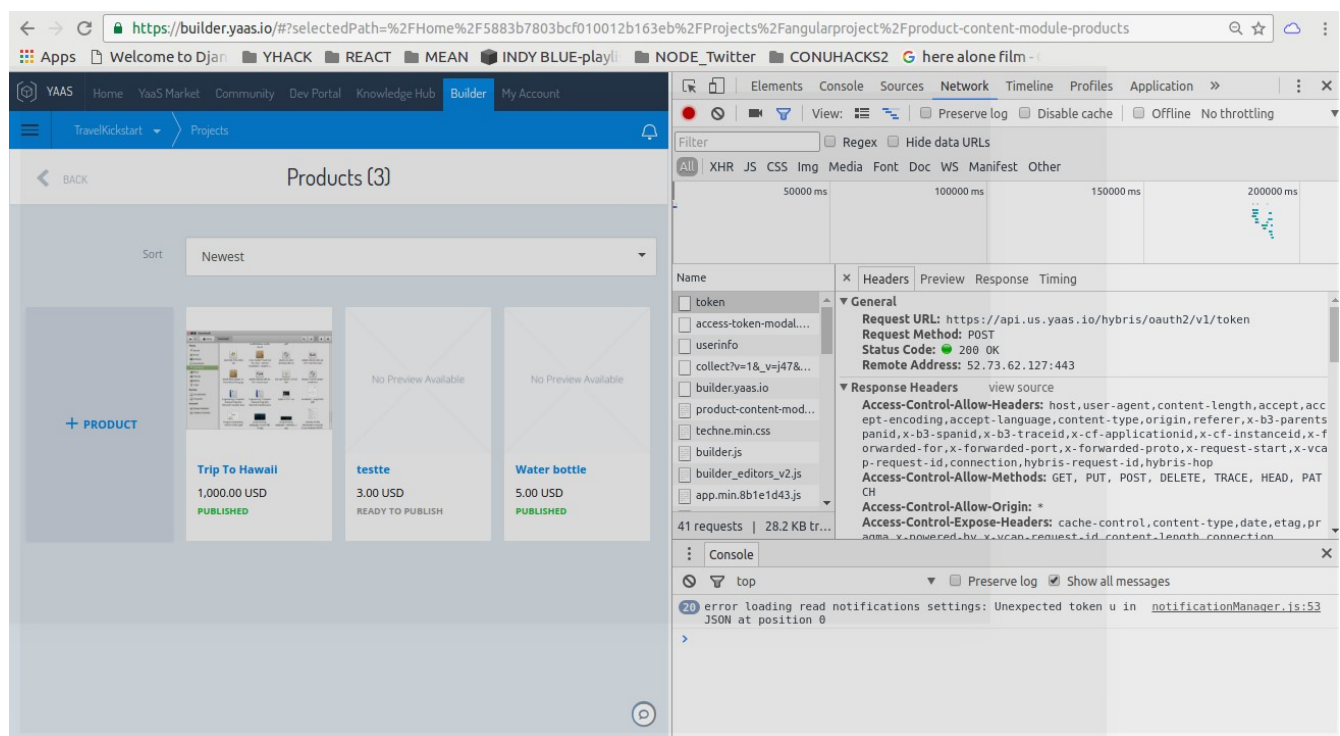
this is good for 1 hour

to try out our token, we can do a simple  
GET all products example

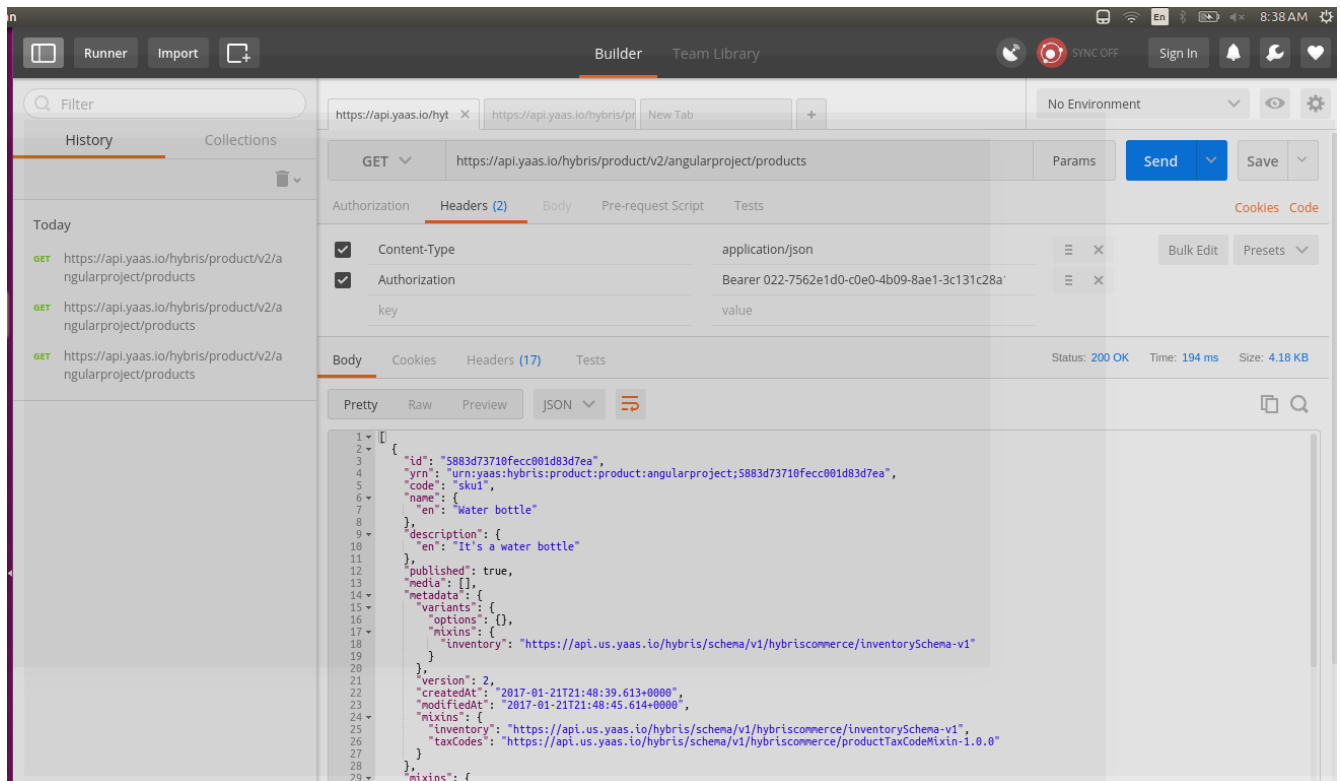
first create products in the product left pane



at this point we could open Chrome dev tools  
and examine the Network tab  
and see all the HTTP response headers  
to observe Auth headers and scopes



now that we hvae products, we can try the YaaS REST API  
by opening Postman



use this URL for most stuff we need to do

<https://api.yaas.io/hybris/product/v2/{projectname}/products>

important:

must add headers

- Content-type: application/json // www might also work
  - Authorization: **Bearer 022-7562e1d0-c0e0-4b09-8ae1-3c131c28a1b8**
- (this is the one we got before)

NOTE:

we dont explicitly supply the Client\_ID and Client\_Secret since we just use the Auth header

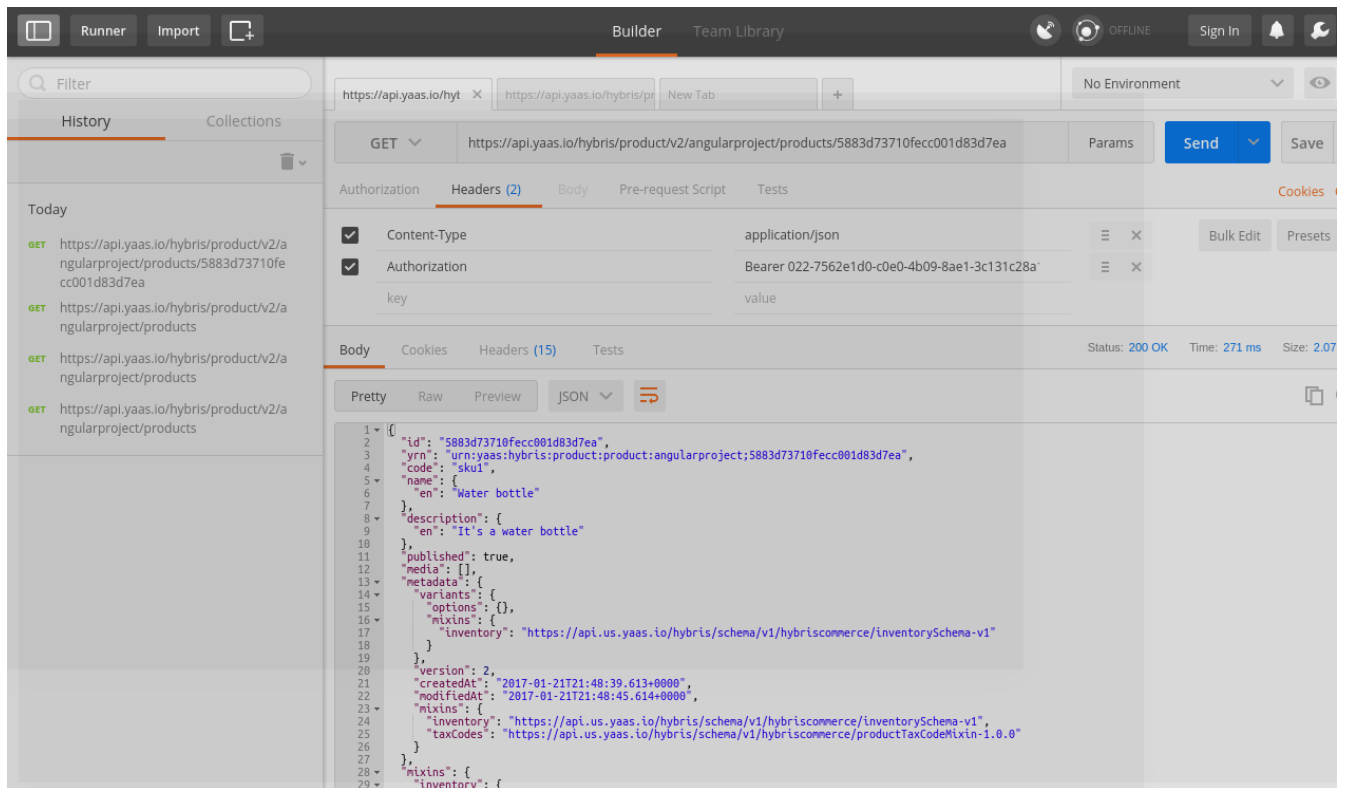
of course we can also do

all GET and POST stuff

like this, getting details of a product, giving the :productID in the URL

<https://api.yaas.io/hybris/product/v2/angularproject/products/5883d73710fecc001d83d7ea>





To see the detailed API details for all packages, go to <https://devportal.yaas.io/services/>

then search for the package name e.g. **Product**

API console is also pretty helpful

The screenshot shows the SAP Hybris Dev Portal interface. The top navigation bar includes links for Home, YaaS Market, Community, Dev Portal (active), Knowledge Hub, Builder, and My Account. A secondary navigation bar lists Home, Overview, Getting Started, Solutions, API Docs, Tools & Resources, News, and UI Design Guidelines. The left sidebar contains a 'Product' section with links to Overview, API Reference, Events, Details, Tutorial, and Glossary, along with a 'RELEASE NOTES' button. The main content area is titled 'Product' and shows the 'Latest - v2' and 'Beta' versions. It includes an 'Overview' section with a description of the Product service, its REST API capabilities, and a list of use cases: Storefronts, Product Content Management, and Systems Integration. An 'API Reference' section is also visible at the bottom.

The screenshot shows the SAP Hybris Dev Portal interface, specifically the 'API Console' for the 'Product' service. The top navigation bar is the same as the previous screenshot. The left sidebar is also the same. The main content area is titled 'Product' and shows the 'Latest - v2' and 'Beta' versions. It includes an 'API Console' section with a table of API endpoints and their corresponding HTTP methods. The table has columns for the endpoint, the HTTP method, and a button to execute the request. The endpoints are listed as follows:

Endpoint	HTTP Method	Action
/tenant/products	DELETE, POST, GET	DELETE, POST, GET
/tenant/products/{productId}	DELETE, PUT, GET	DELETE, PUT, GET
/tenant/products/{productId}/media	POST, GET	POST, GET
/tenant/products/{productId}/media/{mediaId}	DELETE, PUT, GET	DELETE, PUT, GET
/tenant/products/{productId}/media/{mediaId}/commit	POST	POST

\* tenant = project