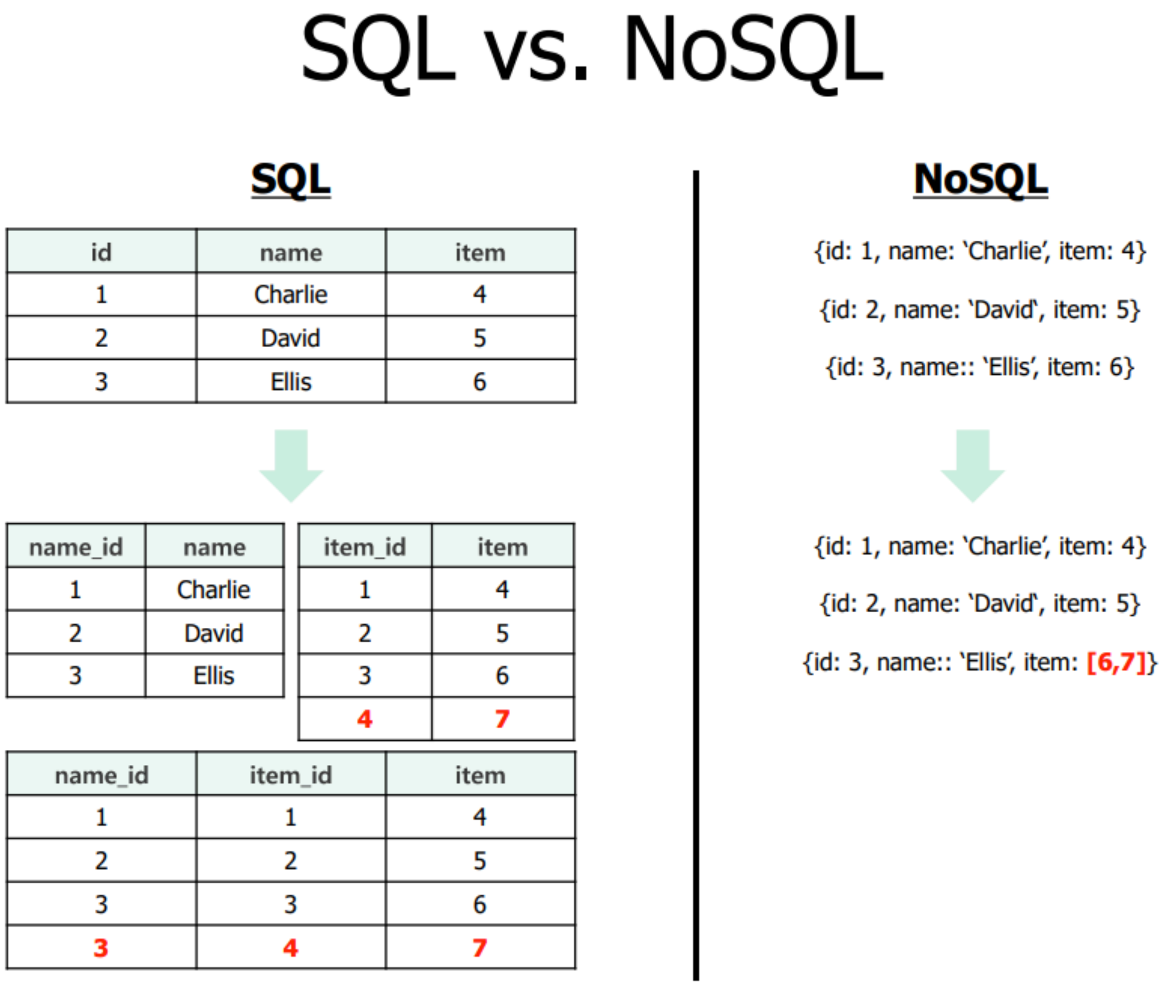
Exercise #1: Introduction to NoSQL

This practical exercise is intended to make you reason about the differences of principles between classic Relational Database Management Systems and NoSQL servers for building and managing data.

Expect to be able to:

- Point out some of their principles concerning: data model, design process, internal and external data management, architecture;
- Compare these principles with those of classic RDBMS.



SQL vs. NoSQL

SQL

id	name	item
1	Charlie	4
2	David	5
3	Ellis	6

name_id	name
1	Charlie
2	David
3	Ellis

item_id	item
1	4
2	5
3	6
4	**7**

name_id	item_id	item
1	1	4
2	2	5
3	3	6
3	**4**	**7**

NoSQL

{id: 1, name: ‘Charlie’, item: 4}

{id: 2, name: ‘David‘, item: 5}

{id: 3, name:: ‘Ellis’, item: 6}

{id: 1, name: ‘Charlie’, item: 4}

{id: 2, name: ‘David‘, item: 5}

{id: 3, name:: ‘Ellis’, item: **[6,7]**}

In this exercise you will populate and query a NoSQL database from a traditional relational database. You will use the classical Northwind example from Microsoft to migrate from a relational database to a NoSQL cloud database.

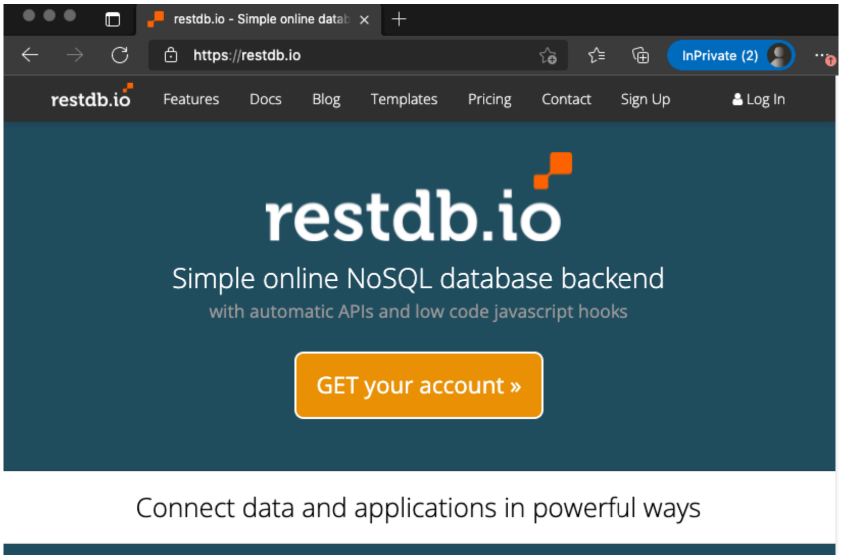
Northwind Traders Access database is a sample database that shipped with Microsoft Office suite. The Northwind database contains the sales data for a fictitious company called Northwind Traders, which imports and exports specialty foods from around the world. Developers (back in the 90's) used it to learn the MS Access product and it has been implemented in other technologies (SQL Server, for example).

Tasks:

- Register to a NoSQL cloud database service.
- Create a NoSQL database
- Create collections
- Map relations between collections
- See JSON data
- Use the API

Task 1. Register to a NoSQL cloud database service
Restdb.io is a NoSQL cloud database service built with Node.js and MongoDB. With restdb.io you get schema, relations, REST API and an efficient multi-user admin UI for working with data.

1.1 Visit RestDB website (https://restdb.io/) and sign up for a free account by clicking on **GET your account**.

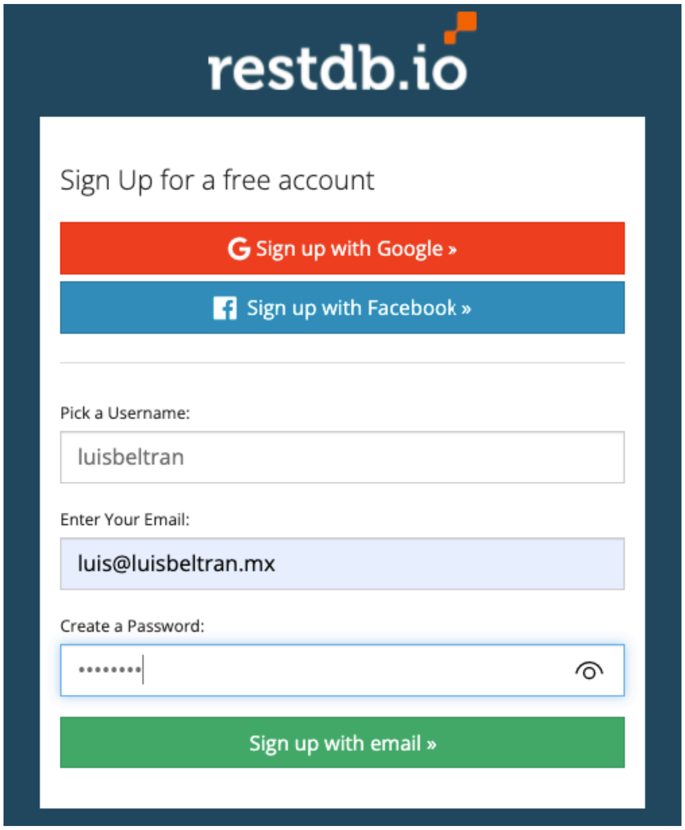


Simple online NoSQL database backend

with automatic APIs and low code javascript hooks

Connect data and applications in powerful ways

1.2 Enter the account information:



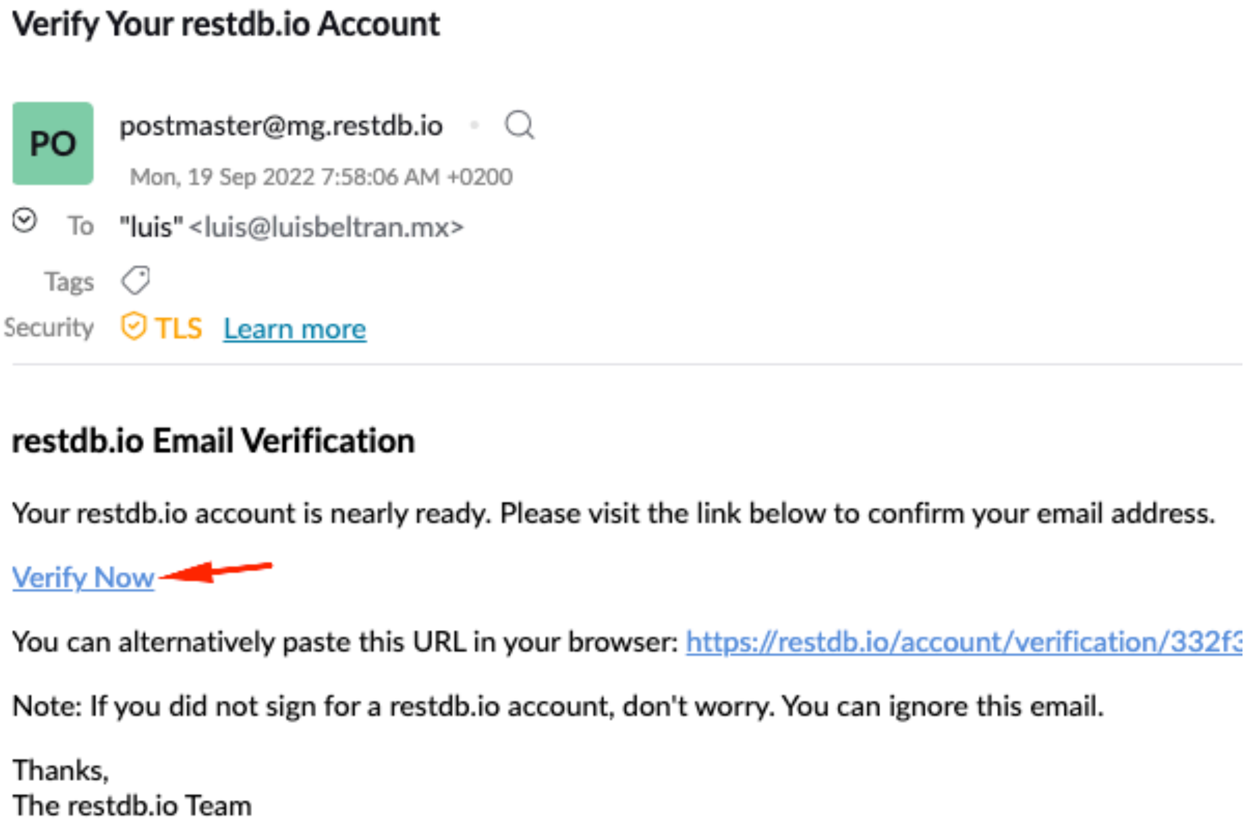
Sign Up for a free account

Pick a Username: luisbeltran

Enter Your Email: luis@luisbeltran.mx

Create a Password: ••••••••

1.3 Verify your account. You will receive an email verification message (check your inbox or spam folder).



Verify Your restdb.io Account

PO postmaster@mg.restdb.io

Mon, 19 Sep 2022 7:58:06 AM +0200

To "luis" <luis@luisbeltran.mx>

Tags

Security TLS Learn more

restdb.io Email Verification

Your restdb.io account is nearly ready. Please visit the link below to confirm your email address.

Verify Now

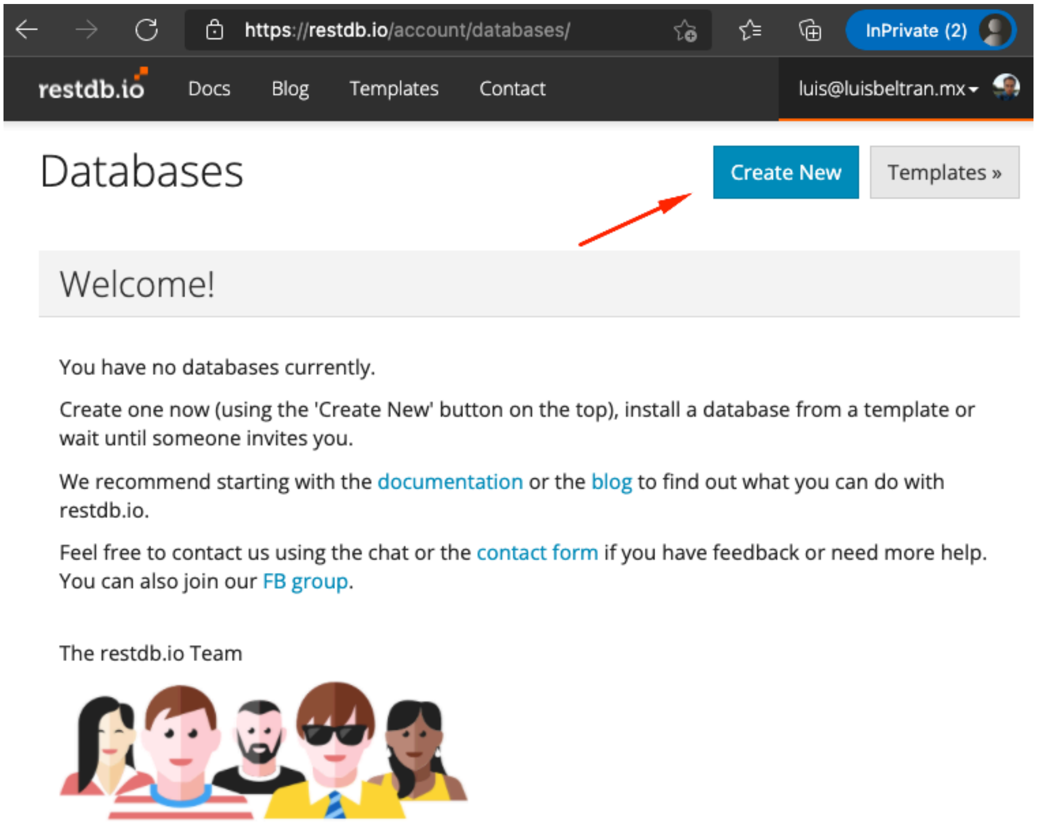
You can alternatively paste this URL in your browser: https://restdb.io/account/verification/332f3

Note: If you did not sign for a restdb.io account, don't worry. You can ignore this email.

Thanks,
The restdb.io Team

Task 2. Create a NoSQL database

2.1 Click on Create New in order to add a new database in your account



Databases

Welcome!

You have no databases currently.

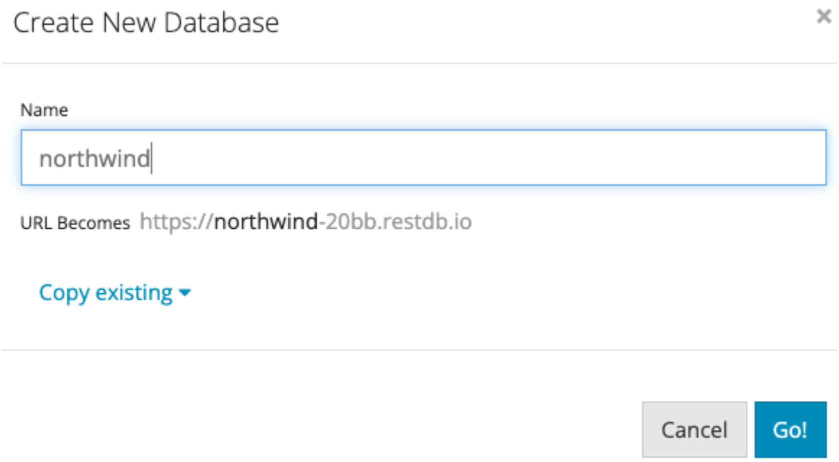
Create one now (using the 'Create New' button on the top), install a database from a template or wait until someone invites you.

We recommend starting with the documentation or the blog to find out what you can do with restdb.io.

Feel free to contact us using the chat or the contact form if you have feedback or need more help. You can also join our FB group.

The restdb.io Team

2.2 The database name will be **northwind**



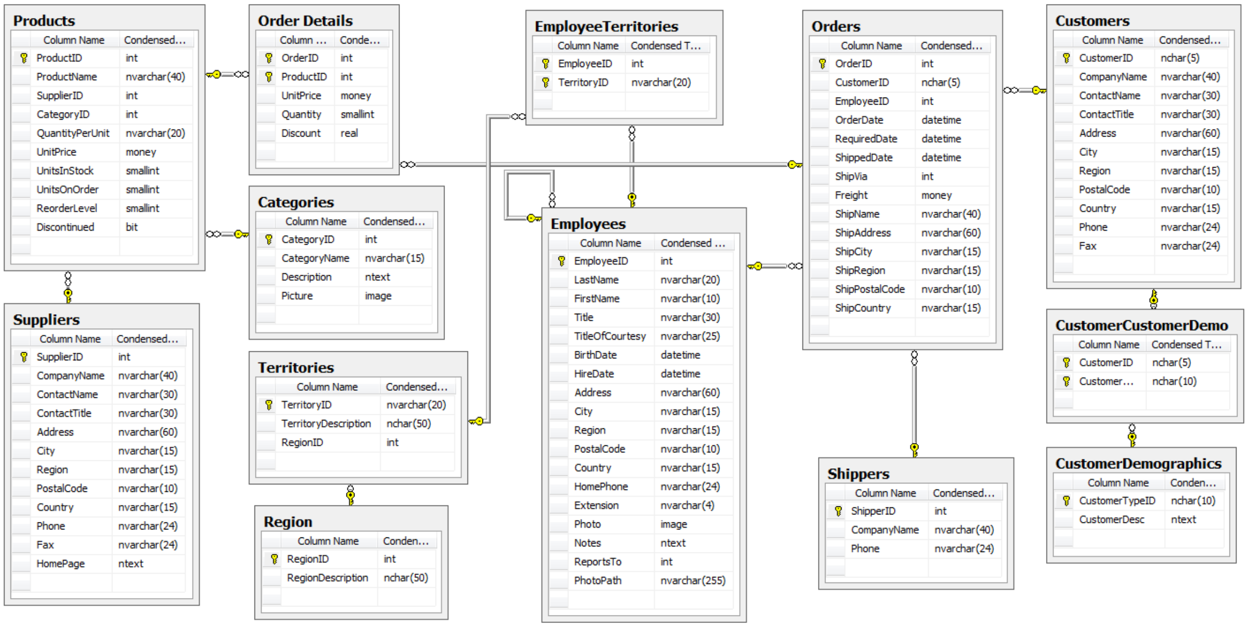
Create New Database

Name: northwind

URL Becomes https://northwind-20bb.restdb.io

Task 3. Create collections

The Northwind relational data schema is shown in the picture below.



Products

	Column Name	Condensed…
🔑	ProductID	int
	ProductName	nvarchar(40)
	SupplierID	int
	CategoryID	int
	QuantityPerUnit	nvarchar(20)
	UnitPrice	money
	UnitsInStock	smallint
	UnitsOnOrder	smallint
	ReorderLevel	smallint
	Discontinued	bit

Order Details

	Column …	Conde…
🔑	OrderID	int
🔑	ProductID	int
	UnitPrice	money
	Quantity	smallint
	Discount	real

EmployeeTerritories

	Column Name	Condensed T…
🔑	EmployeeID	int
🔑	TerritoryID	nvarchar(20)

Orders

	Column Name	Condensed…
🔑	OrderID	int
	CustomerID	nchar(5)
	EmployeeID	int
	OrderDate	datetime
	RequiredDate	datetime
	ShippedDate	datetime
	ShipVia	int
	Freight	money
	ShipName	nvarchar(40)
	ShipAddress	nvarchar(60)
	ShipCity	nvarchar(15)
	ShipRegion	nvarchar(15)
	ShipPostalCode	nvarchar(10)
	ShipCountry	nvarchar(15)

Customers

	Column Name	Condensed…
🔑	CustomerID	nchar(5)
	CompanyName	nvarchar(40)
	ContactName	nvarchar(30)
	ContactTitle	nvarchar(30)
	Address	nvarchar(60)
	City	nvarchar(15)
	Region	nvarchar(15)
	PostalCode	nvarchar(10)
	Country	nvarchar(15)
	Phone	nvarchar(24)
	Fax	nvarchar(24)

Categories

	Column Name	Condensed…
🔑	CategoryID	int
	CategoryName	nvarchar(15)
	Description	ntext
	Picture	image

Employees

	Column Name	Condensed …
🔑	EmployeeID	int
	LastName	nvarchar(20)
	FirstName	nvarchar(10)
	Title	nvarchar(30)
	TitleOfCourtesy	nvarchar(25)
	BirthDate	datetime
	HireDate	datetime
	Address	nvarchar(60)
	City	nvarchar(15)
	Region	nvarchar(15)
	PostalCode	nvarchar(10)
	Country	nvarchar(15)
	HomePhone	nvarchar(24)
	Extension	nvarchar(4)
	Photo	image
	Notes	ntext
	ReportsTo	int
	PhotoPath	nvarchar(255)

Suppliers

	Column Name	Condensed…
🔑	SupplierID	int
	CompanyName	nvarchar(40)
	ContactName	nvarchar(30)
	ContactTitle	nvarchar(30)
	Address	nvarchar(60)
	City	nvarchar(15)
	Region	nvarchar(15)
	PostalCode	nvarchar(10)
	Country	nvarchar(15)
	Phone	nvarchar(24)
	Fax	nvarchar(24)
	HomePage	ntext

CustomerCustomerDemo

	Column Name	Condensed T…
🔑	CustomerID	nchar(5)
🔑	Customer…	nchar(10)

Territories

	Column Name	Condensed…
🔑	TerritoryID	nvarchar(20)
	TerritoryDescription	nchar(50)
	RegionID	int

Shippers

	Column Name	Condensed…
🔑	ShipperID	int
	CompanyName	nvarchar(40)
	Phone	nvarchar(24)

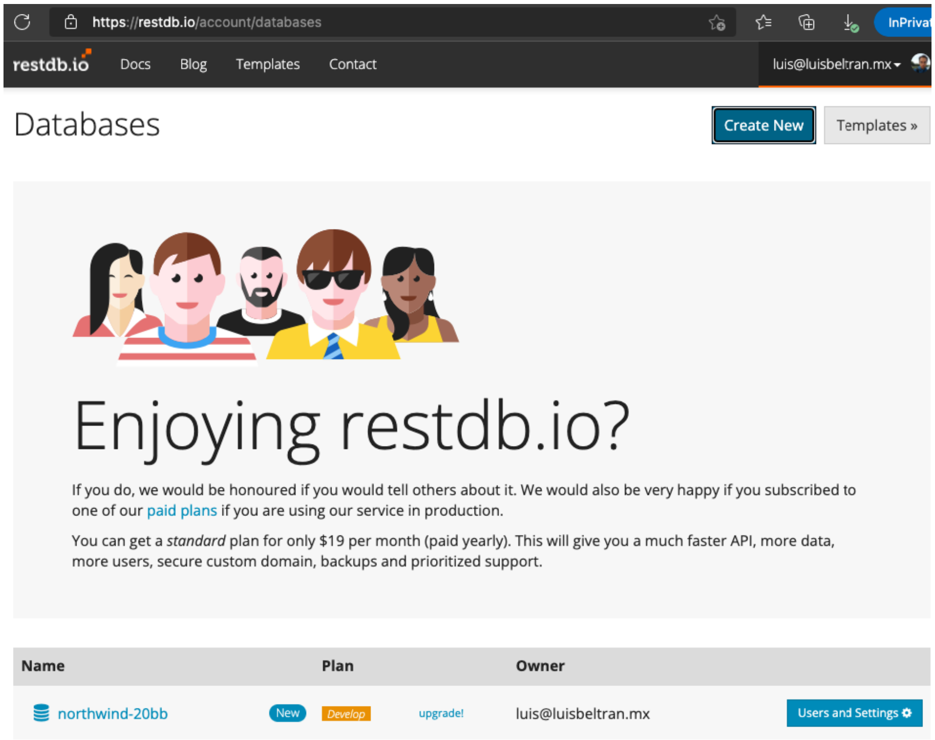
CustomerDemographics

	Column Name	Conden…
🔑	CustomerTypeID	nchar(10)
	CustomerDesc	ntext

Region

	Column Name	Conden…
🔑	RegionID	int
	RegionDescription	nchar(50)

3.1 Click on the northwind database you created earlier.



Databases

Create New Templates »

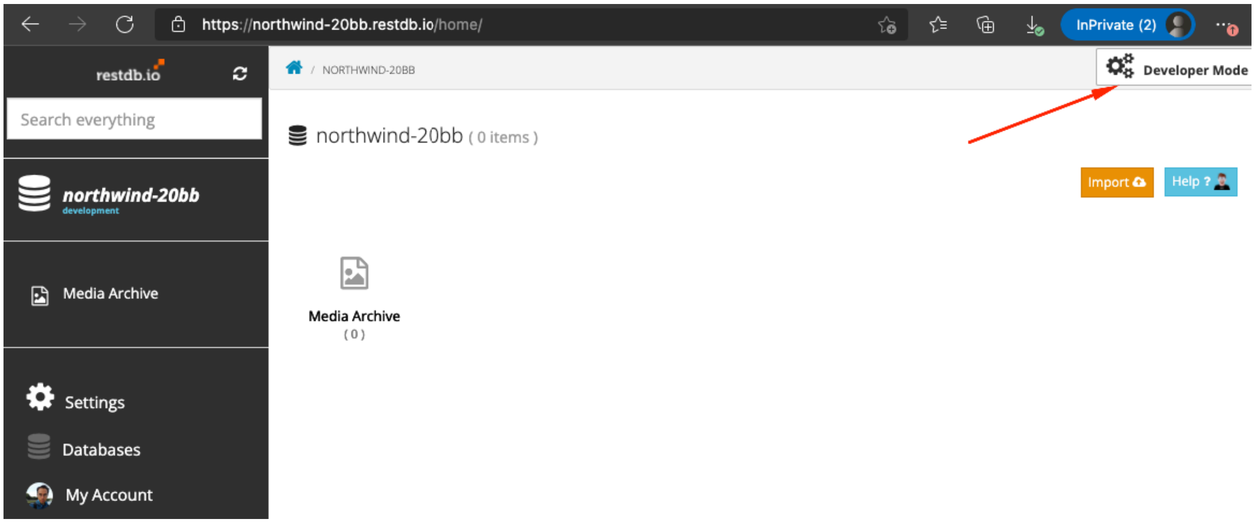
Enjoying restdb.io?

If you do, we would be honoured if you would tell others about it. We would also be very happy if you subscribed to one of our paid plans if you are using our service in production.

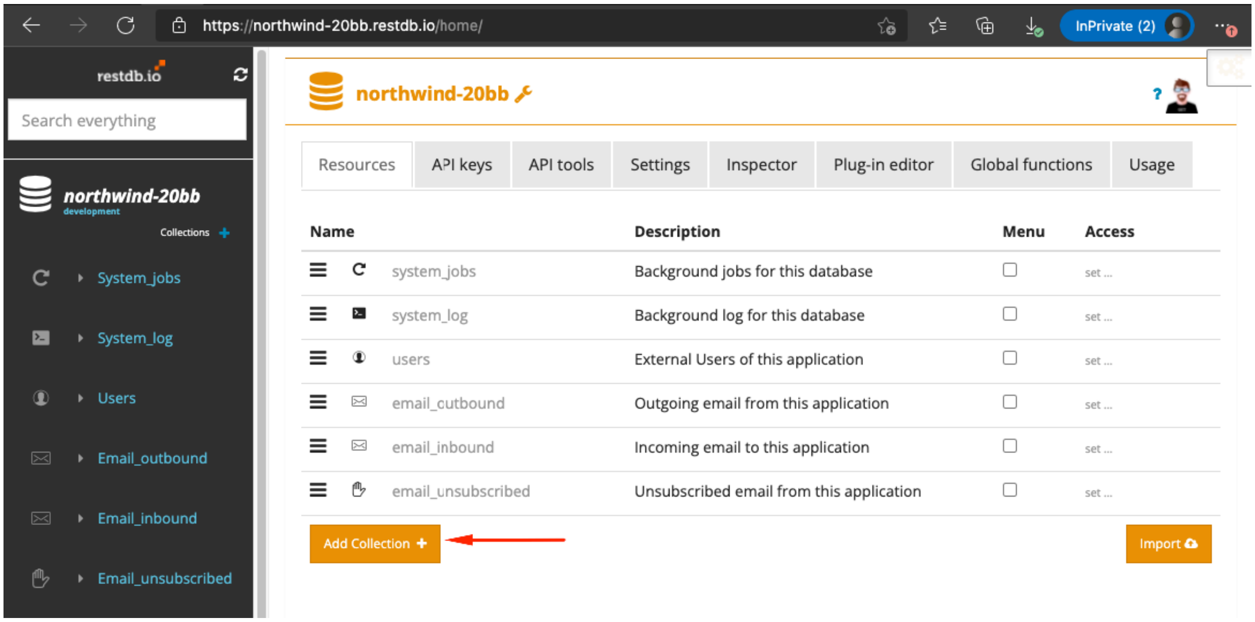
You can get a *standard* plan for only $19 per month (paid yearly). This will give you a much faster API, more data, more users, secure custom domain, backups and prioritized support.

Name	Plan		Owner	
northwind-20bb New	Develop	upgrade!	luis@luisbeltran.mx	Users and Settings ⚙

3.2 Switch to Developer mode

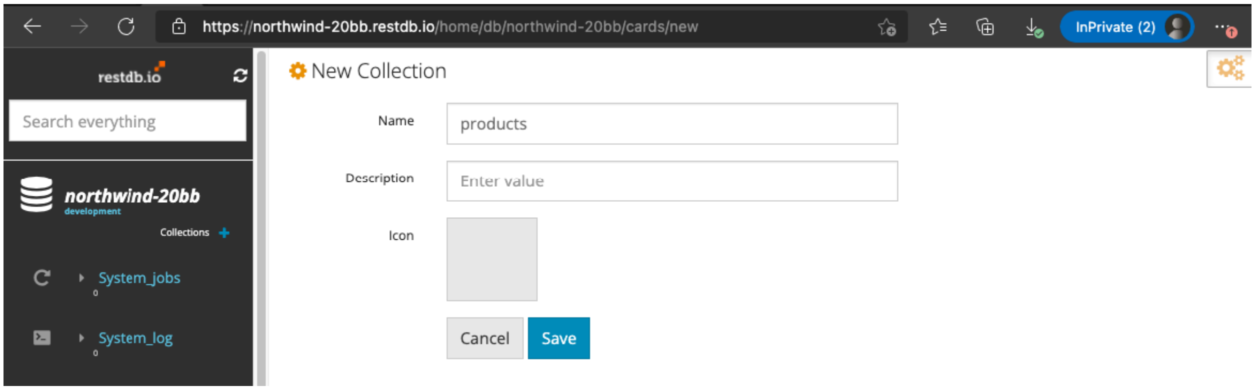


3.3 Click on Add Collection



Name	Description	Menu	Access
system_jobs	Background jobs for this database	□	set ...
system_log	Background log for this database	□	set ...
users	External Users of this application	□	set ...
email_outbound	Outgoing email from this application	□	set ...
email_inbound	Incoming email to this application	□	set ...
email_unsubscribed	Unsubscribed email from this application	□	set ...

3.4 The collection name will be products



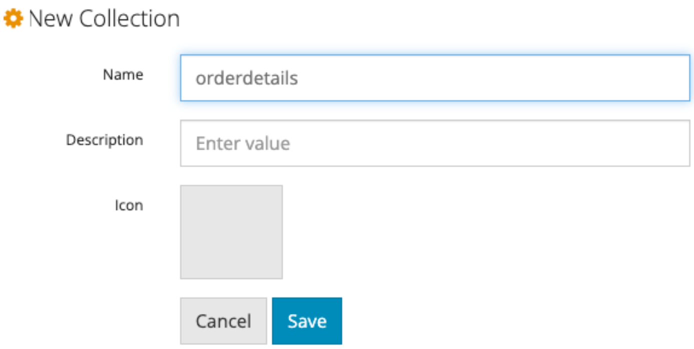
New Collection

Name: products

Description: Enter value

Icon

3.5 Add another collection: orderdetails



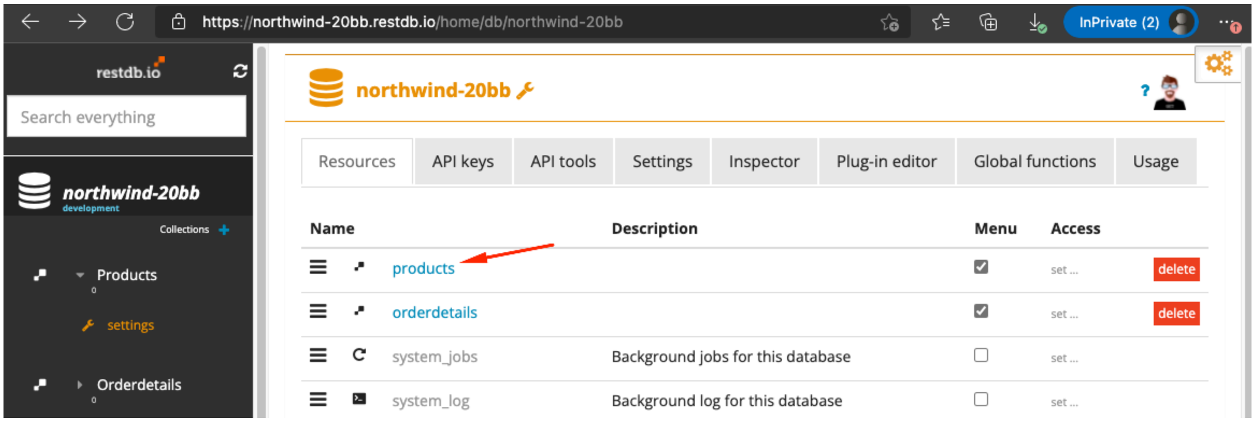
New Collection

Name: orderdetails

Description: Enter value

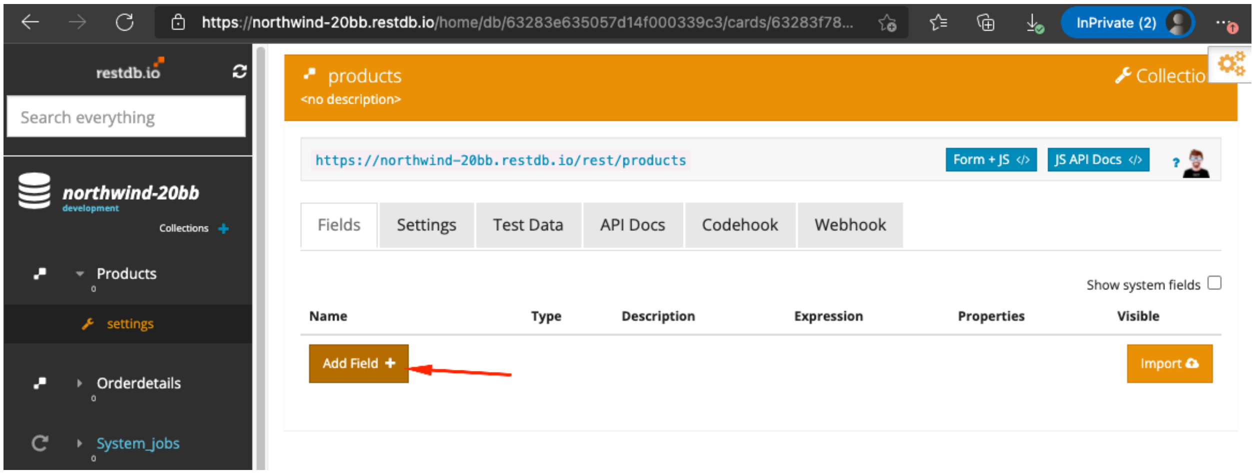
Icon

3.6 Click on Products

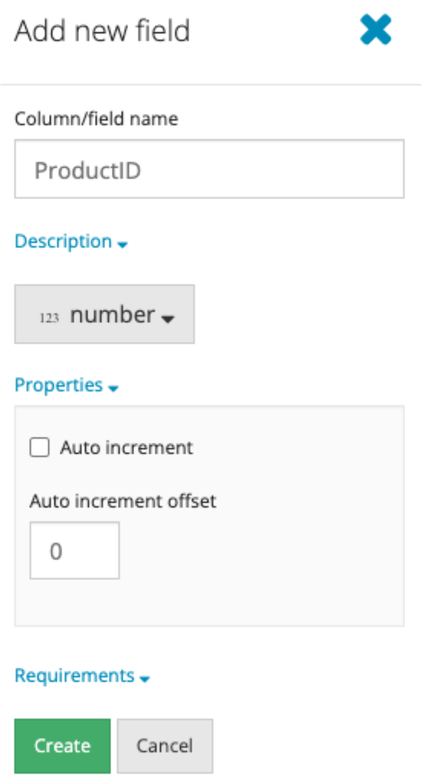


Name	Description	Menu	Access	
products		☑	set ...	delete
orderdetails		☑	set ...	delete
system_jobs	Background jobs for this database	□	set ...	
system_log	Background log for this database	□	set ...	

3.7 Click on Add Field



3.8 Column name is ProductID and data type is number. Click on Create.



Add new field

Column/field name

ProductID

Description

123 number

Properties

□ Auto increment

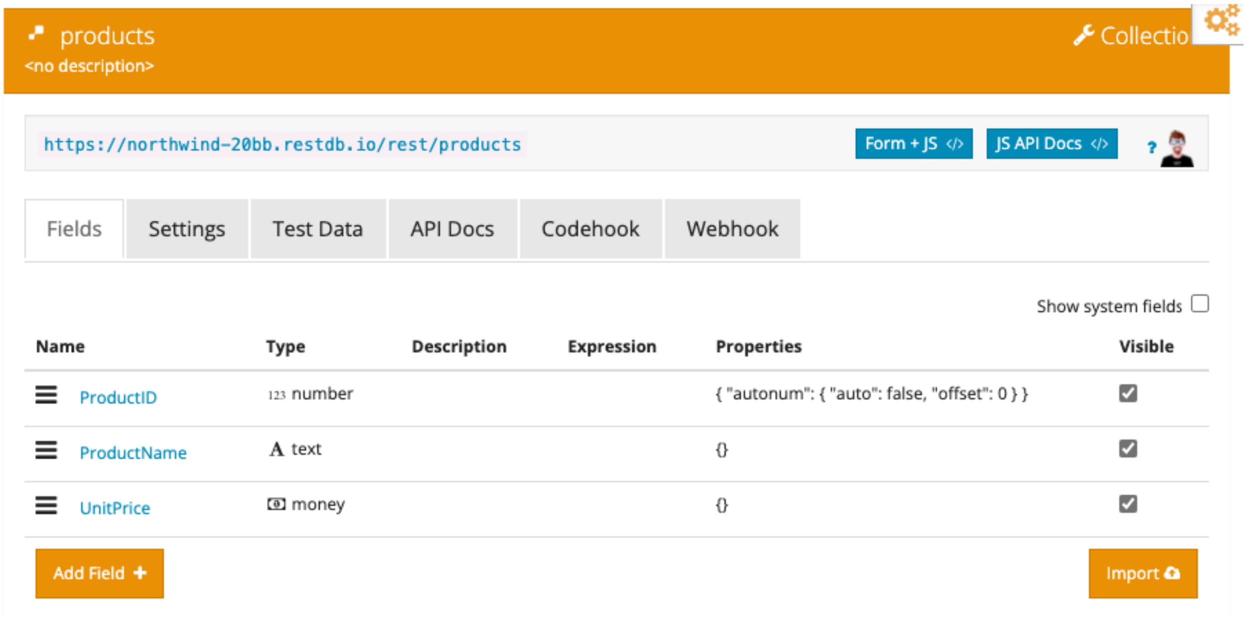
Auto increment offset

0

Requirements

3.9 Add two more fields:

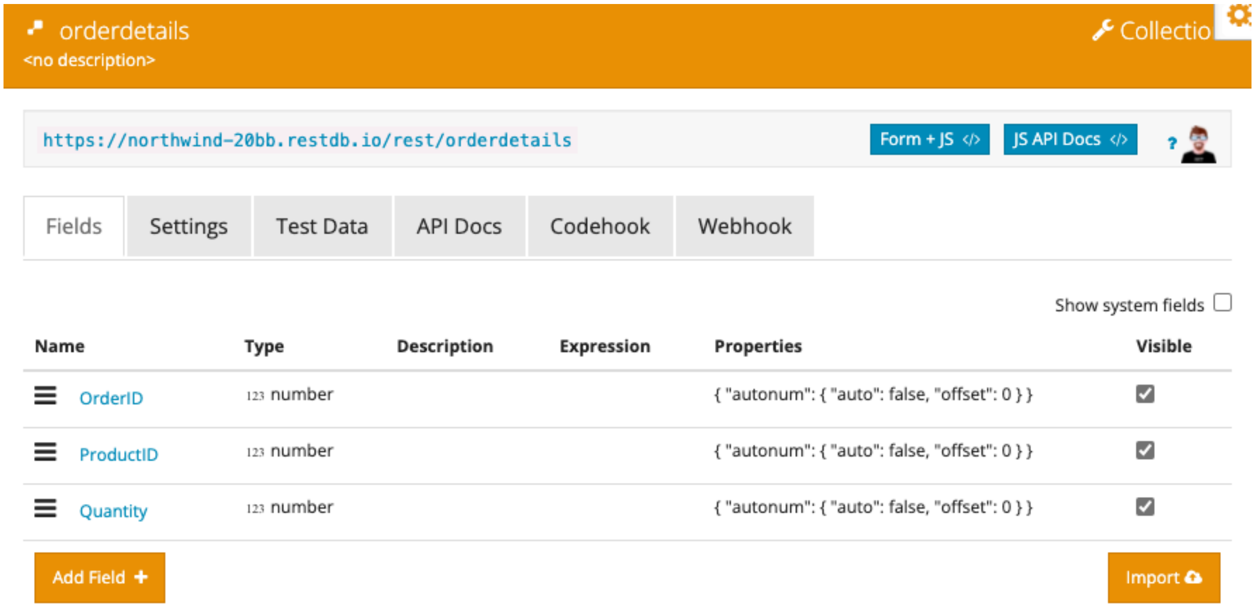
- ProductName (text)
- UnitPrice (money)



https://northwind-20bb.restdb.io/rest/products

Name	Type	Description	Expression	Properties	Visible
ProductID	123 number			{ "autonum": { "auto": false, "offset": 0 } }	☑
ProductName	A text			{}	☑
UnitPrice	money			{}	☑

3.10 Create the following structure in orderdetails collection:



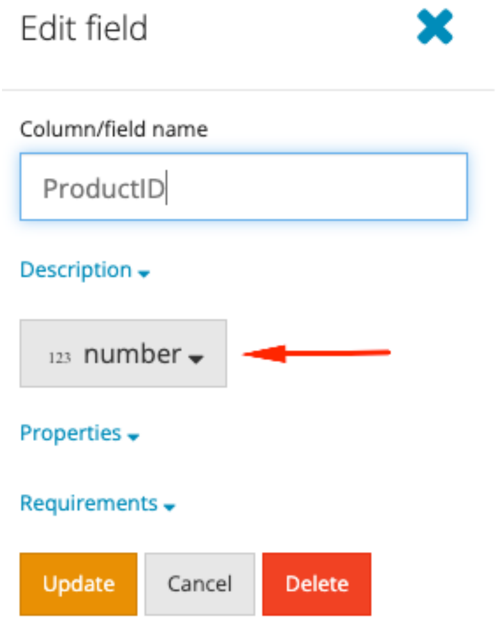
https://northwind-20bb.restdb.io/rest/orderdetails

Name	Type	Description	Expression	Properties	Visible
OrderID	123 number			{ "autonum": { "auto": false, "offset": 0 } }	☑
ProductID	123 number			{ "autonum": { "auto": false, "offset": 0 } }	☑
Quantity	123 number			{ "autonum": { "auto": false, "offset": 0 } }	☑

Task 4. Map relations between collections

In order to create “relations”, you must change the database schema.

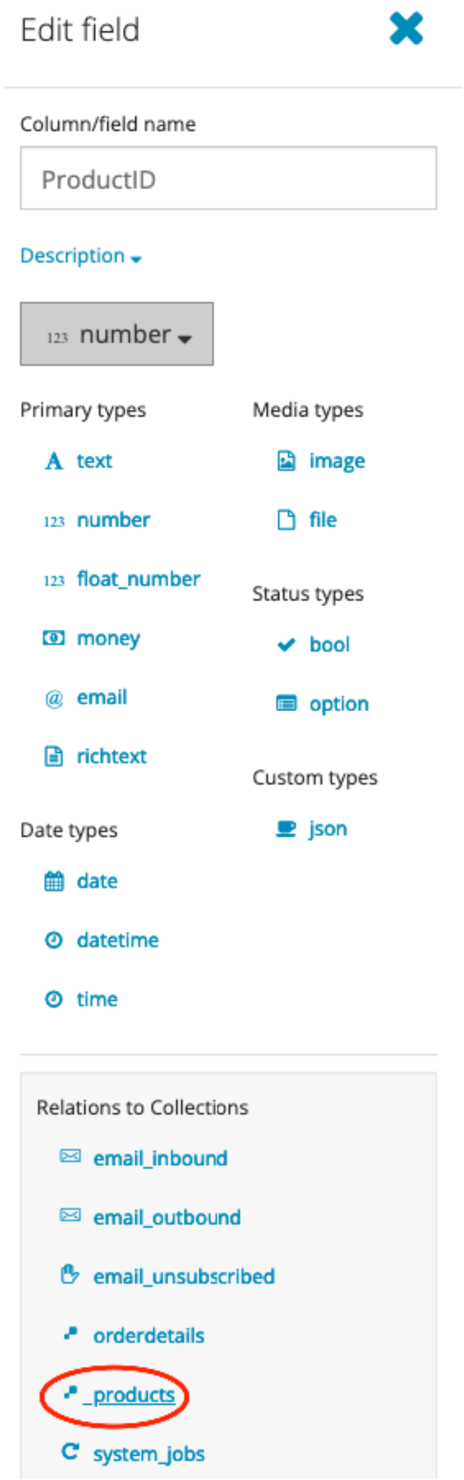
4.1 In orderdetails collection, click on ProductID to see its data type.



Column/field name: ProductID

123 number

4.2 Change its data type, from number to Products.

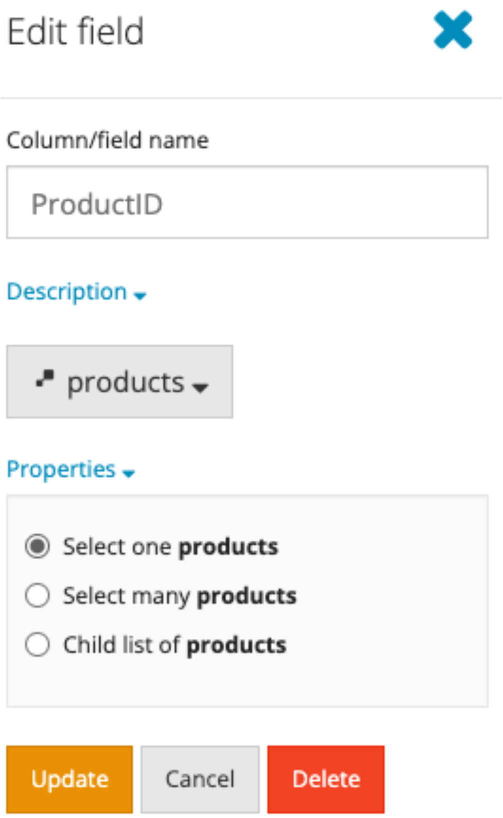


Edit field

Column/field name

ProductID

4.3 Select “Select one product” and Confirm the operation by clicking on Update



Edit field

Column/field name

ProductID

Description

products

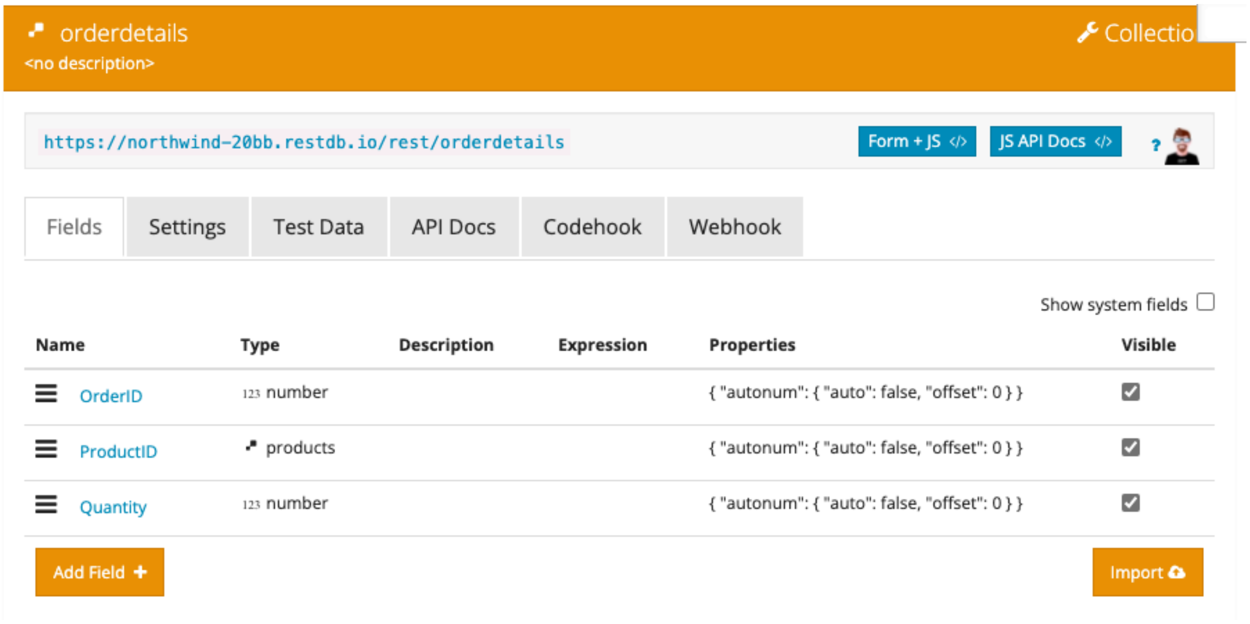
Properties

● Select one **products**

○ Select many **products**

○ Child list of **products**

Expected output



orderdetails

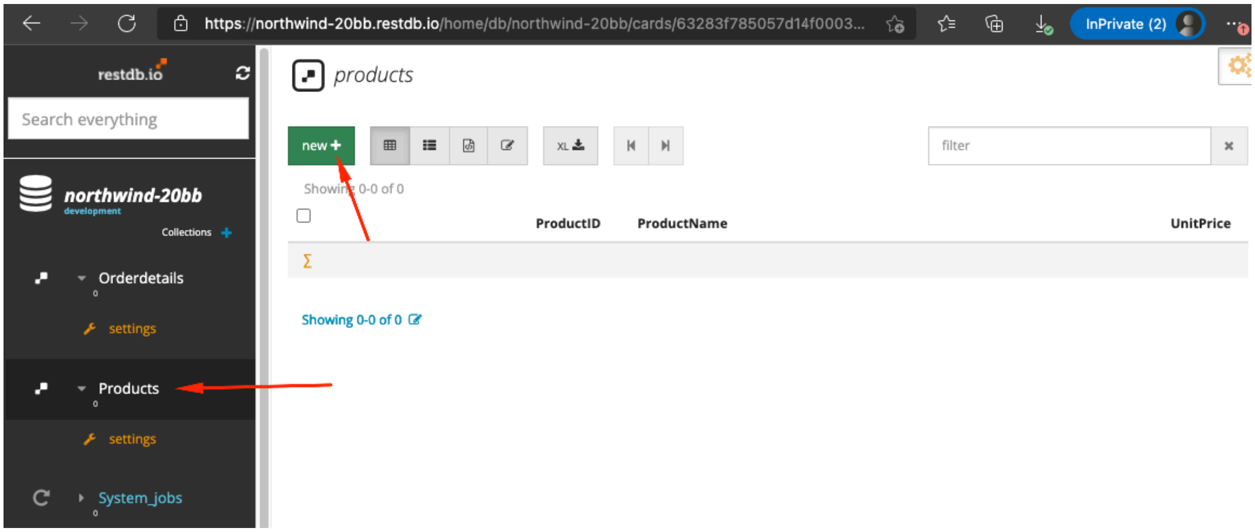
<no description>

https://northwind-20bb.restdb.io/rest/orderdetails

Show system fields □

Name	Type	Description	Expression	Properties	Visible
OrderID	123 number			{ "autonum": { "auto": false, "offset": 0 } }	☑
ProductID	products			{ "autonum": { "auto": false, "offset": 0 } }	☑
Quantity	123 number			{ "autonum": { "auto": false, "offset": 0 } }	☑

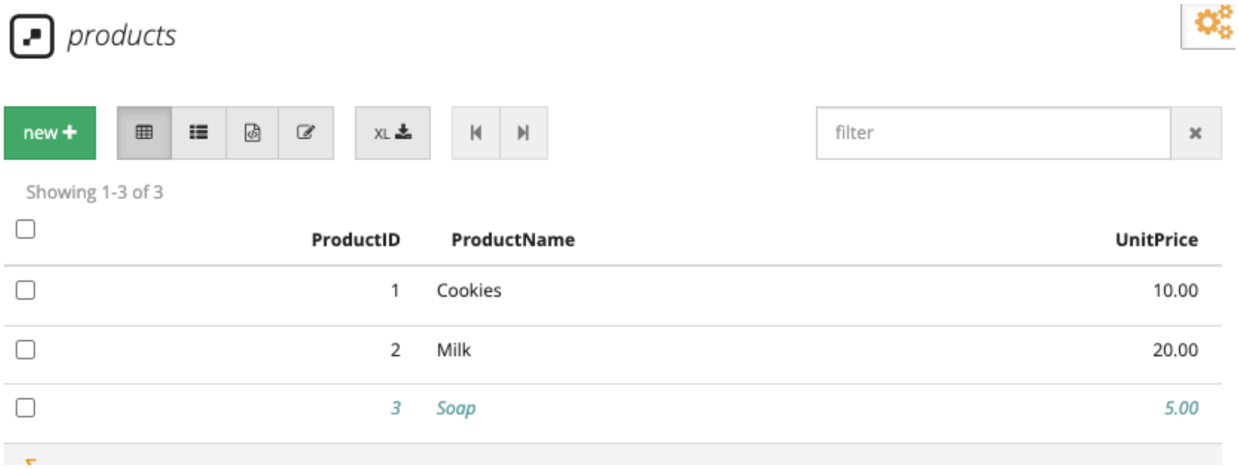
4.4 Now let's add some data. Click on Products and then on New + button



4.5 Add the following products

ProductID	ProductName	UnitPrice
1	Cookies	10
2	Milk	20
3	Soap	5

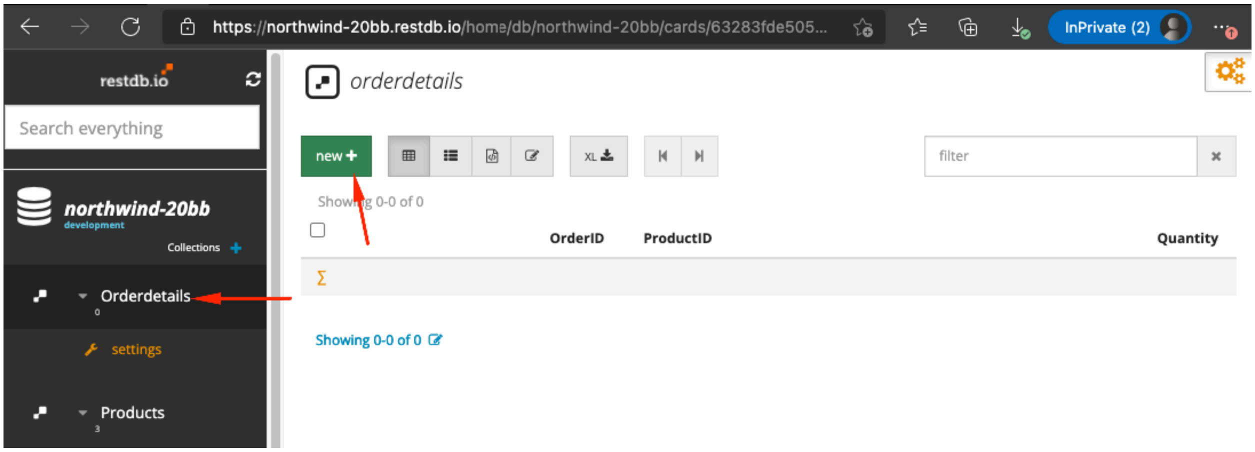
Expected result



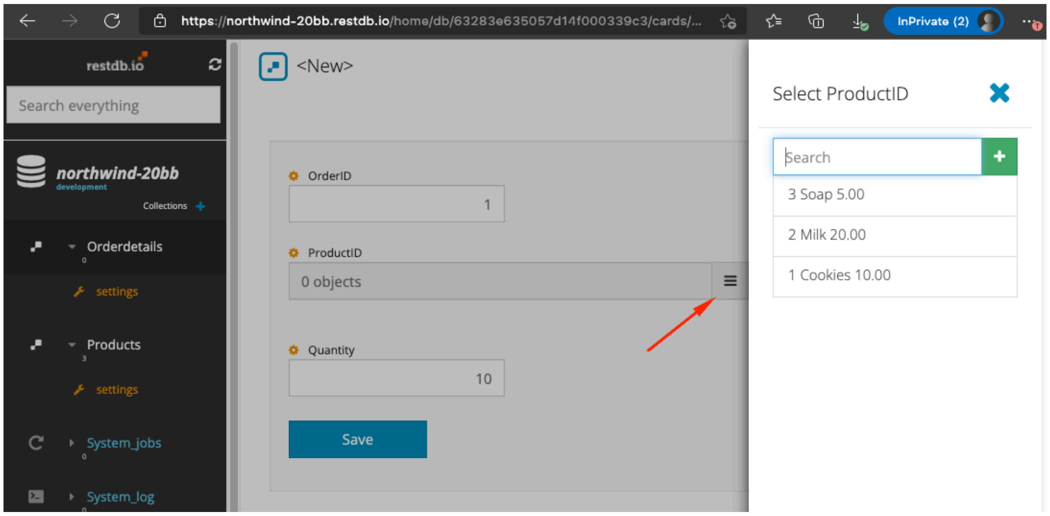
	ProductID	ProductName	UnitPrice
□	1	Cookies	10.00
□	2	Milk	20.00
□	3	Soap	5.00

You have just added 3 documents to the products collection.

4.6 Now try to add an orderdetails item



4.7 When you enter data, for ProductID you actually have to select a member from products collection



OrderID: 1

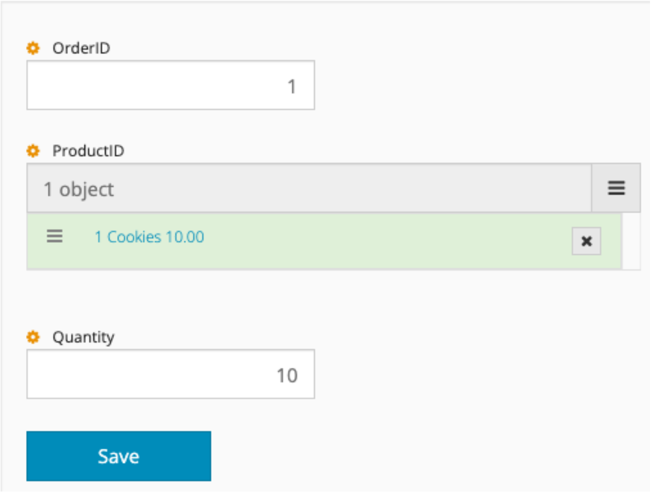
ProductID: 0 objects

Quantity: 10

3 Soap 5.00

2 Milk 20.00

1 Cookies 10.00



OrderID: 1

ProductID: 1 object

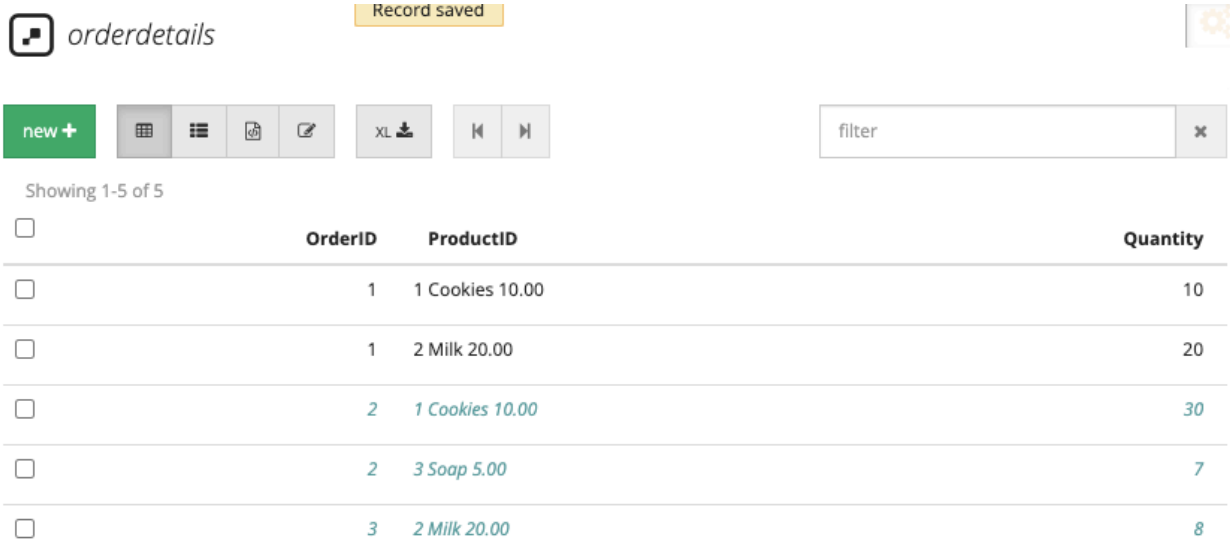
1 Cookies 10.00

Quantity: 10

Add the following data to the collection

OrderID	Product	Quantity
1	Cookies	10
1	Milk	20
2	Cookies	30
2	Soap	7
3	Milk	8

Expected output

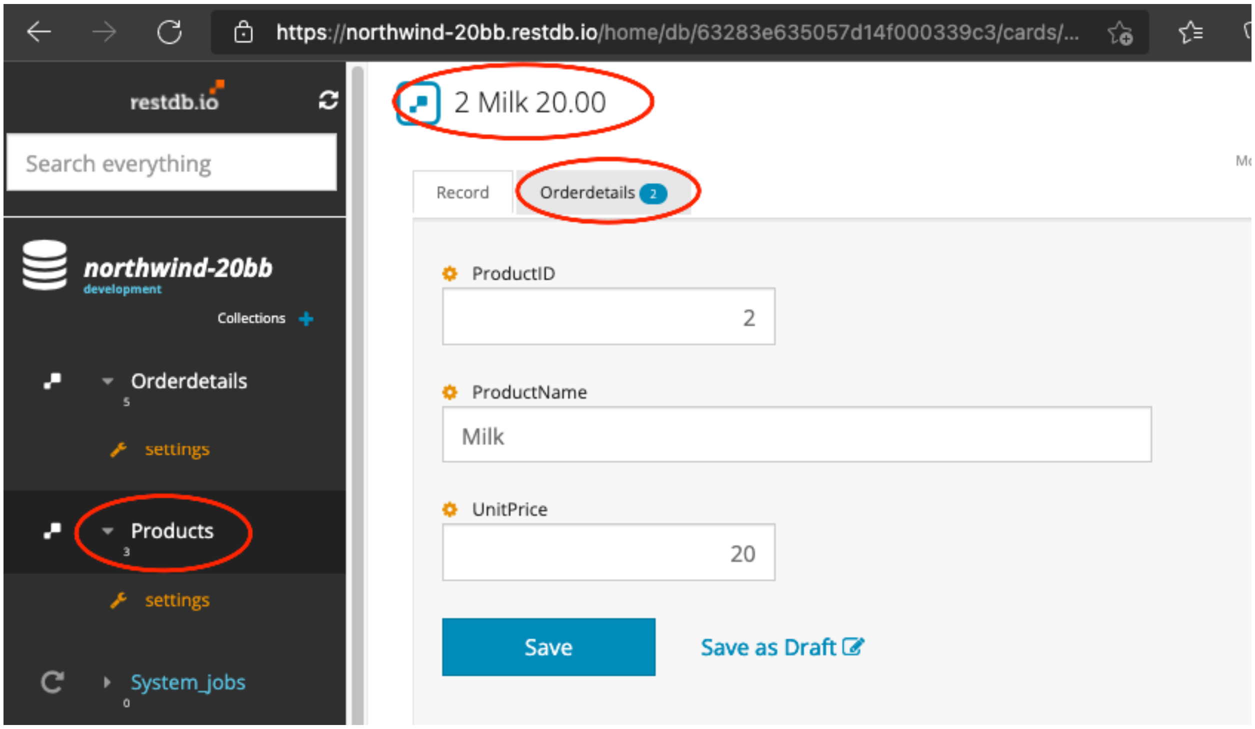


Showing 1-5 of 5

□	OrderID	ProductID	Quantity
□	1	1 Cookies 10.00	10
□	1	2 Milk 20.00	20
□	*2*	*1 Cookies 10.00*	*30*
□	*2*	*3 Soap 5.00*	*7*
□	*3*	*2 Milk 20.00*	*8*

As we now have a data schema that "knows" about the relations between our collections, restdb.io provides you with some really useful productivity features.

4.8 Navigate to any Product and you will see all orderdetails for that particular product.

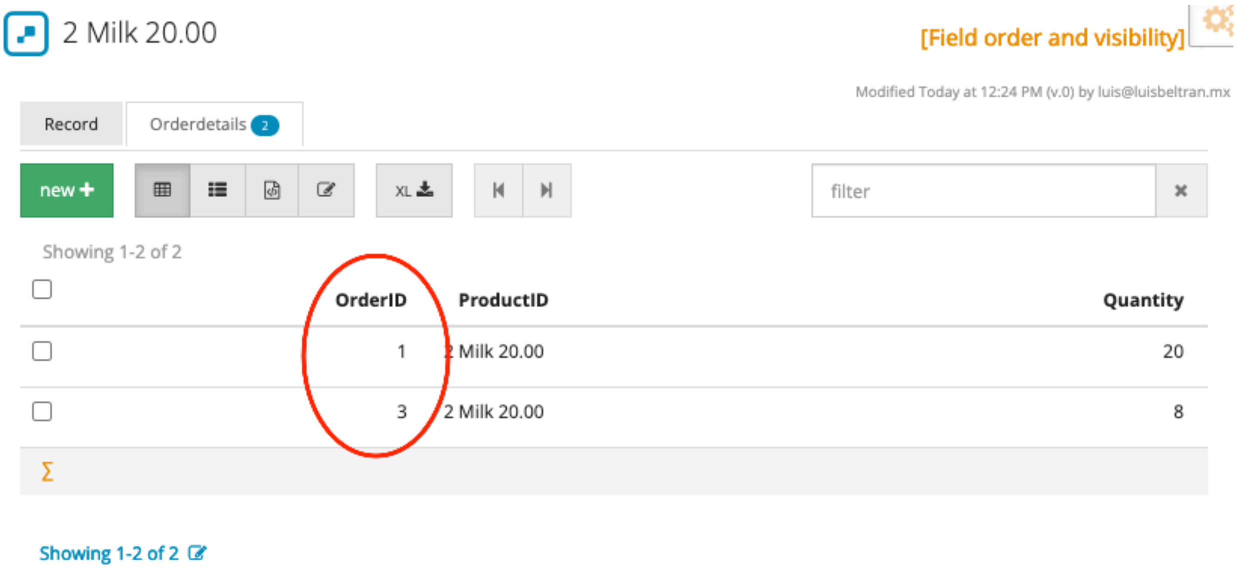


2 Milk 20.00

ProductID: 2

ProductName: Milk

UnitPrice: 20



2 Milk 20.00

Modified Today at 12:24 PM (v.0) by luis@luisbeltran.mx

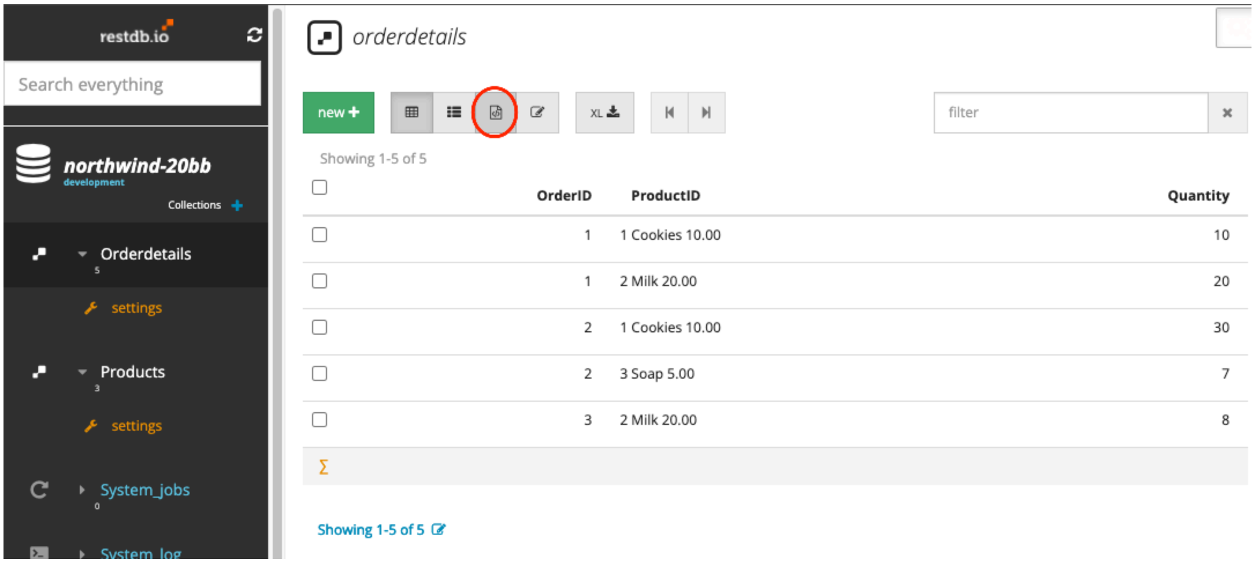
Showing 1-2 of 2

□	OrderID	ProductID	Quantity
□	1	2 Milk 20.00	20
□	3	2 Milk 20.00	8

Showing 1-2 of 2

Task 5. See JSON data

5.1 Firstly, go back to the orderdetails collection and click on the json data icon



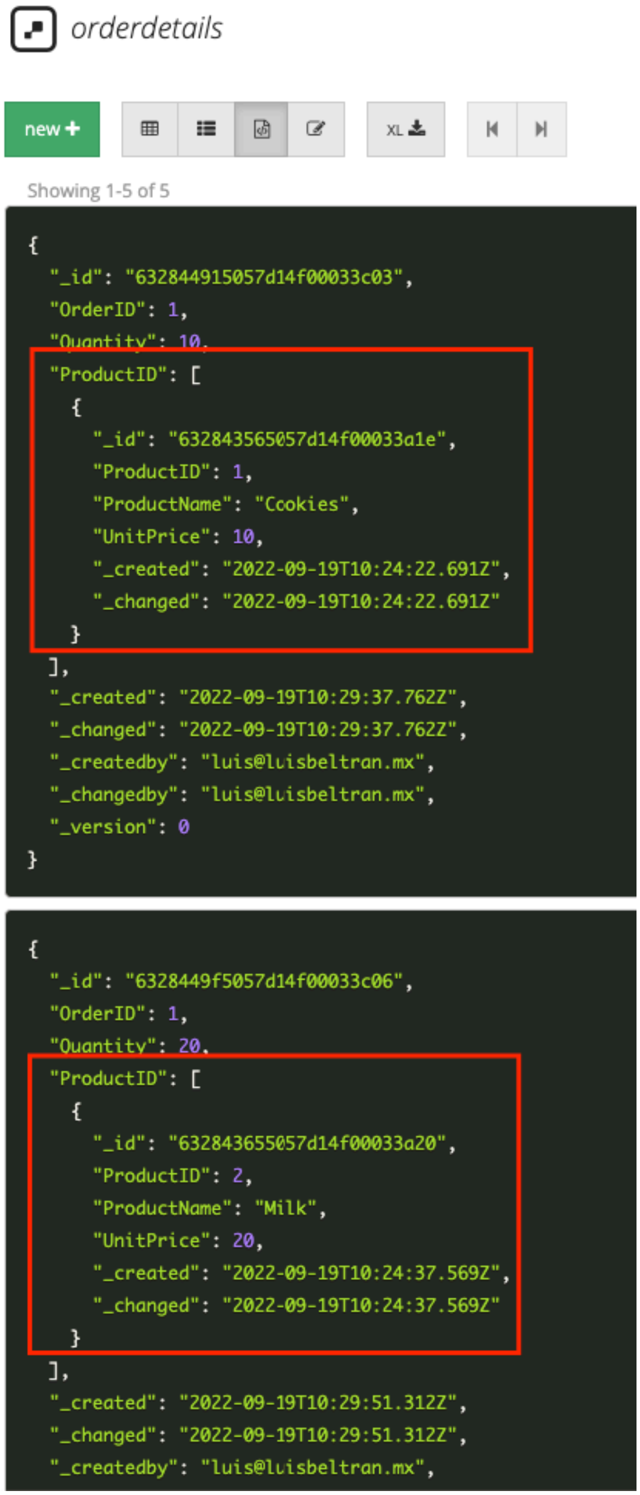
Showing 1-5 of 5

☐	OrderID	ProductID	Quantity
☐	1	1 Cookies 10.00	10
☐	1	2 Milk 20.00	20
☐	2	1 Cookies 10.00	30
☐	2	3 Soap 5.00	7
☐	3	2 Milk 20.00	8

Σ

Showing 1-5 of 5

5.2 You will see that for each element, you can immediately access its products information (a document that contains other documents).



```
{
  "_id": "632844915057d14f00033c03",
  "OrderID": 1,
  "Quantity": 10,
  "ProductID": [
    {
      "_id": "632843565057d14f00033a1e",
      "ProductID": 1,
      "ProductName": "Cookies",
      "UnitPrice": 10,
      "_created": "2022-09-19T10:24:22.691Z",
      "_changed": "2022-09-19T10:24:22.691Z"
    }
  ],
  "_created": "2022-09-19T10:29:37.762Z",
  "_changed": "2022-09-19T10:29:37.762Z",
  "_createdby": "luis@luisbeltran.mx",
  "_changedby": "luis@luisbeltran.mx",
  "_version": 0
}
```

```
{
  "_id": "6328449f5057d14f00033c06",
  "OrderID": 1,
  "Quantity": 20,
  "ProductID": [
    {
      "_id": "632843655057d14f00033a20",
      "ProductID": 2,
      "ProductName": "Milk",
      "UnitPrice": 20,
      "_created": "2022-09-19T10:24:37.569Z",
      "_changed": "2022-09-19T10:24:37.569Z"
    }
  ],
  "_created": "2022-09-19T10:29:51.312Z",
  "_changed": "2022-09-19T10:29:51.312Z",
  "_createdby": "luis@luisbeltran.mx",
```

We see that the field ProductID which used to be just a number, has been replaced with the actual document (record) it refers.

This differs from SQL databases, where we would have to join the two collections to produce a similar result.

In the NoSQL world we embed and duplicate, and that's ok.

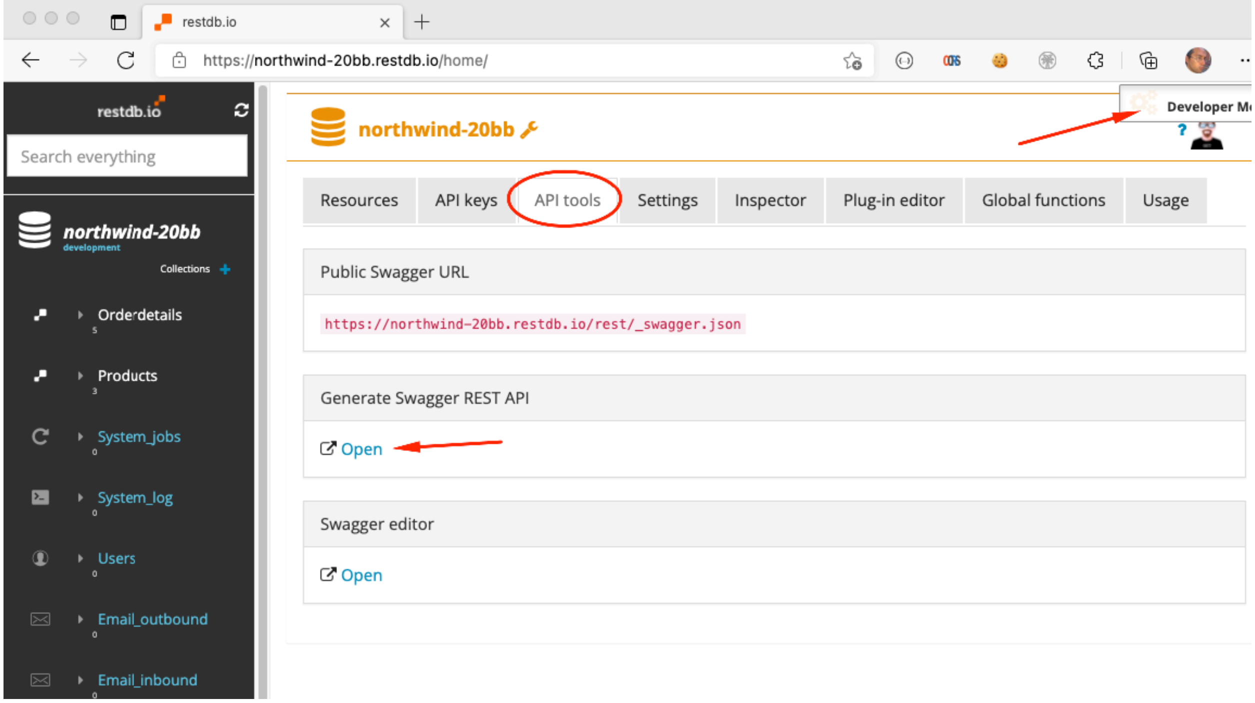
Task 6. Use the API

Querying your database is an essential part of any application. restdb.io uses plain URLs with simple parameters and JSON documents to query your database.

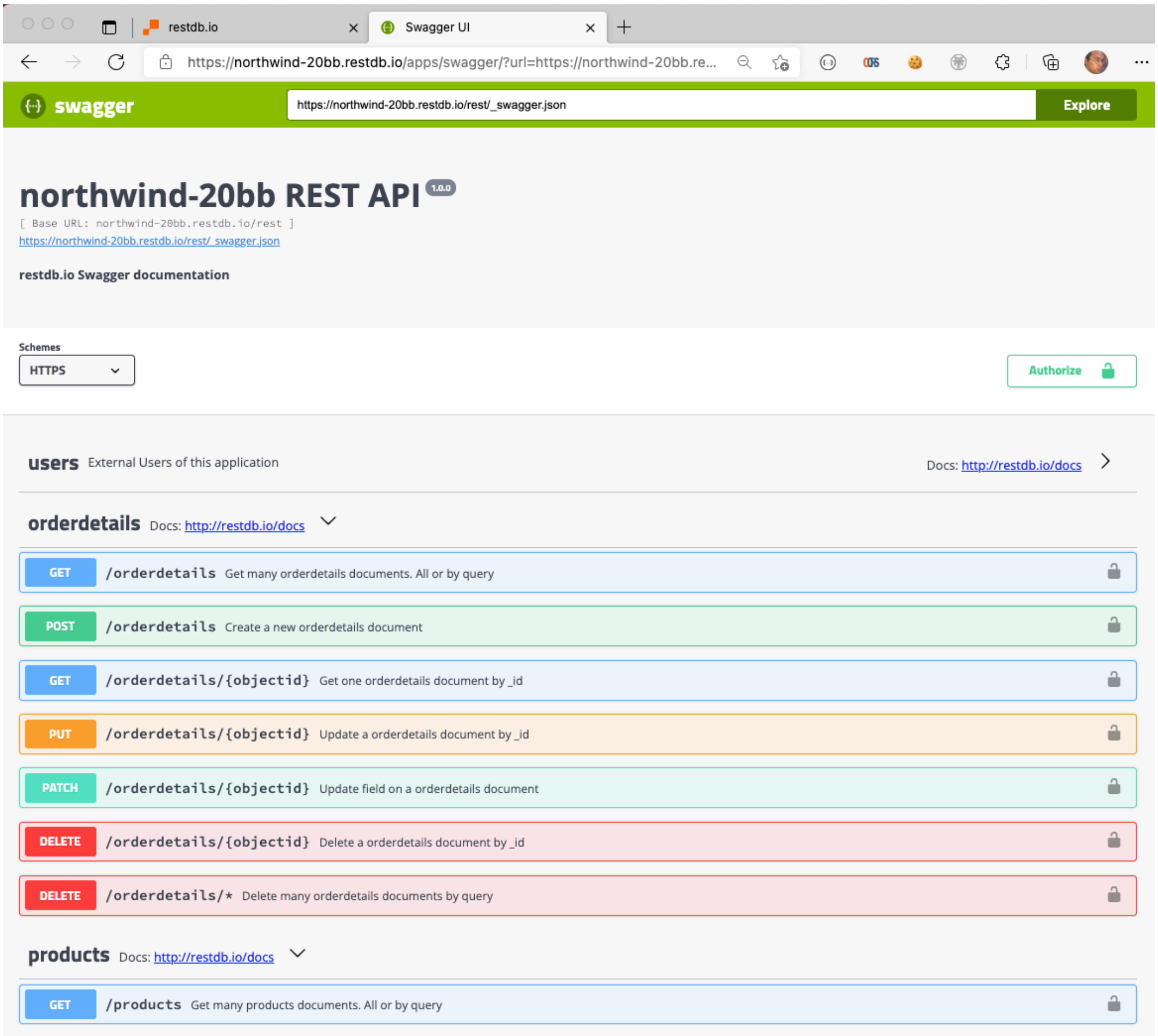
Database queries are created as valid JSON documents. A query object consists of fields and operators that make up a complete query. Please note that query fields are case sensitive.

RestDB.io autogenerates a complete JavaScript API for your database.

6.1 Enter Development mode, click on API tools and then Open the generated Swagger REST API



You will see access the Swagger documentation of the database REST API in order to easily test basic CRUD operations and more which directly impact the collections.



restdb.io
Swagger UI
https://northwind-20bb.restdb.io/apps/swagger/?url=https://northwind-20bb.re...
swagger
Explore

https://northwind-20bb.restdb.io/rest/_swagger.json

northwind-20bb REST API 1.0.0

[Base URL: northwind-20bb.restdb.io/rest]

https://northwind-20bb.restdb.io/rest/_swagger.json

restdb.io Swagger documentation

Schemes

HTTPS

users External Users of this application Docs: http://restdb.io/docs

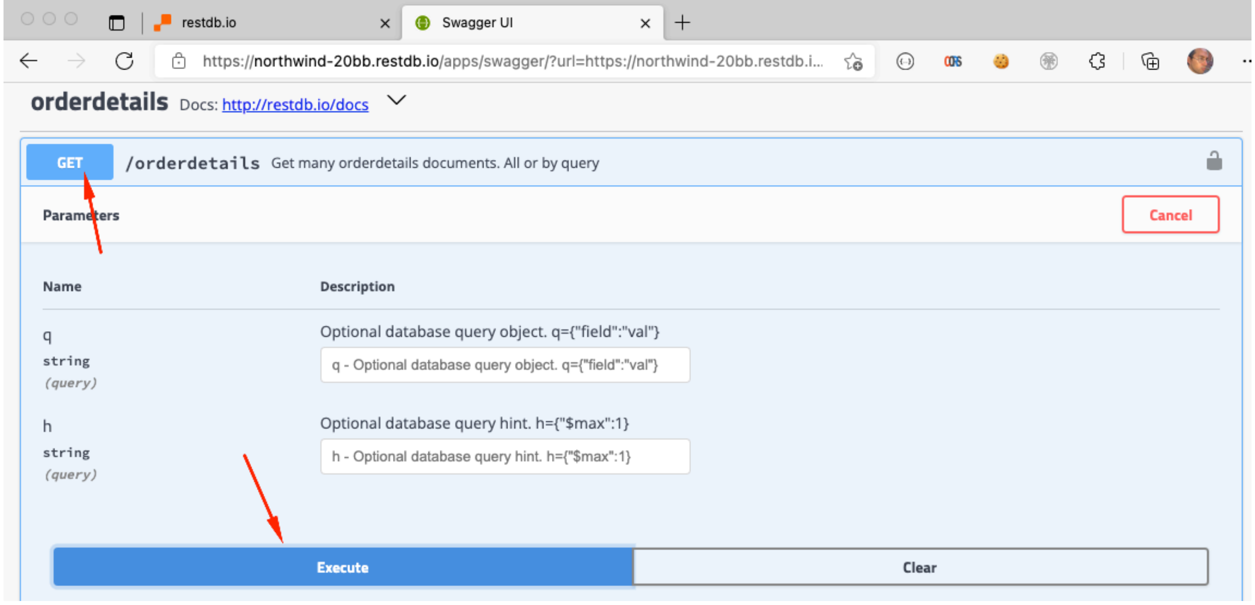
orderdetails Docs: http://restdb.io/docs

- GET `/orderdetails` Get many orderdetails documents. All or by query
- POST `/orderdetails` Create a new orderdetails document
- GET `/orderdetails/{objectid}` Get one orderdetails document by _id
- PUT `/orderdetails/{objectid}` Update a orderdetails document by _id
- PATCH `/orderdetails/{objectid}` Update field on a orderdetails document
- DELETE `/orderdetails/{objectid}` Delete a orderdetails document by _id
- DELETE `/orderdetails/*` Delete many orderdetails documents by query

products Docs: http://restdb.io/docs

- GET `/products` Get many products documents. All or by query

6.2 Click on the GET service of /orderdetails and then on Execute

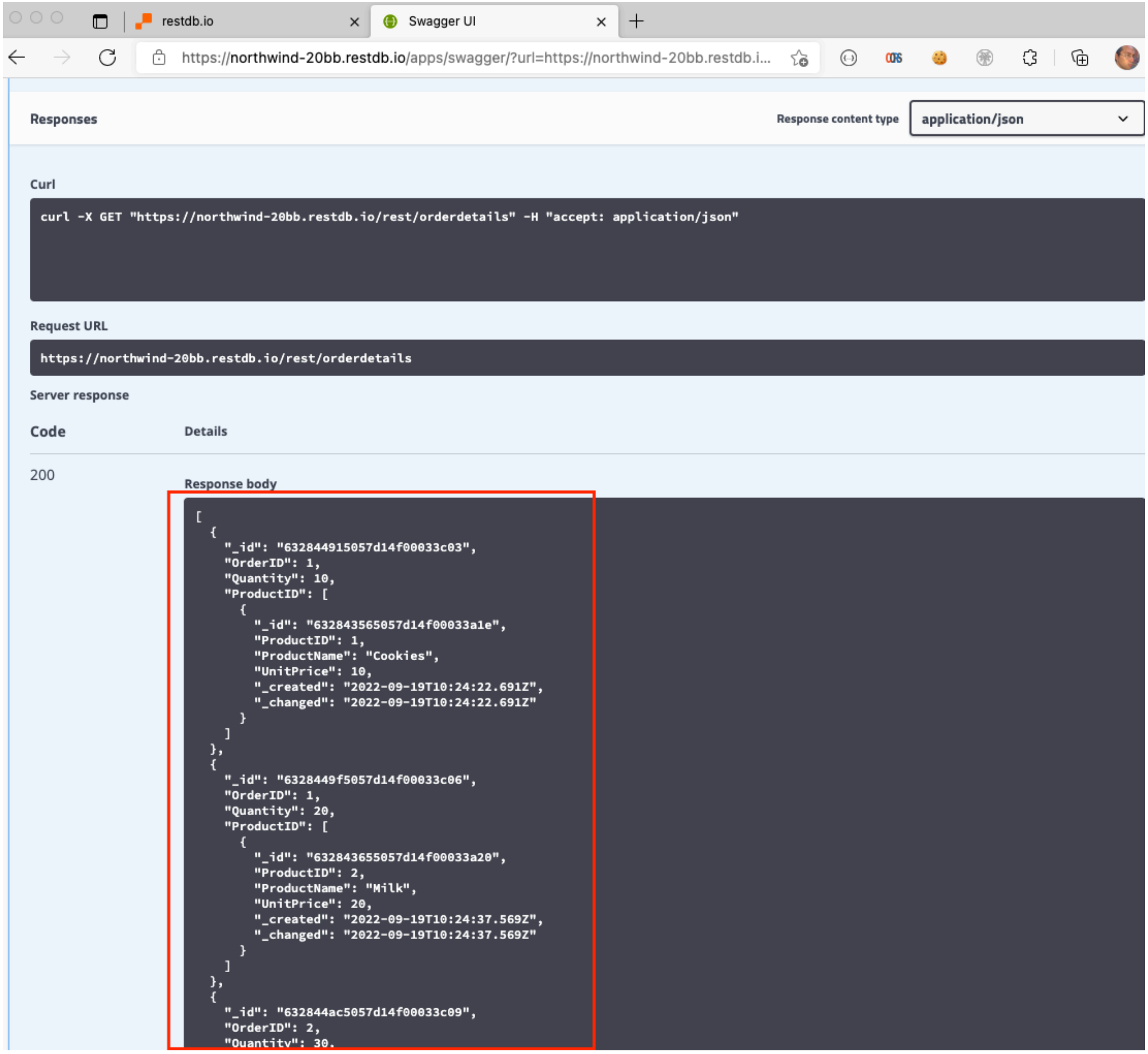


restdb.io
Swagger UI
https://northwind-20bb.restdb.io/apps/swagger/?url=https://northwind-20bb.restdb.i...
orderdetails
Docs: http://restdb.io/docs
GET
/orderdetails
Parameters
Cancel
Name
Description
Execute
Clear

/orderdetails Get many orderdetails documents. All or by query

Name	Description
q string *(query)*	Optional database query object. q={"field":"val"} q - Optional database query object. q={"field":"val"}
h string *(query)*	Optional database query hint. h={"$max":1} h - Optional database query hint. h={"$max":1}

You will see the results below along with the information for request (URL, expected server response, etc)



restdb.io
Swagger UI
https://northwind-20bb.restdb.io/apps/swagger/?url=https://northwind-20bb.restdb.i...
Responses
Response content type
application/json

Curl

```
curl -X GET "https://northwind-20bb.restdb.io/rest/orderdetails" -H "accept: application/json"
```

Request URL

```
https://northwind-20bb.restdb.io/rest/orderdetails
```

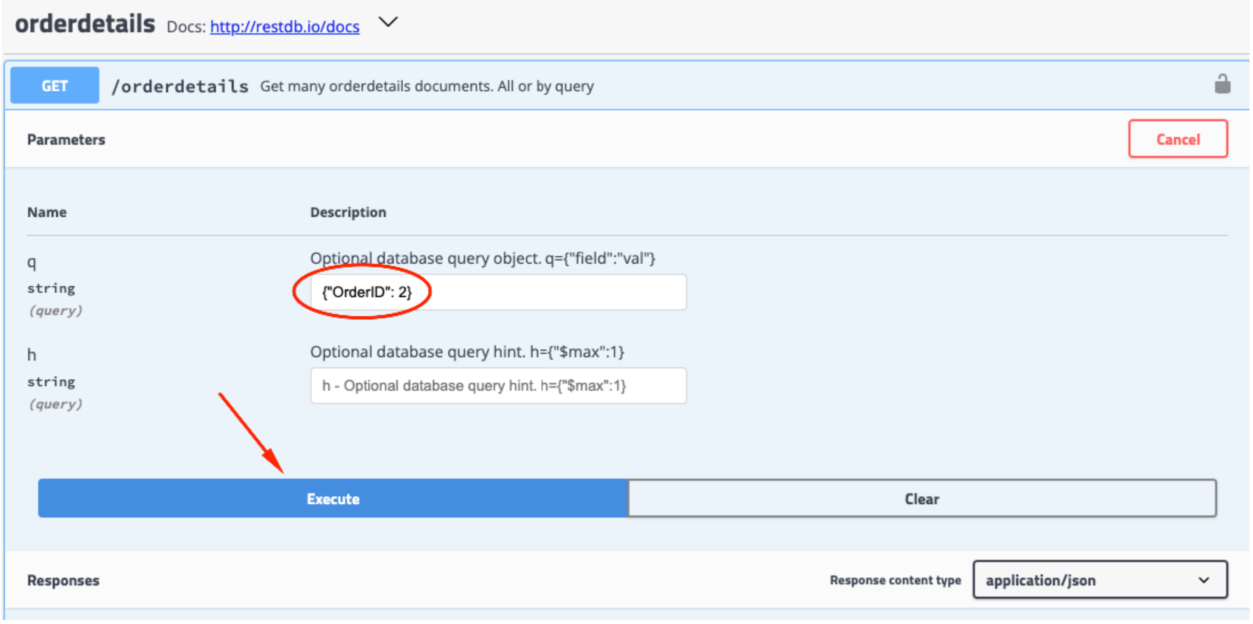
Server response

Code	Details
200	Response body

```
[
  {
    "_id": "632844915057d14f00033c03",
    "OrderID": 1,
    "Quantity": 10,
    "ProductID": [
      {
        "_id": "632843565057d14f00033a1e",
        "ProductID": 1,
        "ProductName": "Cookies",
        "UnitPrice": 10,
        "_created": "2022-09-19T10:24:22.691Z",
        "_changed": "2022-09-19T10:24:22.691Z"
      }
    ]
  },
  {
    "_id": "6328449f5057d14f00033c06",
    "OrderID": 1,
    "Quantity": 20,
    "ProductID": [
      {
        "_id": "632843655057d14f00033a20",
        "ProductID": 2,
        "ProductName": "Milk",
        "UnitPrice": 20,
        "_created": "2022-09-19T10:24:37.569Z",
        "_changed": "2022-09-19T10:24:37.569Z"
      }
    ]
  },
  {
    "_id": "632844ac5057d14f00033c09",
    "OrderID": 2,
    "Quantity": 30,
```

6.3 Now go back to the /GET method and in the q Parameter textbox, enter the following value (and click on Execute):

{"OrderID": 2}



orderdetails Docs: http://restdb.io/docs

GET /orderdetails Get many orderdetails documents. All or by query

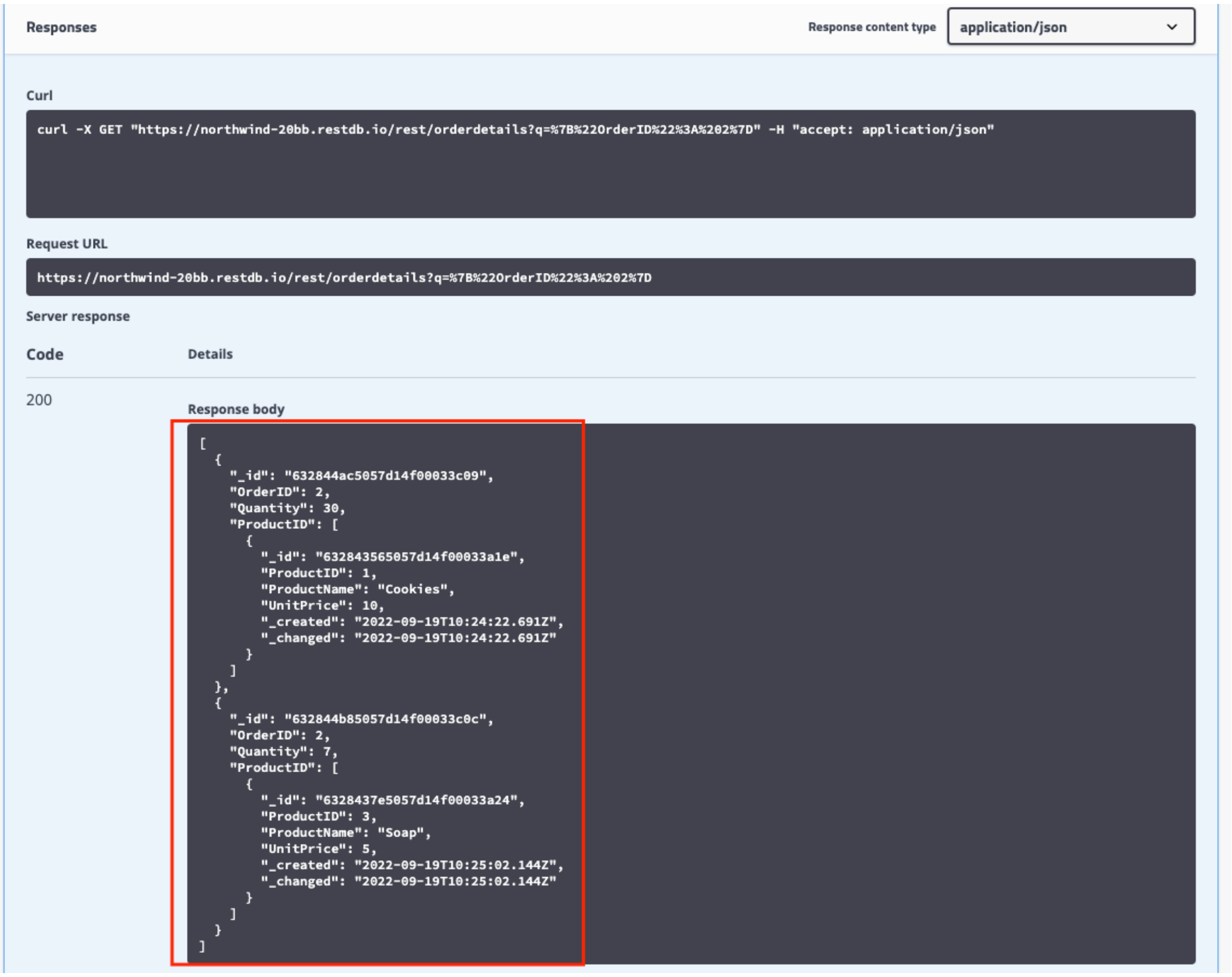
Parameters

Name	Description
q string *(query)*	Optional database query object. q={"field":"val"} {"OrderID": 2}
h string *(query)*	Optional database query hint. h={"$max":1} h - Optional database query hint. h={"$max":1}

Responses

Response content type: application/json

As expected, only the Order Details from Order 2 will appear:



Curl

```
curl -X GET "https://northwind-20bb.restdb.io/rest/orderdetails?q=%7B%22OrderID%22%3A%202%7D" -H "accept: application/json"
```

Request URL

```
https://northwind-20bb.restdb.io/rest/orderdetails?q=%7B%22OrderID%22%3A%202%7D
```

Server response

Code: 200

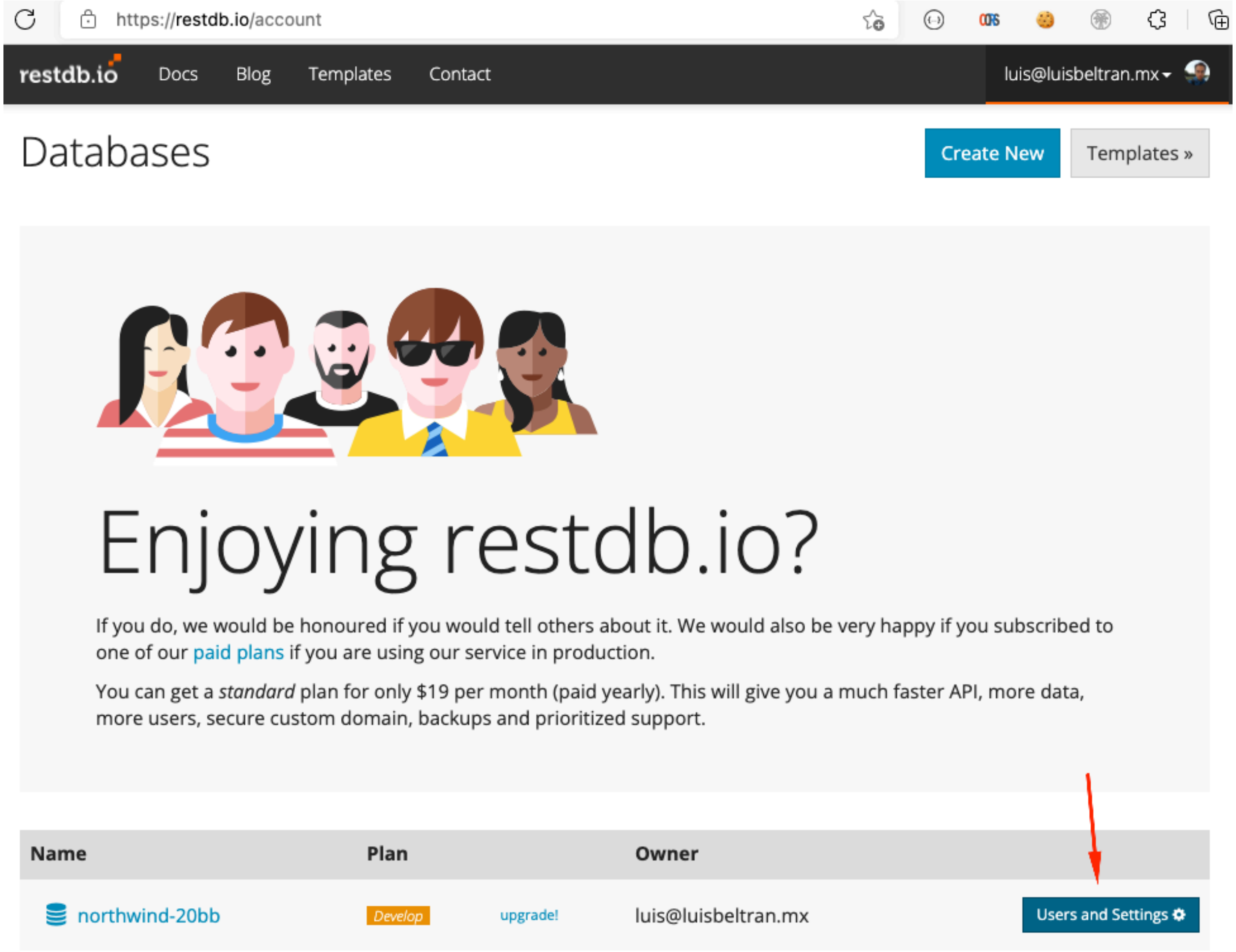
Details

Response body

```
[
  {
    "_id": "632844ac5057d14f00033c09",
    "OrderID": 2,
    "Quantity": 30,
    "ProductID": [
      {
        "_id": "632843565057d14f00033a1e",
        "ProductID": 1,
        "ProductName": "Cookies",
        "UnitPrice": 10,
        "_created": "2022-09-19T10:24:22.691Z",
        "_changed": "2022-09-19T10:24:22.691Z"
      }
    ]
  },
  {
    "_id": "632844b85057d14f00033c0c",
    "OrderID": 2,
    "Quantity": 7,
    "ProductID": [
      {
        "_id": "6328437e5057d14f00033a24",
        "ProductID": 3,
        "ProductName": "Soap",
        "UnitPrice": 5,
        "_created": "2022-09-19T10:25:02.144Z",
        "_changed": "2022-09-19T10:25:02.144Z"
      }
    ]
  }
]
```

You can use the API to develop an application that queries your NoSQL database. If you are developing a web application, you must enable an additional permission.

6.4 Go back to your main restdb.io account page and click on Users and Settings from your database.



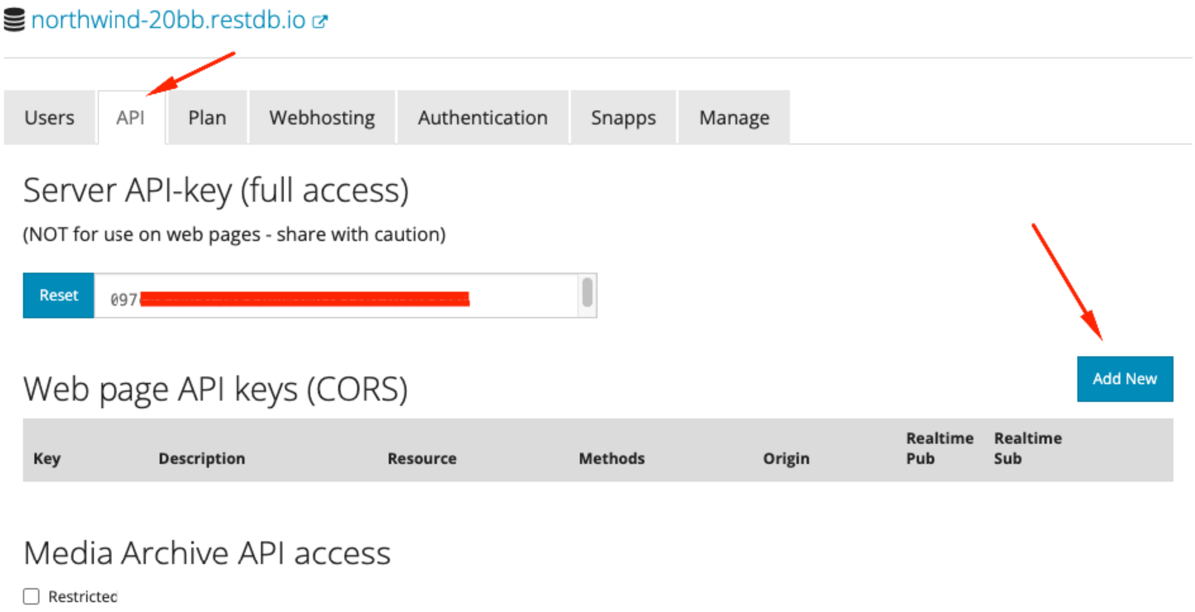
Enjoying restdb.io?

If you do, we would be honoured if you would tell others about it. We would also be very happy if you subscribed to one of our paid plans if you are using our service in production.

You can get a *standard* plan for only $19 per month (paid yearly). This will give you a much faster API, more data, more users, secure custom domain, backups and prioritized support.

Name	Plan		Owner	
northwind-20bb	Develop	upgrade!	luis@luisbeltran.mx	Users and Settings

6.5 Click on the API tab and click on Add New CORS API Key.



northwind-20bb.restdb.io

Server API-key (full access)

(NOT for use on web pages - share with caution)

Reset | 097[redacted]

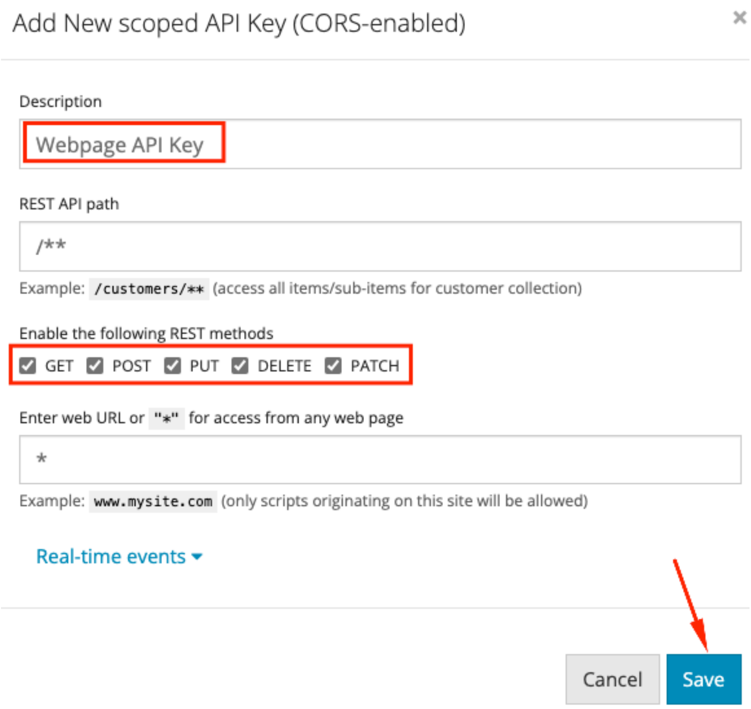
Web page API keys (CORS)

Key	Description	Resource	Methods	Origin	Realtime Pub	Realtime Sub

Media Archive API access

□ Restrictec

6.5 Enter a description and choose some REST methods. Leave the rest of the fields with their default values. Click on Save.



Add New scoped API Key (CORS-enabled)

Description: Webpage API Key

REST API path: /**

Example: `/customers/**` (access all items/sub-items for customer collection)

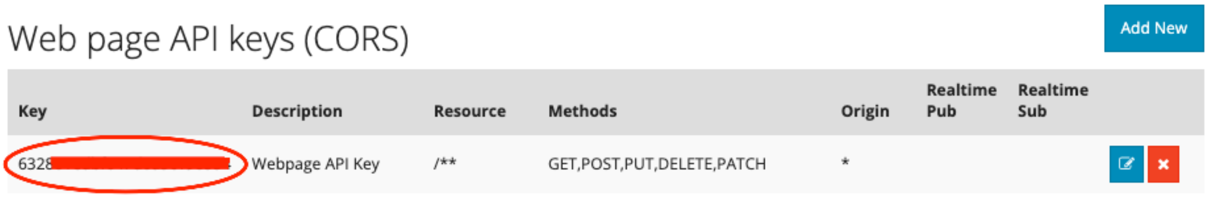
Enable the following REST methods

☑ GET ☑ POST ☑ PUT ☑ DELETE ☑ PATCH

Enter web URL or `"*"` for access from any web page: *

Example: `www.mysite.com` (only scripts originating on this site will be allowed)

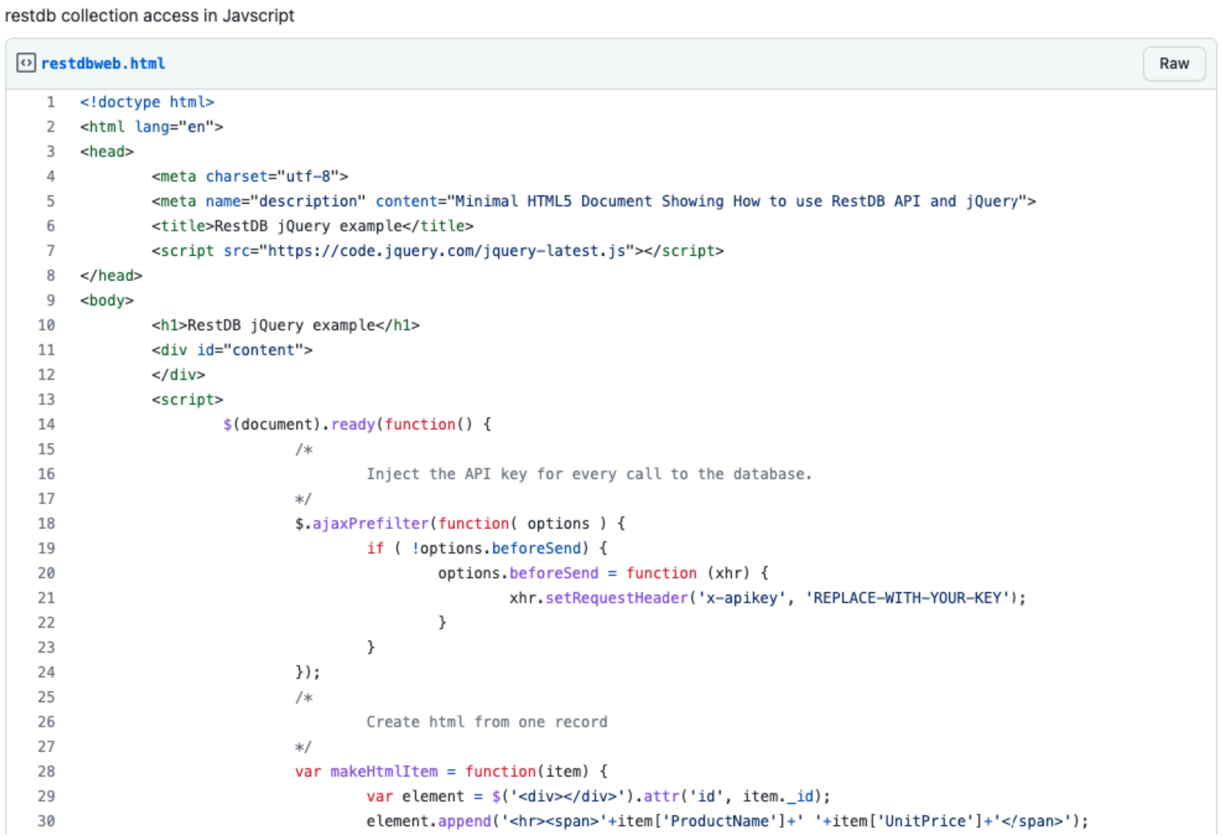
6.6 Copy the key that has been created for you.



Web page API keys (CORS)

Key	Description	Resource	Methods	Origin	Realtime Pub	Realtime Sub
6328[redacted]	Webpage API Key	/**	GET,POST,PUT,DELETE,PATCH	*		

6.7 Access the following code and click on Raw. Then right click and Save the file
https://gist.github.com/icebeam7/cc012e71c22a106ae776ef1370edeb6e

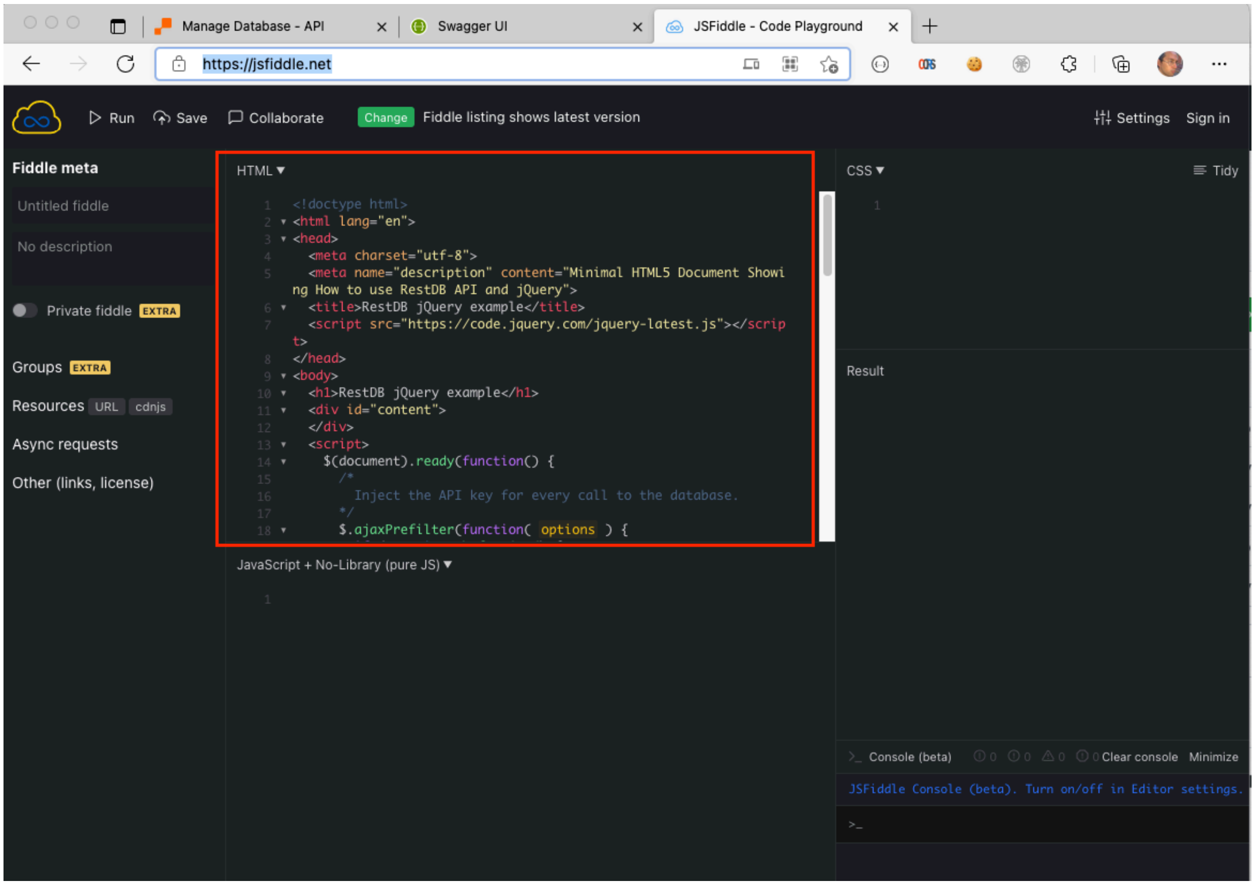


restdb collection access in Javscript

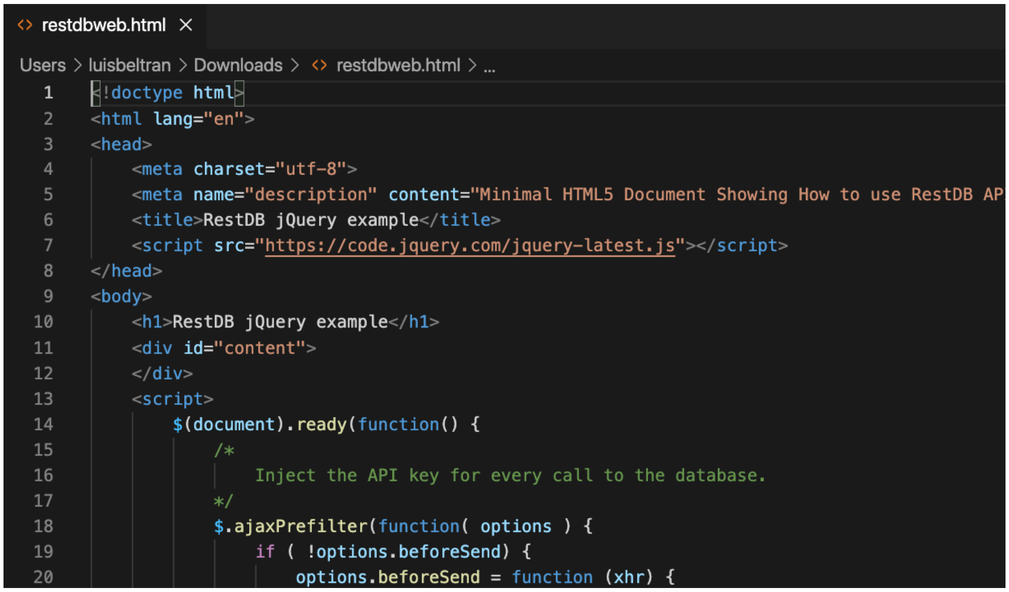
```
<!doctype html>
<html lang="en">
<head>
	<meta charset="utf-8">
	<meta name="description" content="Minimal HTML5 Document Showing How to use RestDB API and jQuery">
	<title>RestDB jQuery example</title>
	<script src="https://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
	<h1>RestDB jQuery example</h1>
	<div id="content">
	</div>
	<script>
		$(document).ready(function() {
			/*
				Inject the API key for every call to the database.
			*/
			$.ajaxPrefilter(function( options ) {
				if ( !options.beforeSend) {
					options.beforeSend = function (xhr) {
						xhr.setRequestHeader('x-apikey', 'REPLACE-WITH-YOUR-KEY');
					}
				}
			});
			/*
				Create html from one record
			*/
			var makeHtmlItem = function(item) {
				var element = $('<div></div>').attr('id', item._id);
				element.append('<hr><span>'+item['ProductName']+' '+item['UnitPrice']+'</span>');
```

6.8 Either:

- Open the file using an Editor (Visual Studio Code, Notepad, etc)
- Go to https://jsfiddle.net/ and paste the content in the HTML section



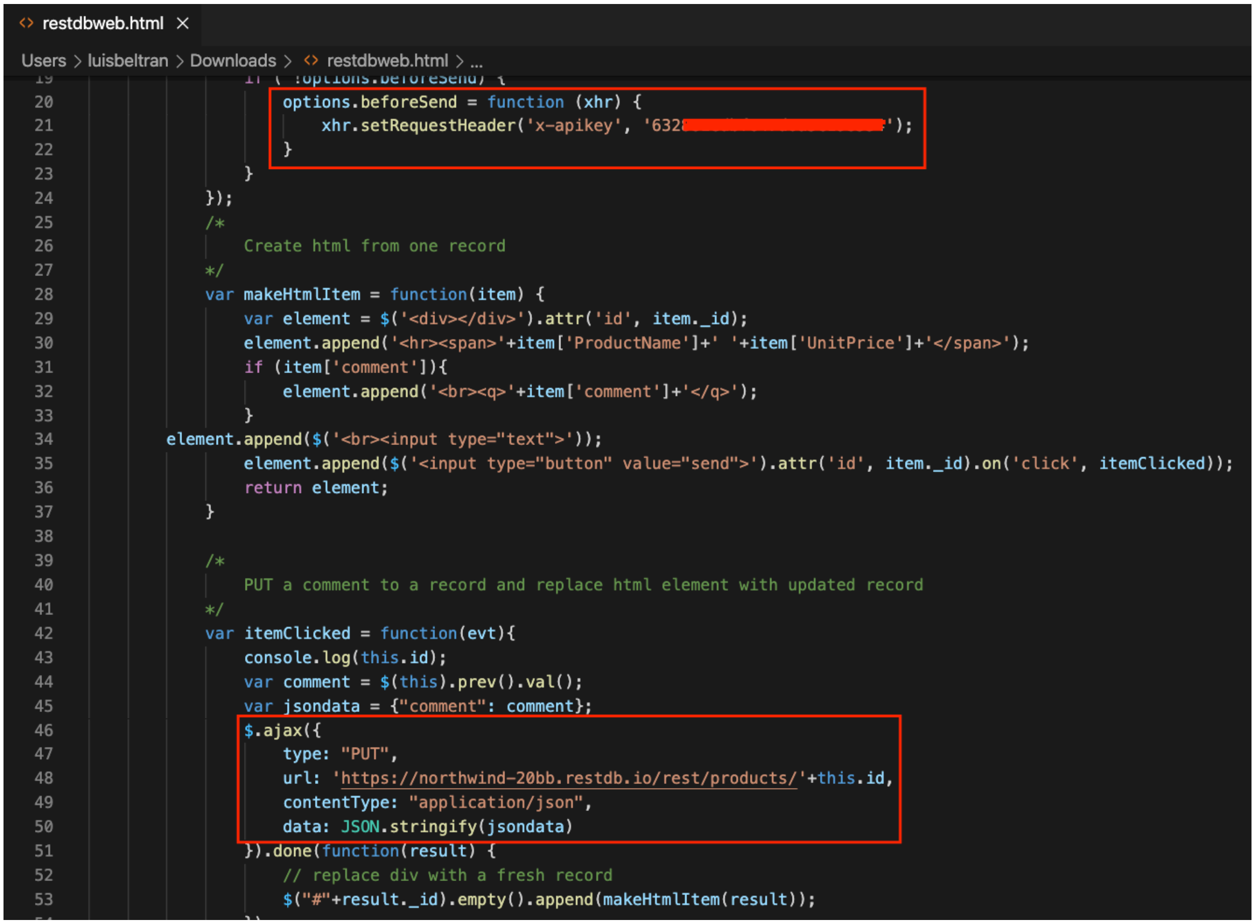
```html
<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="description" content="Minimal HTML5 Document Showing How to use RestDB API and jQuery">
    <title>RestDB jQuery example</title>
    <script src="https://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
    <h1>RestDB jQuery example</h1>
    <div id="content">
    </div>
    <script>
      $(document).ready(function() {
        /*
          Inject the API key for every call to the database.
        */
        $.ajaxPrefilter(function( options ) {
```



```html
<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta name="description" content="Minimal HTML5 Document Showing How to use RestDB AP
    <title>RestDB jQuery example</title>
    <script src="https://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
    <h1>RestDB jQuery example</h1>
    <div id="content">
    </div>
    <script>
        $(document).ready(function() {
            /*
                Inject the API key for every call to the database.
            */
            $.ajaxPrefilter(function( options ) {
                if ( !options.beforeSend) {
                    options.beforeSend = function (xhr) {
```

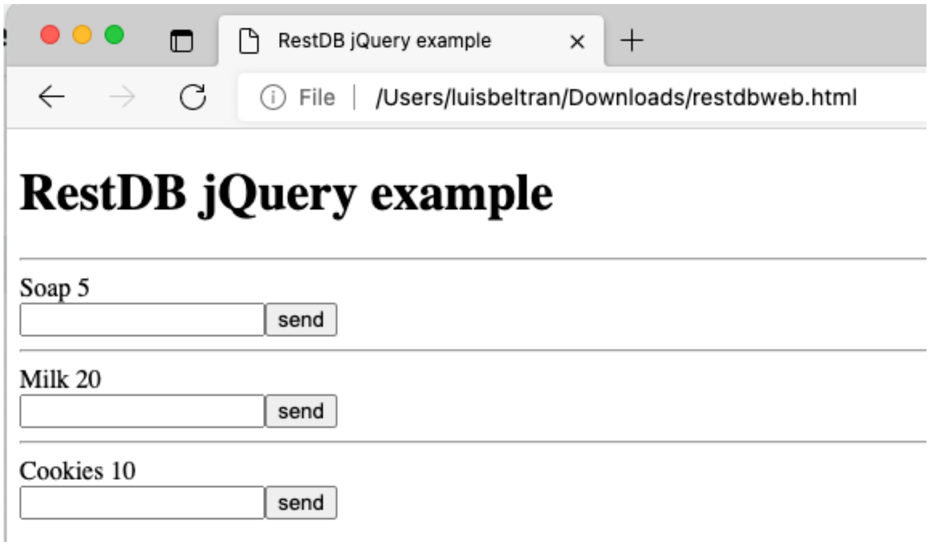
6.9 Find the following lines and replace:

Line	Replace with
21	The key from step 6.6
48	Your database name and suffix (example: northwind-20bb)
61	Your database name and suffix (example: northwind-20bb)



```
19                  if ( !options.beforeSend) {
20                      options.beforeSend = function (xhr) {
21                          xhr.setRequestHeader('x-apikey', '632[redacted]');
22                      }
23                  }
24              });
25              /*
26                  Create html from one record
27              */
28              var makeHtmlItem = function(item) {
29                  var element = $('<div></div>').attr('id', item._id);
30                  element.append('<hr><span>'+item['ProductName']+' '+item['UnitPrice']+'</span>');
31                  if (item['comment']){
32                      element.append('<br><q>'+item['comment']+'</q>');
33                  }
34          element.append($('<br><input type="text">'));
35                  element.append($('<input type="button" value="send">').attr('id', item._id).on('click', itemClicked));
36                  return element;
37              }
38
39              /*
40                  PUT a comment to a record and replace html element with updated record
41              */
42              var itemClicked = function(evt){
43                  console.log(this.id);
44                  var comment = $(this).prev().val();
45                  var jsondata = {"comment": comment};
46                  $.ajax({
47                      type: "PUT",
48                      url: 'https://northwind-20bb.restdb.io/rest/products/'+this.id,
49                      contentType: "application/json",
50                      data: JSON.stringify(jsondata)
51                  }).done(function(result) {
52                      // replace div with a fresh record
53                      $("#"+result._id).empty().append(makeHtmlItem(result));
```

6.10 Access the webpage and observe the results
If you are using Visual Studio, open a browser and access the file



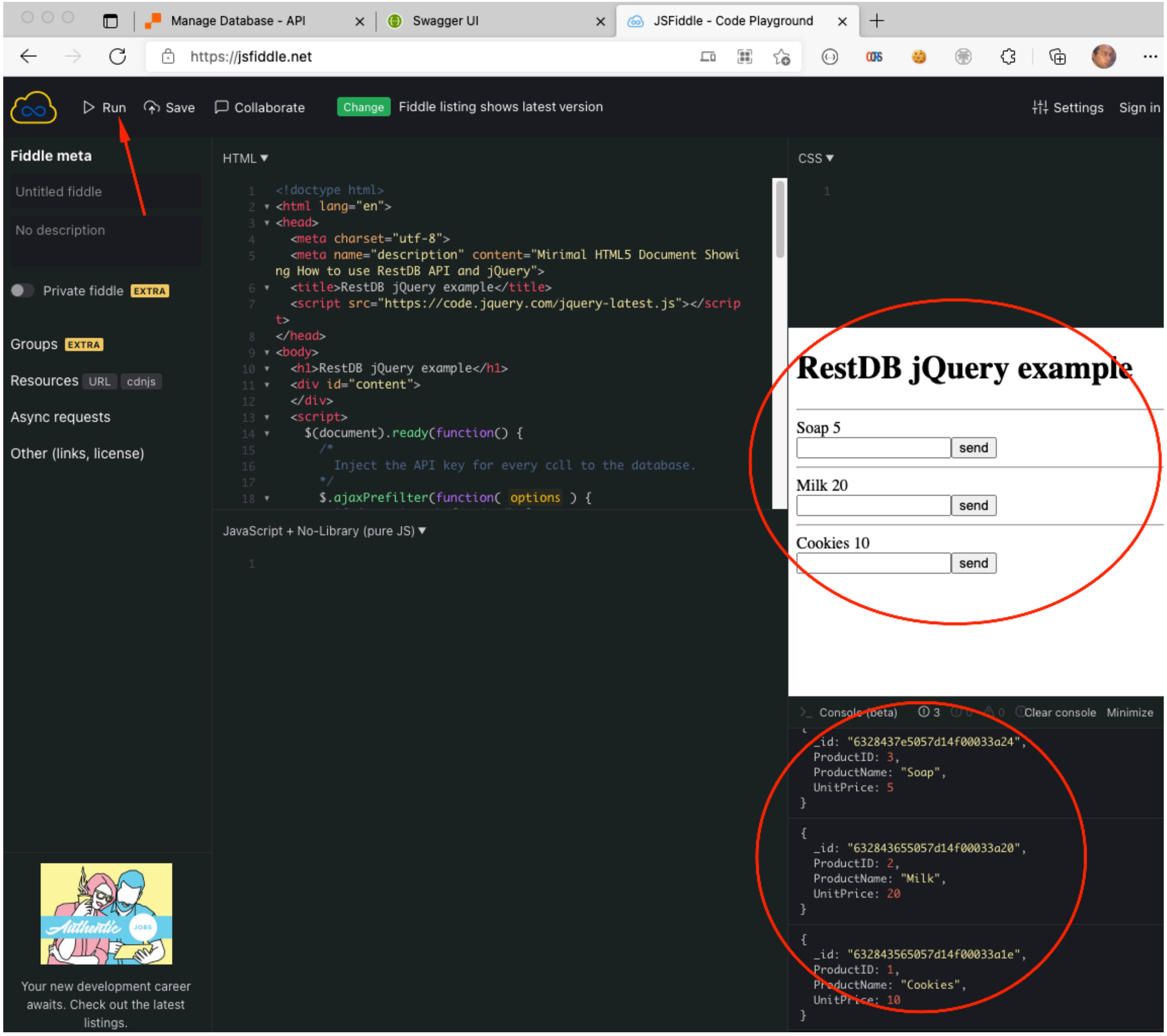
RestDB jQuery example

Soap 5

Milk 20

Cookies 10

If you are using JSFiddle, click on **Run**



```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="description" content="Mirimal HTML5 Document Showing How to use RestDB API and jQuery">
  <title>RestDB jQuery example</title>
  <script src="https://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
  <h1>RestDB jQuery example</h1>
  <div id="content">
  </div>
  <script>
    $(document).ready(function() {
      /*
        Inject the API key for every ccll to the database.
      */
      $.ajaxPrefilter(function( options ) {
```

RestDB jQuery example

Soap 5

send

Milk 20

send

Cookies 10

send

```
  _id: "6328437e5057d14f00033a24",
  ProductID: 3,
  ProductName: "Soap",
  UnitPrice: 5
}

{
  _id: "632843655057d14f00033a20",
  ProductID: 2,
  ProductName: "Milk",
  UnitPrice: 20
}

{
  _id: "632843565057d14f00033a1e",
  ProductID: 1,
  ProductName: "Cookies",
  UnitPrice: 10
}
```