# Guide: Firebase/Backendless/Postman

Introduction to Firebase, Kinvey and Postman for the ["JavaScript Applications" course@SoftUni](https://softuni.bg/courses/js-applications).

## Postman

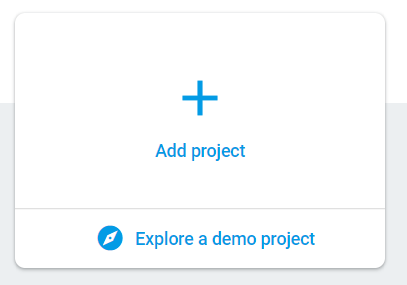
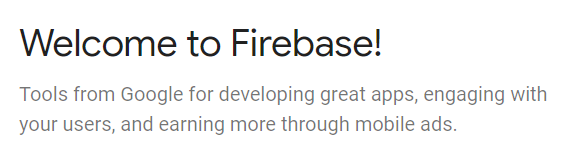
Postman is an application for **testing APIs**, by sending **request** to the **web server** and getting the **response** back. It allows users to set up all the **headers** and **cookies** the **API** expects, and checks the response. You can download it from [here](https://www.getpostman.com/downloads/).

**Firebase**

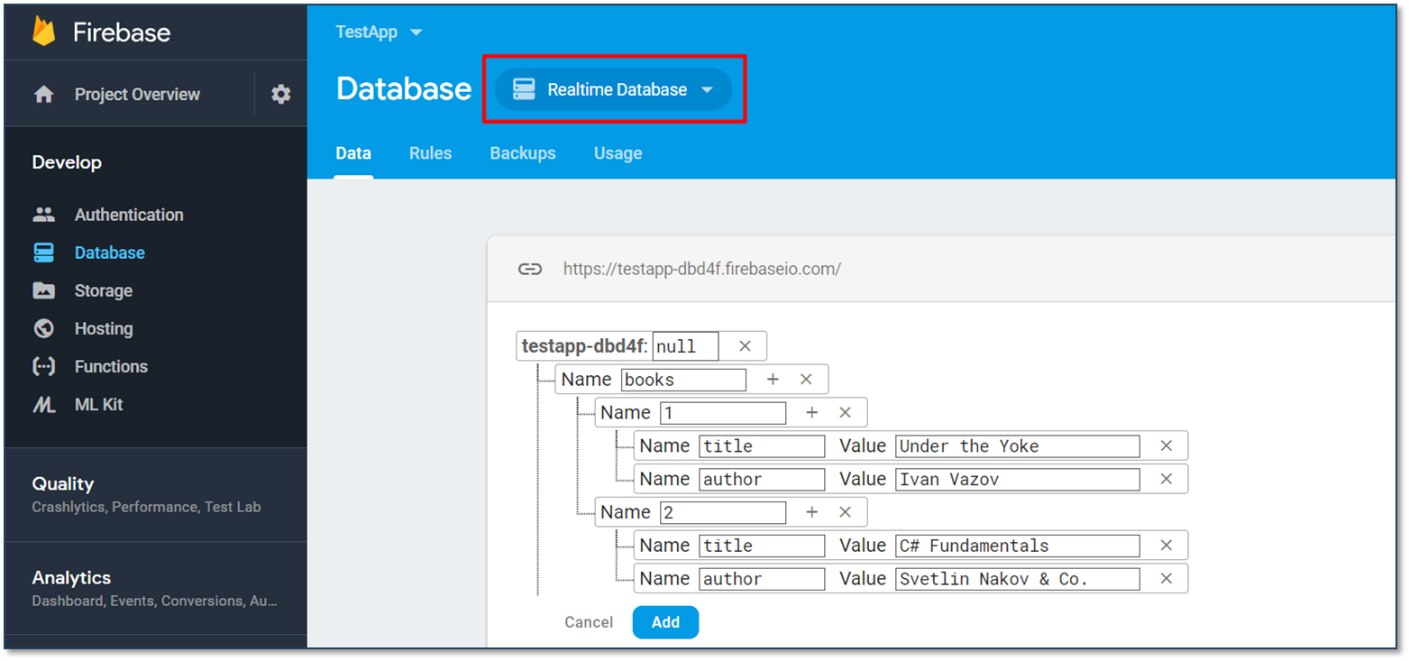
Firebase is a **mobile** and **web** development platform. It provides a **realtime database** and **backend** as a service. The service provides developers an **API** that allows application data to be **synchronized** across clients and **stored** on Firebase's cloud. The **data** is **structured** as a **JSON** tree.

### Registration

**Register** at <https://console.firebase.google.com>. Afterwards, **create a new project** and start playing around with it in order to understand how the database works.



## Put Some Data in the Database



## REST API

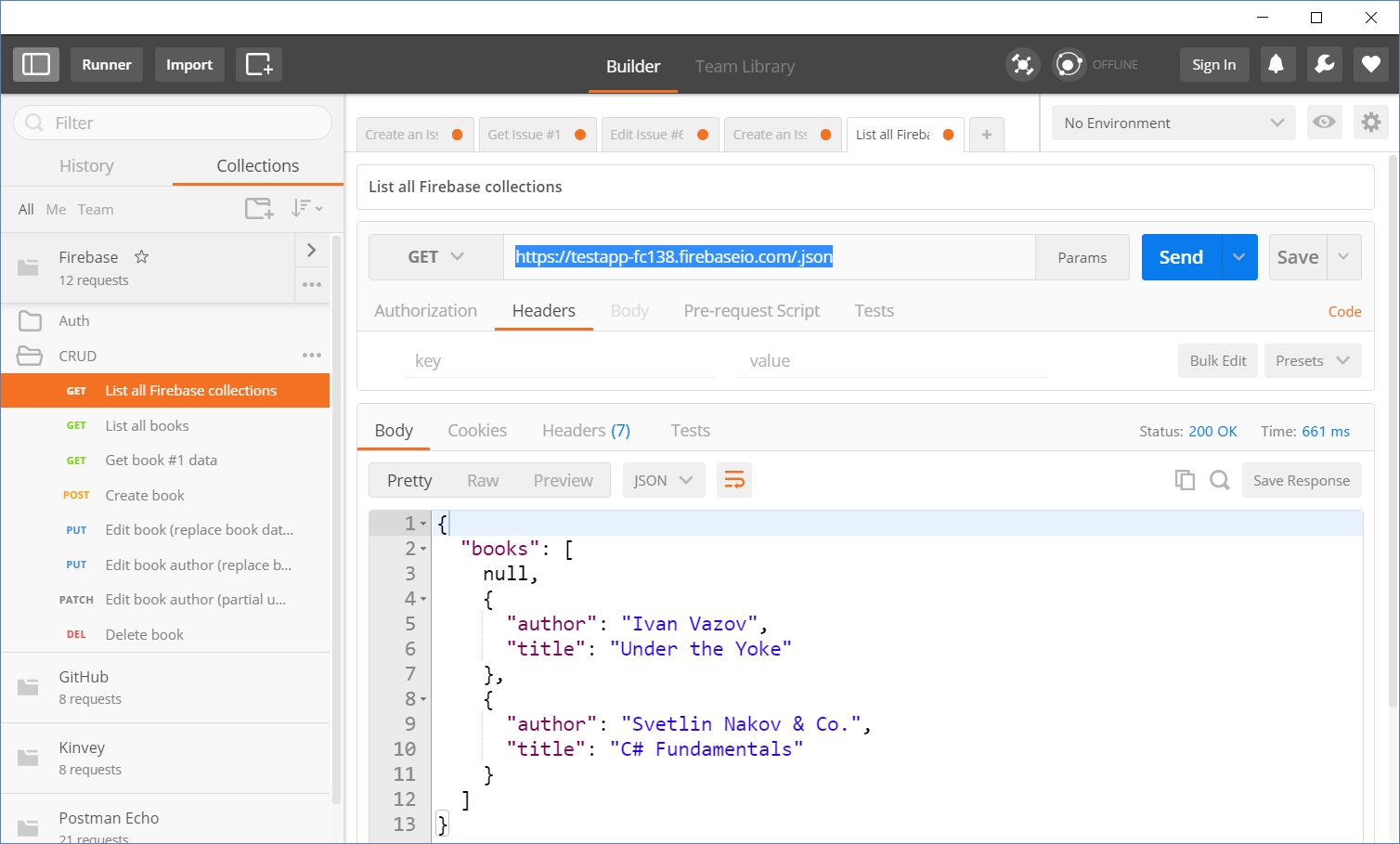
Make sure to enable **unauthorized access** to your database. Note that this is for **educational purposes** only and you should **NOT** do it in real apps as it is a **security hole**! After you have done that, access your data through the REST API.

**GET**

<https://testapp-fc138.firebaseio.com/.json>



## Accessing Firebase REST API with Postman

Open **Postman** and make a **GET** request to receive all of the information in your database. In our case that would be a list of all the available books. 

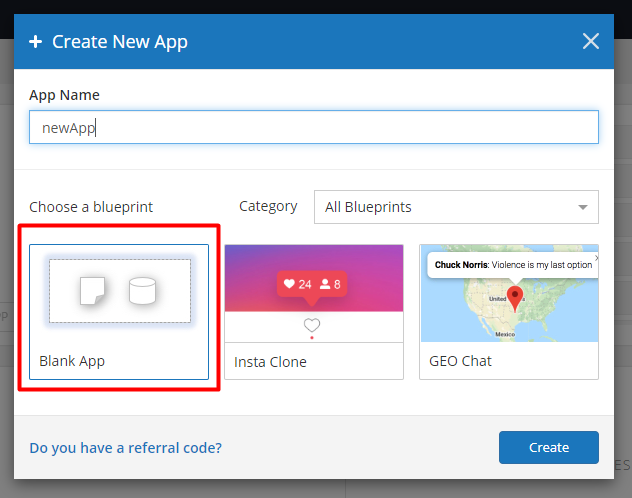
## Backendless

Backendless is a **BaaS** provider that makes it easy for developers to set up, use, and operate a **cloud back-end** for their apps. It holds **users** (API for creating an account) and **data collections** (API for CRUD operations).

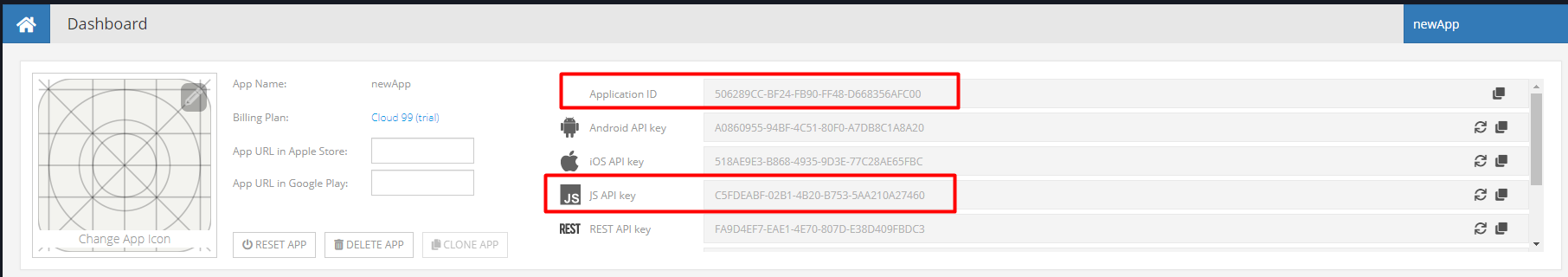
## Register

The first thing to do is create an account in **Backendless**, followed by creating an app.



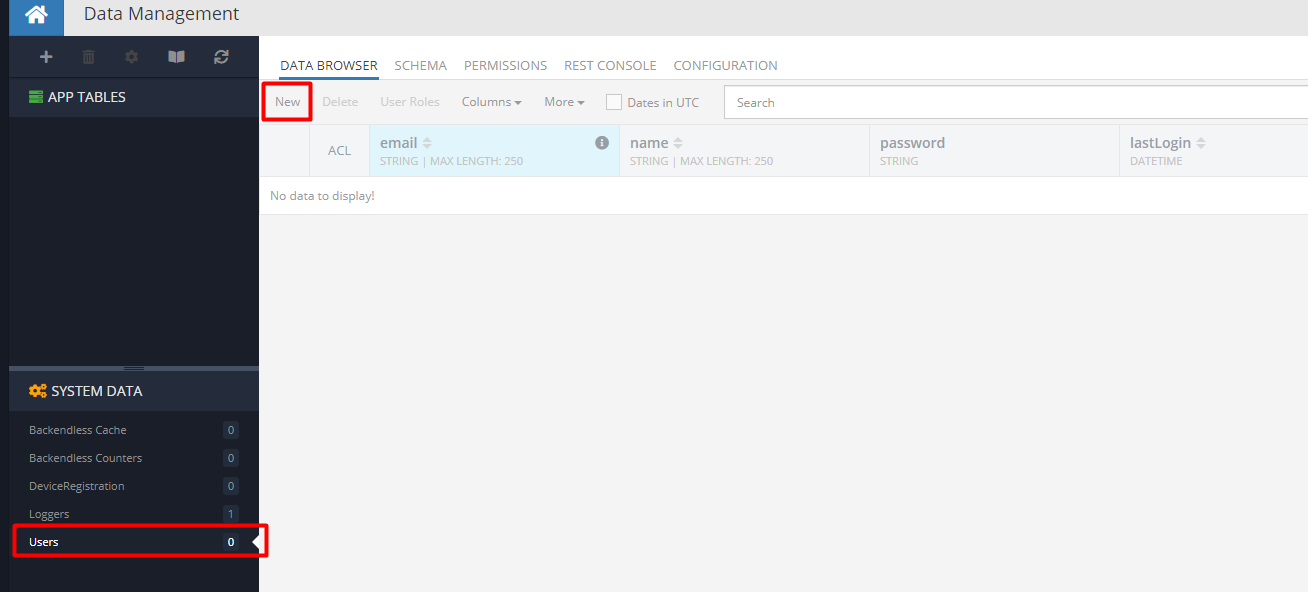


The **App ID** and **JS API key** of your app are at the main page - **Dashboard**

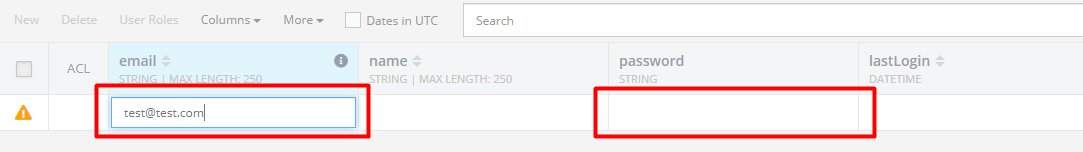


## Create a User

In order to **create a user**, click on "**Users**" choose the user menu. Create new user from the "**New**" button:

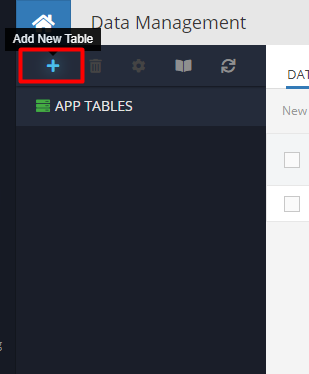


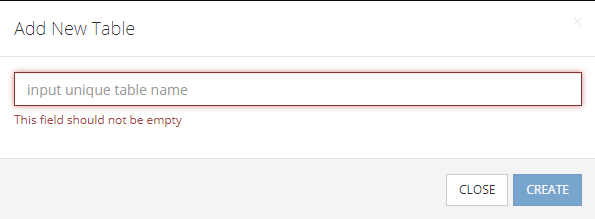
After that, you just enter the new email and password.

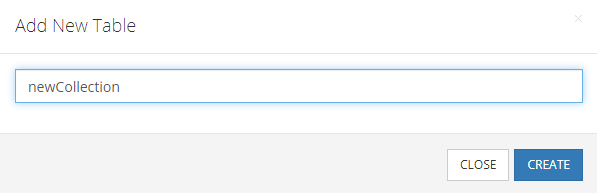


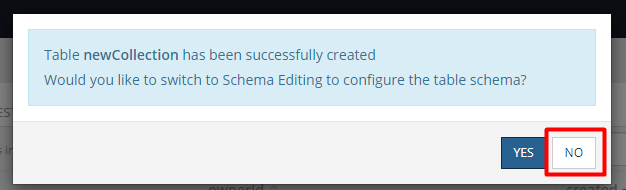
## 2.3 Create a Data Collection

In order to **create a collection**, click on "**+**" right above "**APP TABLES**" in the menu.

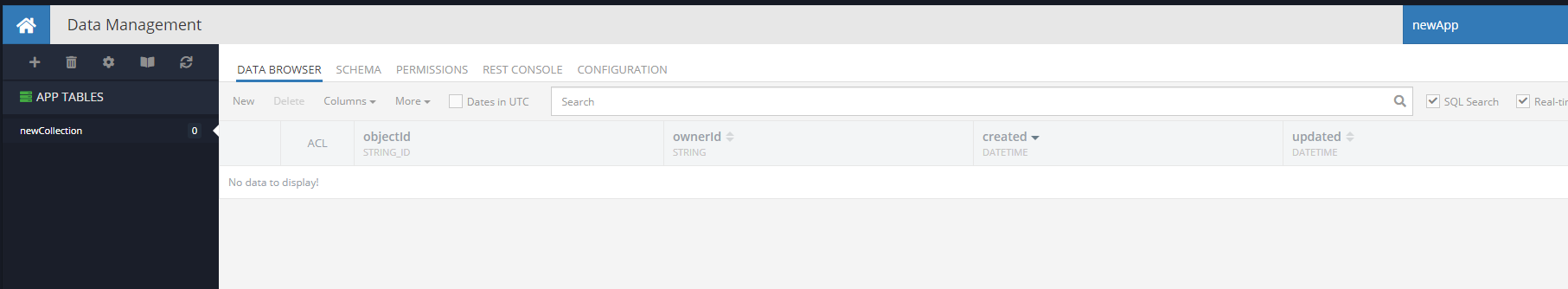


This will open a new window where you enter the collection name. 



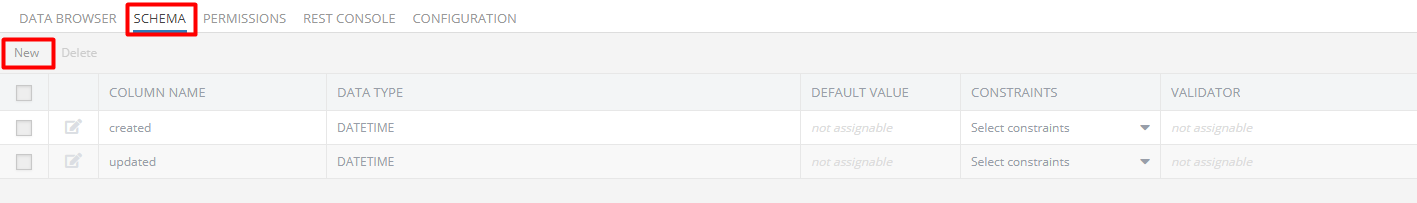


Now we have our new collection with no data init.

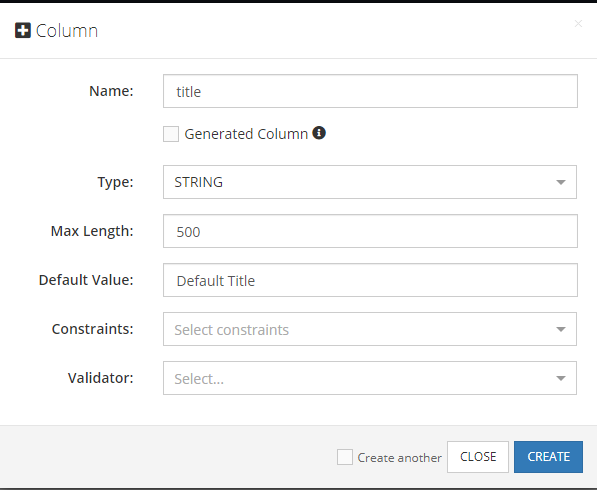


## Create Data Columns

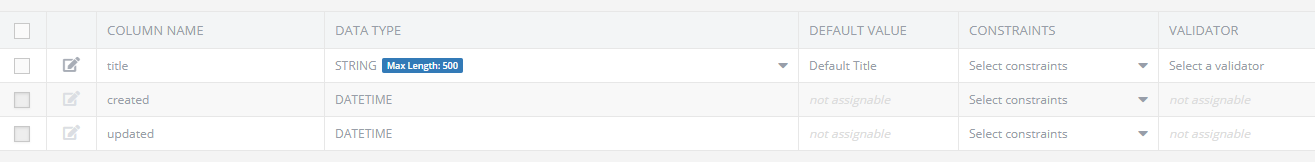
Now it is time to **create** some **data columns** for our collection. Click on the "**SCHEMA**".



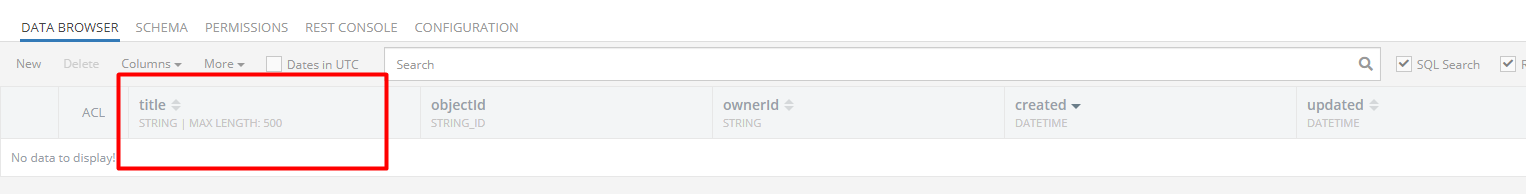
We will make the example with columns **title** and **body**. Clicking the "**New**" button, it will open a form for us:



We fill the form like it shown into the example. With the button "**CREATE**" we create the column that appear like:

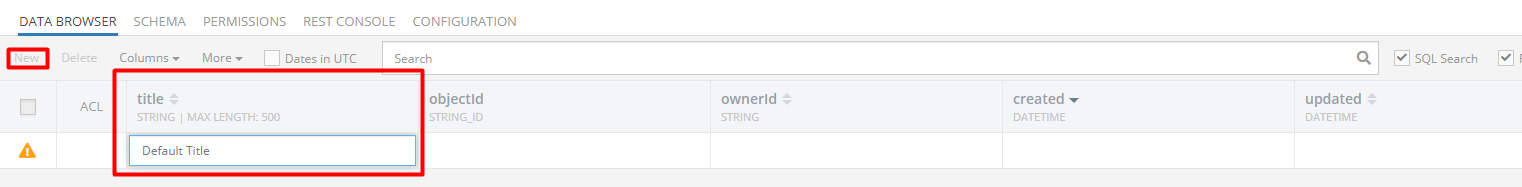


In the DATA BROWSER looks like new column:

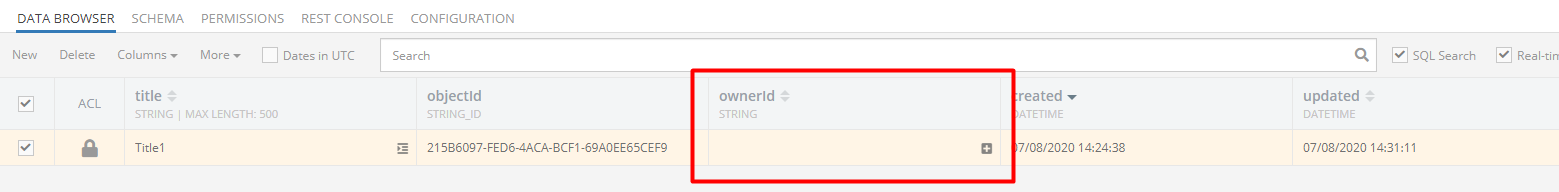


## Create Data Rows

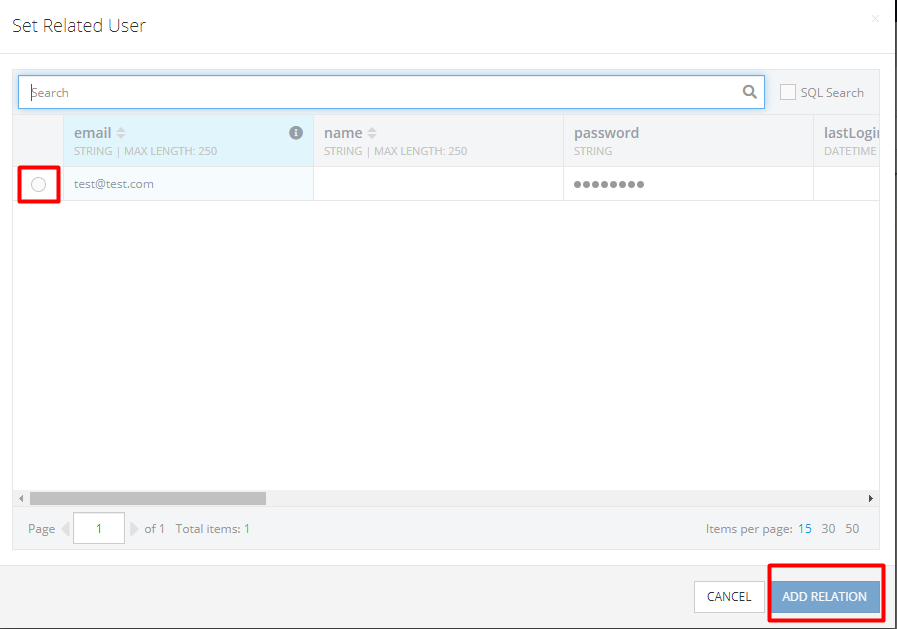
Here with click on "**New**" button, it will add new element to the collection with the defaulted value, which we can change with entering the new value into the opened input.



For the example, we enter "Title1" here and then with click on the input **ownerId** here:



It will load a window where we can choose a user that will be relate like an owner of the current element of the collection. We mark the user ant then click "ADD RELATION":



Now we have the finished the new row:

