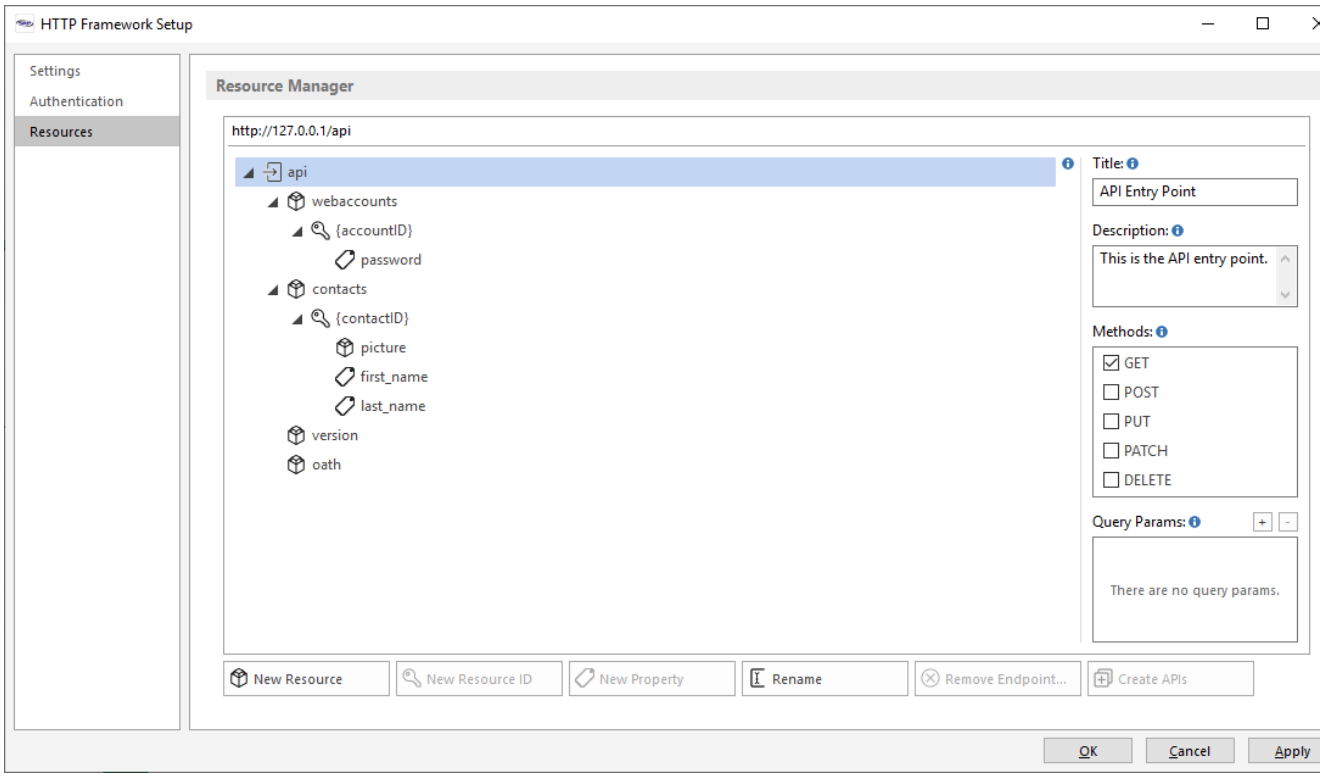
How do I create an API?

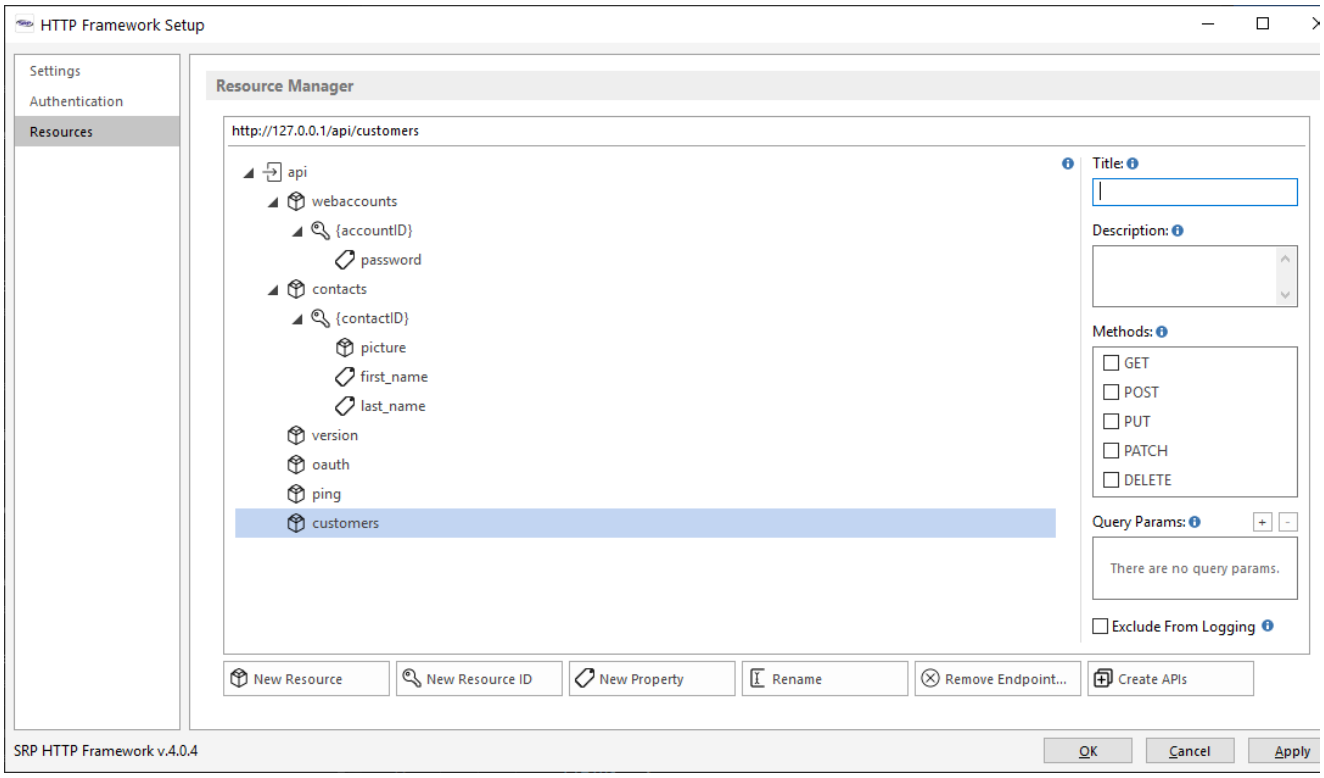
For this article, we will explain how to create an API capable of responding to HTTP requests. The code to prepare a valid response will be a topic for another another article (e.g., *How do I create a resource?*).

First, we need to run the **HTTP Framework Setup** form (e.g., `EXEC NDW_HTTP_FRAMEWORK_SETUP` from the System Monitor) and then click on the *Resources* menu to open the **Resource Manager**:

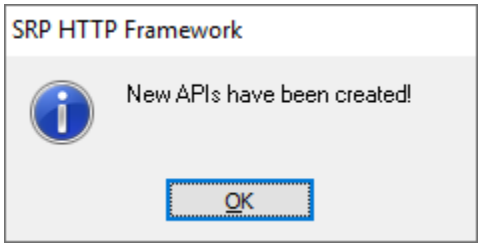


When creating an API you are either creating a *new* resource coupled with one or more supported HTTP methods or you are updating an *existing* resource by adding new supported HTTP methods. The below documentation creates an API for new resource but the process of creating an API for an updated resource is very similar.

Let's plan on creating a *customers* resource with an endpoint of `/api/customers`. Since this resource is directly underneath our `/api` endpoint, we start by selecting the *api* resource item in the Resource Manager tree (which is already demonstrated in the above screenshot). We then click on the **New Resource** button, type **`customers`** in the prompt, and finish by pressing the *Enter* key:



This *defines* the resource within the SRP HTTP Framework environment but it does not yet create the API. This requires two more actions. First, click on one or more methods to support. For now we'll just click on the *GET* method. Second, click on the **Create APIs** button. If successful, the following message will appear:



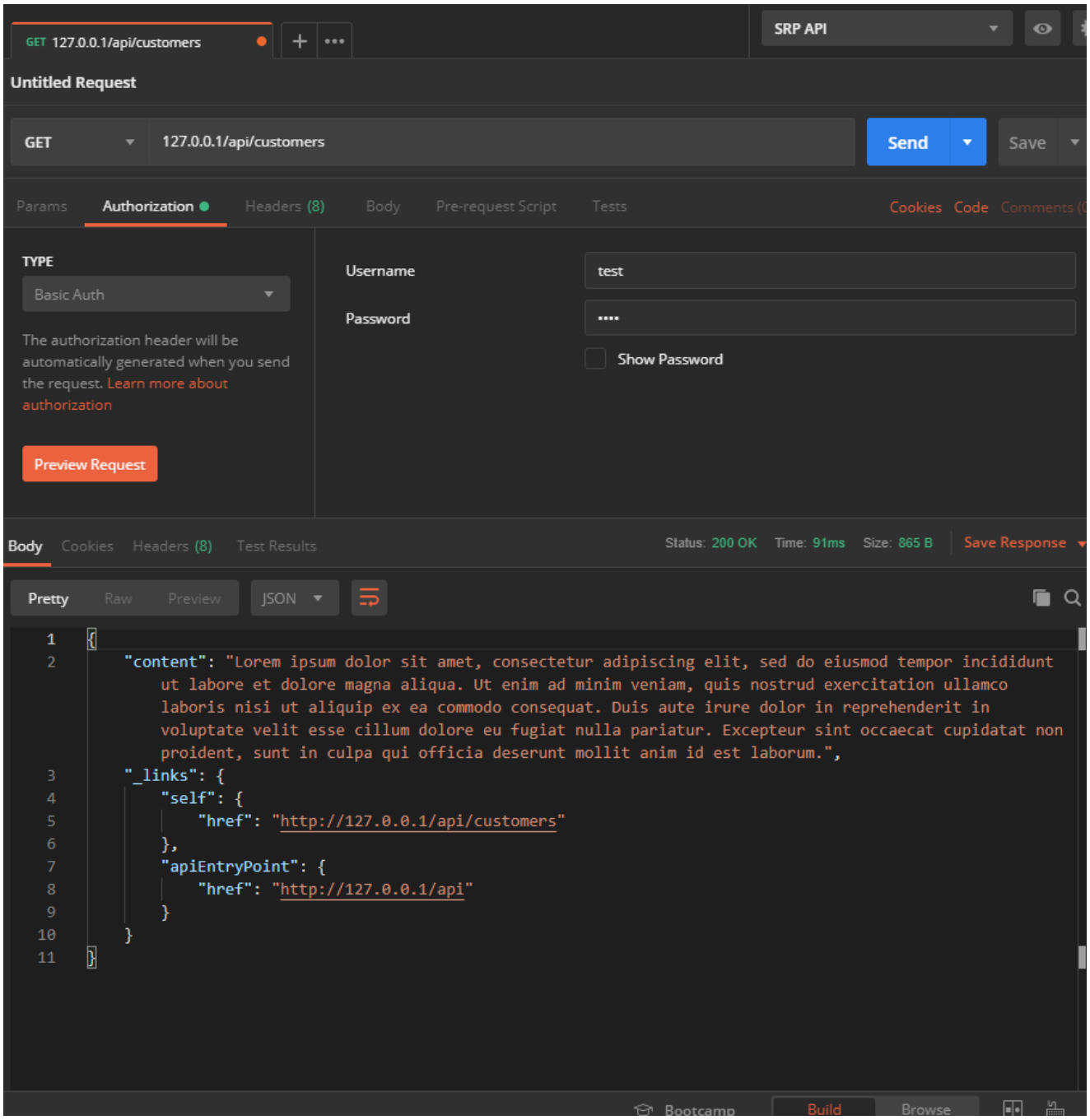
A new stored procedure (aka *API module*) will be added to your repository and this can be opened with any code editor (we recommend using the SRP Editor because of its native support for enhanced BASIC+ syntax, such as the *API* keyword). API modules are named after the resource using this pattern: ***<ResourceName>*_API**. Thus, since our resource is named "customers", we will look for a stored procedure named **CUSTOMERS_API**. It should look like this:

```
Function Customers_API(@API)
/*****************************************************************************************************
********

    This program is proprietary and is not to be used by or disclosed to others, nor is it to be copied without
written
    permission from SRP Computer Solutions, Inc.

    Name        :   Customers_API

    Description :   API logic for the Customers resource.

    Notes       :   All web APIs should include the API_SETUP insert. This will provide several useful
variables:

                        HTTPMethod              - The HTTP Method (Verb) submitted by the client (e.g., GET,
POST, etc.)
                        APIURL                  - The URL for the API entry point (e.g., api.mysite.com/v1).
                        FullEndpointURL         - The URL submitted by the client, including query params.
                        FullEndpointURLNoQuery  - The URL submitted by the client, excluding query params.
                        ParentURL               - The URL path preceeding the current endpoint.
```

```
                        CurrentAPI              - The name of this stored procedure.

    Parameters  :
        API             [in] -- Web API to process. Format is [APIPattern].[HTTPMethod]:
                                    - APIPattern must follow this structure Customers[.ID.[<Property>]]
                                    - HTTPMethod can be any valid HTTP method, e.g., GET, POST, PUT, DELETE,
etc.
                                Examples:
                                    - Customers.POST
                                    - Customers.ID.PUT
                                    - Customers.ID.firstName.GET
        Response       [out] -- Response to be sent back to the Controller (HTTP_MCP) or requesting procedure.
Web API
                                services do not rely upon anything being returned in the response. This is what
the
                                various services like SetResponseBody and SetResponseStatus services are for. A
response
                                value is only helpful if the developers want to use it for debug purposes.

    History     :   (Date, Initials, Notes)
        05/15/19    xxx     Original programmer.

*****************************************************************************************************************
*******/

#pragma precomp SRP_PreCompiler

$insert APP_INSERTS
$insert API_SETUP
$insert HTTP_INSERTS

GoToAPI else
    // The specific resource endpoint doesn't have a API handler yet.
    HTTP_Services('SetResponseStatus', 200, 'This is a valid endpoint but a web API handler has not yet been
created.')
end

Return Response OR ''


/////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////
// Endpoint Handlers
/////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////


API customers.GET

    HTTP_Resource_Services('LoremIpsum')

end api
```

The actual APIs are identified by the **API** keyword followed by the resource name and the HTTP method (e.g., **`API customers.GET`**). All API logic terminates when the **`end api`** line is reached.

When new APIs are defined, the SRP HTTP Framework automatically adds a call to the **LoremIpsum** service (a member of the HTTP_Resource_Services module). This provides functionality for the API so that the endpoint can be tested immediately. For instance, we can now use an API test utility like Postman to make a request:



Note, the above API was tested with the default credentials. If you are not sure how credentials are handled then please read the *How do I authenticate my API?* article.