

Table of Contents

1	<i>Abstract</i>	2
2	<i>Railisa Rest API proof of concept</i>	2
3	<i>Functionalities of Railisa Rest API</i>	2
4	<i>Open API Schema</i>	3
5	<i>Rest Api View</i>	3
6	<i>Authentication mechanism based on Basic Authentication or Token Authentication</i>	4
7	<i>APIs for reading nomenclatures for Selections</i>	7
7.1	List of domains	7
7.2	List of filterable variables by domain	8
7.3	List of regions	10
7.4	List of filterable countries by regions	11
7.5	List of filterable company by countries.....	13
7.6	List of activities.....	14
7.7	List of unit.....	16
7.8	List of company activity by company.....	17
8	<i>API for selections data with dynamic filtering capabilities</i>	17
9	<i>API for CSV export based on API for selections data</i>	20

1 Abstract

The purpose of this document is to explain to the end user how to use the Rest API provided by Railisa.

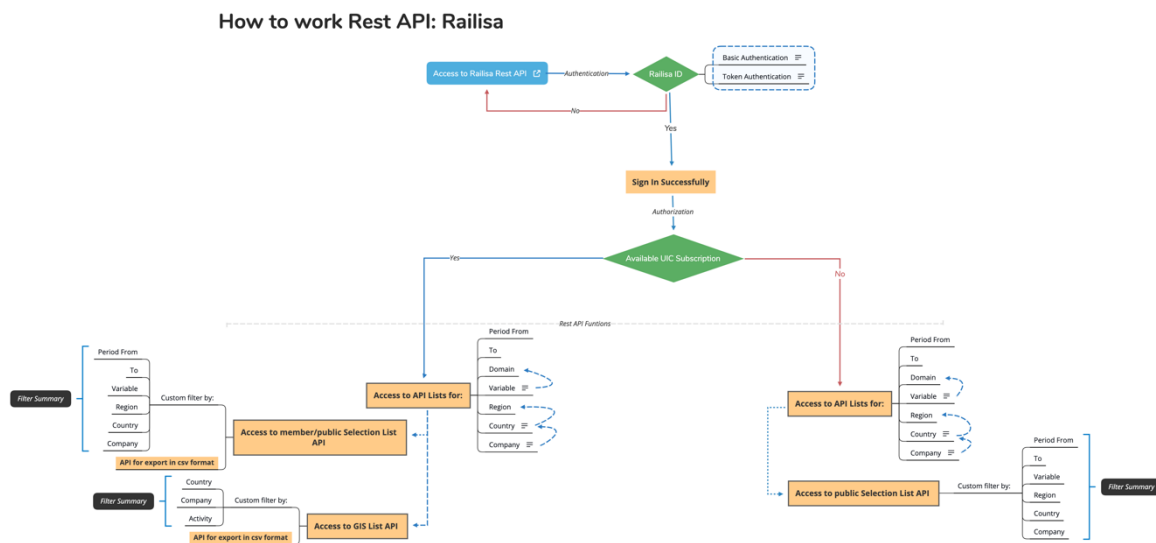
2 Railisa Rest API proof of concept

The Railisa Rest API implemented as the Railisa web application module.

This module provides to the end user with all the functionalities offered by Railisa Application, but all the functionalities will be provided as REST APIs.

Practically the end user will have the possibility to integrate these functionalities in his own application, having the possibility to make queries and to obtain the data in JSON or csv format.

The figure below shows the logical flow of Railisa Rest API.



3 Functionalities of Railisa Rest API

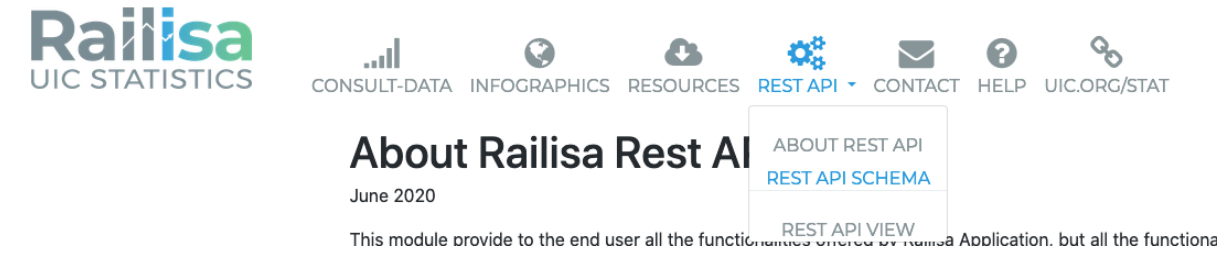
Railisa Rest API provide the following functionalities:

- Authentication mechanism based on Basic Authentication or Token Authentication
- Authorization mechanism based on groups permissions
- APIs for reading nomenclatures for Selections
 - List of domains
 - List of filterable variables by domain
 - List of regions
 - List of filterable countries by regions
 - List of filterable company by countries
 - List of unit
 - List of countries
- API for selections data with dynamic filtering capabilities based on APIs for reading nomenclatures for Selections
- API for CSV export based on API for selections data
- APIs for reading nomenclatures for Infographics (will be available in the next version of Rest API)
 - List of countries
 - List of filterable company by countries
 - List of filterable activities by company
- GIS API for selections data with dynamic filtering capabilities based on APIs for reading nomenclatures for Infographics

4 Open API Schema

This Rest API was developed in accordance with the specifications of [OpenAPI version 3](#).

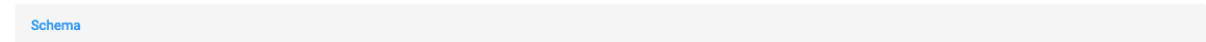
The Rest API Schema Provide the Rest API definitions. This definition can be used by documentation generation tools to display the API, code generation tools to generate servers and clients in various programming languages, testing etc.



The screenshot shows the Railisa website header with the logo 'Railisa UIC STATISTICS' on the left. A navigation menu contains icons and labels for 'CONSULT-DATA', 'INFOGRAPHICS', 'RESOURCES', 'REST API', 'CONTACT', 'HELP', and 'UIC.ORG/STAT'. The 'REST API' menu is open, showing a dropdown with three options: 'ABOUT REST API', 'REST API SCHEMA', and 'REST API VIEW'. Below the navigation, the page title is 'About Railisa Rest API' with a date 'June 2020' and a brief description: 'This module provide to the end user all the functionalities offered by Railisa Application, but all the functiona'.



The screenshot shows a blue header bar with the text 'Railisa API v1' on the left and a user name 'lucian' with a dropdown arrow on the right.



The screenshot shows a light gray bar with the word 'Schema' in blue text on the left.

Schema

OPTIONS GET

```
GET /schema/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/vnd.oai.openapi
Vary: Accept

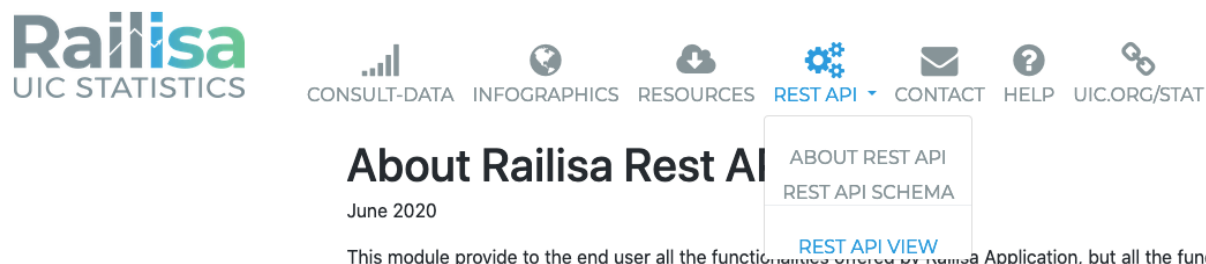
openapi: 3.0.2
info:
  title: Railisa API
  version: 1
  description: An API Railisa system.
paths:
  /api/domains/:
    get:
      operationId: listDomains
      description: ''
      parameters:
        - name: page
          required: false
          in: query
          description: A page number within the paginated result set.
          schema:
            type: integer
        - name: domain_name
          required: false
          in: query
          description: domain name
          schema:
```

5 Rest Api View

It offers a web interface, through which you can perform queries in visual mode.

This will help you test the features offered by this Rest API.

As well as how to integrate this service in your applications.



This screenshot is identical to the one in section 4, showing the Railisa website header with the 'REST API' menu open and the 'REST API VIEW' option selected.

Api Root

Api Root

OPTIONS

GET ▾

The default basic root view for DefaultRouter

GET /api/

HTTP 200 OK

Allow: GET, HEAD, OPTIONS

Content-Type: application/json

Vary: Accept

```
{
  "domains": "https://uic-stats.uic.org/api/domains/",
  "variables": "https://uic-stats.uic.org/api/variables/",
  "regions": "https://uic-stats.uic.org/api/regions/",
  "countries": "https://uic-stats.uic.org/api/countries/",
  "companies": "https://uic-stats.uic.org/api/companies/",
  "unit": "https://uic-stats.uic.org/api/unit/",
  "company-activities": "https://uic-stats.uic.org/api/company-activities/",
  "visualisation": "https://uic-stats.uic.org/api/visualisation/"
}
```

6 Authentication mechanism based on Basic Authentication or Token Authentication

Access to the Rest API functions is only available to users registered in the system.

Rest API provides two authentication mechanisms: Basic Authentication or Token Authentication.

If you use the Basic Authentication method, the user must provide his Username and Password when calling each function called

Java Script example for use Basic Authentication:

```
1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.   "async": true,
7.   "crossDomain": true,
8.   "url": "https://uic-stats.uic.org/api/domains/",
9.   "method": "GET",
10.  "headers": {
11.    "Authorization": "Basic bHVjaWFuMjphbGU1MjE5NzM=",
12.    "User-Agent": "PostmanRuntime/7.19.0",
13.    "Accept": "*/*",
14.    "Cache-Control": "no-cache",
15.    "Postman-Token": "0f7cca40-5b18-425f-a5c9-a9b0971c4f7a,53332392-be0b-4d50-8dc7-65e9abebb64a",
16.    "Host": "uic-stats.uic.org",
17.    "Content-Type": "multipart/form-data; boundary=-----156797782820618184521152",
18.    "Accept-Encoding": "gzip, deflate",
19.    "Content-Length": "286",
20.    "Connection": "keep-alive",
21.    "cache-control": "no-cache"
22.  },
23.  "processData": false,
24.  "contentType": false,
25.  "mimeType": "multipart/form-data",
26.  "data": form
27. }
```

```

28.
29. $.ajax(settings).done(function (response) {
30.     console.log(response);
31. });
32.

```

If you use the Token Authentication, this involves 2 steps

- Obtaining the token
- Token use when calling each function called

Java Script example for obtain Token:

```

1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.     "async": true,
7.     "crossDomain": true,
8.     "url": " https://uic-stats.uic.org/api/api-token-auth/",
9.     "method": "POST",
10.    "headers": {
11.        "cache-control": "no-cache",
12.        "Postman-Token": "04655c10-4bc9-45e8-8efe-a13f02785981"
13.    },
14.    "processData": false,
15.    "contentType": false,
16.    "mimeType": "multipart/form-data",
17.    "data": form
18. }
19.
20. $.ajax(settings).done(function (response) {
21.     console.log(response);
22. });
23.

```

Java Script example for use Token Authentication:

```

1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.     "async": true,
7.     "crossDomain": true,
8.     "url": "https://uic-stats.uic.org/api/domains/",
9.     "method": "GET",
10.    "headers": {
11.        "Authorization": "Token tutyu567567gfhgfhgj6765713gfhgfh908809",
12.        "User-Agent": "PostmanRuntime/7.19.0",
13.        "Accept": "*/*",
14.        "Cache-Control": "no-cache",
15.        "Postman-Token": "df4a65b2-50d8-4496-b5f0-fe5d604c1403,206a097a-aa01-4b16-
16.        a75f-f64144c9ba6f",
17.        "Host": "uic-stats.uic.org",
18.        "Content-Type": "multipart/form-data; boundary=-----
19.        876253164551911932612168",
20.        "Accept-Encoding": "gzip, deflate",
21.        "Content-Length": "286",
22.        "Connection": "keep-alive",
23.        "cache-control": "no-cache"

```

```

22.   },
23.   "processData": false,
24.   "contentType": false,
25.   "mimeType": "multipart/form-data",
26.   "data": form
27. }
28.
29. $.ajax(settings).done(function (response) {
30.   console.log(response);
31. });
32.

```

R example for use Token Authentication:

```

1. # Title      : Railisa R Client for Rest API. Obtain Token for Authenticatiobn
2. # Objective  : Demo. Haw to consuming Railisa Rest API in R.
3. # Created by: lucian
4. # Created on: 07/08/2020
5.
6. #install.packages("httr")
7.
8. #Require the package so you can use it
9. require("httr")
10. #install.packages("jsonlite")
11.
12. #Require the package so you can use it
13. require("jsonlite")
14.
15. #Paste_API_Username_Here. Change with your Railisa Username
16. Username <- "Test"
17. #Paste_API_Password_Here. Change with your Railisa password
18. Password <- "test12345"
19.
20. #Base URL Railisa rest API
21. token_url <- "https://uic-stats.uic.org/api/api-token-auth/"
22.
23. login <- list(
24.   username = Username,
25.   password = Password
26. )
27.
28. #this directive are used because the SSL certificat on Railisa preproduction
   was expired
29. httr::set_config(config(ssl_verifypeer = 0L))
30.
31. token <- POST(token_url, body = login, encode = "form")
32.
33. get_token <- content(token, "text", encoding = "UTF-8")
34.
35. #Obtain Token for Authentication
36. #The results are format "[1]"
   "{\"token\":\"tutyu567567gfhgfhgj6765713gfhgfh908809\"}"
37. # Token values is tutyu567567gfhgfhgj6765713gfhgfh908809
38. #get_token
39. get_token_json <- fromJSON(get_token, flatten = TRUE)
40. get_token_df <- as.data.frame(get_token_json)
41. get_token_df
42.

```

Python example for use Token Authentication:

```

1. def login(username, password):
2.     url = 'https://uic-stats.uic.org/api/api-token-auth/'
3.     payload = {'username': username,
4.               'password': password}

```

```

5.     files = []
6.     headers = {}
7.     response = requests.request("POST", url, headers=headers, data=payload,
files=files, verify=ssl.CERT_NONE)
8.     return response.json()['token']
9.

```

7 APIs for reading nomenclatures for Selections

7.1 List of domains

For obtain the list of domains the Rest API provide the function “Domain List”

Method: **GET**

URL: **/api/domains/**

Filter by (optional): **domain_name** (eg. /api/domains/?domain_name=Accidents)

JavaScript example:

```

1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.   "async": true,
7.   "crossDomain": true,
8.   "url": "http://uic-stats.uic.org/api/domains/?domain_name=Accidents",
9.   "method": "GET",
10.  "headers": {
11.    "Authorization": "Token tutyu567567gfhgfhgj6765713gfhgfh908809",
12.    "User-Agent": "PostmanRuntime/7.19.0",
13.    "Accept": "*/*",
14.    "Cache-Control": "no-cache",
15.    "Postman-Token": "32473d98-e0eb-4731-b95a-1f293ef412ef,e641d27c-07d4-4ad1-
89d2-51a97d89344d",
16.    "Host": "127.0.0.1:8000",
17.    "Content-Type": "multipart/form-data; boundary=-----
286063985509474678866833",
18.    "Accept-Encoding": "gzip, deflate",
19.    "Content-Length": "286",
20.    "Connection": "keep-alive",
21.    "cache-control": "no-cache"
22.  },
23.  "processData": false,
24.  "contentType": false,
25.  "mimeType": "multipart/form-data",
26.  "data": form
27. }
28.
29. $.ajax(settings).done(function (response) {
30.   console.log(response);
31. });
32.

```

Also, this function can be executed by Rest API View (see chapter 5)

R example:

```

1. #Require the package so you can use it
2. require("httr")
3. #install.packages("jsonlite")

```

```

4.
5. #Require the package so you can use it
6. require("jsonlite")
7.
8. require("dplyr")
9.
10. # Required to run Loghin.R for obtain Token and paset here
11. Token <- "tutyu567567gfhgfhgj6765713gfhgfh908809"
12. #this directive are used because the SSL certificat on Railisa preproduction
    was expired
13. httr::set_config(config(ssl_verifypeer = 0L))
14.
15. #Obtain Domain List
16. api_url_1 <- "https://uic-stats.uic.org/api/domains"
17. get_domain <- GET(api_url_1, add_headers(Authorization = paste("Token",
    Token)))
18. get_domain_text <- content(get_domain, "text", encoding = "UTF-8")
19. get_domain_json <- fromJSON(get_domain_text, flatten = FALSE)
20. get_domain_df <- as.data.frame(get_domain_json[4]$results)
21. message("Domain List")
22. get_domain_df

```

Python example:

```

1. def getDomain(token):
2.     url = 'https://uic-stats.uic.org/api/domains/'
3.     headers = {
4.         'Authorization': 'Token ' + token
5.     }
6.     response = requests.request("GET", url, headers=headers,
    verify=ssl.CERT_NONE)
7.     return response.json()
8.

```

7.2 List of filterable variables by domain

For obtain the list of variables the Rest API provide the function “Variable List”

Method: **GET**

URL: **/api/variables/**

Filter by (optional): **domain_id, variable_id, variable_name**

(eg. /api/variables?domain_id=FRET&variable_id=6105&variable_name=Revenue-earning freight traffic on the national territory - Full+wagonloads - total number of tonnes)

JavaScript example:

```

1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.     "async": true,
7.     "crossDomain": true,
8.     "url": "http://uic-stats.uic.org/api/variables/?domain_id
    =FRET&variable_id=6105&variable_name =Revenue-
    earning%20freight%20traffic%20on%20the%20national%20territory%20-
    %20Full%20wagonloads%20-%20total%20number%20of%20tonnes",
9.     "method": "GET",
10.    "headers": {
11.        "Authorization": "Token tutyu567567gfhgfhgj6765713gfhgfh908809",
12.        "User-Agent": "PostmanRuntime/7.19.0",
13.        "Accept": "*/*",
14.        "Cache-Control": "no-cache",

```



```

15.     "Postman-Token": "97012ec2-4058-4da3-898f-517fa1051e8e,964afece-10e0-402f-
    b53b-22ffe08a6e14",
16.     "Host": "uic-stats.uic.org",
17.     "Content-Type": "multipart/form-data; boundary=-----
    782184182105230076277231",
18.     "Accept-Encoding": "gzip, deflate",
19.     "Content-Length": "286",
20.     "Connection": "keep-alive",
21.     "cache-control": "no-cache"
22. },
23. "processData": false,
24. "contentType": false,
25. "mimeType": "multipart/form-data",
26. "data": form
27. }
28.
29. $.ajax(settings).done(function (response) {
30.     console.log(response);
31. });
32.

```

Also, this function can be executed by Rest API View (see chapter 5)

R example:

```

1. #Obtain Variable List
2. ## set parameter Domain ID. The list of Domain ID can be obtained by
   get_domain_df Eg. ACCID
3.
4. Dom_ID <- "ACCID"
5. api_url_2 <- "https://uic-stats.uic.org//api/variables/"
6. call_0 <- paste0(api_url_2, "?", "domain_id", "=", Dom_ID)
7. Items <- GET(call_0,add_headers(Authorization = paste("Token", Token)))
8. Items_text <- content(Items, "text", encoding = "UTF-8")
9. Items_json <- fromJSON(Items_text, flatten = TRUE)
10.
11. count <- Items_json[1]$count
12. get_variable_json <- Items_json[4]$results
13. get_variable_df <- as.data.frame(get_variable_json)
14. pages <- round(count/50)
15. for(i in 2:pages){
16.     call_1 <- paste0(api_url_2, "?", "domain_id", "=", Dom_ID, "&", "page", "=",
   i)
17.     get_variable_2 <- GET(call_1,add_headers(Authorization = paste("Token",
   Token)))
18.     get_variable_2_text <- content(get_variable_2, "text", encoding = "UTF-8")
19.     get_variable_2_json <- fromJSON(get_variable_2_text)
20.     get_variable_2_df <- as.data.frame(get_variable_2_json[4]$results)
21.     get_variable_df <- rbind(get_variable_df, get_variable_2_df)
22. }
23. message("Variable List")
24. get_variable_df

```

Python example:

```

1. def getVariable(dom_id, token):
2.     url = 'https://uic-stats.uic.org/api/variables/?domain_id=' + dom_id
3.     headers = {
4.         'Authorization': 'Token ' + token
5.     }
6.     results = {}
7.     results['results'] = []
8.     response = requests.request("GET", url, headers=headers,
   verify=ssl.CERT_NONE)

```

```

9.     raw = response.json()
10.
11.     for i in raw['results']:
12.         results['results'].append(i)
13.
14.     pg = int(raw['count'] / 50) + 1
15.     p = 2
16.     while p <= pg:
17.         try:
18.             r = requests.request("GET", url + '&page=' + str(p),
19. headers=headers, verify=ssl.CERT_NONE)
20.             raw2 = r.json()
21.             for j in raw2['results']:
22.                 results['results'].append(j)
23.         finally:
24.             p += 1
25.
26.     return results
27.

```

7.3 List of regions

For obtain the list of regions the Rest API provide the function “Region List”

Method: **GET**

URL: **/api/regions/**

Filter by (optional): **region_name** (eg. /api/regions/?region_name=Africa)

JavaScript example:

```

1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.     "async": true,
7.     "crossDomain": true,
8.     "url": "https://uic-stats.uic.org/api/regions/?region_name=Africa",
9.     "method": "GET",
10.    "headers": {
11.        "Authorization": "Token tutyu567567gfhgfhghj6765713gfhgfh908809",
12.        "User-Agent": "PostmanRuntime/7.19.0",
13.        "Accept": "*/*",
14.        "Cache-Control": "no-cache",
15.        "Postman-Token": "c1d2c71d-a511-42dc-bf8b-ddda54797c12,c5c5d377-cef8-4a15-
16. 9a61-1c0400895e05",
17.        "Host": "uic-stats.uic.org ",
18.        "Content-Type": "multipart/form-data; boundary=-----
19. 418282051346309597066817",
20.        "Accept-Encoding": "gzip, deflate",
21.        "Content-Length": "286",
22.        "Connection": "keep-alive",
23.        "cache-control": "no-cache"
24.    },
25.    "processData": false,
26.    "contentType": false,
27.    "mimeType": "multipart/form-data",
28.    "data": form
29. }
30. $.ajax(settings).done(function (response) {
31.     console.log(response);
32. });

```

Also, this function can be executed by Rest API View (see chapter 5)

R example:

```
1. #Obtain Region List
2. api_url_3 <- "https://uic-stats.uic.org/api/regions"
3. get_regions <- GET(api_url_3, add_headers(Authorization = paste("Token", Token)))
4. get_regions_text <- content(get_regions, "text", encoding = "UTF-8")
5. get_regions_json <- fromJSON(get_regions_text, flatten = FALSE)
6. get_regions_df <- as.data.frame(get_regions_json[4]$results)
7. message("Region List")
8. get_regions_df
```

Python example:

```
1. def getRegion(token):
2.     url = 'https://uic-stats.uic.org/api/regions/'
3.     headers = {
4.         'Authorization': 'Token ' + token
5.     }
6.     response = requests.request("GET", url, headers=headers,
7.     verify=ssl.CERT_NONE)
8.     return response.json()
```

7.4 List of filterable countries by regions

For obtain the list of countries the Rest API provide the function “Country List”

Method: **GET**

URL: **/api/countries/**

Filter by (optional): **region_id** (eg. **/api/countries/?region_id=AF_C**)

JavaScript example:

```
1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.     "async": true,
7.     "crossDomain": true,
8.     "url": "http://uic-stats.uic.org/api/countries/?region_id=AF_C",
9.     "method": "GET",
10.    "headers": {
11.        "Authorization": "Token tutyu567567gfhgfhghj6765713gfhgfh908809",
12.        "User-Agent": "PostmanRuntime/7.19.0",
13.        "Accept": "*/*",
14.        "Cache-Control": "no-cache",
15.        "Postman-Token": "2dbd9986-df03-4459-8efe-aad4121377e3,f31c1fae-a085-4f0d-8aaa-ba3d6e96cce3",
16.        "Host": "uic-stats.uic.org",
17.        "Content-Type": "multipart/form-data; boundary=-----267042938833963093941861",
18.        "Accept-Encoding": "gzip, deflate",
19.        "Content-Length": "286",
20.        "Connection": "keep-alive",
21.        "cache-control": "no-cache"
22.    },
23.    "processData": false,
```

```

24.   "contentType": false,
25.   "mimeType": "multipart/form-data",
26.   "data": form
27. }
28.
29. $.ajax(settings).done(function (response) {
30.   console.log(response);
31. });
32.

```

Also, this function can be executed by Rest API View (see chapter 5)

R example:

```

1. #Obtain Country List
2. # set parameter Region ID. The list of Region ID can be obtained by
   get_regions_df Eg. AF_C
3. Rgn_ID <- ""
4. api_url_4 <- "https://uic-stats.uic.org//api/countries/"
5. call_4 <- paste0(api_url_4, "?", "region_id", "=", Rgn_ID)
6. Items4 <- GET(call_4,add_headers(Authorization = paste("Token", Token)))
7. Items4_text <- content(Items4, "text", encoding = "UTF-8")
8. Items4_json <- fromJSON(Items4_text, flatten = TRUE)
9.
10. count <- Items4_json[1]$count
11. get_countries_json <- Items4_json[4]$results
12. get_countries_df <- as.data.frame(get_countries_json)
13. pages <- round(count/50)
14. for(i in 2:pages){
15.   call_5 <- paste0(api_url_4, "?", "region_id", "=", Rgn_ID, "&", "page", "=",
   i)
16.   get_countries_2 <- GET(call_5,add_headers(Authorization = paste("Token",
   Token)))
17.   get_countries_2_text <- content(get_countries_2, "text", encoding = "UTF-8")
18.   get_countries_2_json <- fromJSON(get_countries_2_text)
19.   get_countries_2_df <- as.data.frame(get_countries_2_json[4]$results)
20.   get_countries_df <- rbind(get_countries_df, get_countries_2_df)
21. }
22. message("Countries List")
23. get_countries_df

```

Python example:

```

1. def getCountry(rgn_id, token):
2.     url = 'https://uic-stats.uic.org/api/countries/?region_id=' + rgn_id
3.     headers = {
4.         'Authorization': 'Token ' + token
5.     }
6.     results = {}
7.     results['results'] = []
8.     response = requests.request("GET", url, headers=headers,
   verify=ssl.CERT_NONE)
9.     raw = response.json()
10.
11.     for i in raw['results']:
12.         results['results'].append(i)
13.
14.     pg = int(raw['count'] / 50) + 1
15.     p = 2
16.
17.     while p <= pg:
18.         try:
19.             r = requests.request("GET", url + '&page=' + str(p),
   headers=headers, verify=ssl.CERT_NONE)
20.             raw2 = r.json()

```

```

21.         for j in raw2['results']:
22.             results['results'].append(j)
23.     finally:
24.         p += 1
25.     return results

```

7.5 List of filterable company by countries

For obtain the list of companies the Rest API provide the function “Company List”

Method: **GET**

URL: **/api/companies/**

Filter by (optional): **country_id, company_name** (eg. /api/companies/? country_id=DZ& company_name=SNTF)

JavaScript example:

```

1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.     "async": true,
7.     "crossDomain": true,
8.     "url": "http://uic-
stats.uic.org/api/companies/?country_id=DZ&company_name=SNTF",
9.     "method": "GET",
10.    "headers": {
11.        "Authorization": "Token tutyu567567gfhgfhghj6765713gfhgfh908809",
12.        "User-Agent": "PostmanRuntime/7.19.0",
13.        "Accept": "*/*",
14.        "Cache-Control": "no-cache",
15.        "Postman-Token": "1a72c877-a5e7-40d8-9687-2fe2e519235e,f19d7c45-5c20-49ee-
8ffb-a15461e316cb",
16.        "Host": "uic-stats.uic.org",
17.        "Content-Type": "multipart/form-data; boundary=-----
204517922753159705986767",
18.        "Accept-Encoding": "gzip, deflate",
19.        "Content-Length": "286",
20.        "Connection": "keep-alive",
21.        "cache-control": "no-cache"
22.    },
23.    "processData": false,
24.    "contentType": false,
25.    "mimeType": "multipart/form-data",
26.    "data": form
27. }
28.
29. $.ajax(settings).done(function (response) {
30.     console.log(response);
31. });

```

Also, this function can be executed by Rest API View (see chapter 5)

R example:

```

1. #Obtain Company List
2. # set parameter Country ID. The list of Country ID can be obtained by
get_countries_df Eg. DE
3. Ctry_ID <- ""
4. api_url_6 <- "https://uic-stats.uic.org//api/companies/"

```

```

5. call_6 <- paste0(api_url_6, "?", "country_id", "=", Ctry_ID)
6. Items6 <- GET(call_6, add_headers(Authorization = paste("Token", Token)))
7. Items6_text <- content(Items6, "text", encoding = "UTF-8")
8. Items6_json <- fromJSON(Items6_text, flatten = TRUE)
9.
10. count <- Items6_json[1]$count
11. get_companies_json <- Items6_json[4]$results
12. get_companies_df <- as.data.frame(get_companies_json)
13. pages <- round(count/50)
14. for(i in 2:pages){
15.   call_7 <- paste0(api_url_6, "?", "country_id", "=", Ctry_ID, "page", "=", i)
16.   get_companies_2 <- GET(call_7, add_headers(Authorization = paste("Token",
Token)))
17.   get_companies_2_text <- content(get_companies_2, "text", encoding = "UTF-8")
18.   get_companies_2_json <- fromJSON(get_companies_2_text)
19.   get_companies_2_df <- as.data.frame(get_companies_2_json[4])
20.   get_companies_df <- rbind(get_companies_df, get_companies_2_df)
21. }
22. message("Company List")
23. get_companies_df
24.

```

Python example:

```

1. def getCompany(ctry_id, token):
2.     url = 'https://uic-stats.uic.org/api/companies/?country_id' + ctry_id
3.     headers = {
4.         'Authorization': 'Token ' + token
5.     }
6.     results = {}
7.     results['results'] = []
8.     response = requests.request("GET", url, headers=headers,
verify=ssl.CERT_NONE)
9.     raw = response.json()
10.
11.     for i in raw['results']:
12.         results['results'].append(i)
13.
14.     pg = int(raw['count'] / 50) + 1
15.     print("start call " + str(pg) + "API pages")
16.     p = 2
17.
18.     print("Page 1 loaded")
19.
20.     while p <= pg:
21.         print("Page " + str(p) + " loaded")
22.         try:
23.             r = requests.request("GET", url + '&page=' + str(p),
headers=headers, verify=ssl.CERT_NONE)
24.             raw2 = r.json()
25.             for j in raw2['results']:
26.                 results['results'].append(j)
27.         finally:
28.             p += 1
29.     return results

```

7.6 List of activities

For obtain the list of activities the Rest API provide the function “Activity List”

Method: **GET**

URL: **/api/activities/**

Filter by (optional): **activity_name** (eg. /api/activities/?activity_name=Freight Operator)

JavaScript example:

```

33. var form = new FormData();
34. form.append("username", "test");
35. form.append("password", "test12345");
36.
37. var settings = {
38.   "async": true,
39.   "crossDomain": true,
40.   "url": "https://uic-stats.uic.org/api/activities/?activity_name=Freight
Operator",
41.   "method": "GET",
42.   "headers": {
43.     "Authorization": "Token tutyu567567gfhgfhgj6765713gfhgfh908809",
44.     "User-Agent": "PostmanRuntime/7.19.0",
45.     "Accept": "*/*",
46.     "Cache-Control": "no-cache",
47.     "Postman-Token": "c1d2c71d-a511-42dc-bf8b-ddda54797c12,c5c5d377-cef8-4a15-
9a61-1c0400895e05",
48.     "Host": "uic-stats.uic.org ",
49.     "Content-Type": "multipart/form-data; boundary=-----
418282051346309597066817",
50.     "Accept-Encoding": "gzip, deflate",
51.     "Content-Length": "286",
52.     "Connection": "keep-alive",
53.     "cache-control": "no-cache"
54.   },
55.   "processData": false,
56.   "contentType": false,
57.   "mimeType": "multipart/form-data",
58.   "data": form
59. }
60.
61. $.ajax(settings).done(function (response) {
62.   console.log(response);
63. });
64.

```

Also, this function can be executed by Rest API View (see chapter 5)

R example:

```

9. #Obtain Activity List
10. api_url_3 <- "https://uic-stats.uic.org/api/activities"
11. get_activity <- GET(api_url_3, add_headers(Authorization = paste("Token",
Token)))
12. get_activity_text <- content(get_activity, "text", encoding = "UTF-8")
13. get_activity_json <- fromJSON(get_activity_text, flatten = FALSE)
14. get_activity_df <- as.data.frame(get_activity_json[4]$results)
15. message("Activity List")
16. get_regions_df

```

Python example:

```

8. def getActivity(token):
9.     url = 'https://uic-stats.uic.org/api/activities'
10.     headers = {
11.         'Authorization': 'Token ' + token
12.     }
13.     response = requests.request("GET", url, headers=headers,
verify=ssl.CERT_NONE)
14.     return response.json()

```

7.7 List of unit

For obtain the list of units the Rest API provide the function “Unit List”

Method: **GET**

URL: **/api/unit/**

Filter by (optional): **unit_name** (eg. /api/unit/?unit_name=cubic metre)

JavaScript example:

```
65. var form = new FormData();
66. form.append("username", "test");
67. form.append("password", "test12345");
68.
69. var settings = {
70.   "async": true,
71.   "crossDomain": true,
72.   "url": "https://uic-stats.uic.org/api/unit/?unit_name=cubic metre",
73.   "method": "GET",
74.   "headers": {
75.     "Authorization": "Token tutyu567567gfhgfhgj6765713gfhgfh908809",
76.     "User-Agent": "PostmanRuntime/7.19.0",
77.     "Accept": "*/*",
78.     "Cache-Control": "no-cache",
79.     "Postman-Token": "c1d2c71d-a511-42dc-bf8b-ddda54797c12,c5c5d377-cef8-4a15-
9a61-1c0400895e05",
80.     "Host": "uic-stats.uic.org ",
81.     "Content-Type": "multipart/form-data; boundary=-----
418282051346309597066817",
82.     "Accept-Encoding": "gzip, deflate",
83.     "Content-Length": "286",
84.     "Connection": "keep-alive",
85.     "cache-control": "no-cache"
86.   },
87.   "processData": false,
88.   "contentType": false,
89.   "mimeType": "multipart/form-data",
90.   "data": form
91. }
92.
93. $.ajax(settings).done(function (response) {
94.   console.log(response);
95. });
96.
```

Also, this function can be executed by Rest API View (see chapter 5)

R example:

```
17. #Obtain Unit List
18. api_url_3 <- "https://uic-stats.uic.org/api/unit"
19. get_unit <- GET(api_url_3, add_headers(Authorization = paste("Token", Token)))
20. get_unit_text <- content(get_unit, "text", encoding = "UTF-8")
21. get_unit_json <- fromJSON(get_unit_text, flatten = FALSE)
22. get_unit_df <- as.data.frame(get_unit_json[4]$results)
23. message("Unit List")
24. get_regions_df
```

Python example:

```
15. def getUnit(token):
```



```

16.     url = 'https://uic-stats.uic.org/api/unit'
17.     headers = {
18.         'Authorization': 'Token' + token
19.     }
20.     response = requests.request("GET", url, headers=headers,
    verify=ssl.CERT_NONE)

```

7.8 List of company activity by company

8 API for selections data with dynamic filtering capabilities

This function is representations of “Selection View” by Railisa web.

For obtain selection result the Rest API provide the function “Visualization List”

Method: **GET**

URL: **/api/visualisation/**

Filter by (optional): **companies, regions, countries, variables, max_year, min_year, value_visibility, value_last_update, unit_id, domain_id** (eg.

/api/visualisation/?companies=RFF,CFR®ions=EUR_C&countries=RO,FR&variables=1004,1110&min_year=1995&max_year=2018&value_visibility=1&value_last_update=2012-11-26&unit_id=4&domain_id=INFRA)

Note:

Variables = it is a composite get parameter represented by list of company of **variable_id** separated by **,**

Companies = it is a composite get parameter represented by list of company **company_name** separated by **,**

Countries = it is a composite get parameter represented by list of country **country_id** separated by **,**

Visibility = it is level of visibility (1=Public, 2=Public + Member, 3=Member)

JavaScript example:

```

1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.     "async": true,
7.     "crossDomain": true,
8.     "url": "http://uic-
stats.uic.org/api/visualisation/?companies=RFF,CFR&regions=EUR_C&countries=RO,FR
&variables=1004,1110&min_year=1995&max_year=2018&value_visibility=1&value_last_up
date=2012-11-26&unit_id=4&domain_id=INFRA",
9.     "method": "GET",
10.    "headers": {
11.        "Authorization": "Token tutyu567567gfhgfhgj6765713gfhgfh908809",
12.        "User-Agent": "PostmanRuntime/7.19.0",
13.        "Accept": "*/*",
14.        "Cache-Control": "no-cache",
15.        "Postman-Token": "182c698d-4805-4cfc-950c-74656a654174,6402153b-73ca-426d-
80f9-2db5f8db7246",
16.        "Host": "uic-stats.uic.org",
17.        "Content-Type": "multipart/form-data; boundary=-----
085594795765079419433806",
18.        "Accept-Encoding": "gzip, deflate",
19.        "Content-Length": "286",
20.        "Connection": "keep-alive",
21.        "cache-control": "no-cache"
22.    },
23.    "processData": false,
24.    "contentType": false,

```

```

25.   "mimeType": "multipart/form-data",
26.   "data": form
27. }
28.
29. $.ajax(settings).done(function (response) {
30.   console.log(response);
31. });
32.

```

Also, this function can be executed by Rest API View (see chapter 5)

R example:

```

1. # Title      : Railisa R Client for Rest API
2. # Objective  : Demo. How to consuming Railisa Rest API in R
3. # Created by: lucian
4. # Created on:30/08/2021
5.
6. #install.packages("httr")
7.
8. #Require the package so you can use it
9. require("httr")
10. #install.packages("jsonlite")
11.
12. #Require the package so you can use it
13. require("jsonlite")
14.
15. # Required to run Loghin.R for obtain Token and paset here
16. Token <- "tutyu567567gfhgfhj6765713gfhgfh908809"
17. #this directive are used because the SSL certificat on Railisa preproduction
    was expired
18. httr::set_config(config(ssl_verifypeer = 0L))
19.
20. ## set parameters
21. ### Variables (variable ID) Eg. 1110,2306
22. Var_ID <- ""
23.
24. ### Domain (domain ID) Eg. INFRA
25. Dom_ID <- ""
26.
27. ### Unit (unit ID) Eg. 4
28. Unit_ID <- ""
29.
30.
31. ## min_year
32. Min_year <- "2018"
33.
34. ## max_year
35. Max_year <- "2018"
36.
37. # countries (list of countries ID separaded by , ) Eg. FR,DE
38. Countries <- "IT"
39.
40. # regions (list of region ID separaded by , ) Eg. EUR_C,AF_C
41. Regions <- "EUR_C"
42.
43. # companies (list of company ID separaded by , ) Eg. EUROTUNNEL,SNCF,AAE,DB
44. Companies <- ""
45.
46. # visibility (level ov visibility) Eg. 1 - only Public, 2 - Mamber and Public,
    3 - only Member
47. Visibility <- ""
48.
49. ## Value Last Uodate
50. Val_Last_Upd <- "2018"
51.
52. api_url <- "https://uic-stats.uic.org/api/visualisation"

```

```

53. call_0 <- paste0(api_url, "?", "variables", "=", Var_ID, "&", "min_year", "=",
Min_year, "&", "max_year", "=", Max_year, "&", "countries", "=", Countries, "&",
"companies", "=", Companies, "&", "value_visibility", "=", Visibility, "&",
"value_last_update", "=", Val_Last_Upd, "&", "regions", "=", Regions, "&",
"unit_id", "=", Unit_ID, "&", "domain_id", "=", Dom_ID)
54. Items <- GET(call_0, add_headers(Authorization = paste("Token", Token)))
55. Items_text <- content(Items, "text", encoding = "UTF-8")
56. Items_json <- fromJSON(Items_text, flatten = TRUE)
57.
58. count <- Items_json[1]$count
59. pages <- ceiling(count/50)
60.
61. get_selections_json <- Items_json[4]$results
62. get_selections_df <- as.data.frame(get_selections_json)
63. if (pages > 1) {
64.   for(i in 2:pages){
65.     call_1 <- paste0(api_url, "?", "variables", "=", Var_ID, "&", "min_year",
"=", Min_year, "&", "max_year", "=", Max_year, "&", "countries", "=", Countries,
"&", "companies", "=", Companies, "&", "value_visibility", "=", Visibility, "&",
"value_last_update", "=", Val_Last_Upd, "&", "regions", "=", Regions, "&",
"unit_id", "=", Unit_ID, "&", "domain_id", "=", Dom_ID, "&", "page", "=", i)
66.     Items2 <- GET(call_1, add_headers(Authorization = paste("Token", Token)))
67.     Items2_text <- content(Items2, "text", encoding = "UTF-8")
68.     Items2_json <- fromJSON(Items2_text, flatten = TRUE)
69.     get_selections2_json <- Items2_json[4]$results
70.     get_selections2_df <- as.data.frame(get_selections2_json)
71.     get_selections_df <- rbind(get_selections_df, get_selections2_df)
72.   }
73. }
74. get_selections_df
75.

```

Python example:

```

1. def getSelection(var_id, min_year, max_year, countries, regions, companies,
visibility, value_last_update, unit_id, domain_id, token):
2.     url = 'https://uic-
stats.uic.org//api/visualisation/?variables='+var_id+'&min_year='+min_year+'&max_
year='+max_year+'&countries='+ countries+'&companies=' + companies +
'&value_visibility=' + visibility + '&value_last_update =' + value_last_update +
'&unit_id=' + unit_id + '&domain_id=' + domain_id
3.     headers = {
4.         'Authorization': 'Token ' + token
5.     }
6.     results = {}
7.     results['results'] = []
8.     response = requests.request("GET", url, headers=headers,
verify=ssl.CERT_NONE)
9.     raw = response.json()
10.
11.     for i in raw['results']:
12.         results['results'].append(i)
13.
14.     pg = int(raw['count'] / 50) + 1
15.     p = 2
16.
17.     while p <= pg:
18.         try:
19.             r = requests.request("GET", url + '&page=' + str(p),
headers=headers, verify=ssl.CERT_NONE)
20.             raw2 = r.json()
21.             for j in raw2['results']:
22.                 results['results'].append(j)
23.         finally:
24.             p += 1
25.     return results
26.

```

9 API for CSV export based on API for selections data

This function is representations of “Download View” by Railisa web.

For obtain selection result the Rest API provide the function “CSV export of Visualization List”

Method: **GET**

URL: `/api/download/`

Filter by (optional): `companies, regions, countries, variables, max_year, min_year, value_visibility, value_last_update, unit_id, domain_id` (eg. `/api/download/?companies=RFF,CFR®ions=EUR_C&countries=RO,FR&variables=1004,1110&min_year=1995&max_year=2018&value_visibility=1&value_last_update=2012-11-26&unit_id=4&domain_id=INFRA`)

Note:

Variables = it is a composite get parameter represented by list of company of `variable_id` separated by `,`

Companies = it is a composite get parameter represented by list of company `company_name` separated by `,`

Countries = it is a composite get parameter represented by list of country `country_id` separated by `,`

Visibility = it is level of visibility (1=Public, 2=Public + Member, 3=Member)

JavaScript example:

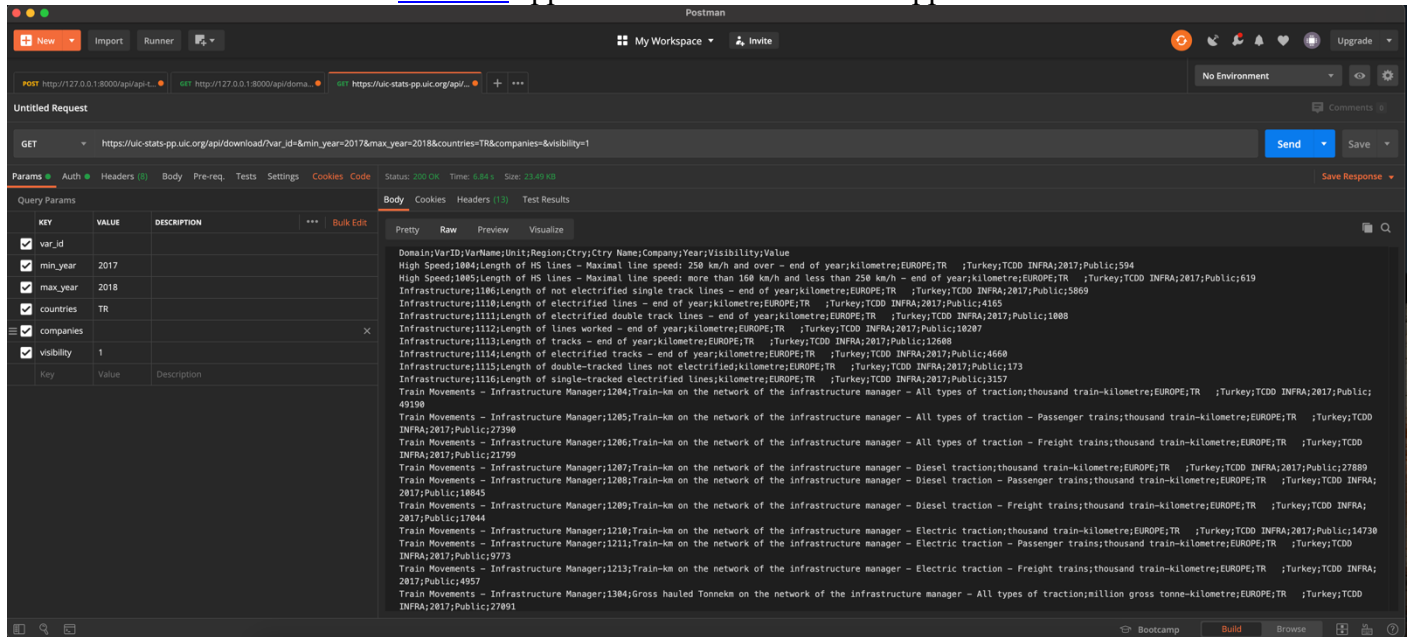
```
1. var form = new FormData();
2. form.append("username", "test");
3. form.append("password", "test12345");
4.
5. var settings = {
6.   "async": true,
7.   "crossDomain": true,
8.   "url": "http://uic-
stats.uic.org//api/visualisation/?companies=RFF,CFR&regions=EUR_C&countries=RO,FR
&variables=1004,1110&min_year=1995&max_year=2018&value_visibility=1&value_last_up
date=2012-11-26&unit_id=4&domain_id=INFRA",
9.   "method": "GET",
10.  "headers": {
11.    "Authorization": "Token tutyu567567gfhgfgghj6765713gfhgfh908809",
12.    "User-Agent": "PostmanRuntime/7.19.0",
13.    "Accept": "*/*",
14.    "Cache-Control": "no-cache",
15.    "Postman-Token": "182c698d-4805-4cfc-950c-74656a654174,6402153b-73ca-426d-
80f9-2db5f8db7246",
16.    "Host": "uic-stats.uic.org",
17.    "Content-Type": "multipart/form-data; boundary=-----
085594795765079419433806",
18.    "Accept-Encoding": "gzip, deflate",
19.    "Content-Length": "286",
20.    "Connection": "keep-alive",
21.    "cache-control": "no-cache"
22.  },
23.  "processData": false,
24.  "contentType": false,
25.  "mimeType": "multipart/form-data",
26.  "data": form
27. }
28.
29. $.ajax(settings).done(function (response) {
30.   console.log(response);
31. });
32.
33.
```

Python example:

```
1. import requests
2.
3. url = "http://uic-
stats.uic.org//api/visualisation/?companies=RFF,CFR&regions=EUR_C&countries=RO,FR
&variables=1004,1110&min_year=1995&max_year=2018&value_visibility=1&value_last_up
date=2012-11-26&unit_id=4&domain_id=INFRA"
4.
5. payload = {}
6. headers = {
7.     'Authorization': 'Token tutyu567567gfhgfhgj6765713gfhgfh908809'
8. }
9.
10. response = requests.request("GET", url, headers=headers, data = payload)
11.
12. print(response.text.encode('utf8'))
13.
```

This function can't be executed by Rest API View because the result is in CSV format, and this result can't be rendered by web browser.

This function can be tested by [Postman](#) application or other Test API applications.



The screenshot shows the Postman interface with a REST client request and its response. The request is a GET call to `https://uic-stats.pp.uic.org/api/download/var_id=&min_year=2017&max_year=2018&countries=TR&companies=&visibility=1`. The response is a large CSV file containing infrastructure and train movement data. The response body is displayed in a table format with columns for Domain, VarID, VarName, Unit, Region, Ctry, Ctry Name, Company, Year, and Visibility. The data includes various infrastructure metrics such as High Speed, Length of HS Lines, Length of not electrified single track lines, Length of electrified lines, Length of electrified double track lines, Length of lines worked, Length of tracks, Length of electrified tracks, Length of double-tracked lines not electrified, Length of single-tracked electrified lines, and Train Movements (Infrastructure Manager, Train-km on the network of the infrastructure manager - All types of traction - Passenger trains, Freight trains, Diesel traction - Passenger trains, Diesel traction - Freight trains, Electric traction - Passenger trains, Electric traction - Freight trains, Gross hauled Tonnekm on the network of the infrastructure manager - All types of traction).