31-01-2021

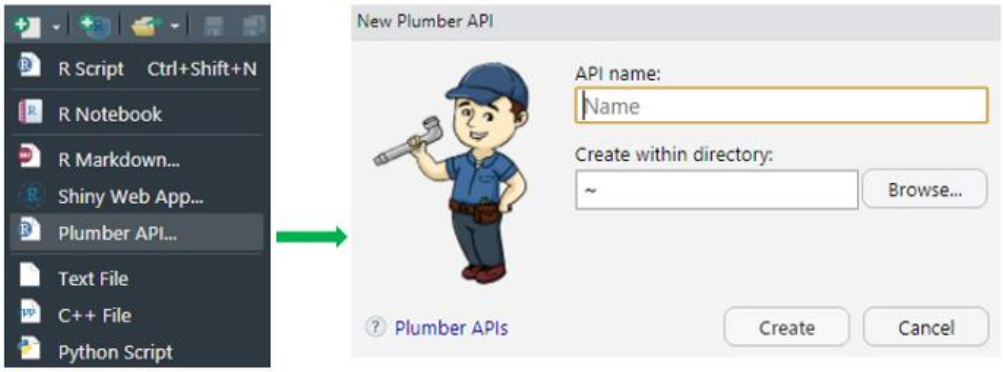
R ile REST API Nasıl Yazılır ?

API (Application Programming Interface) bir uygulamanın özelliklerinin, başka bir uygulama tarafından kullanılmasına imkan sunan bir arayüzdür.

REST (**RE**presentational **S**tate **T**ransfer), istemci-sunucu arasındaki haberleşmeyi sağlayan, HTTP metotları kullanılarak isteklerde bulunup,bu isteklere çeşitli formatlarda yanıt alınan esnek yapılı bir mimaridir.

Plumber, mevcut R kodlarını API yazım formatına uyarlayarak bir web hizmeti olarak sunmaya izin veren ve REST API yazımı için kullanılan R kütüphanesidir. Plumber, R'da yapılan işlemlerin diğer platformlarda veya programlama dillerinden kullanma imkanını sunar. Plumber ile R'da yapılan analiz veya görselleştirmelerinizi bir web uygulamasına entegre edebilirsiniz.

R'da API yazmak için plumber dosyasının aşağıdaki resimde görüldüğü gibi eklenmesi gerekir.



31-01-2021

Kullanılan Kütüphaneler

```
library(plumber)
library(dplyr)
library(plotly)
library(gridExtra)
library(leaflet)
library(grid)
library(jsonlite)
```

Kullanılan veri setine buradan ulaşabilirsiniz.

Png Formatı İçin Get Metodu Kullanımı

```
# * getPng
# * @png
# * @get / png
function () {

 attack_type_data <- turkey_data%>%
  select (attacktype1_txt)%>%
  filter (attacktype1_txt! = "Unknown")%>%
  group_by (attacktype1_txt)%>%
  summarise (count = n ())

 plot <- ggplot (attack_type_data,
  aes (x = "", y = count, fill = attacktype1_txt)) +
  geom_bar (stat = "identity", width = 1) +
  coord_polar ("y" , start = 0) +
  theme_void () +
  scale_fill_brewer (palette = "Set2") +
  labs (fill = "Attack Type")

 print (plot)
}
```

Pdf Formatı İçin Get Metodu Kullanımı

```
# * getPdf
# * @serializer contentType list (type = “application / pdf”)
# * @get / pdf
function () {
 attack_type_data <- turkey_data%>%
                select (attacktype1_txt)%>%
                filter (attacktype1_txt! = “Unknown”) ))%>%
                group_by (attacktype1_txt)%>%
                summarise (count = n ())

 total_name <- rbind (attack_type_data [, 1], “Toplam”)
 sum <- addmargins (as.matrix (attack_type_data [-1]) , 1)
 attack_type_data <- cbind (total_name, sum)

 attack_type_data <- attack_type_data%>%
  mutate (Yuzde = round ((attack_type_data $ count /
  attack_type_data [length (attack_type_data [,) 1]), 2]), 2) * 100)
  attack_type_data $ Yuzde <- paste0 (attack_type_data $ Yuzde, “%”)

 colnames (attack_type_data) [colnames (attack_type_data) ==
                    “attacktype1_txt ] <- “Attack Type”
 colnames (attack_type_data) [colnames (attack_type_data) ==
                    “count”] <- “Sayi”

 tmp <- tempfile ()
 pdf (tmp)
 pdf_title <- textGrob (“List Percentages of Attack Types”,
                  gp = gpar (fontsize = 14, col = ‘
                  Darkblue’, fontface = ‘bold’))

 tema <- ttheme_minimal (
  core = list (bg_params = list (fill = blues9 [1: 4], col = NA),
         fg_params = list (fontface = 3)),
         colhead = list (fg_params = list (col = “navyblue”,
         fontface = 4L) ),
  rowhead = list (fg_params = list (col = “darkblue”, fontface = 3L)))

 grid.arrange (pdf_title,
          tableGrob (attack_type_data, theme = thema))
 dev.off ()

 readBin (tmp, “raw”, n = file.info (tmp) $ size)
}
```

Json Formatı İçin Get Metodu Kullanımı

```
# * getJson
# * @serializer contentType list (type = "application / json")
# * @get / json
function () {
 attack_type_data <- turkey_data%>%
  select (attacktype1_txt)%>%
  filter (attacktype1_txt! = "Unknown") ))%>%
  group_by (attacktype1_txt)%>%
  summarise (count = n ())

 df <- jsonlite :: toJSON (attack_type_data)
 return (df)

}
```

Html Widget Formatı İçin Get Metodu Kullanımı

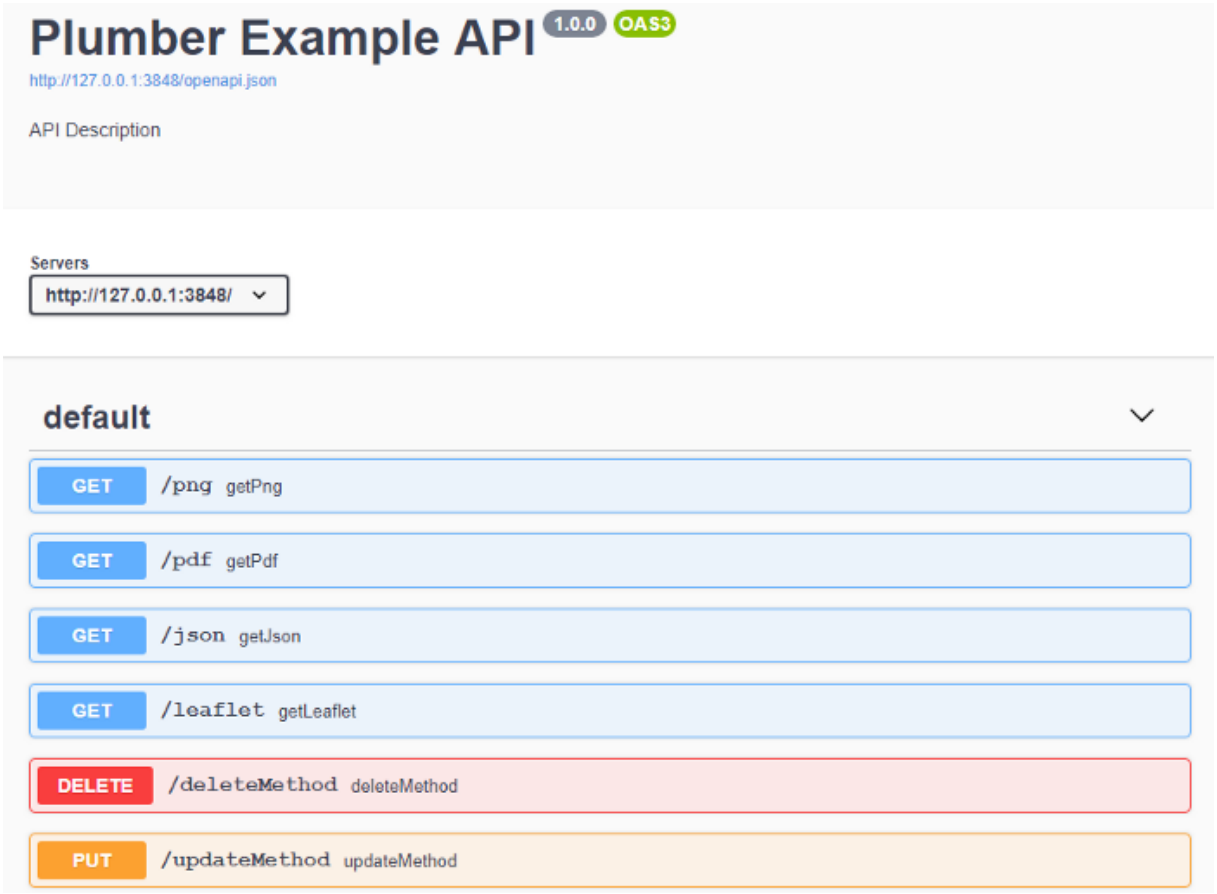
```
# * getLeaflet
# * @serializer htmlwidget
# * @get / leaflet
function () {
 leaflet ()%>%
  addTiles ()%>%
  addMarkers (data = turkey_data, clusterOptions =
          markerClusterOptions (),
          popup = ~ paste (country_txt, " / ", city))

}
```

Delete Metodunun Kullanımı

```
# * deleteMethod
# * @serializer contentType list (type = "application / json")
# * @delete / deleteMethod
function (attack_type) {
 attack_type_data <- turkey_data%>%
  select (attacktype1_txt)%>%
  filter (attacktype1_txt! = "Unknown")%>%
  group_by (attacktype1_txt)%>%
  summarise (count = n ())

 attack_type_data_delete <- attack_type_data %>%
  filter (attacktype1_txt != attack_type)

 df <- jsonlite :: toJSON (attack_type_data_delete)
 return (df)

}
```

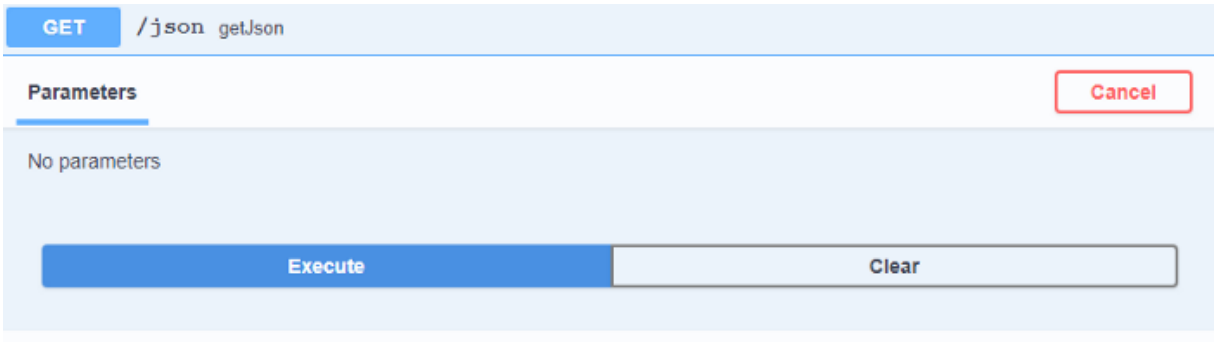
Put Metodunun Kullanımı

```
# * updateMethod
# * @serializer contentType list (type = "application / json")
# * @put / updateMethod
function (attack_type, countValue) {
 countValue <- as.numeric (countValue)
 attack_type_data <- turkey_data%>%
  select (attacktype1_txt )%>%
  filter (attacktype1_txt! = "Unknown")%>%
  group_by (attacktype1_txt)%>%
  summarise (count = n ())

 attack_type_data [attack_type_data $ attacktype1_txt == attack_type, "count"] <- countValue

 attack_type_data_update <-  attack_type_data

 df <- jsonlite :: toJSON (attack_type_data_update)
 return (df)

}
```



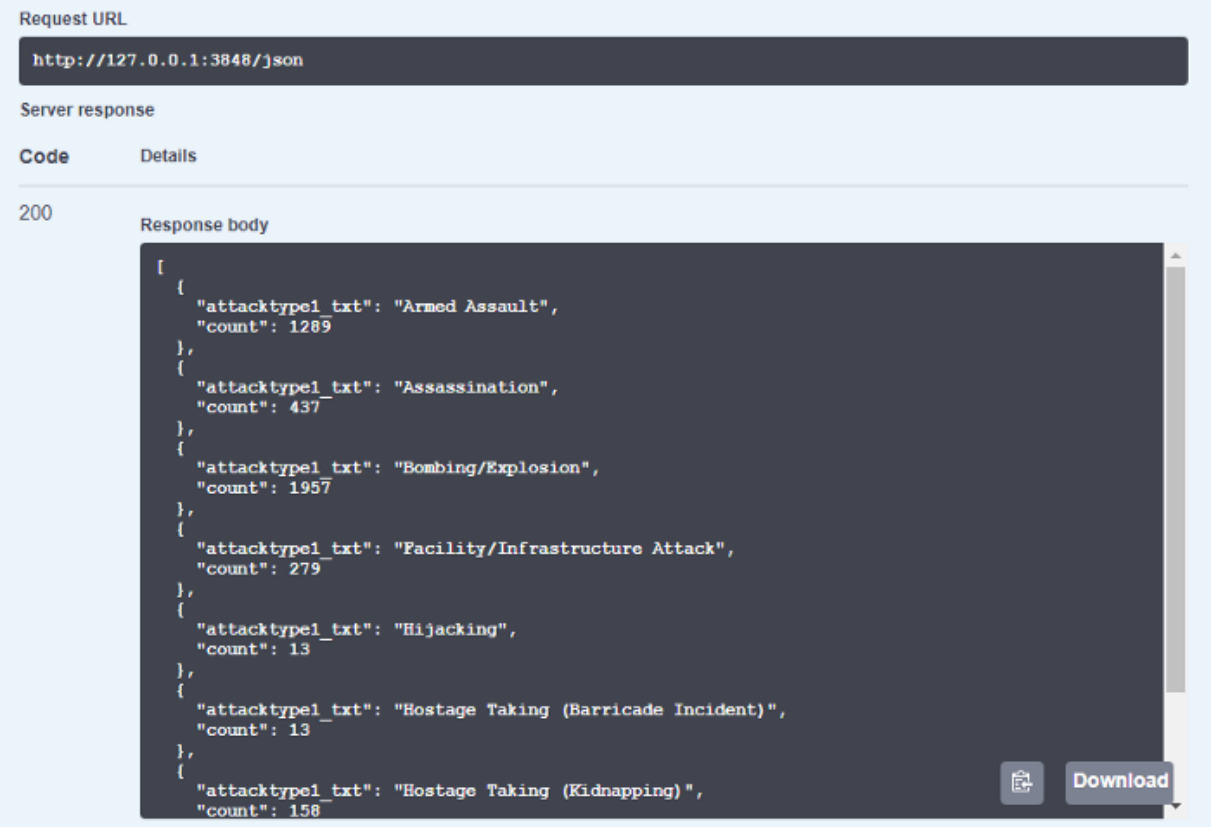
Şekil 1: Swagger Dokümantasyonu

Yazılan kodların çalıştırılması ile Swagger ekranı açılır. Bu ekranda metodların çalıştırılması ile kodların testi yapılabilir.

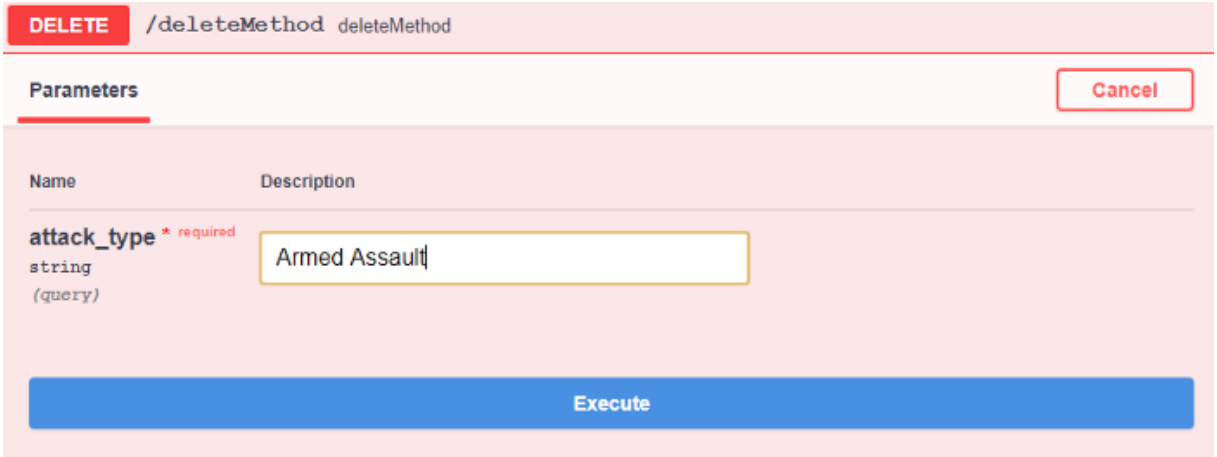


Şekil 2: Get Metodu

Swagger ekranında ilgili metod seçilerek Execute butonuna tıklanarak sonuç görüntülenebilir. Server response kısmında 200 dönmesi metodun başarılı olduğunu göstermektedir.

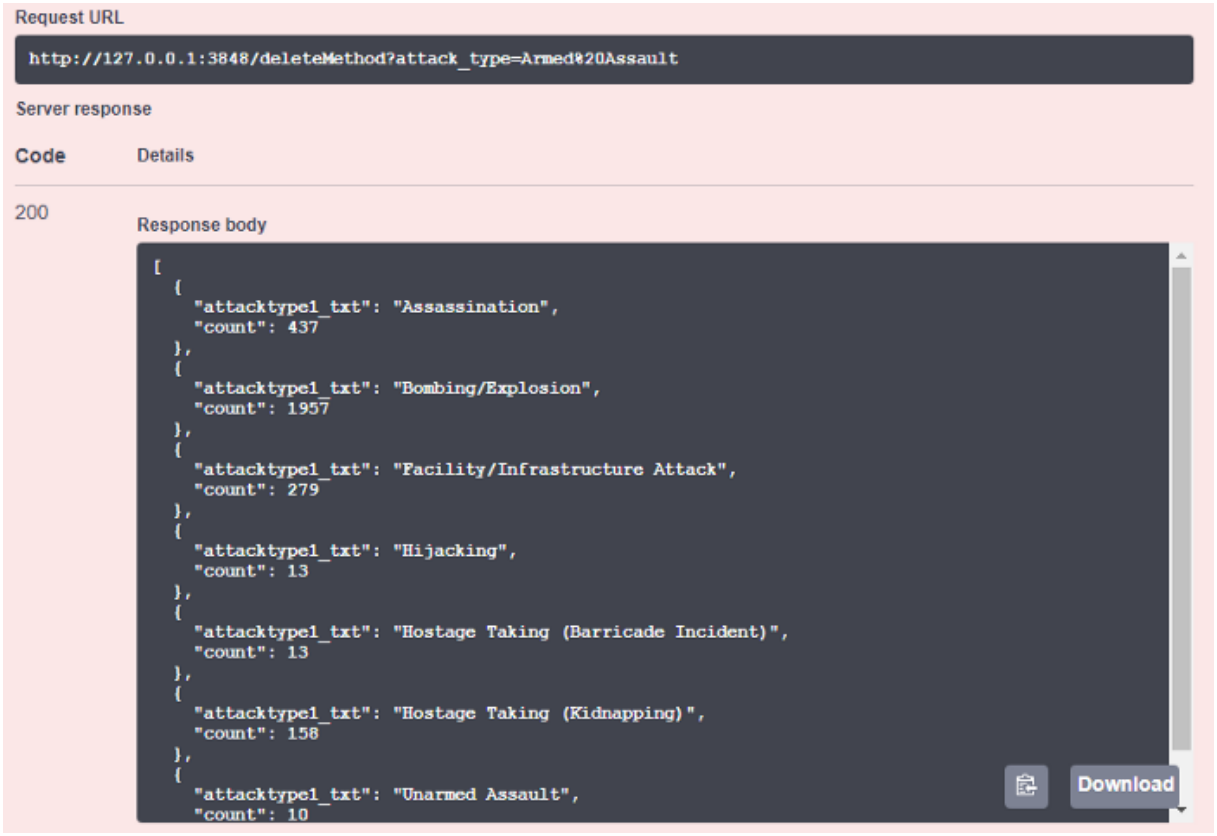


Şekil 3: Get Metodu Sonucu

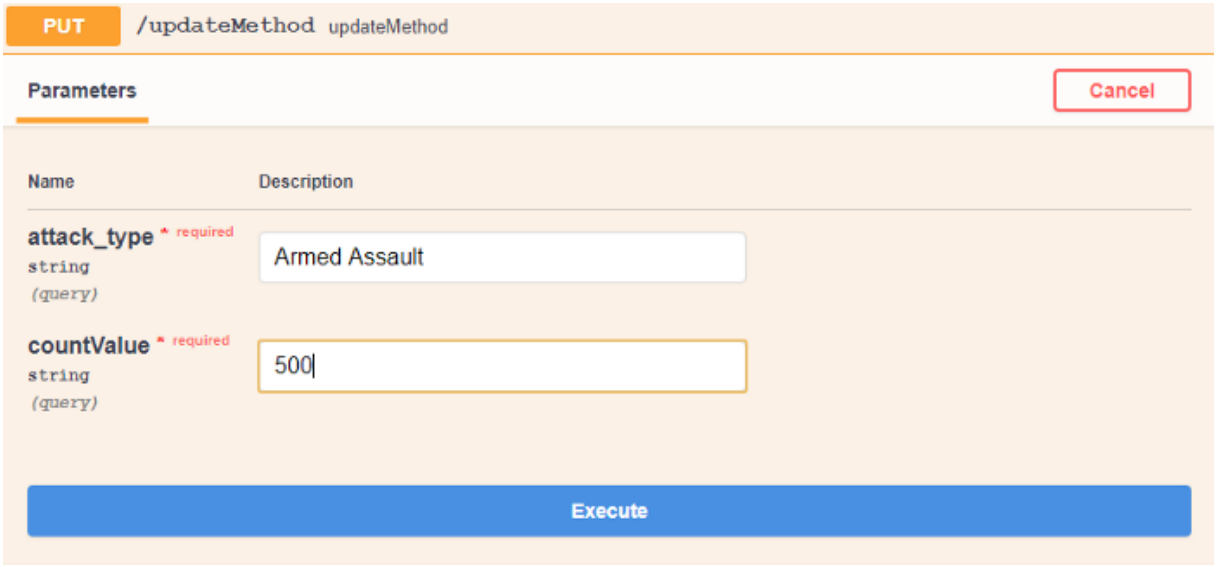


Şekil 4: Delete Metodu

Delete metodunda fonksiyondaki parametreye göre girilen değer veri içinden silinir, verinin son hali elde edilir.



Şekil 5: Delete Metodu Sonucu



Şekil 6: Update Metodu

Put metodunda fonksiyondaki parametreye göre girilen değer veri içinde güncellenir, verinin son hali elde edilir.

Request URL

```
http://127.0.0.1:3848/updateMethod?attack_type=Armed%20Assault&countValue=500
```

Server response

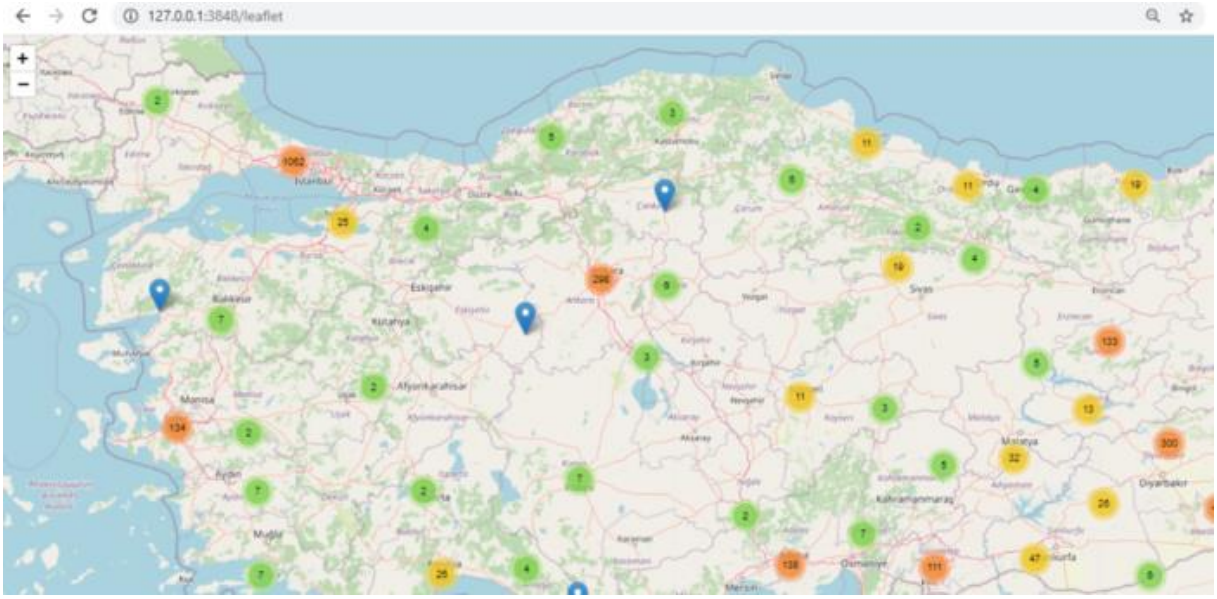
Code	Details
200	Response body


```
[
  {
    "attacktype1_txt": "Armed Assault",
    "count": 500
  },
  {
    "attacktype1_txt": "Assassination",
    "count": 437
  },
  {
    "attacktype1_txt": "Bombing/Explosion",
    "count": 1957
  },
  {
    "attacktype1_txt": "Facility/Infrastructure Attack",
    "count": 279
  },
  {
    "attacktype1_txt": "Hijacking",
    "count": 13
  },
  {
    "attacktype1_txt": "Hostage Taking (Barricade Incident)",
    "count": 13
  },
  {
    "attacktype1_txt": "Hostage Taking (Kidnapping)",
    "count": 158
```


Download

Şekil 7: Update Metodu Sonucu

Ek olarak **Request URL** ile tarayıcıdan da sonuçları görebilirsiniz.



Şekil 8: Requesst Url ile Tarayıcıdan Dönen Sonuç