# **Problem & Solution**

**Problem**: when uploading file via a program it has a payload limit of 36 MB.

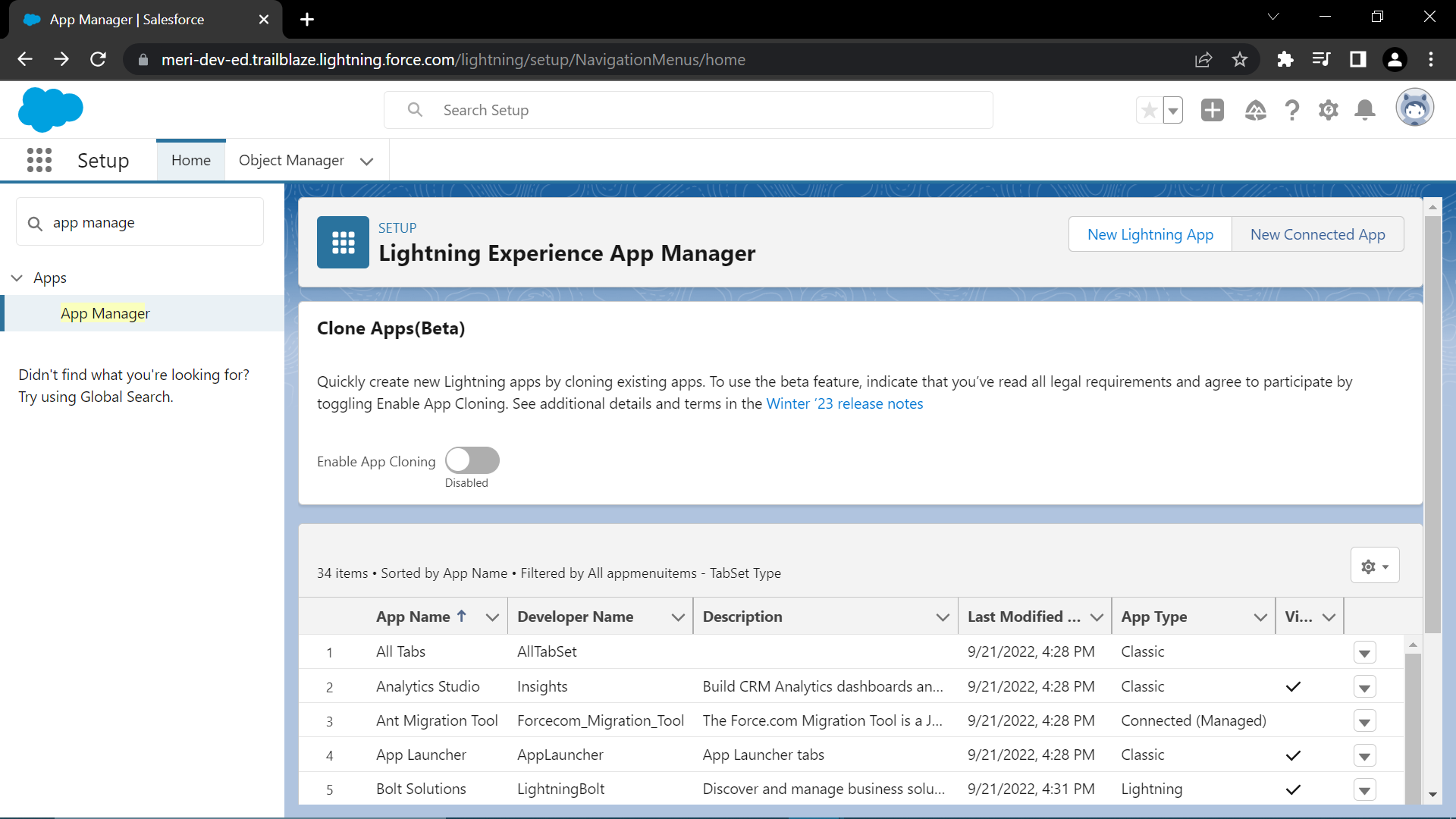
**Solution**: this limit can be overcome by uploading the file as multi-part/form data.

# **Step 1:**

Upload the file to salesforce via postman, the file should be uploaded as a muti-part/form data.

## **Create new connected app**

1. On setup search app manager
2. Click **New Connected App**



1. **Fill in the details:**

Connected App Name,

API Name,

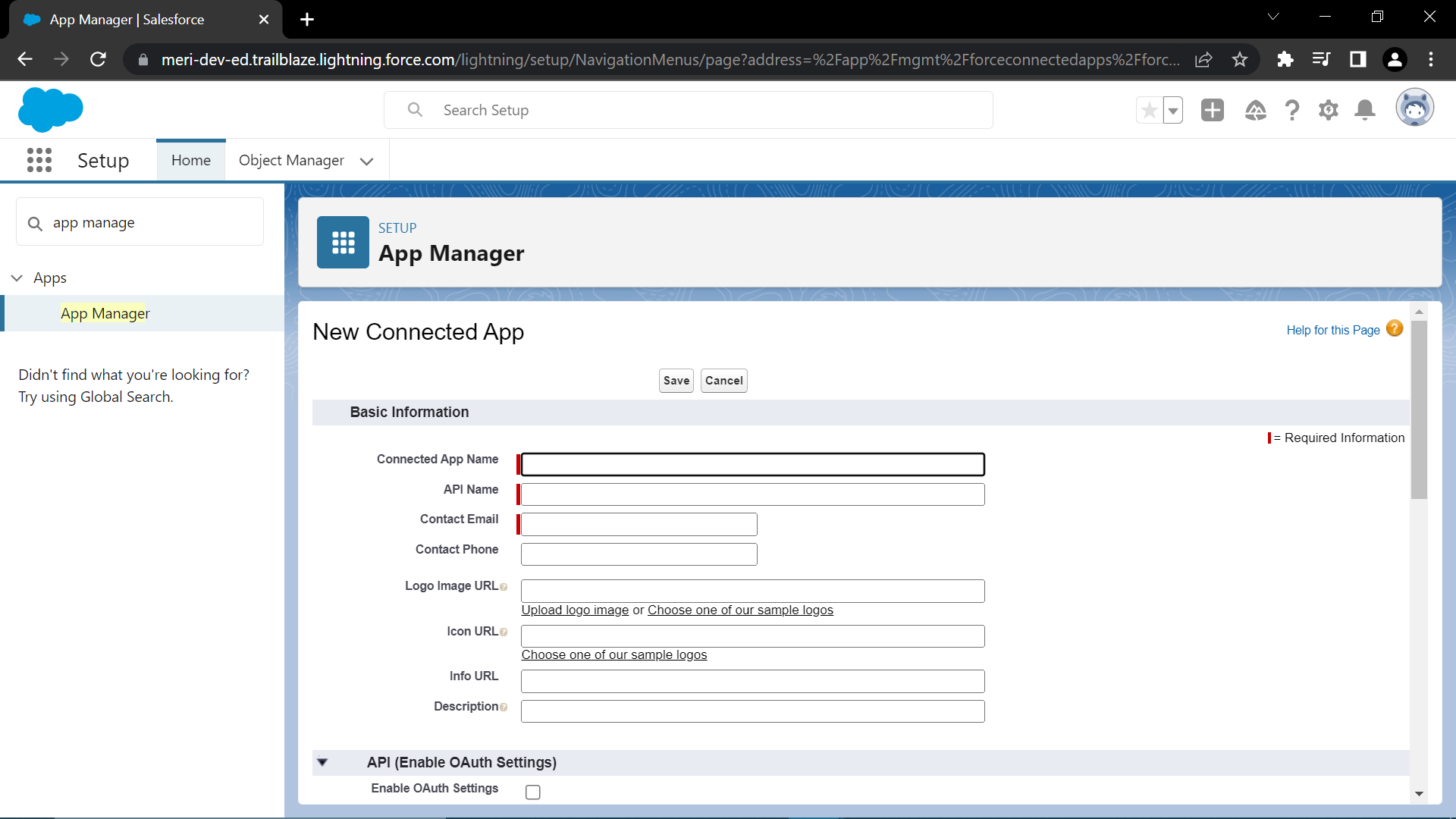
Contact Email,

Enable the OAuth settings checkbox,

In callback URL put the link  <https://login.salesforce.com>

In selected OAuth scopes, select **full access**

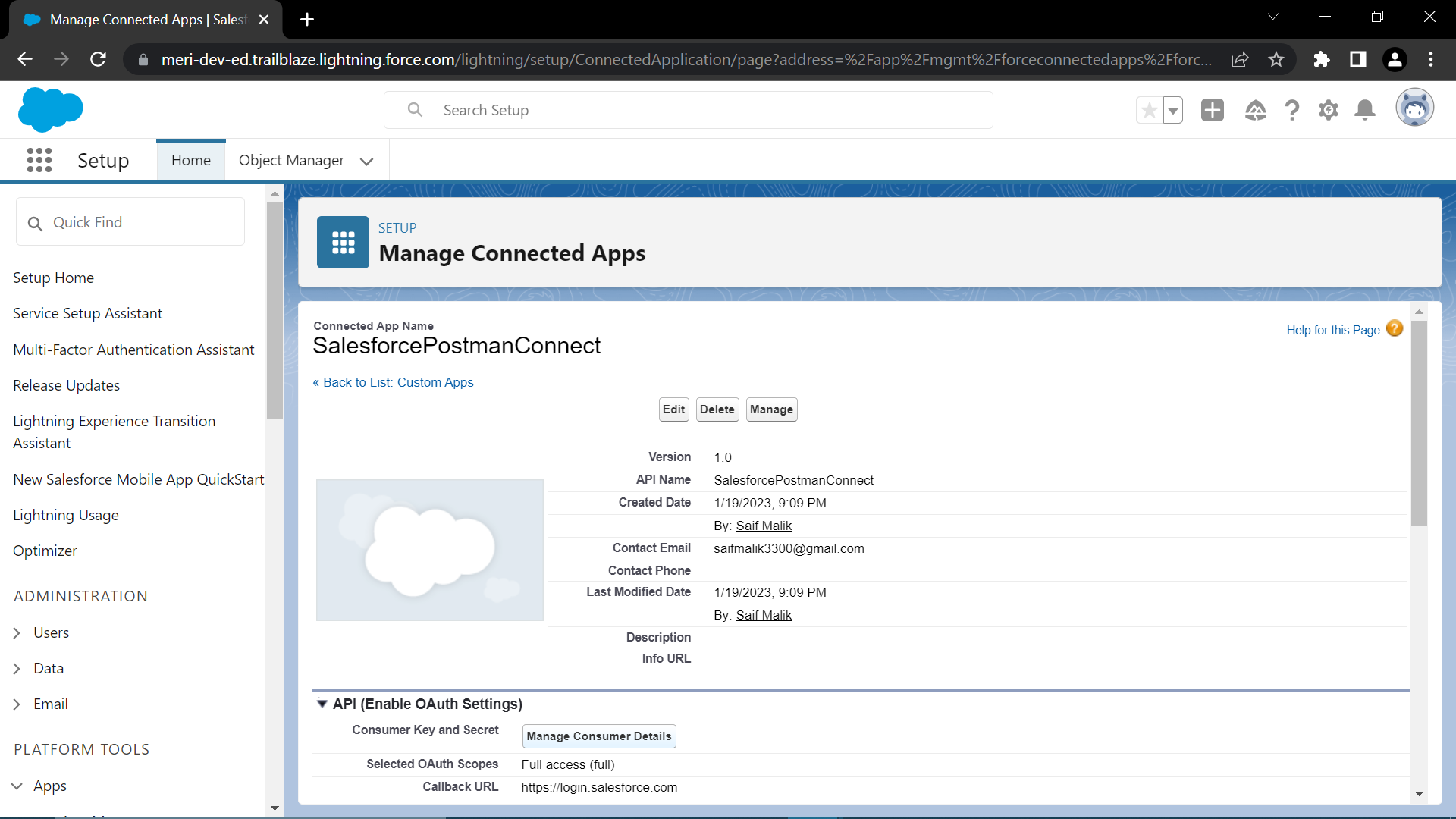
And, then **save**



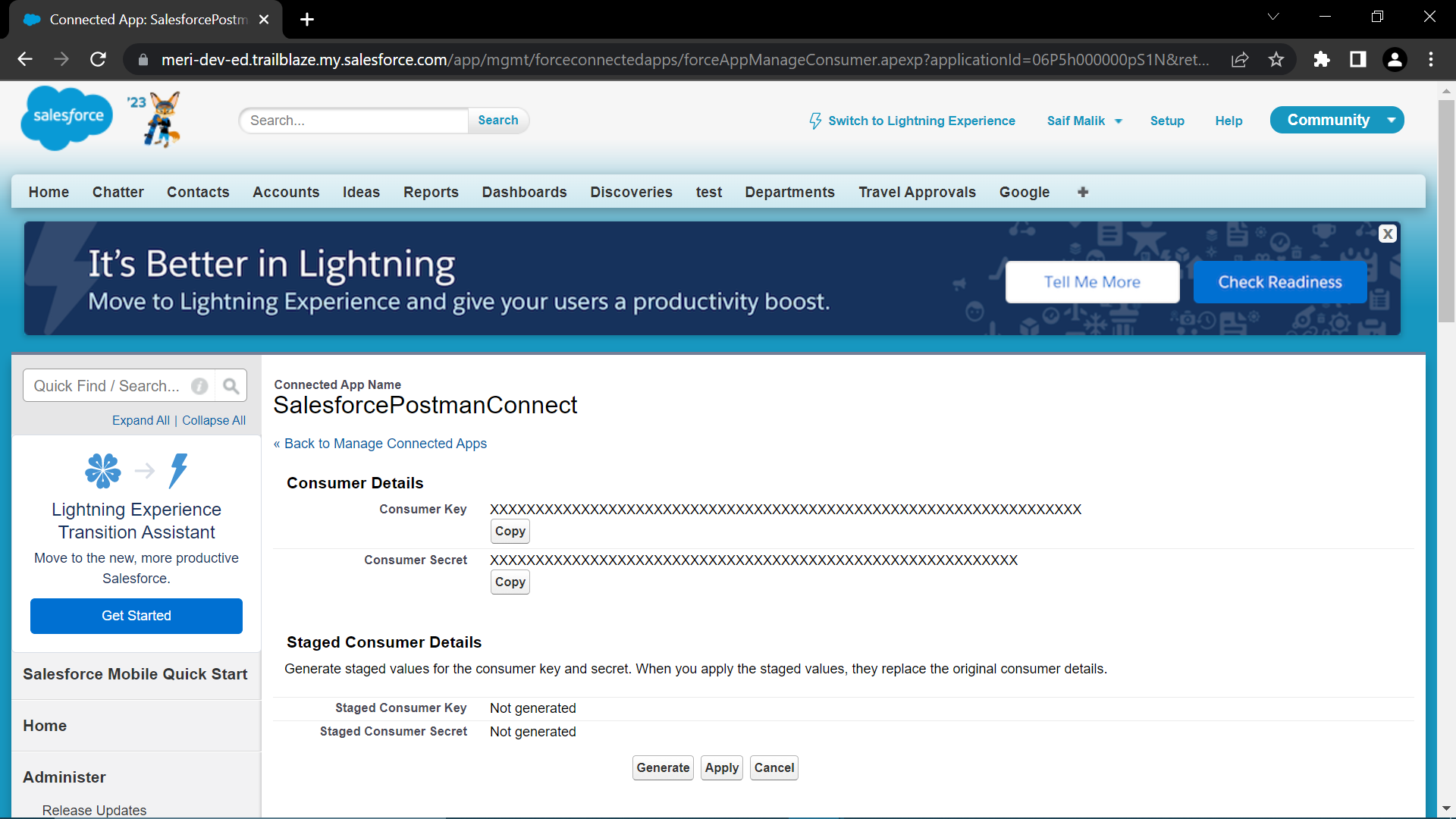
1. After the connected app is created, click **Manage Customer Details** to get client id & client secret

## **Get Client Id & Client Secret**

1. After the connected app is created,
2. Click manage customer details to get **client id** & **client secret**
3. **Customer key** is **Client Id** & **Customer Secret** is **Client Secret**



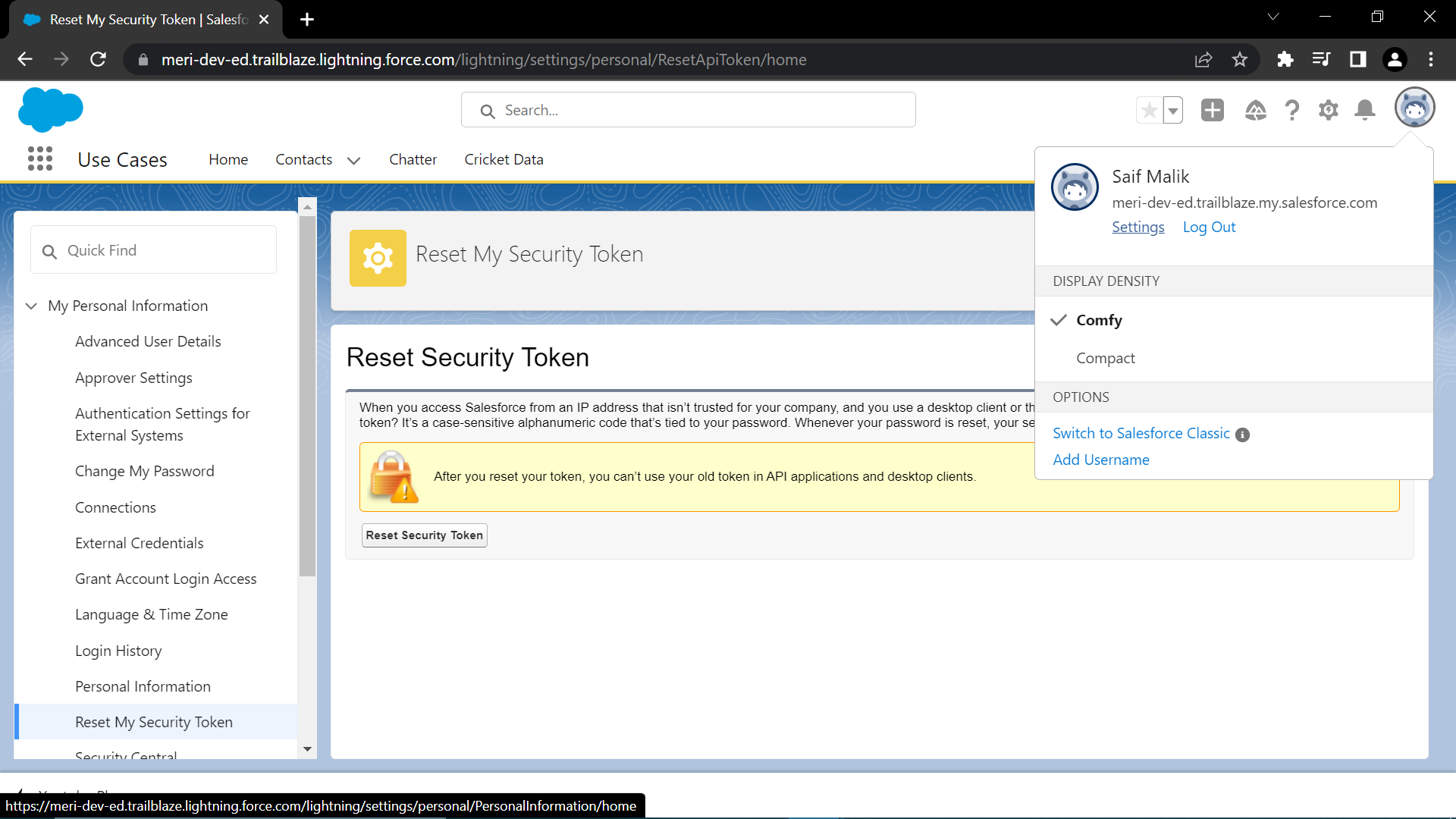
1. After **Manage Customer Details** is clicked this page will open.



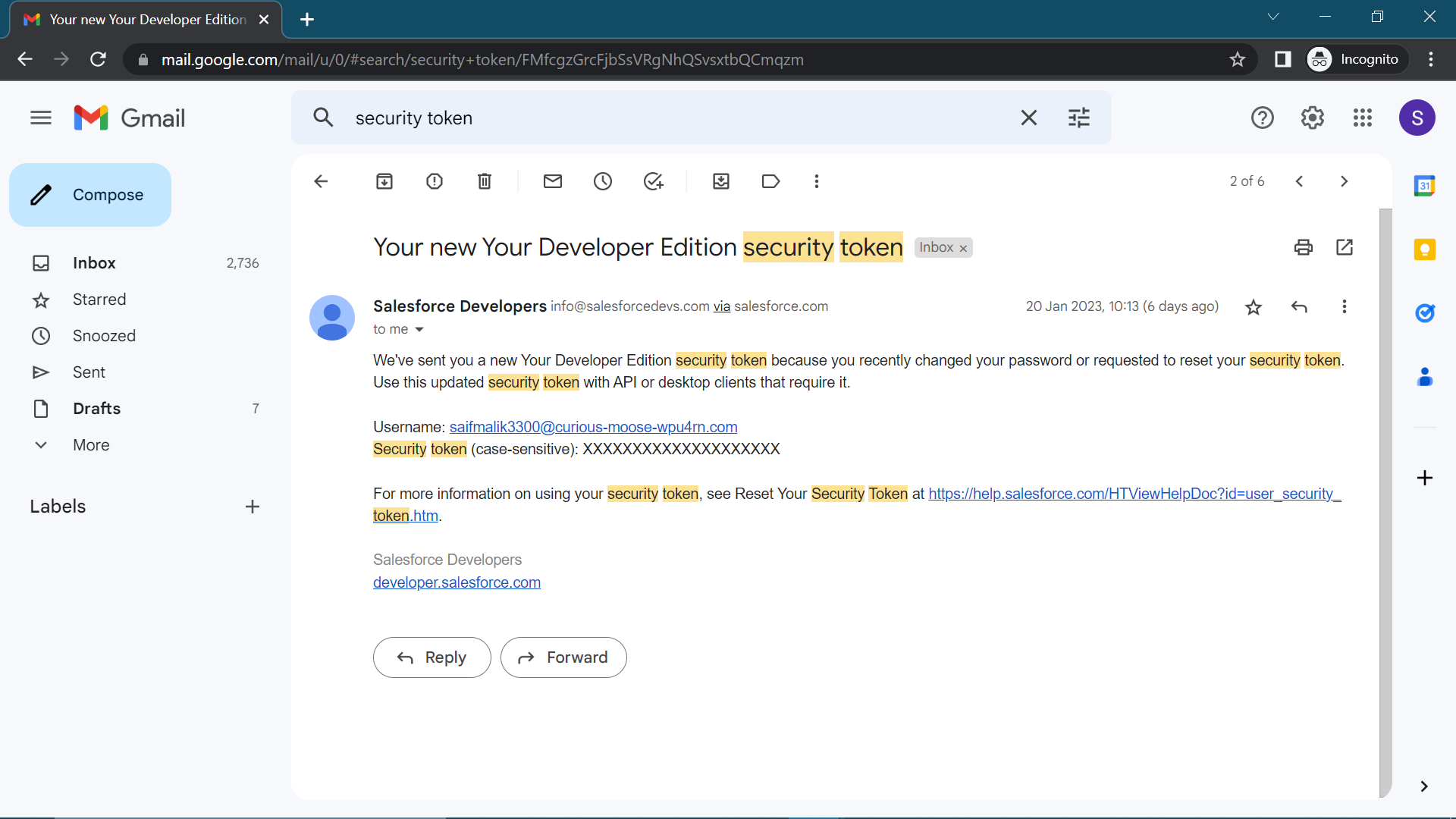
1. Copy **Customer Key** & **Customer Secret**

## **Get security token**

1. Click view profile
2. Click settings
3. Under my personal information, open **Reset My Security Token**



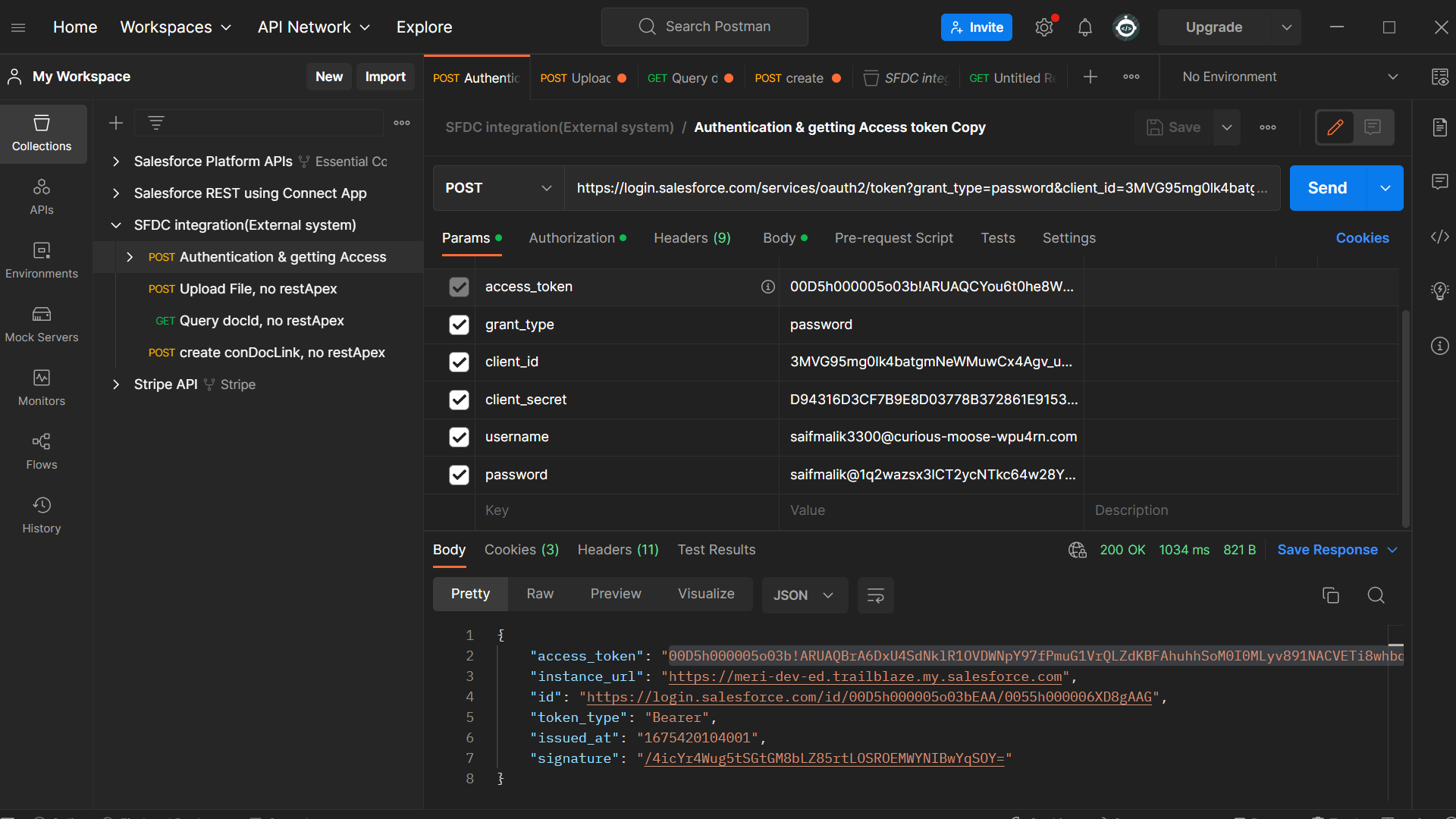
1. Then **Click Reset Security Token** button, this will send a mail with security token in it.



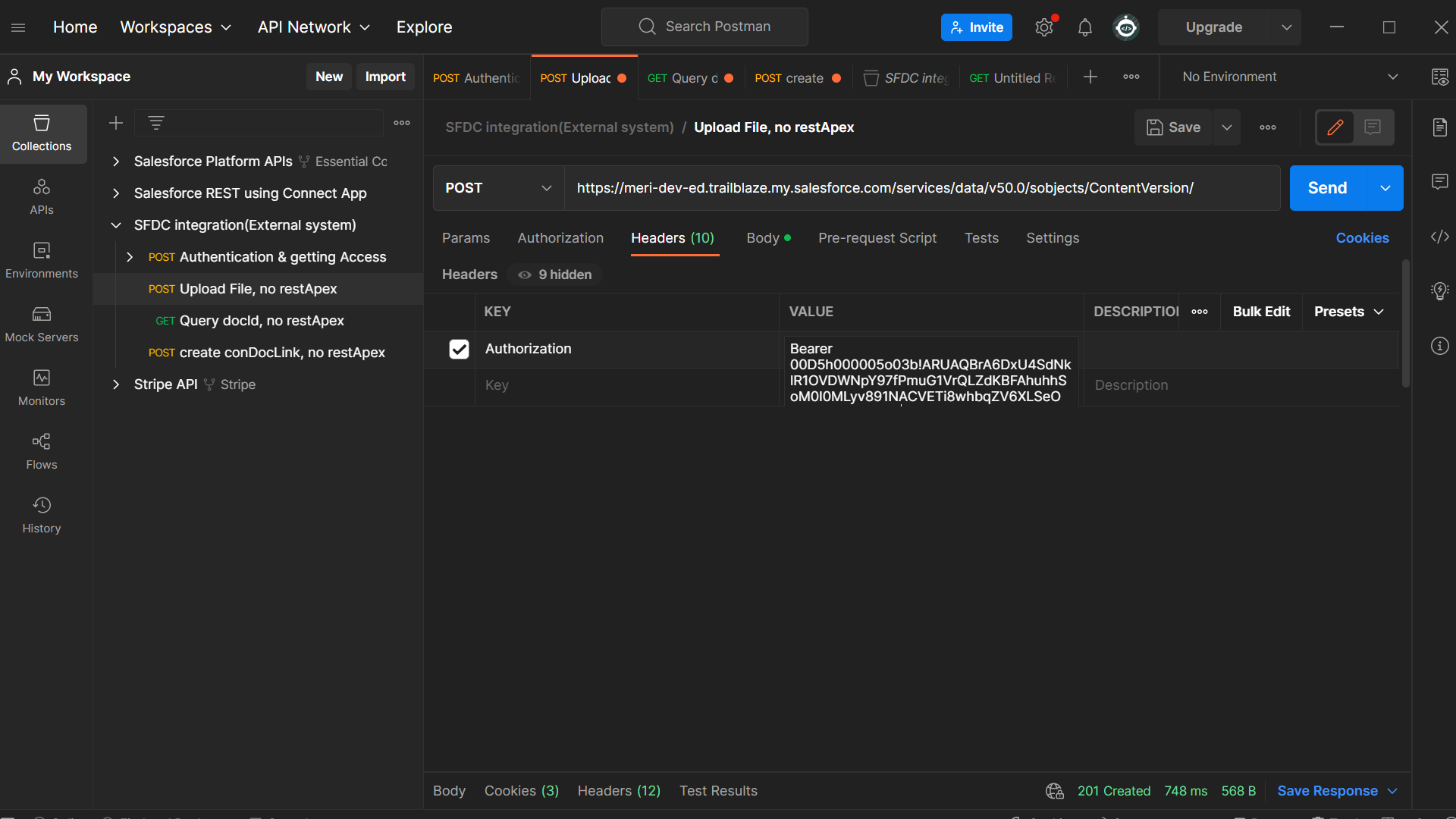
1. Since we have got the **Client Id, Client Secret** and **Secret token.**
2. get **username** & **password** handy
3. now I have got all the credentials, I can authenticate external system & send https requests.

## **Authentication (before sending requests to SFDC):**

1. I will login to salesforce from external system, and I will get the access token so that I won’t have to login again & again.
2. For logging in I will send a POST request to this([https://login.salesforce.com/services/oauth2/token?**grant\_type=**password&**client\_id=**<clientID>&**client\_secret=**<clientSecret>&**username=**<username>&**password=**<password><securityToken](https://login.salesforce.com/services/oauth2/token?grant_type=password&client_id=%3cclientID%3e&client_secret=%3cclientSecret%3e&username=%3cusername%3e&password=%3cpassword%3e%3csecurityToken)>) URL.



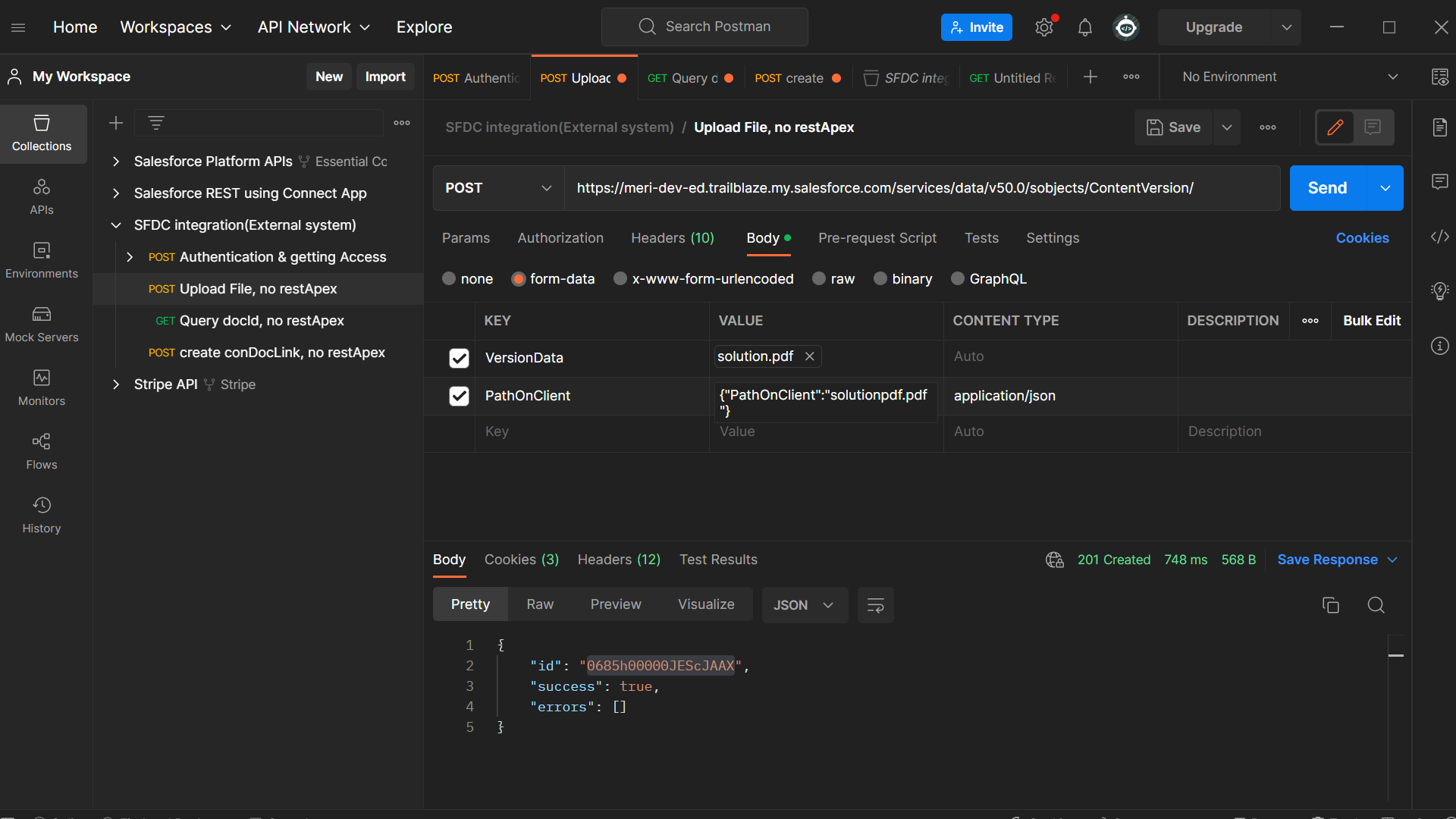
1. Now I have got the access token I will add it to authentication header of every request that will be send to this salesforce org.

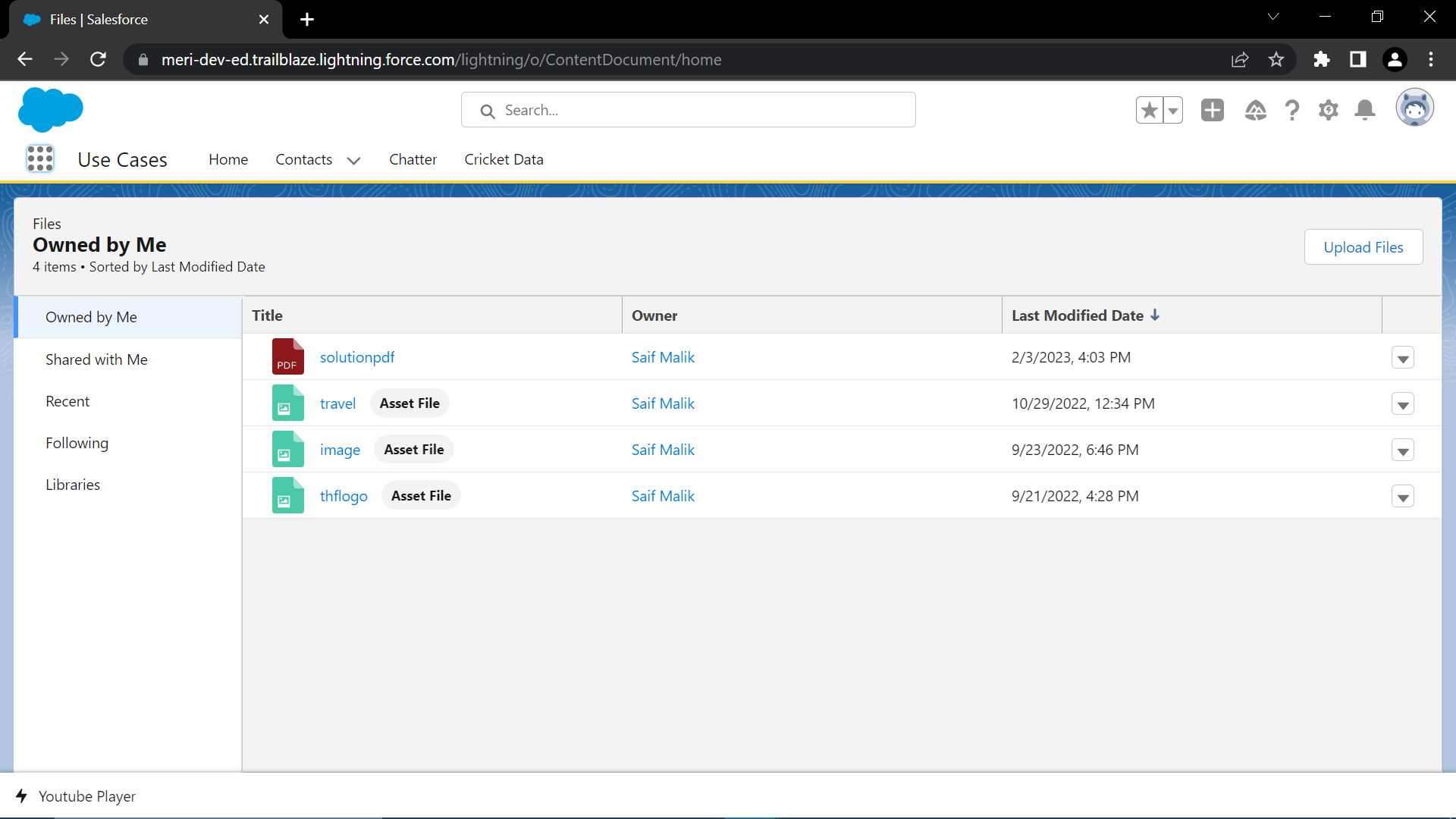


1. Now we can send https requests to our salesforce org.

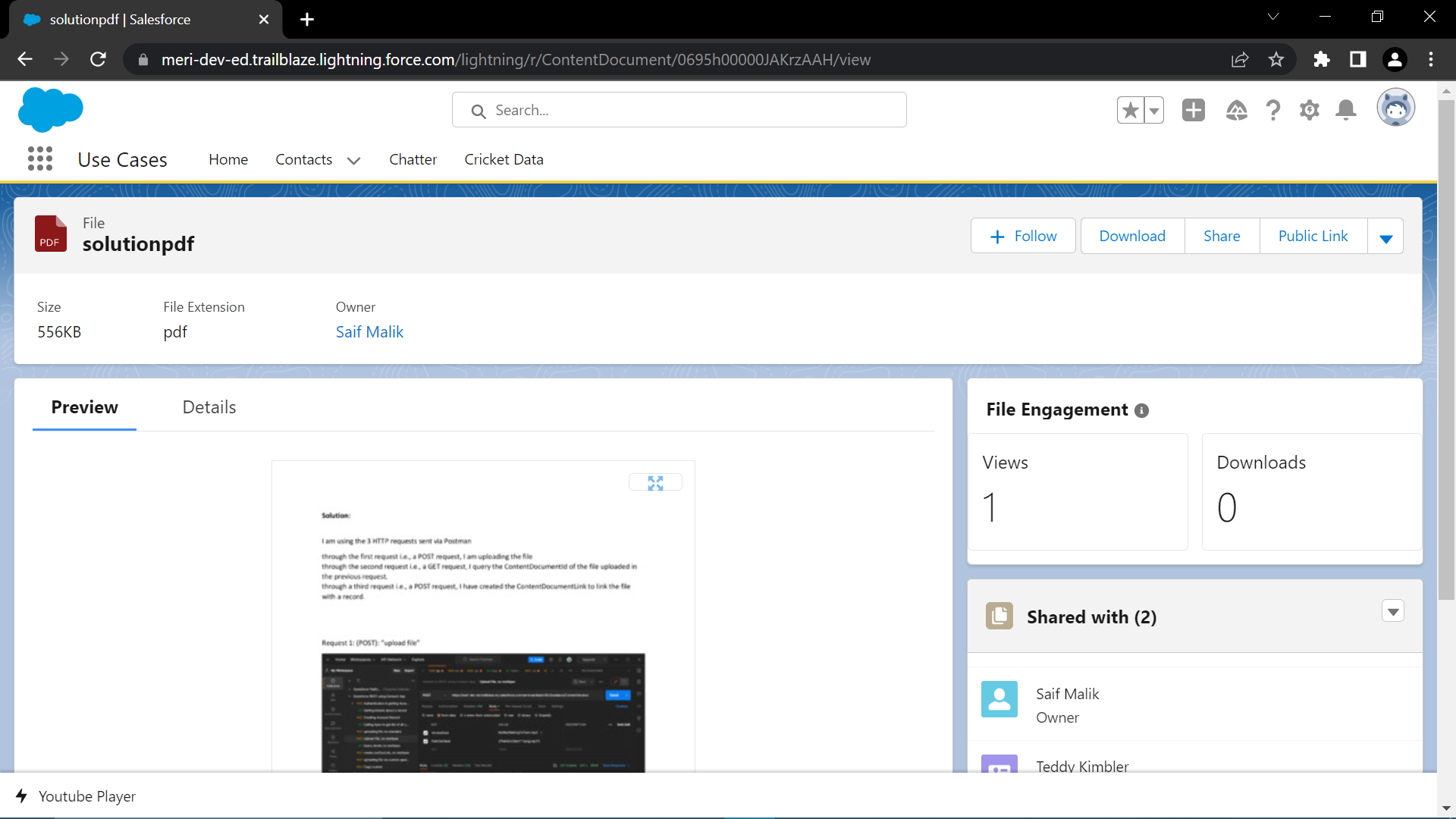
## **Solution:**

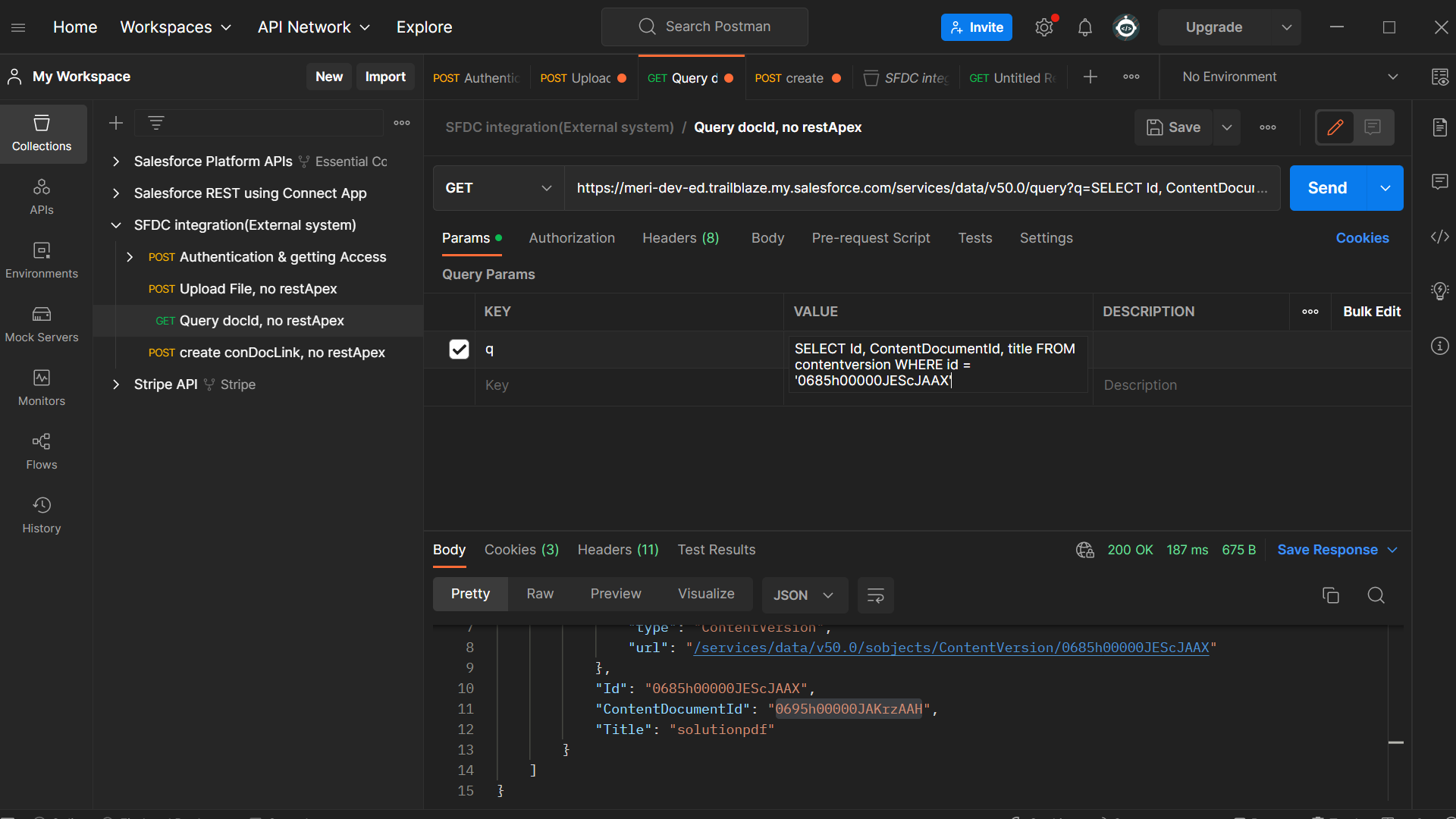
### **Request 1: (POST): “upload file”**



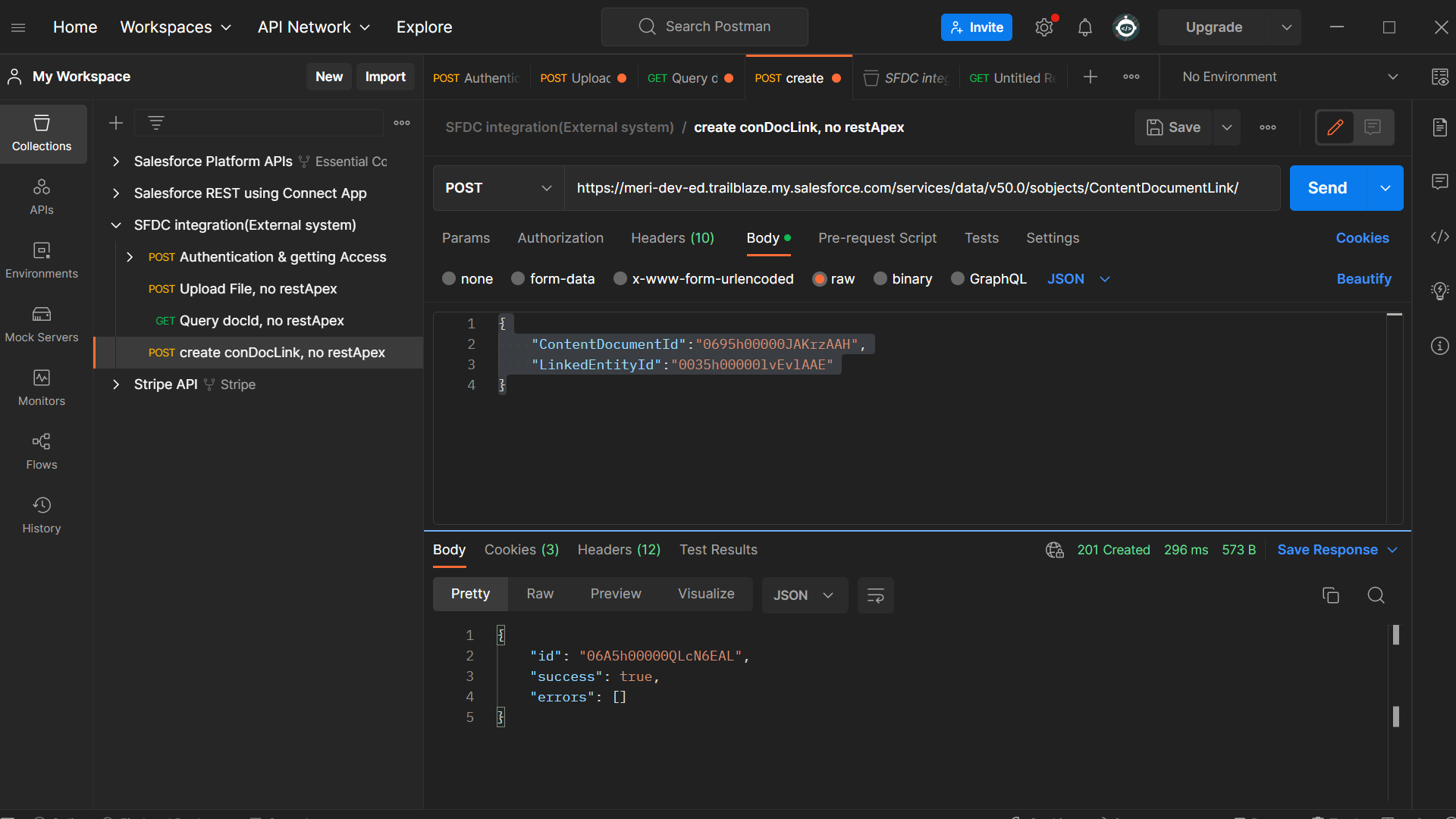


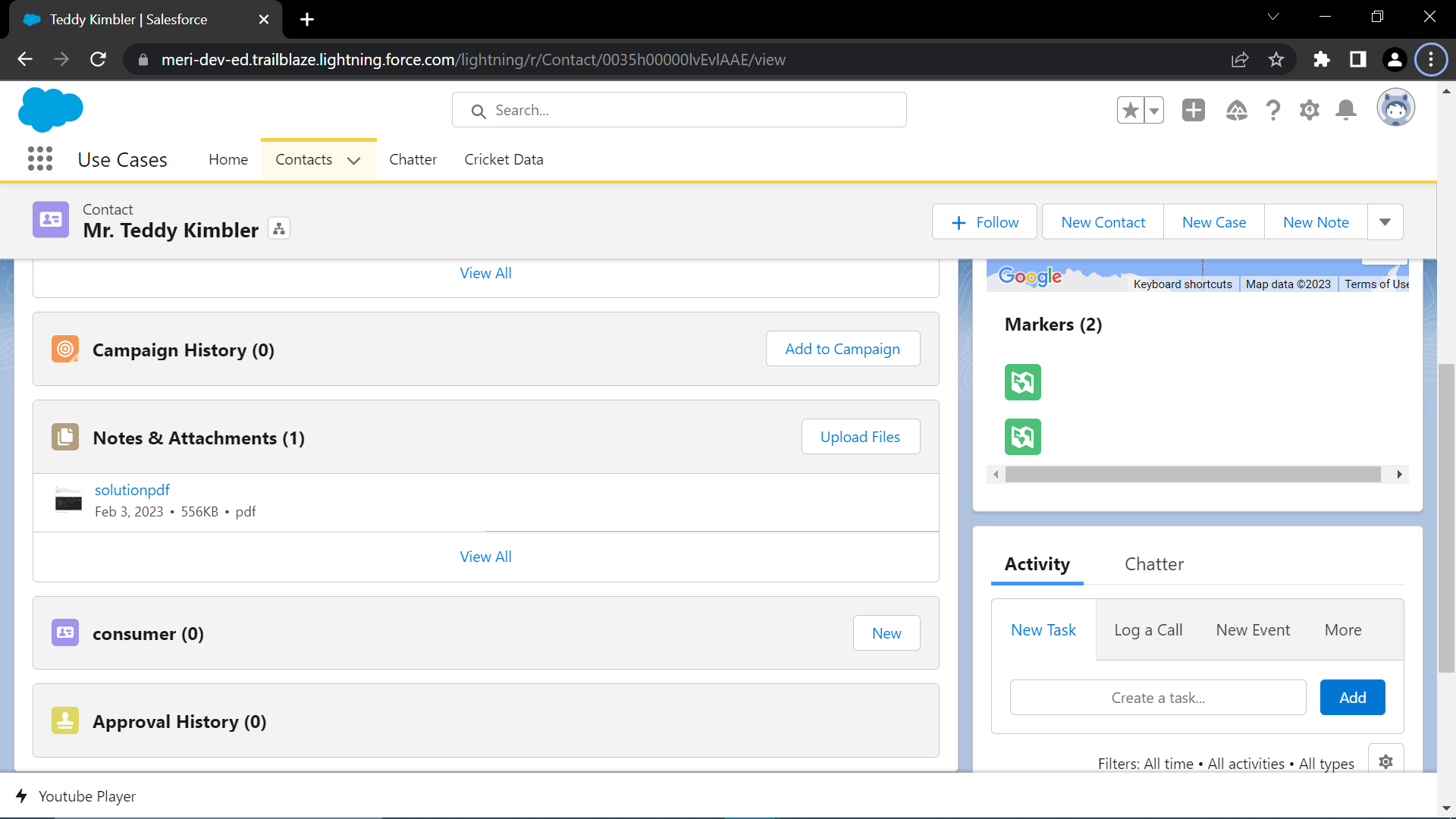
### **Request 2: (GET): “query ContentDocumentId”**





### **Request 3: (POST): “Create ContentDocumentLink”**





# **Step 2:**

Use Node JS and upload file using a JS program.

## **Environment file for Credentials**

SF\_LOGIN\_URL = https://login.salesforce.com

SF\_USERNAME = usernameXXXX@usernameXXXX.com

SF\_PASSWORD = passwordHereXXXXX

SF\_SECURITY\_TOKEN = XXXXXXXXXXXXXXXXXXXX

I have saved it as .env file

## **Program to upload file**

Change FILE\_PATH & LinkedEntityIdd according to your need and then run the code

// upload file and link it to record using firstpublishlocationid

// required

const jsforce = require("jsforce");

require("dotenv").config();

const fs = require("fs");

// required for multi-part file upload

var FormData = require("form-data");

const mime = require("mime-types");

const axios = require("axios");

// required credentials and parameters

const { SF\_LOGIN\_URL, SF\_USERNAME, SF\_PASSWORD, SF\_SECURITY\_TOKEN } =

  process.env;

// about file & record to link file to

FILE\_PATH = "/Users/Dell/Desktop/HorseRide.mp4"; // file to be uploaded

const LinkedEntityIdd = "0035h00000lvEvlAAE"; // record Id of record. to which we want to link file

// main method that log into SFDC, uploads file and links it to a record

const main = async (

*SF\_LOGIN\_URL*,

*SF\_USERNAME*,

*SF\_PASSWORD*,

*SF\_SECURITY\_TOKEN*,

*FILE\_PATH*,

*LinkedEntityIdd*

) => {

  const conn = new jsforce.Connection({

    loginUrl: SF\_LOGIN\_URL,

  });

  // log into salesforce

  console.log("logging in to salesforce ....");

  await conn.login(SF\_USERNAME, SF\_PASSWORD + SF\_SECURITY\_TOKEN, (*err*, *res*) => {

    if (err) {

      console.log(err);

    } else {

      console.log("logged in!");

      console.log(res);

      // read file size scope start

      console.log("reading file size ....");

      fs.stat(FILE\_PATH, (*err*, *stats*) => {

        if (err) {

          console.error(err);

        } else {

          // read file size scope start

          fileSizeInBytes = stats.size;

          console.log(`File size: ${fileSizeInBytes} bytes`);

          if (fileSizeInBytes > 36000000) {

            // create multi-part form data to send file

            console.log("file size greater than 36 MB");

            const createFormData = (*file*) => {

              // File

              //console.log("File received:", file);

              // linking it with a record

              console.log("creating the multi-part form data ....");

              const contentVersion = {

                FirstPublishLocationId: LinkedEntityIdd, //Id to which the content version needs to be linked

                Title: FILE\_PATH.replace(/^.\*[\\\/]/, ""), //get file name from file path

                PathOnClient: FILE\_PATH.replace(/^.\*[\\\/]/, ""), //get file name from file path

                Origin: "H",

              };

              const form = new FormData();

              form.setBoundary("boundary\_string");

              form.append("entity\_content", JSON.stringify(contentVersion), {

                contentType: "application/json",

              });

              form.append("VersionData", file, {

                filename: FILE\_PATH.replace(/^.\*[\\\/]/, ""), //get file name from file path

                contentType: mime.lookup(FILE\_PATH.replace(/^.\*[\\\/]/, "")), //get file name from file path

              });

              return form;

            };

            // read file, to create multi-part form data

            console.log("reading file, to create multi-part form data ....");

            fs.readFile(FILE\_PATH, { encoding: "base64" }, (*err*, *data*) => {

              if (err) {

                console.error(err);

                return;

              } else {

                const formData = createFormData(data);

                // upload file as multi-part form data

                console.log("uploading as mult-part form data ....");

                const URL =

                  conn.instanceUrl +

                  "/services/data/v51.0/sobjects/ContentVersion";

                axios({

                  method: "post",

                  maxContentLength: Infinity,

                  maxBodyLength: Infinity,

                  url: URL,

                  headers: {

                    Authorization: "Bearer " + conn.accessToken,

                    "Content-Type": `multipart/form-data; boundary=\"boundary\_string\"`,

                  },

                  data: formData,

                });

                console.log("File Uploading ....");

                // uploading as multi-part scope over, inside if

              }

            });

          } else {

            // file size < 36 MB. hence, no need of multi-part form data

            // read file, convert to base64 scope start

            console.log("reading file, converting to base64 ....");

            fs.readFile(FILE\_PATH, { encoding: "base64" }, (*err*, *data*) => {

              if (err) {

                console.error(err);

                return;

              } else {

                base64Data = data;

                //console.log("base64 Data:", base64Data);

                // upload file scope start

                console.log("uploading file ....");

                conn.sobject("ContentVersion").insert(

                  {

                    Title: FILE\_PATH.replace(/^.\*[\\\/]/, ""), //get file name from file path

                    PathOnClient: FILE\_PATH.replace(/^.\*[\\\/]/, ""), //get file name from file path

                    VersionData: base64Data,

                    FirstPublishLocationId: LinkedEntityIdd, //Id to which the content version needs to be linked

                    Origin: "H",

                  },

                  (*err*, *res*) => {

                    if (err) {

                      console.log("error:", err);

                    } else {

                      // file uploaded

                      console.log("File uploaded!");

                      console.log(res);

                      // file uploading scope over, inside else

                    }

                  }

                );

                // reading file and converting to base64 scope over, inside else

              }

            });

          } // else scope i.e if file size < 36 MB. that scope over

          // read file size scope over

        }

      });

      // logged in scope over

    }

  });

}; // main method scope over

// calling main method with all the required parameters

main(

  SF\_LOGIN\_URL,

  SF\_USERNAME,

  SF\_PASSWORD,

  SF\_SECURITY\_TOKEN,

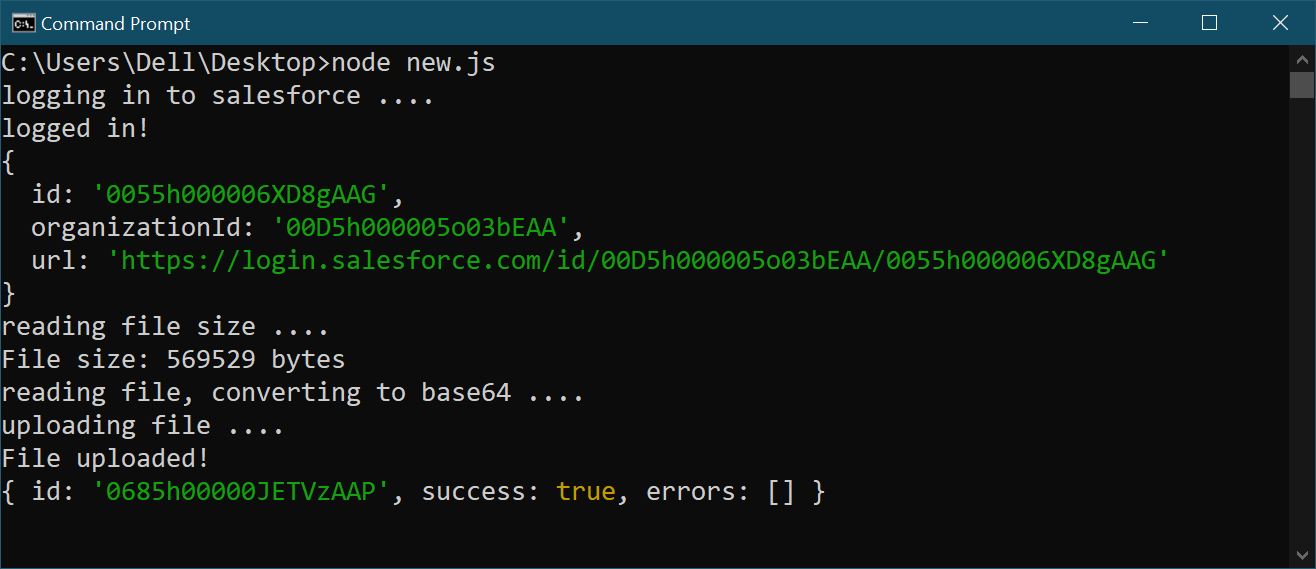
  FILE\_PATH,

  LinkedEntityIdd

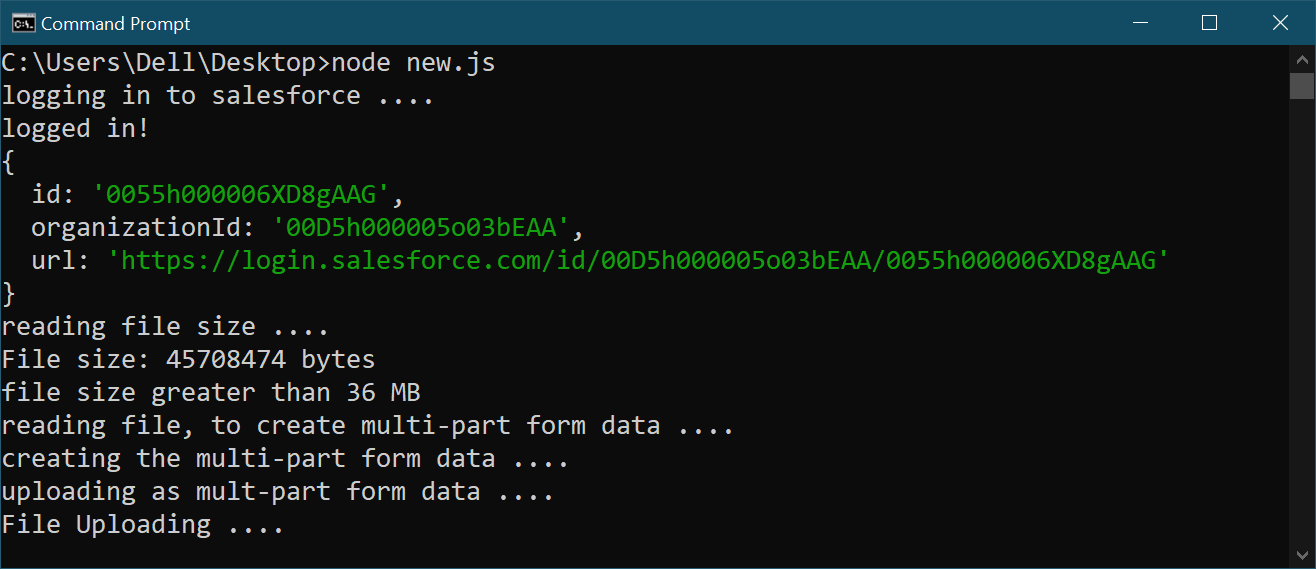
);

I have installed NodeJS and I have saved the code as “new.js”

## **Executing code for payload less than 36 MB**



## **Executing code for payload size greater than 36 MB**



# **Notes:**

* The default limit for Salesforce Classic is 25MB. Support can increase the limit up to 65MB. Attachments larger than 36MB can be attached only via User Interface.
* This limit increase does not apply to OnDemand Email-to-Case as that 25mb limit is hard coded
* The Salesforce Lightning attachment limit is set at 2GB (a hard-coded limit).
* This limit controls the maximum size of an attachment when uploaded using the standard related list, 'Notes & Attachments.'

# **References:**