www.bytescout.com

How to split PDF from uploaded file (node for PDF splitting API in JavaScript with ByteScout Cloud API Server

Step-by-step tutorial:How to split PDF from uploaded file (node to have PDF splitting API in JavaScript

These source code samples are listed and grouped by their programming language and functions they use. ByteScout Cloud API Server helps with PDF splitting API in JavaScript. ByteScout Cloud API Server is the ready to use Web API Server that can be deployed in less than 30 minutes into your own in-house server or into private cloud server. Can store data on in-house local server based storage or in Amazon AWS S3 bucket. Processing data solely on the server using buil-in ByteScout powered engine, no cloud services are used to process your data!.

Use the code displayed below in your application to save a lot of time on writing and testing code. Follow the tutorial and copy - paste code for JavaScript into your project's code editor. Want to see how it works with your data then code testing will allow the function to be tested and work properly.

Free! Free! Free! ByteScout free trial version is available for FREE download from our website. Programming tutorials along with source code samples are assembled.

FOR MORE INFORMATION AND FREE TRIAL:

Download Free Trial SDK (on-premise version)

Read more about ByteScout Cloud API Server

Explore API Documentation

Get Free Training for ByteScout Cloud API Server

Get Free API key for Web API

visit www.ByteScout.com

Source Code Files:

SplitPdfFromUploadedFile.js

```
/*jshint esversion: 6 */

// Please NOTE: In this sample we're assuming Cloud Api Server is hosted at "https://l
// If it's not then please replace this with with your hosting url.

var https = require("https");
var path = require("path");
var fs = require("fs");

// `request` module is required for file upload.
// Use "npm install request" command to install.
var request = require("request");

// Source PDF file to split
const SourceFile = "./sample.pdf";
// Comma-separated list of page numbers (or ranges) to process. Example: '1,3-5,7-'.
const Pages = "1-2,3-";

// 1. RETRIEVE PRESIGNED URL TO UPLOAD FILE.
getPresignedUrl(SourceFile)
    .then(([uploadUrl, uploadedFileUrl]) => {
        // 2. UPLOAD THE FILE TO CLOUD.
        uploadFile(SourceFile, uploadUrl)
            .then(() => {
                // 3. SPLIT UPLOADED PDF
                splitPdf(uploadedFileUrl, Pages);
            })
            .catch(e => {
                console.log(e);
            });
    })
    .catch(e => {
        console.log(e);
    });


function getPresignedUrl(localFile) {
    return new Promise(resolve => {
        // Prepare request to `Get Presigned URL` API endpoint
        let queryPath = `/file/upload/get-presigned-url?contenttype=application/octet-s
        let reqOptions = {
            host: "localhost",
            path: encodeURI(queryPath)
        };
        // Send request
        https.get(reqOptions, (response) => {
            response.on("data", (d) => {
                let data = JSON.parse(d);
                if (data.error == false) {
                    // Return presigned url we received
                    resolve([data.presignedUrl, data.url]);
                }
                else {
                    // Service reported error
                    console.log("getPresignedUrl(): " + data.message);
```

```
                }
            });
        })
            .on("error", (e) => {
                // Request error
                console.log("getPresignedUrl(): " + e);
            });
    });
}

function uploadFile(localFile, uploadUrl) {
    return new Promise(resolve => {
        fs.readFile(SourceFile, (err, data) => {
            request({
                method: "PUT",
                url: uploadUrl,
                body: data,
                headers: {
                    "Content-Type": "application/octet-stream"
                }
            }, (err, res, body) => {
                if (!err) {
                    resolve();
                }
                else {
                    console.log("uploadFile() request error: " + err);
                }
            });
        });
    });
}

function splitPdf(uploadedFileUrl, pages) {
    // Prepare request to `Make Searchable PDF` API endpoint
    var queryPath = `/pdf/split?pages=${pages}&url=${uploadedFileUrl}&async=True`;
    let reqOptions = {
        host: "localhost",
        path: encodeURI(queryPath),
        method: "GET"
    };
    // Send request
    https.get(reqOptions, (response) => {
        response.on("data", (d) => {
            response.setEncoding("utf8");
            // Parse JSON response
            let data = JSON.parse(d);
            if (data.error == false) {
                console.log(`Job #${data.jobId} has been created!`);
                checkIfJobIsCompleted(data.jobId, data.url);
            }
            else {
                // Service reported error
                console.log("splitPdf(): " + data.message);
            }
        });
    })
        .on("error", (e) => {
            // Request error
            console.log("splitPdf(): " + e);
        });
```

```
}

function checkIfJobIsCompleted(jobId, resultFileUrlJson) {
    console.log("Here..");
    let queryPath = `/job/check?jobid=${jobId}`;
    let reqOptions = {
        host: "localhost",
        path: encodeURI(queryPath),
        method: "GET"
    };

    https.get(reqOptions, (response) => {
        response.on("data", (d) => {
            response.setEncoding("utf8");

            // Parse JSON response
            let data = JSON.parse(d);
            console.log(`Checking Job #${jobId}, Status: ${data.status}, Time: ${new Da

            if (data.status == "working") {
                // Check again after 3 seconds
                setTimeout(function () { checkIfJobIsCompleted(jobId, resultFileUrlJson
            }
            else if (data.status == "success") {

                request({ method: 'GET', uri: resultFileUrlJson, gzip: true },
                    function (error, response, body) {

                        // Parse JSON response
                        let respJsonFileArray = JSON.parse(body);
                        let part = 1;

                        respJsonFileArray.forEach((url) => {
                            var localFileName = `./part${part}.pdf`;
                            var file = fs.createWriteStream(localFileName);
                            https.get(url, (response2) => {
                                response2.pipe(file)
                                    .on("close", () => {
                                        console.log(`Generated PDF file saved as "${loc
                                    });
                            });
                            part++;
                        }, this);

                    });
            }
            else {
                console.log(`Operation ended with status: "${data.status}".`);
            }
        })
    });
}
```

VIDEO

https://www.youtube.com/watch?v=NEwNs2b9YN8

ON-PREMISE OFFLINE SDK

60 Day Free Trial or Visit ByteScout Cloud API Server Home Page
Explore ByteScout Cloud API Server Documentation
Explore Samples
Sign Up for ByteScout Cloud API Server Online Training

ON-DEMAND REST WEB API

Get Your API Key
Explore Web API Docs
Explore Web API Samples

visit www.ByteScout.com

visit www.PDF.co