How to split PDF from uploaded file (node for PDF splitting API in JavaScript and PDF.co Web API

Step By Step Tutorial: how to split PDF from uploaded file (node for PDF splitting API in JavaScript

Today we will explain the steps and algorithm of how to split PDF from uploaded file (node and how to make it work in your application. PDF.co Web API was made to help with PDF splitting API in JavaScript. PDF.co Web API is the flexible Web API that includes full set of functions from e-signature requests to data extraction, OCR, images recognition, pdf splitting and pdf splitting. Can also generate barcodes and read barcodes from images, scans and pdf.

JavaScript code snippet like this for PDF.co Web API works best when you need to quickly implement PDF splitting API in your JavaScript application. This JavaScript sample code should be copied and pasted into your project. After doing this just compile your project and click Run. Use of PDF.co Web API in JavaScript is also explained in the documentation included along with the product.

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

JavaScript - SplitPdfFromUploadedFile.js

```
/*jshint esversion: 6 */

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");

// The authentication key (API Key).
// Get your own by registering at https://app.pdf.co/documentation/api
const API_KEY = "**********************************";


// 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(API_KEY, SourceFile)
```

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


function getPresignedUrl(apiKey, localFile) {
    return new Promise(resolve => {
        // Prepare request to `Get Presigned URL` API endpoint
        let queryPath = `/v1/file/upload/get-presigned-url?
contenttype=application/octet-stream&name=${path.basename(SourceFile)}`;
        let reqOptions = {
            host: "api.pdf.co",
            path: encodeURI(queryPath),
            headers: { "x-api-key": API_KEY }
        };
        // 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(apiKey, 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(apiKey, uploadedFileUrl, pages) {
    // Prepare request to `Make Searchable PDF` API endpoint
    var queryPath = `/v1/pdf/split?pages=${pages}&url=${uploadedFileUrl}&async=True`;
    let reqOptions = {
        host: "api.pdf.co",
        path: encodeURI(queryPath),
        method: "GET",
        headers: { "x-api-key": API_KEY }
    };
    // 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 = `/v1/job/check?jobid=${jobId}`;
    let reqOptions = {
        host: "api.pdf.co",
        path: encodeURI(queryPath),
        method: "GET",
        headers: { "x-api-key": API_KEY }
    };

    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
Date().toLocaleString()}`);

            if (data.status == "working") {
```

```
                // Check again after 3 seconds
                setTimeout(function () { checkIfJobIsCompleted(jobId,
resultFileUrlJson) }, 3000);
            }
            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${part}.pdf`;
                            var file = fs.createWriteStream(localFileName);
                            https.get(url, (response2) => {
                                response2.pipe(file)
                                    .on("close", () => {
                                        console.log(`Generated PDF file saved as
"${localFileName} file."`);
                                    });
                            });
                            part++;
                        }, this);

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