Send file as response using NextJS API

Asked 2 years, 6 months ago Modified 1 month ago Viewed 25k times

26

As titled,

In NodeJs + Express, i can return a file as response with the following line

```
res.sendFile(absolute_path_to_the_file)
```

How can i achieve this with NextJs API, assuming if i want to return a single image from output folder inside NextJs directory? I can only see res.send() and res.json() as ways to return response, and im not sure how can i leverage it to return image as response back to the caller.

if i do like this

```
res.send(absolute_path_to_the_file)
```

It will just send me the string of the directory path. What i expect is the image send from the directory denoted by the directory path.

Need help here for this.

node.js express next.js

Share Improve this question Follow

asked Jul 24, 2020 at 4:44

Fred A

1,532 1 20 40

2 aw it's not answered... I'm stuck at the same thing,, You got any luck? – rakesh shrestha Jul 25, 2020 at 21:03

2 @rakeshshrestha i asked the same question at Vercel Github and they responded me with this - github.com/vercel/next.js/discussions/... .. havent tested yet but answer given looks good – Fred A

```
var filePath = path.join(__dirname, 'myfile.mp3');
var stat = fileSystem.statSync(filePath);

response.writeHead(200, {
    'Content-Type': 'audio/mpeg',
    'Content-Length': stat.size
});

var readStream = fileSystem.createReadStream(filePath);
// We replaced all the event handlers with a simple call to readStream.pipe()
readStream.pipe(response);
```

or change the object into buffer and use send method

```
/*
Project structure:
.
├── images_folder
│   └── next.jpg
├── package.json
├── pages
│   ├── api
│   │   └── image.js
│   └── index.js
├── README.md
└── yarn.lock
*/

// pages/api/image.js

import fs from 'fs'
import path from 'path'

const filePath = path.resolve('.', 'images_folder/next.jpg')
const imageBuffer = fs.readFileSync(filePath)

export default function(req, res) {
  res.setHeader('Content-Type', 'image/jpg')
  res.send(imageBuffer)
}
```

Both answers work in my case. Use `process.cwd()` to navigate to the files / image that needed to be send as response.

Share Improve this answer Follow edited Nov 20, 2021 at 22:30 answered May 18, 2021 at 4:15

0

Here is a real-world example, much more advanced, with Sentry for debugging, and returning a stream with dynamic CSV filename. (and TypeScript types)

It's probably not as helpful as the other answer (due to its complexity) but it might be interesting to have a more complete real-world example.

Note that I'm not familiar with streams, I'm not 100% what I'm doing is the most efficient way to go, but it does work.

`src/pages/api/webhooks/downloadCSV.ts`

```
import { logEvent } from '@/modules/core/amplitude/amplitudeServerClient';
import {
  AMPLITUDE_API_ENDPOINTS,
  AMPLITUDE_EVENTS,
} from '@/modules/core/amplitude/events';
import { createLogger } from '@/modules/core/logging/logger';
import { ALERT_TYPES } from '@/modules/core/sentry/config';
import { configureReq } from '@/modules/core/sentry/server';
import { flushSafe } from '@/modules/core/sentry/universal';
import * as Sentry from '@sentry/node';
import {
  NextApiRequest,
  NextApiResponse,
} from 'next';
import stream, { Readable } from 'stream';
import { promisify } from 'util';

const fileLabel = 'api/webhooks/downloadCSV';
const logger = createLogger({
  fileLabel,
});

const pipeline = promisify(stream.pipeline);

type EndpointRequestQuery = {
  /**
   * Comma-separated CSV string.
   *
   * Will be converted into an in-memory stream and sent back to the browser so
it can be downloaded as an actual CSV file.
   */
  csvAsString: string;

  /**
   * Name of the file to be downloaded.
```

```
 *
 * @method GET
 *
 * @example https://753f-80-215-115-17.ngrok.io/api/webhooks/downloadCSV?
downloadAs=bulk-orders-for-student-ambroise-dhenain-
27.csv&csvAsString=beneficiary_name%2Ciban%2Camount%2Ccurrency%2Creference%0AAmbrois
 */
export const downloadCSV = async (req: EndpointRequest, res: NextApiResponse):
Promise<void> => {
  try {
    configureReq(req, { fileLabel });
    const {
      csvAsString,
      downloadAs = 'data.csv',
    } = req?.query as EndpointRequestQuery;

    await logEvent(AMPLITUDE_EVENTS.API_INVOKED, null, {
      apiEndpoint: AMPLITUDE_API_ENDPOINTS.WEBHOOK_DOWNLOAD_CSV,
    });

    Sentry.withScope((scope): void => {
      scope.setTag('alertType', ALERT_TYPES.WEBHOOK_DOWNLOAD_CSV);

      Sentry.captureEvent({
        message: `[downloadCSV] Received webhook callback.`,
        level: Sentry.Severity.Log,
      });
    });

    await flushSafe();
    res.setHeader('Content-Type', 'application/csv');
    res.setHeader('Content-Disposition', `attachment; filename=${downloadAs}`);

    res.status(200);
    await pipeline(Readable.from(new Buffer(csvAsString)), res);
  } catch (e) {
    Sentry.captureException(e);
    logger.error(e.message);

    await flushSafe();

    res.status(500);
    res.end();
  }
};

export default downloadCSV;
```

0

putting the full answer here for better visibility.

```
import fs from "fs";

export default function(req, res) {
  const filePath = path.join(__dirname, 'myfile.mp3');

  const { size } = fs.statSync(filePath);

  res.writeHead(200, {
    'Content-Type': 'audio/mpeg',
    'Content-Length': size,
  });

  const readStream = fs.createReadStream(filePath);

  await new Promise(function (resolve) {
    readStream.pipe(res);

    readStream.on("end", resolve);
  });
});
```

Share Improve this answer Follow

answered Jan 10 at 5:00

Daniel Loureiro

4,065 31 45