

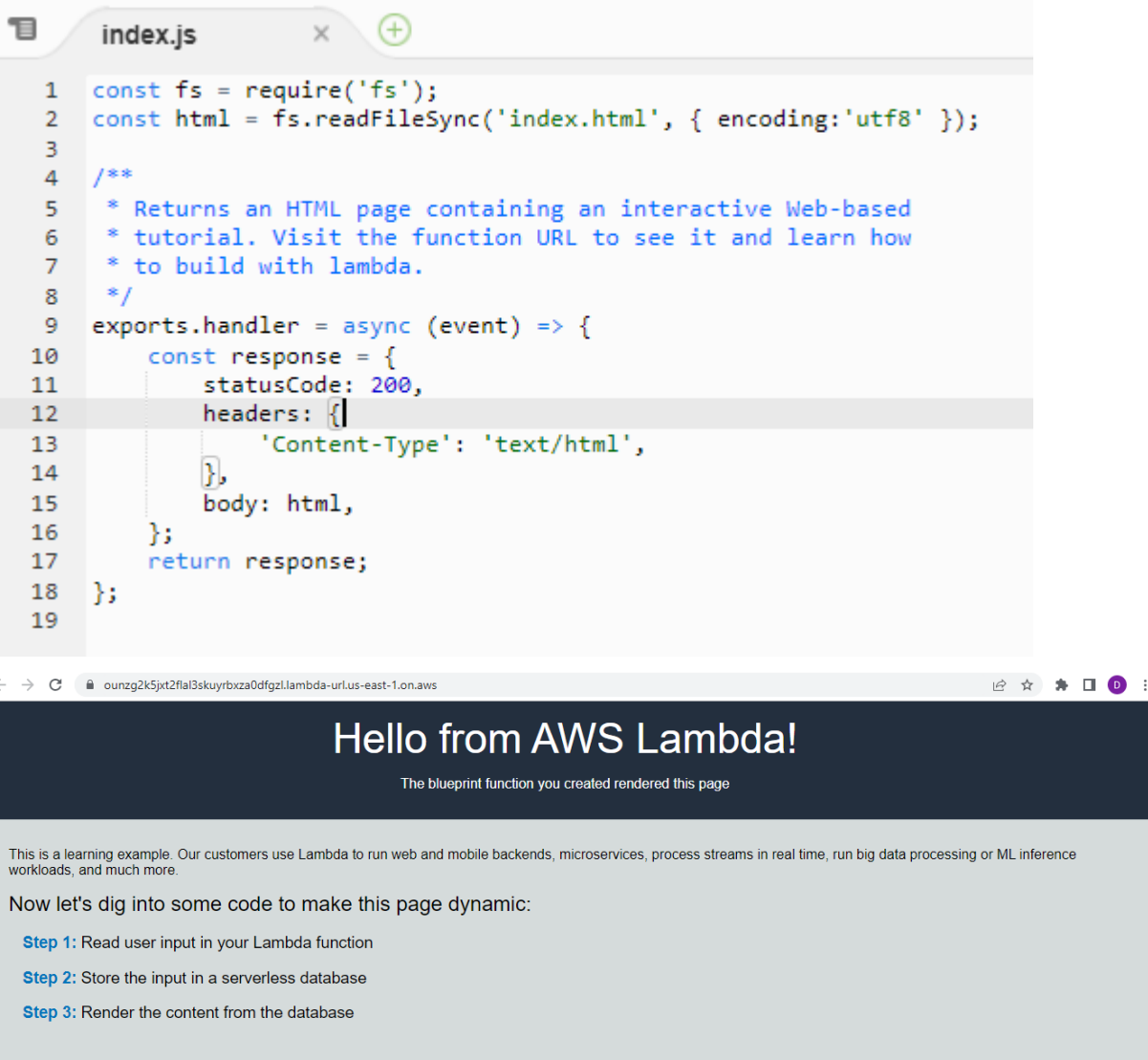
LSC raport  
Lab 6  
Dawid Białka  
24.11.22



**AGH**

## Task 1.

a)



The image shows a code editor window with a file named `index.js`. The code is a JavaScript function that reads a file and returns an HTML response. Below the code editor, a web browser window displays the rendered output of the function.

```
1  const fs = require('fs');
2  const html = fs.readFileSync('index.html', { encoding: 'utf8' });
3
4  /**
5   * Returns an HTML page containing an interactive Web-based
6   * tutorial. Visit the function URL to see it and learn how
7   * to build with lambda.
8   */
9  exports.handler = async (event) => {
10     const response = {
11         statusCode: 200,
12         headers: {
13             'Content-Type': 'text/html',
14         },
15         body: html,
16     };
17     return response;
18 };
19
```

The browser window shows the following content:

# Hello from AWS Lambda!

The blueprint function you created rendered this page

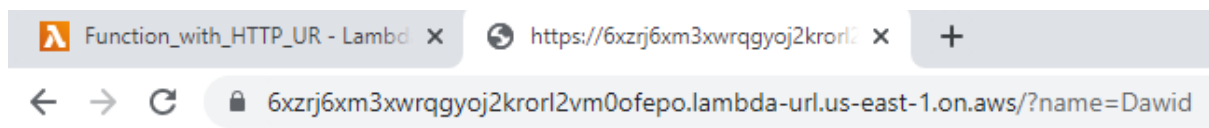
This is a learning example. Our customers use Lambda to run web and mobile backends, microservices, process streams in real time, run big data processing or ML inference workloads, and much more.

Now let's dig into some code to make this page dynamic:

- Step 1:** Read user input in your Lambda function
- Step 2:** Store the input in a serverless database
- Step 3:** Render the content from the database

b)

```
index.js
1 exports.handler = async (event) => {
2   const name = event["queryStringParameters"]["name"]
3   const response = {
4     statusCode: 200,
5     headers: {
6       'Content-Type': 'text/html',
7     },
8     body: `<!DOCTYPE html>
9           <html>
10            <body>
11              <h1>Hello ${name}</h1>
12            </body>
13          </html>`,
14   };
15   return response;
16 };
17
```



**Hello Dawid!**

Task 2.

Upload succeeded  
View details below.

Upload: status

The information below will no longer be available after you navigate away from this page.

Summary

Destination s3://dboutput-a	Succeeded 1 file, 26.0 B (100.00%)	Failed 0 files, 0 B (0.00%)
--------------------------------	---------------------------------------	--------------------------------

Files and folders

Configuration

Files and folders (1 Total, 26.0 B)

dboutput-b [Info](#)[Objects](#)[Properties](#)[Permissions](#)[Metrics](#)[Management](#)[Access Points](#)

## Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. If you want to share objects, you'll need to explicitly grant them permissions. [Learn more](#)

[Copy S3 URI](#)[Copy URL](#)[Download](#)[Open](#)[Delete](#)[Create folder](#)[Upload](#)

Name



Type



Last modified



a.txt

txt

November 26, 2022, 20:37:32 (UTC+01:00)

```
[output-a]
dokument
abcd
uploaded at: 2022-11-27 12:15:36.342576
```

lambda\_function ×



```
1 import json
2 import boto3
3 from datetime import datetime
4
5 def lambda_handler(event, context):
6     s3 = boto3.client('s3')
7     bucket_name = event['Records'][0]['s3']['bucket']['name']
8     file = event['Records'][0]['s3']['object']['key']
9     response = s3.get_object(Bucket=bucket_name, Key=file)
10    content = response['Body'].read()
11    line=content.splitlines()[0]
12    new_content = content + bytes(f'\nuploaded at: {datetime.now()}', 'utf-8')
13
14    if bucket_name == 'dboutput-a' and line == b'[output-a]':
15        s3.put_object(Bucket='dboutput-b', Key=file, Body=new_content)
16    elif bucket == 'dboutput-b' and line == b'[output-b]':
17        s3.put_object(Bucket='dboutput-a', Key=file, Body=new_content)
18
19    return {
20        'statusCode': 200,
21        'body': json.dumps('Completed!')
22    }
23
```