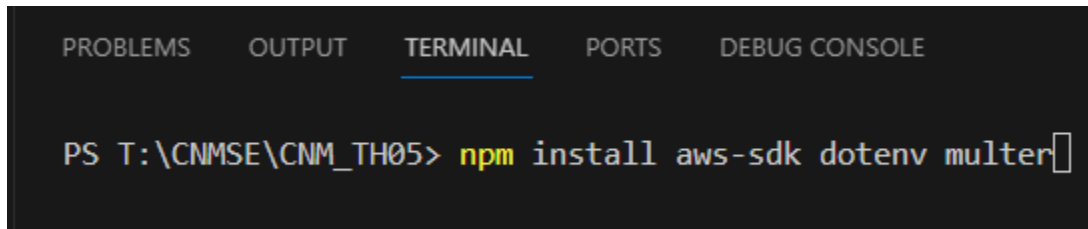
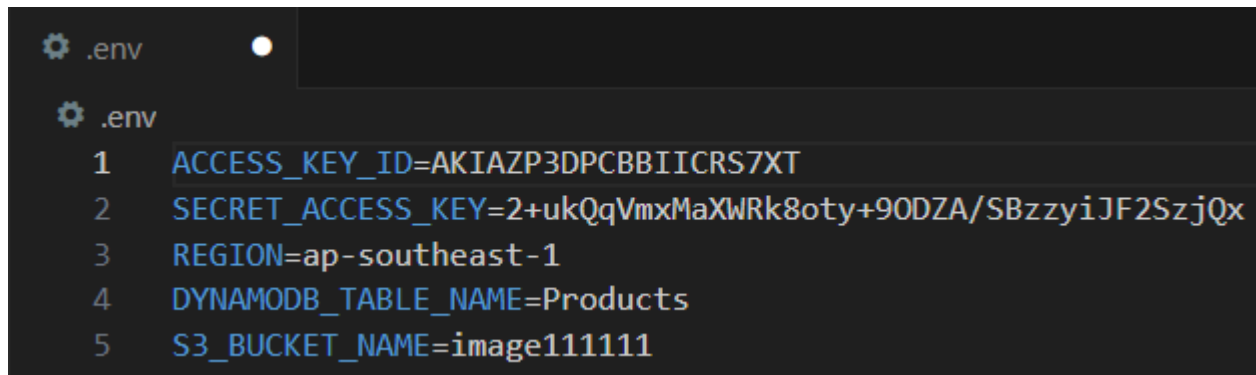


1. Cài đặt thư viện **aws-sdk** và **dotenv** và **multer** để chuẩn bị kết nối lên Cloud AWS:



The screenshot shows a terminal window with tabs for PROBLEMS, OUTPUT, TERMINAL, PORTS, and DEBUG CONSOLE. The TERMINAL tab is active, displaying the command: `PS T:\CNMSE\CNM_TH05> npm install aws-sdk dotenv multer`

2. Tạo file **.env** trong root folder để lưu trữ các giá trị AccessKey, SecretKey tài khoản IAM User của bạn:



The screenshot shows a code editor with a file named `.env` open. The file contains the following environment variables:

```
1 ACCESS_KEY_ID=AKIAZP3DPCBBIICRS7XT
2 SECRET_ACCESS_KEY=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
3 REGION=ap-southeast-1
4 DYNAMODB_TABLE_NAME=Products
5 S3_BUCKET_NAME=image111111
```

- Chú ý region **ap-southeast-1** là khu vực của Singapore.

3. Trong trang **index.js**, khai báo thư viện **aws**, **dotenv**, **multer** và cấu hình AWS kết nối tới Cloud thông qua **accessKey** và **secretAccessKey**.

```

5   const multer = require("multer"); // Khai báo thư viện multer
6   const AWS = require("aws-sdk"); // Khai báo thư viện aws sdk
7   require("dotenv").config(); // Khai báo thư viện dotenv để đọc biến môi trường
8   const path = require("path");
9
10  // Cấu hình AWS
11  process.env.AWS_SDK_JS_SUPPRESS_MAINTENANCE_MODE_MESSAGE = "1"; // Kể từ 2023 v2 đã deprecated, ta chọn sử dụng aws-sdk javascript v2 thay vì v3
12
13  // Cấu hình aws sdk để truy cập vào Cloud Aws thông qua tài khoản IAM user
14  AWS.config.update({
15    region: process.env.REGION,
16    accessKeyId: process.env.ACCESS_KEY_ID,
17    secretAccessKey: process.env.SECRET_ACCESS_KEY,
18  });
19  const s3 = new AWS.S3(); // Khai báo service S3
20  const dynamodb = new AWS.DynamoDB.DocumentClient(); // Khai báo service DynamoDB
21
22  const bucketName = process.env.S3_BUCKET_NAME;
23  const tableName = process.env.DYNAMODB_TABLE_NAME;

```

- **tableName** và **bucketName** là tên bảng DynamoDB và tên Bucket mà bạn đã tạo trên AWS để lưu trữ.

4. Tạo thêm input chọn file Image để upload lên S3 trong file **index.ejs**:

```

<h4>Books:</h4>
<form action="/save" method="post" enctype="multipart/form-data">
  <input type="text" name="maSanPham" id="maSanPham" placeholder="Nhập mã sản phẩm" />
  <br />
  <input type="text" name="tenSanPham" id="tenSanPham" placeholder="Nhập tên sản phẩm" />
  <br />
  <input type="text" name="soLuong" id="soLuong" placeholder="Nhập số lượng" />
  <br />
  <input type="file" name="image" id="image" accept="image/*" />
  <br />
  <input type="submit" value="Thêm" />
</form>

```

- Bổ sung thuộc tính **accept="images/\*"** chỉ cho phép file upload là ảnh.

5. Cấu hình multer để quản lý việc upload hình ảnh:

```

30 // Cấu hình multer quản lý upload image
31 const storage = multer.memoryStorage({
32   destination(req, file, callback) {
33     callback(null, "");
34   },
35 });
36 const upload = multer({
37   storage,
38   limits: { fileSize: 2000000 }, // Chỉ cho phép file tối đa là 2MB
39   fileFilter(req, file, cb) {
40     checkFileType(file, cb);
41   },
42 });
43 function checkFileType(file, cb) {
44   const fileTypes = /jpeg|jpg|png|gif/;
45
46   const extname = fileTypes.test(path.extname(file.originalname).toLowerCase());
47   const mimetype = fileTypes.test(file.mimetype);
48   if (extname && mimetype) {
49     return cb(null, true);
50   }
51   return cb("Error: Pls upload images /jpeg|jpg|png|gif/ only!");
52 }

```

- Hàm **checkFileType** sẽ validate định dạng file upload có phải là hình ảnh hay không.

6. Render data lên trang **index.ejs** từ mảng data lấy từ cloud dymanodb:

```

app.get("/", async (req, res) => {
  try {
    const params = { TableName: tableName };
    const data = await dynamodb.scan(params).promise(); // Dùng hàm scan để lấy toàn bộ dữ
    liệu trong table DynamoDB
    console.log("data=", data.Items);
    return res.render("index.ejs", { data: data.Items }); // Dùng biến response để render
    trang `index.ejs` đồng thời truyền biến `data`
  } catch (error) {
    console.error("Error retrieving data from DynamoDB:", error);
    return res.status(500).send("Internal Server Error");
  }
});

```

```

<form action="/delete" method="post" enctype="multipart/form-data">
  <table border="true">
    <tr>
      <th>Mã sản phẩm</th>
      <th>Tên sản phẩm</th>
      <th>Số lượng</th>
      <th>Hình ảnh</th>
      <th>Chọn</th>
    </tr>
    <% for(let i = 0; i < data.length; i++){ %>
      <tr>
        <td><%= data[i].maSanPham %></td>
        <td><%= data[i].tenSanPham %></td>
        <td><%= data[i].soLuong %></td>
        <td>"
          width="48" height="48" /></td>
        <td><input type="checkbox" name="<%= data[i].maSanPham %>" id="<%= data[i].
          maSanPham %>" /></td>
      </tr>
    <% } %>
  </table>
  <input type="submit" value="Xoá">
</form>

```

7. Lưu data item lên Cloud DynamoDB:

```

app.post("/save", upload.single("image"), (req, res) => {
  // Middleware uploadSingle('image') chính định rằng field có name `image` trong request sẽ được xử lý (lọc, phân
  try {
    const maSanPham = Number(req.body.maSanPham); // Lấy ra các tham số từ body của form
    const tenSanPham = req.body.tenSanPham; // Lấy ra các tham số từ body của form
    const soLuong = Number(req.body.soLuong); // Lấy ra các tham số từ body của form

    const image = req.file?.originalname.split(".");
    const fileType = image[image.length - 1];
    const filePath = `${maSanPham}_${Date.now().toString()}.${fileType}`; // Đặt tên cho hình ảnh sẽ lưu trong s3

    const paramsS3 = {
      Bucket: bucketName,
      Key: filePath,
      Body: req.file.buffer,
      ContentType: req.file.mimetype,
    };

    s3.upload(paramsS3, async (err, data) => { // Upload ảnh lên S3 trước
      if (err) {
        console.error("error=", err);
        return res.send("Internal server error!");
      } else { // Khi upload S3 thành công
        const imageURL = data.Location; // Gán URL S3 trả về vào field trong table DynamoDB
        const paramsDynamoDb = {
          TableName: tableName,
          Item: {
            maSanPham: Number(maSanPham), // Include maSanPham attribute with its value
            tenSanPham: tenSanPham,
            soLuong: soLuong,
            image: imageURL,
          },
        };

        await dynamodb.put(paramsDynamoDb).promise();
        return res.redirect("/"); // Render lại trang index để cập nhật dữ liệu table
      }
    });
  } catch (error) {
    console.error("Error saving data from DynamoDB:", error);
    return res.status(500).send("Internal Server Error");
  }
});

```

- Lưu ý, cần thêm middleware **upload.single('image')** để validate hình ảnh từ form gửi lên trước.
- S3 sẽ trả về urlImage sau khi upload thành công.

Amazon S3 > Buckets > image111111 > 333\_1709520987442.jpg

## 333\_1709520987442.jpg Info

[Copy S3 URI](#) [Download](#) [Open](#) [Object actions](#)

[Properties](#) [Permissions](#) [Versions](#)

### Object overview

Owner	tonlongphuocvn	S3 URI	<a href="s3://image111111/333_1709520987442.jpg">s3://image111111/333_1709520987442.jpg</a>
AWS Region	Asia Pacific (Singapore) ap-southeast-1	Amazon Resource Name (ARN)	<a href="arn:aws:s3:::image111111/333_1709520987442.jpg">arn:aws:s3:::image111111/333_1709520987442.jpg</a>
Last modified	March 4, 2024, 09:56:28 (UTC+07:00)	Entity tag (Etag)	<a href="#">7b3698683ae7777a0e51957838224855</a>
Size	62.0 KB	Object URL	<a href="https://image111111.s3.ap-southeast-1.amazonaws.com/333_1709520987442.jpg">https://image111111.s3.ap-southeast-1.amazonaws.com/333_1709520987442.jpg</a>
Type	jpg		
Key	<a href="#">333_1709520987442.jpg</a>		

## Products

► **Scan or query items**  
Expand to query or scan items.

✔ Completed. Read capacity units consumed: 0.5

### Items returned (2)

	maSanPham...	image	soLuong	tenSanPham
<input type="checkbox"/>	<a href="#">333</a>	<a href="https://image111111.s3.ap-southeast-1.amazonaws.com/333_1709520987442.jpg">https://image111111.s3.ap-southeast-1.amazonaws.com/333_1709520987442.jpg</a>	30	Dien Thoai Iphone 15
<input type="checkbox"/>	<a href="#">123</a>	<a href="https://image111111.s3.ap-southeast-1.amazonaws.com/123_1709520641093.png">https://image111111.s3.ap-southeast-1.amazonaws.com/123_1709520641093.png</a>	13	Tivi LG

8. Xóa item trên cloud DynamoDB:

```

app.post("/delete", upload.fields([]), (req, res) => {
  const listCheckboxSelected = Object.keys(req.body); // Lấy ra tất cả checkboxes
  // req.body trả về 1 object chứa các cặp key & value định dạng:
  // '123456': 'on',
  // '123458': 'on',
  //listCheckboxSelected trả về 1 array: [ '123456', '123458', '96707133' ]
  if (!listCheckboxSelected || listCheckboxSelected.length <= 0) { // Nếu không có gì để xóa
    return res.redirect("/");
  }
  try {
    function onDeleteItem(length) { // Định nghĩa hàm đệ quy xóa

      const params = {
        TableName: tableName,
        Key: {
          maSanPham: Number(listCheckboxSelected[length]),
        },
      };

      dynamodb.delete(params, (err, data) => { // Dùng hàm .delete của aws-sdk
        if (err) {
          console.error("error=", err);
          return res.send("Internal Server Error!");
        } else if (length > 0) onDeleteItem(length - 1); // Nếu vị trí cần xóa vẫn > 0 thì gọi Đệ Quy xóa tiếp
        else return res.redirect("/"); // Render lại trang index.ejs để cập nhật dữ liệu table
      });
    }

    onDeleteItem(listCheckboxSelected.length - 1); // Gọi hàm đệ quy xóa
  } catch (error) {
    console.error("Error deleting data from DynamoDB:", error);
    return res.status(500).send("Internal Server Error");
  }
}

```

- Chạy **npm run start** để xem kết quả