EcommerceSite main

Code Analysis

Overview History Settings

Created 9th Dec 2022 | Snapshot taken by snyk.io a few seconds ago | Retest now

IMPORTED BY
Hashir Sarwar

PROJECT OWNER
Add a project owner

ANALYSIS SUMMARY
128 analyzed files (86%)
Repo breakdown

Issues 31

Search...

SEVERITY
- [ ] High 13
- [ ] Medium 5
- [ ] Low 13

PRIORITY SCORE
Scored between 0 - 1000

STATUS
- [x] Open 31
- [ ] Ignored 0

LANGUAGES
- [ ] JavaScript 27
- [ ] TypeScript 4

VULNERABILITY TYPES
- [ ] Path Traversal 10
- [ ] Improper Type Validation 9
- [ ] Use of Hardcoded Crede... 4
- [ ] Allocation of Resources ... 3
- [ ] Hardcoded Secret 3
- [ ] Cross-Site Request Forg... 1
- [ ] Information Exposure 1

Share your feedback!

31 of 31 issues

Group by none Sort by highest severity

H Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
54      !pCategory |
55      !pOffer |
56      !pStatus
57    ) {
58      Product.deleteImages(images, "file");
```

Unsanitized input from *an uploaded file flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**products.js** 11 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore Full details

H Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
59      return res.json({ error: "All filled must be required" });
60    }
61    // Validate Name and description
62    else if (pName.length > 255 || pDescription.length > 3000) {
63      Product.deleteImages(images, "file");
```

Unsanitized input from *an uploaded file flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**products.js** 11 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore Full details

H Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
66      });
67    }
68    // Validate Images
69    else if (images.length !== 2) {
70      Product.deleteImages(images, "file");
```

Unsanitized input from *an uploaded file flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**products.js** 12 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore Full details

Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
129     });
130   }
131   // Validate Update Images
132   else if (editImages && editImages.length == 1) {
133     Product.deleteImages(editImages, "file");
```

Unsanitized input from *an uploaded file flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**products.js** 13 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore | Full details

Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
19    let cImage = req.file.filename;
20    const filePath = `../server/public/uploads/categories/${cImage}`;
21
22    if (!cName || !cDescription || !cStatus || !cImage) {
23      fs.unlink(filePath, (err) => {
```

Unsanitized input from *an uploaded file flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**categories.js** 7 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore | Full details

Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
30      cName = toTitleCase(cName);
31      try {
32        let checkCategoryExists = await categoryModel.findOne({ cName: cName });
33        if (checkCategoryExists) {
34          fs.unlink(filePath, (err) => {
```

Unsanitized input from *an uploaded file flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**categories.js** 7 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore | Full details

Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
147       for (const img of editImages) {
148         allEditImages.push(img.filename);
149       }
150       editData = { ...editData, pImages: allEditImages };
151       Product.deleteImages(pImages.split(","), "string");
```

Unsanitized input from *the HTTP request body flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**products.js** 14 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore | Full details

Path Traversal

SCORE

SNYK CODE | CWE-23

SCORE 828

```
171     let deleteProductObj = await productModel.findById(pId);
172     let deleteProduct = await productModel.findByIdAndDelete(pId);
173     if (deleteProduct) {
174       // Delete Image from uploads -> products folder
175       Product.deleteImages(deleteProductObj.pImages, "string");
```

Unsanitized input from *the HTTP request body flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**products.js** 17 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore Full details

H Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
88      let deleteCategory = await categoryModel.findByIdAndDelete(cId);
89      if (deleteCategory) {
90        // Delete Image from uploads -> categories folder
91        fs.unlink(filePath, (err) => {
```

Unsanitized input from *the HTTP request body flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**categories.js** 12 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore Full details

H Path Traversal

SNYK CODE | CWE-23

SCORE 828

```
47      let deleteImage = await customizeModel.findByIdAndDelete(id);
48      if (deleteImage) {
49        // Delete Image from uploads -> customizes folder
50        fs.unlink(filePath, (err) => {
```

Unsanitized input from *the HTTP request body flows* into *fs.unlink*, where it is used as a path. This may result in a Path Traversal vulnerability and allow an attacker to delete arbitrary files.

server/controller/**customize.js** 12 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore Full details

H Hardcoded Secret

SNYK CODE | CWE-547

SCORE 806

```
54        return res.json({ error });
55      } else {
56        // If Email & Number exists in Database then:
57        try {
58          password = bcrypt.hashSync(password, 10);
```

Hardcoded *value* is used as a *salt (in bcryptjs.hashSync)*. Generate the value with a cryptographically strong random number generator and do not hardcode it in source code.

server/controller/**auth.js** 2 steps in 1 file

Ignore Full details

H Hardcoded Secret

SNYK CODE | CWE-547

SCORE 756

```
111     });
112   } else {
113     const oldPassCheck = await bcrypt.compare(oldPassword, data.password);
```

```
114     if (oldPassCheck) {
115       newPassword = bcrypt.hashSync(newPassword, 10);
```

Hardcoded *value* is used as a *salt (in bcryptjs.hashSync)*. Generate the value with a cryptographically strong random number generator and do not hardcode it in source code.

server/controller/**users.js** 2 steps in 1 file

Ignore | Full details

H Hardcoded Secret

SNYK CODE | CWE-547

SCORE 753

```
1 module.exports = {
2   JWT_SECRET: "SecretKey",
```

Avoid hardcoding values that are meant to be secret. Found *a hardcoded string* used in *here*.

server/config/**keys.js** 2 steps in 1 file

Ignore | Full details

M Allocation of Resources Without Limits or Throttling

SNYK CODE | CWE-770

SCORE 659

```
13      console.log(err);
14    }
15  }
16
17  async postAddCategory(req, res) {
```

This *endpoint handler* performs *a file system operation* and does not use a rate-limiting mechanism. It may enable the attackers to perform Denial-of-service attacks. Consider using a rate-limiting middleware such as express-limit.

server/controller/**categories.js** 2 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore | Full details

M Allocation of Resources Without Limits or Throttling

SNYK CODE | CWE-770

SCORE 659

```
75      console.log(err);
76    }
77  }
78
79  async getDeleteCategory(req, res) {
```

This *endpoint handler* performs *a file system operation* and does not use a rate-limiting mechanism. It may enable the attackers to perform Denial-of-service attacks. Consider using a rate-limiting middleware such as express-limit.

server/controller/**categories.js** 2 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore | Full details

M Allocation of Resources Without Limits or Throttling

SNYK CODE | CWE-770

SCORE 659

```
34      console.log(err);
35    }
36  }
37
38  async deleteSlideImage(req, res) {
```

This *endpoint handler* performs *a file system operation* and does not use a rate-limiting mechanism. It may enable the attackers to perform Denial-of-service attacks. Consider using a rate-limiting middleware such as express-limit.

server/controller/**customize.js** 2 steps in 1 file

NEW Learn about this type of vulnerability and how to fix it

Ignore Full details

M Cross-Site Request Forgery (CSRF)

SNYK CODE | CWE-352

SCORE 553

```
17 */
18
19 const express = require("express");
20 const app = express();
```

CSRF protection is disabled for your *Express app*. This allows the attackers to execute requests on a user's behalf.

server/app.js 1 step in 1 file

Ignore Full details

M Information Exposure

SNYK CODE | CWE-200

SCORE 553

```
17 */
17 */
18
19 const express = require("express");
20 const app = express();
```

Disable X-Powered-By header for your *Express app* (consider using Helmet middleware), because it exposes information about the used framework to potential attackers.

server/app.js 1 step in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
58     Product.deleteImages(images, "file");
59     return res.json({ error: "All filled must be required" });
60   }
61   // Validate Name and description
62   else if (pName.length > 255 || pDescription.length > 3000) {
```

The type of this *object*, coming from *body* and the value of its *length* property can be controlled by the user. An attacker may craft the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/products.js 3 steps in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
58     Product.deleteImages(images, "file");
59     return res.json({ error: "All filled must be required" });
60   }
61   // Validate Name and description
62   else if (pName.length > 255 || pDescription.length > 3000) {
```

The type of this *object*, coming from *body* and the value of its *length* property can be controlled by the user. An attacker may craft the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/products.js 3 steps in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
122     ) {
123       return res.json({ error: "All filled must be required" });
124     }
125     // Validate Name and description
126     else if (pName.length > 255 || pDescription.length > 3000) {
```

The type of this *object*, coming from *body* and the value of its *length* property can be controlled by the user. An attacker may craft the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/**products.js** 3 steps in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
122     ) {
123       return res.json({ error: "All filled must be required" });
124     }
125     // Validate Name and description
126     else if (pName.length > 255 || pDescription.length > 3000) {
```

The type of this *object*, coming from *body* and the value of its *length* property can be controlled by the user. An attacker may craft the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/**products.js** 3 steps in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
37        cPassword: "Filed must not be empty",
38      };
39      return res.json({ error });
40    }
41    if (name.length < 3 || name.length > 25) {
```

The type of this *object*, coming from *body* and the value of its *length* property can be controlled by the user. An attacker may craft the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/**auth.js** 3 steps in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
37        cPassword: "Filed must not be empty",
38      };
39      return res.json({ error });
40    }
41    if (name.length < 3 || name.length > 25) {
```

The type of this *object*, coming from *body* and the value of its *length* property can be controlled by the user. An attacker may craft the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/**auth.js** 3 steps in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
43      return res.json({ error });
44    } else {
45      if (validateEmail(email)) {
46        name = toTitleCase(name);
47        if ((password.length > 255) | (password.length < 8)) {
```

The type of this *object*, coming from *body* and the value of its *length* property can be controlled by the user. An attacker may craft

the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/**auth.js** 3 steps in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
43      return res.json({ error });
44    } else {
45      if (validateEmail(email)) {
46        name = toTitleCase(name);
47        if ((password.length > 255) | (password.length < 8)) {
```

The type of this *object*, coming from *body* and the value of its *length* property can be controlled by the user. An attacker may craft the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/**auth.js** 3 steps in 1 file

Ignore Full details

L Improper Type Validation

SNYK CODE | CWE-1287

SCORE 425

```
147       for (const img of editImages) {
148         allEditImages.push(img.filename);
149       }
150       editData = { ...editData, pImages: allEditImages };
151       Product.deleteImages(pImages.split(","), "string");
```

The type of this *object*, coming from *body* and the value of its *split* property can be controlled by the user. An attacker may craft the properties of the object to crash the application or bypass its logic. Consider checking the type of the object.

server/controller/**products.js** 3 steps in 1 file

Ignore Full details

L Use of Hardcoded Credentials

SNYK CODE | CWE-798 + 1 MORE

SCORE 411

```
3 Scenario('register', async ({ I }) => {
4   const response = await I.sendPostRequest('/signup', {
5     name: 'Hashir',
6     email: 'hashir@gmail.com',
7     password: 'HASHIR12345',
```

Do not hardcode passwords in code. Found hardcoded password used in *password*.

tests/**registration_api_test.ts** 1 step in 1 file

Ignore Full details

L Use of Hardcoded Credentials

SNYK CODE | CWE-798 + 1 MORE

SCORE 411

```
4   const response = await I.sendPostRequest('/signup', {
5     name: 'Hashir',
6     email: 'hashir@gmail.com',
7     password: 'HASHIR12345',
8     cPassword: 'HASHIR12345',
```

Do not hardcode passwords in code. Found hardcoded password used in *cPassword*.

tests/**registration_api_test.ts** 1 step in 1 file

Ignore Full details