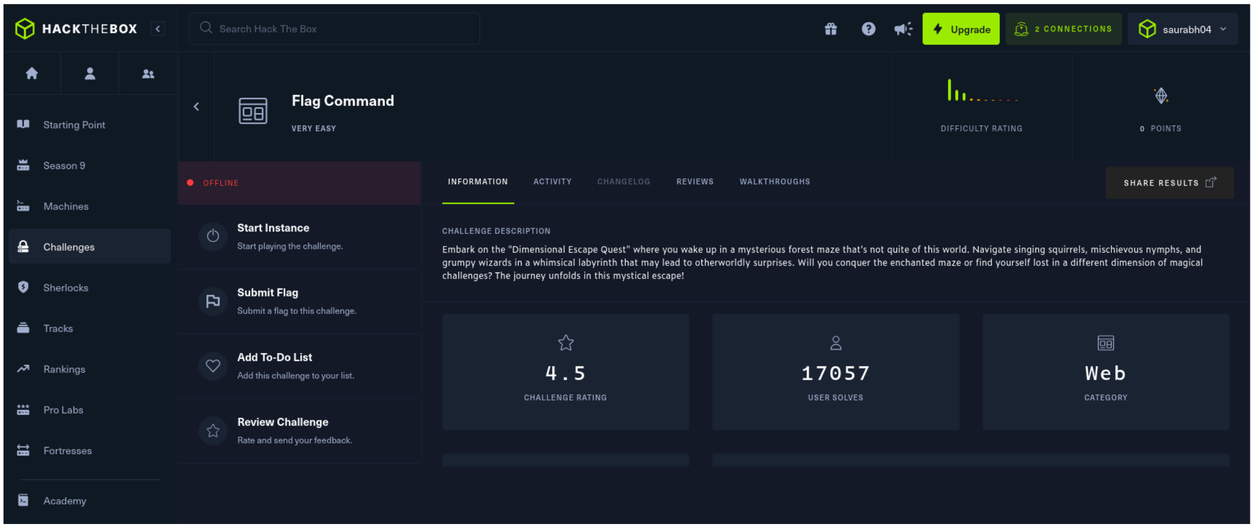
flag command

Let’s go and solve our first challenge from HTB web challenge category

Name: Flag command

Difficulty: very easy



Flag Command

VERY EASY

CHALLENGE DESCRIPTION

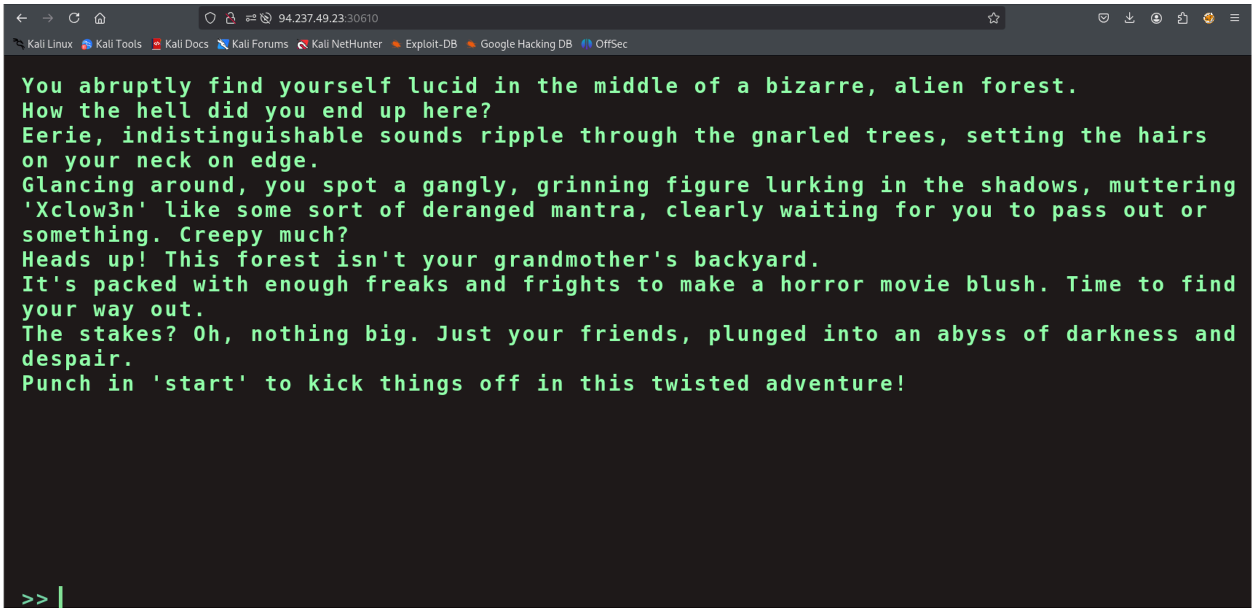
Embark on the "Dimensional Escape Quest" where you wake up in a mysterious forest maze that's not quite of this world. Navigate singing squirrels, mischievous nymphs, and grumpy wizards in a whimsical labyrinth that may lead to otherworldly surprises. Will you conquer the enchanted maze or find yourself lost in a different dimension of magical challenges? The journey unfolds in this mystical escape!

4.5 CHALLENGE RATING

17057 USER SOLVES

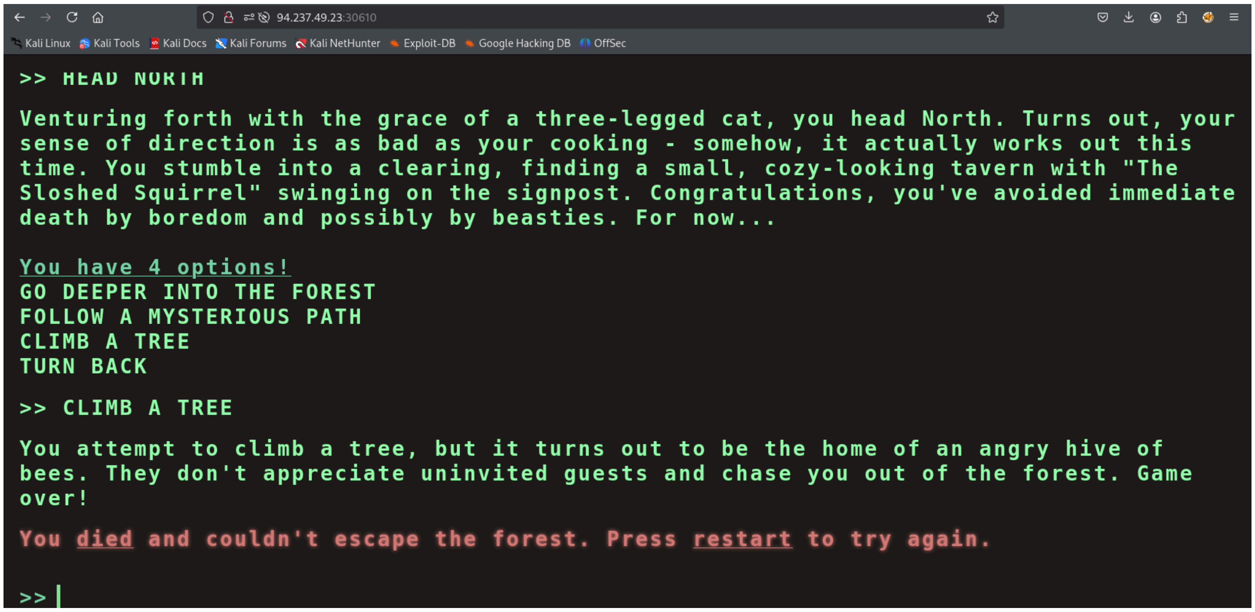
Web CATEGORY

Now Let’s go ahead and start the challenge and access the website



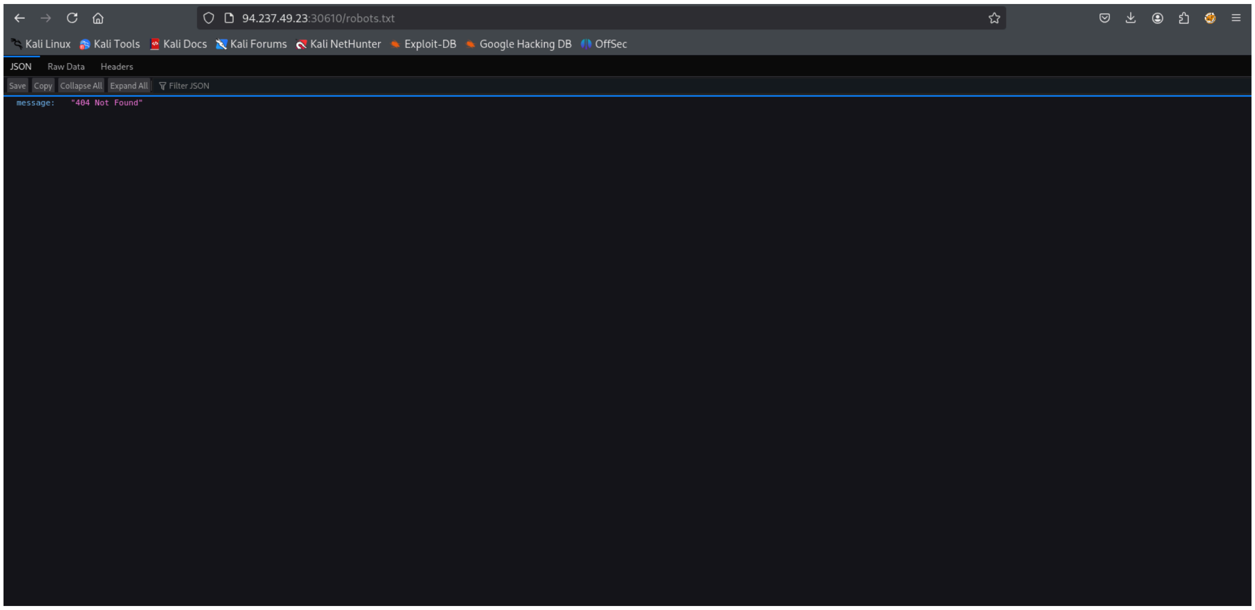
```
You abruptly find yourself lucid in the middle of a bizarre, alien forest.
How the hell did you end up here?
Eerie, indistinguishable sounds ripple through the gnarled trees, setting the hairs
on your neck on edge.
Glancing around, you spot a gangly, grinning figure lurking in the shadows, muttering
'Xclow3n' like some sort of deranged mantra, clearly waiting for you to pass out or
something. Creepy much?
Heads up! This forest isn't your grandmother's backyard.
It's packed with enough freaks and frights to make a horror movie blush. Time to find
your way out.
The stakes? Oh, nothing big. Just your friends, plunged into an abyss of darkness and
despair.
Punch in 'start' to kick things off in this twisted adventure!



>>
```

It looks like a game, and it seems interesting. Let's play the game to find out more.



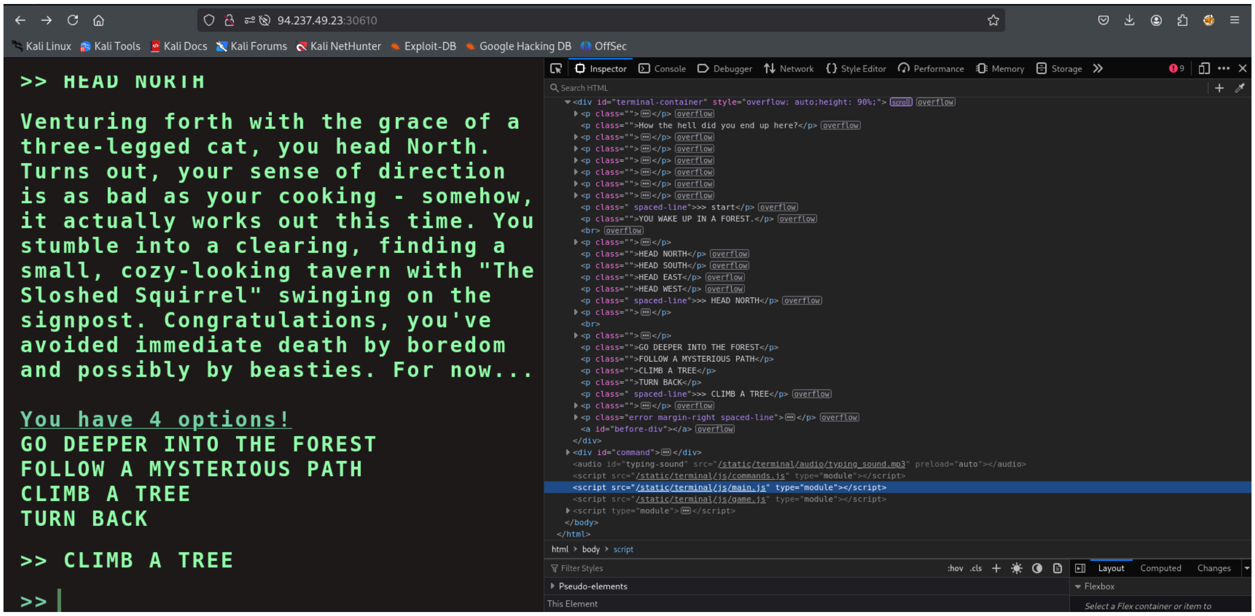
```
>> HEAD NORTH

Venturing forth with the grace of a three-legged cat, you head North. Turns out, your
sense of direction is as bad as your cooking - somehow, it actually works out this
time. You stumble into a clearing, finding a small, cozy-looking tavern with "The
Sloshed Squirrel" swinging on the signpost. Congratulations, you've avoided immediate
death by boredom and possibly by beasties. For now...

You have 4 options!
GO DEEPER INTO THE FOREST
FOLLOW A MYSTERIOUS PATH
CLIMB A TREE
TURN BACK

>> CLIMB A TREE

You attempt to climb a tree, but it turns out to be the home of an angry hive of
bees. They don't appreciate uninvited guests and chase you out of the forest. Game
over!

You died and couldn't escape the forest. Press restart to try again.

>> 
```

I tried each and every option but every time it says you died and game over, so now, as is typical in web challenges, we'll check `robots.txt`.



```
message:    "404 Not Found"
```

Since the request to `robots.txt` returned a **404 Not Found** error, we'll now move on to inspecting the application's source code and network traffic using the browser's **Developer Tools**.

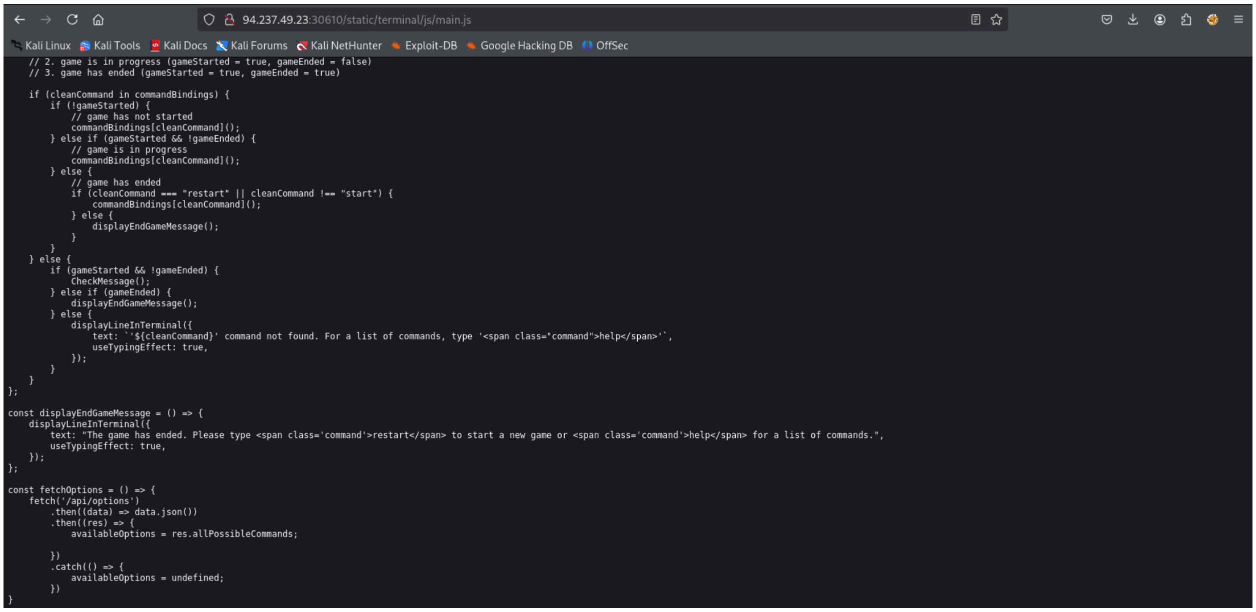


```
>> HEAD NORTH

Venturing forth with the grace of a
three-legged cat, you head North.
Turns out, your sense of direction
is as bad as your cooking - somehow,
it actually works out this time. You
stumble into a clearing, finding a
small, cozy-looking tavern with "The
Sloshed Squirrel" swinging on the
signpost. Congratulations, you've
avoided immediate death by boredom
and possibly by beasties. For now...

You have 4 options!
GO DEEPER INTO THE FOREST
FOLLOW A MYSTERIOUS PATH
CLIMB A TREE
TURN BACK

>> CLIMB A TREE

>>
```

```
<div id="terminal-container" style="overflow: auto;height: 90%;"> scroll overflow
  <p class="">…</p> overflow
  <p class="">How the hell did you end up here?</p> overflow
  <p class="">…</p> overflow
  <p class="">…</p> overflow
  <p class="">…</p> overflow
  <p class="">…</p> overflow
  <p class="">…</p> overflow
  <p class="">…</p> overflow
  <p class=" spaced-line">>> start</p> overflow
  <p class="">YOU WAKE UP IN A FOREST.</p> overflow
  <br> overflow
  <p class="">…</p>
  <p class="">HEAD NORTH</p> overflow
  <p class="">HEAD SOUTH</p> overflow
  <p class="">HEAD EAST</p> overflow
  <p class="">HEAD WEST</p> overflow
  <p class=" spaced-line">>> HEAD NORTH</p> overflow
  <p class="">…</p>
  <br>
  <p class="">…</p>
  <p class="">GO DEEPER INTO THE FOREST</p>
  <p class="">FOLLOW A MYSTERIOUS PATH</p>
  <p class="">CLIMB A TREE</p>
  <p class="">TURN BACK</p>
  <p class=" spaced-line">>> CLIMB A TREE</p> overflow
  <p class="">…</p> overflow
  <p class="error margin-right spaced-line">…</p> overflow
  <a id="before-div"></a> overflow
</div>
<div id="command">…</div>
<audio id="typing-sound" src="/static/terminal/audio/typing_sound.mp3" preload="auto"></audio>
<script src="/static/terminal/js/commands.js" type="module"></script>
<script src="/static/terminal/js/main.js" type="module"></script>
<script src="/static/terminal/js/game.js" type="module"></script>
<script type="module">…</script>
</body>
</html>
```

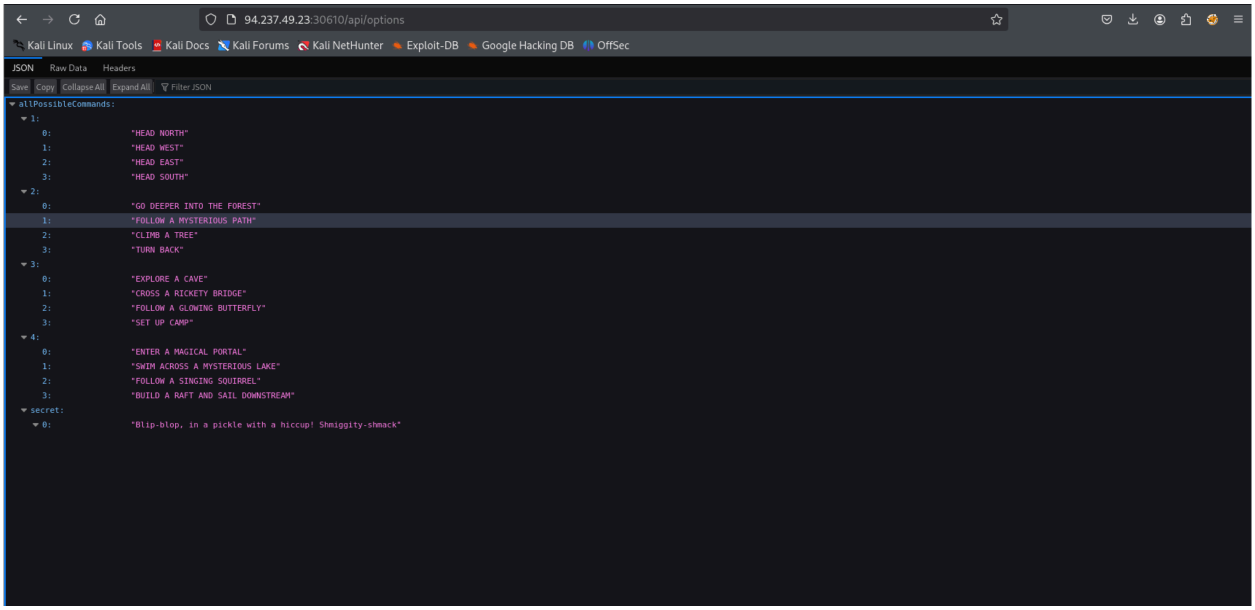
Nice! We found three interesting JS files. Let's review them one by one

- /static/terminal/js/commands.js
- /static/terminal/js/main.js
- /static/terminal/js/game.js



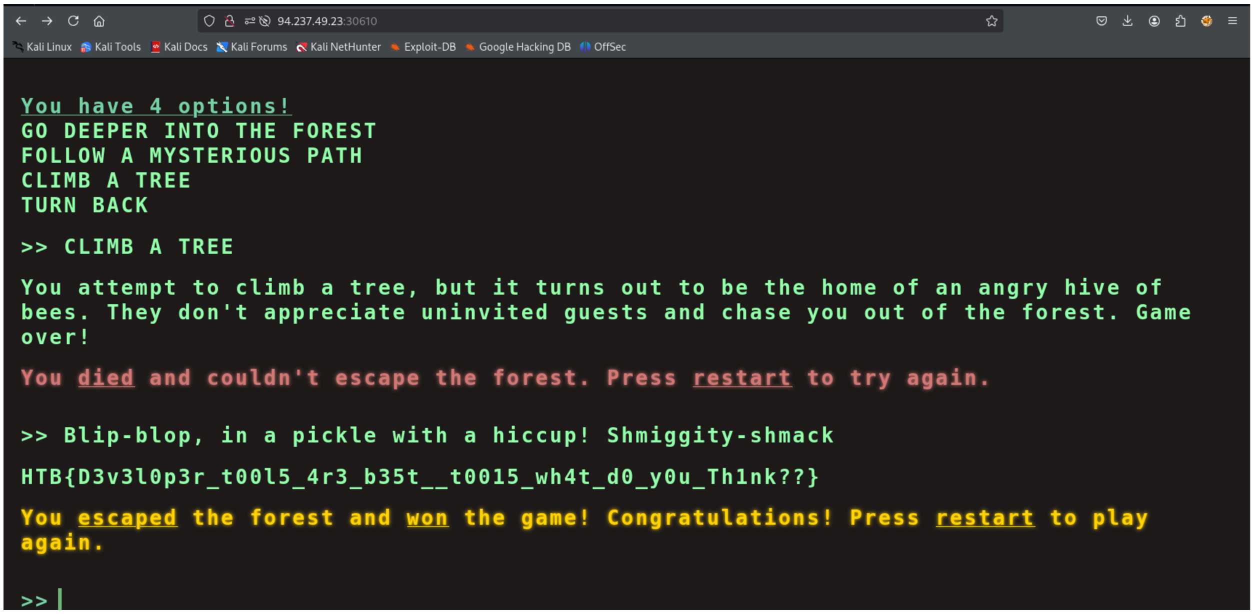
```
    // 2. game is in progress (gameStarted = true, gameEnded = false)
    // 3. game has ended (gameStarted = true, gameEnded = true)

    if (cleanCommand in commandBindings) {
        if (!gameStarted) {
            // game has not started
            commandBindings[cleanCommand]();
        } else if (gameStarted && !gameEnded) {
            // game is in progress
            commandBindings[cleanCommand]();
        } else {
            // game has ended
            if (cleanCommand === "restart" || cleanCommand !== "start") {
                commandBindings[cleanCommand]();
            } else {
                displayEndGameMessage();
            }
        }
    } else {
        if (gameStarted && !gameEnded) {
            CheckMessage();
        } else if (gameEnded) {
            displayEndGameMessage();
        } else {
            displayLineInTerminal({
                text: `'${cleanCommand}' command not found. For a list of commands, type '<span class="command">help</span>'`,
                useTypingEffect: true,
            });
        }
    }
};

const displayEndGameMessage = () => {
    displayLineInTerminal({
        text: "The game has ended. Please type <span class='command'>restart</span> to start a new game or <span class='command'>help</span> for a list of commands.",
        useTypingEffect: true,
    });
};

const fetchOptions = () => {
    fetch('/api/options')
        .then((data) => data.json())
        .then((res) => {
            availableOptions = res.allPossibleCommands;

        })
        .catch(() => {
            availableOptions = undefined;
        })
}
```

In main.js, we found a very interesting API endpoint: /api/options. Let's access this endpoint to see what data it returns.



```
allPossibleCommands:
  1:
    0:            "HEAD NORTH"
    1:            "HEAD WEST"
    2:            "HEAD EAST"
    3:            "HEAD SOUTH"
  2:
    0:            "GO DEEPER INTO THE FOREST"
    1:            "FOLLOW A MYSTERIOUS PATH"
    2:            "CLIMB A TREE"
    3:            "TURN BACK"
  3:
    0:            "EXPLORE A CAVE"
    1:            "CROSS A RICKETY BRIDGE"
    2:            "FOLLOW A GLOWING BUTTERFLY"
    3:            "SET UP CAMP"
  4:
    0:            "ENTER A MAGICAL PORTAL"
    1:            "SWIM ACROSS A MYSTERIOUS LAKE"
    2:            "FOLLOW A SINGING SQUIRREL"
    3:            "BUILD A RAFT AND SAIL DOWNSTREAM"
  secret:
    0:            "Blip-blop, in a pickle with a hiccup! Shmiggity-shmack"
```

It is just a JSON response, but there is one particularly different option given the name 'secret.' Now, let's see what happens when we use this secret value in the game.



```
You have 4 options!
GO DEEPER INTO THE FOREST
FOLLOW A MYSTERIOUS PATH
CLIMB A TREE
TURN BACK

>> CLIMB A TREE

You attempt to climb a tree, but it turns out to be the home of an angry hive of
bees. They don't appreciate uninvited guests and chase you out of the forest. Game
over!

You died and couldn't escape the forest. Press restart to try again.

>> Blip-blop, in a pickle with a hiccup! Shmiggity-shmack

HTB{D3v3l0p3r_t00l5_4r3_b35t__t0015_wh4t_d0_y0u_Th1nk??}

You escaped the forest and won the game! Congratulations! Press restart to play
again.

>> 
```

Boom! we finally got the flag .....

it’s pretty easy challenge where we just have to use devloperoption

It was a pretty easy challenge where we simply had to use the browser's **Developer Tools (F12)**. This exercise reinforced a critical lesson in web exploitation: always inspect the client-side logic and network traffic before trying to play the application as intended. The "secret" was essentially a **hardcoded developer backdoor** exposed via the accessible `/api/options` endpoint.