DOCUMENTATION

How To Implement Server Side Validation

(Drag and Drop CAPTCHA)

Introduction

This document outlines how to integrate CAPTCHA validation on the server side to enhance security against automated bots. The provided CAPTCHA system uses client-side JavaScript; however, for robust security, a server-side verification step is recommended.

Prerequisites

- A web server running Node.js, Python (Flask/Django), PHP, or another backend framework.
- A method to receive CAPTCHA responses from the client.
- A way to verify user-submitted responses against the expected correct answer.

Steps to Implement Server-Side CAPTCHA Validation

1. Modify the Frontend to Send CAPTCHA Data

In the completeCaptcha() function, modify the request to send the CAPTCHA result to the server:

```
function completeCaptcha() {
    clearTimeout(captchaTimeout);
    const modal = document.getElementById("captchaModal");
    modal.style.display = "none";

    const captchaType = document.getElementById("captcha-
content").getAttribute("data-type");
    const selectedAnswer =
document.getElementById("draggable").id;

    fetch("/verify-captcha", {
        method: "POST",
        headers: {
```

```
            "Content-Type": "application/json"
        },
        body: JSON.stringify({
            captchaType: captchaType,
            answer: selectedAnswer
        })
    })
    .then(response => response.json())
    .then(data => {
        if (data.success) {
            window.location.href =
'https://mathewsin.github.io/CaptchaTester/';
        } else {
            Swal.fire({
                title: "Incorrect!",
                text: "Try dragging the correct image!",
                icon: "error",
                confirmButtonText: "OK"
            }).then(() => {
                reloadCaptcha();
            });
        }
    })
    .catch(error => console.error("Error verifying CAPTCHA:",
error));
}
```

2. Create a Server-Side Verification Endpoint

Implement a server-side endpoint to validate the CAPTCHA response.

Here are some examples for 2 types of programming language:

- Python

```
from flask import Flask, request, jsonify

app = Flask(__name__)
  correct_answers = {
     "fruit": "carrot",
     "house": "rabbitHouse",
     "animal": "turtle"
       }

@app.route('/verify-captcha', methods=['POST'])
     def verify_captcha():
```

```
        data = request.json
        captcha_type = data.get("captchaType")
        answer = data.get("answer")

        if correct_answers.get(captcha_type) == answer:
            return jsonify({"success": True})
        return jsonify({"success": False})

        if __name__ == '__main__':
            app.run(debug=True)
```

- PHP

```
<?php
header("Content-Type: application/json");
$data = json_decode(file_get_contents("php://input"), true);

$correctAnswers = [
    "fruit" => "carrot",
    "house" => "rabbitHouse",
    "animal" => "turtle"
];

$captchaType = $data["captchaType"];
$answer = $data["answer"];

if ($correctAnswers[$captchaType] === $answer) {
    echo json_encode(["success" => true]);
} else {
    echo json_encode(["success" => false]);
}
?>
```

Conclusion

By implementing this server-side verification, you ensure that CAPTCHA validation is not bypassed through frontend manipulation. This additional layer of security helps protect your authentication system against bots and automated attacks.