Project NoSpam

Mögliche Prüfungen:

- Brute Force (Login mit Username in x Minuten, Login von IP in x Minuten, Login von gleichem User Agent in x Minuten)
- Wortfilter in (bestimmten) Feldern
- RegEx-Filter in (bestimmten) Feldern
- IP ASN Check (ASN auf Blacklist oder verdächtig? Z.B. Provider X ASN)
- IP Land Check (ist das Land der IP mit dem Land der Website vergleichbar?)
- Registrieren der Tasteneingaben/Mausbewegungen und Auswerten aufgrund der Regelmässigkeit bzw. eher Unregelmässigkeit
- Trash-E-Mail-Adresse-Check
- Sprache erkennen

Datenschutz:

- Input[type='password'] Wert direkt im Browser mit * ersetzen
- Bestätigung, dass Daten von (externer) Seite bearbeitet werden dürfen durch Klicken des Buttons/der Checkbox
- Daten in System werden automatisch nach x Tagen entfernt, als Spam eingestufte Nachrichten werden als Hash gespeichert

Features:

- Möglichkeit, um ein als Spam erkannte Nachricht trotzdem absenden zu können, dafür nur einmal innerhalb X Minuten (pro Seite/Token), ggf. mit 30 Sekunden Wartezeit vor dem Senden

Ablauf:

- User lädt Formular auf Website X.
- Eine spezielle Checkbox (oder Button) wird unterhalb des Formulars aber oberhalb des Senden-Buttons angezeigt.
- Sobald der User das Formular ausgefüllt hat, klickt er auf die Checkbox (oder auf den Button).
- Beim Betätigen der Checkbox (oder Button) werden alle Formularfelder gesammelt. Die Daten der einzelnen Felder sowie der Feldtyp werden in einem JSON-Objekt zusammengefasst.
- Die gesammelten Informationen werden zusammen mit einem Public Key an eine API übermittelt.
- Die API scannt die Daten und versucht einen SPAM-Score zu ermitteln. Dazu werden verschiedene Informationen ausgewertet und bewertet. Alle Informationen des Formulars werden gespeichert.
- Wenn die Überprüfung abgeschlossen ist, wird ein Token erstellt und der Code des Tokens wird zusammen mit dem Ergebnis (Spam/Kein Spam) an den Browser zurückgegeben

- Die Checkbox wird dementsprechend markiert bzw. dem User wird ein allfälliger Spam-Verdacht angezeigt. Der Token wird im Formular gespeichert
- Der User sendet das Formular ab
- Der Backend-Teil überprüft anhand einer API mithilfe des Tokens, ob die Formulareingaben korrekt sind und so verwendet werden dürfen. Falls die Daten ungültig sind, wird entweder der Benutzer darauf aufmerksam gemacht oder der Prozess wird so oder so verworfen.

- Falls der User nach erfolgreicher Überprüfung das Formular erneut verändert, wird der Token ungültig gesetzt

- Über eine zusätzliche API kann der Entwickler mitteilen, ob z.B. ein Login-Vorgang mit dem Token X gültig war oder nicht und kann so die Funktion des Brute Force ermöglichen.