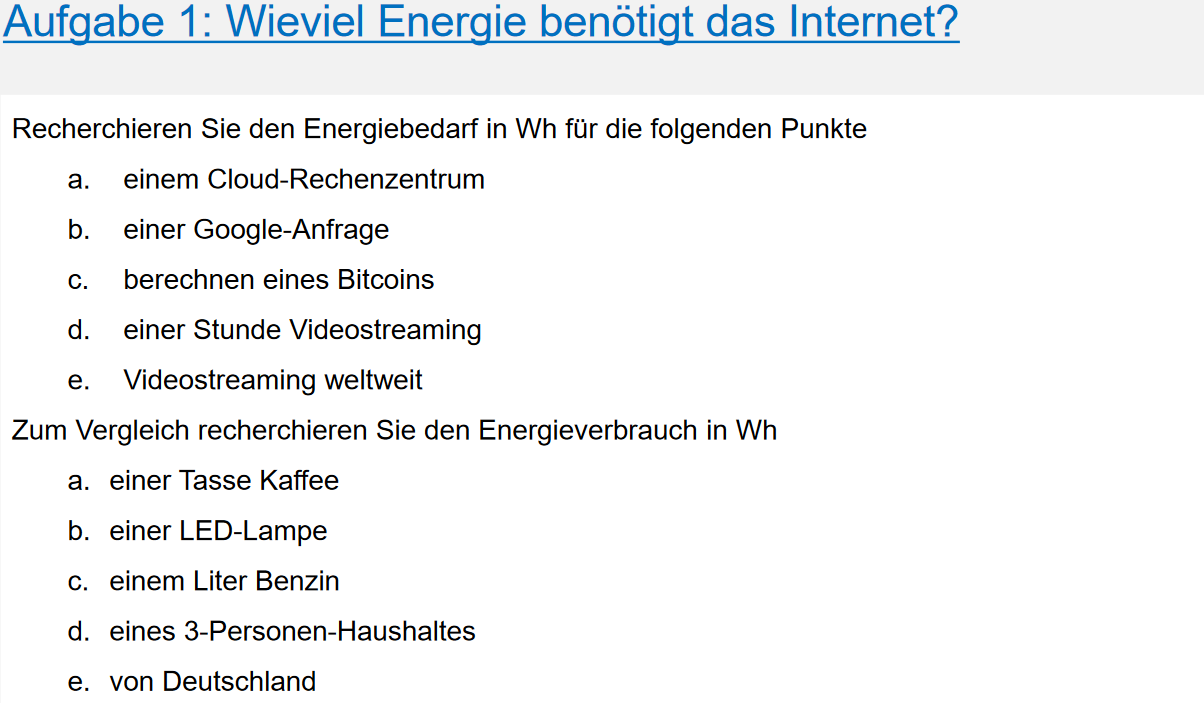
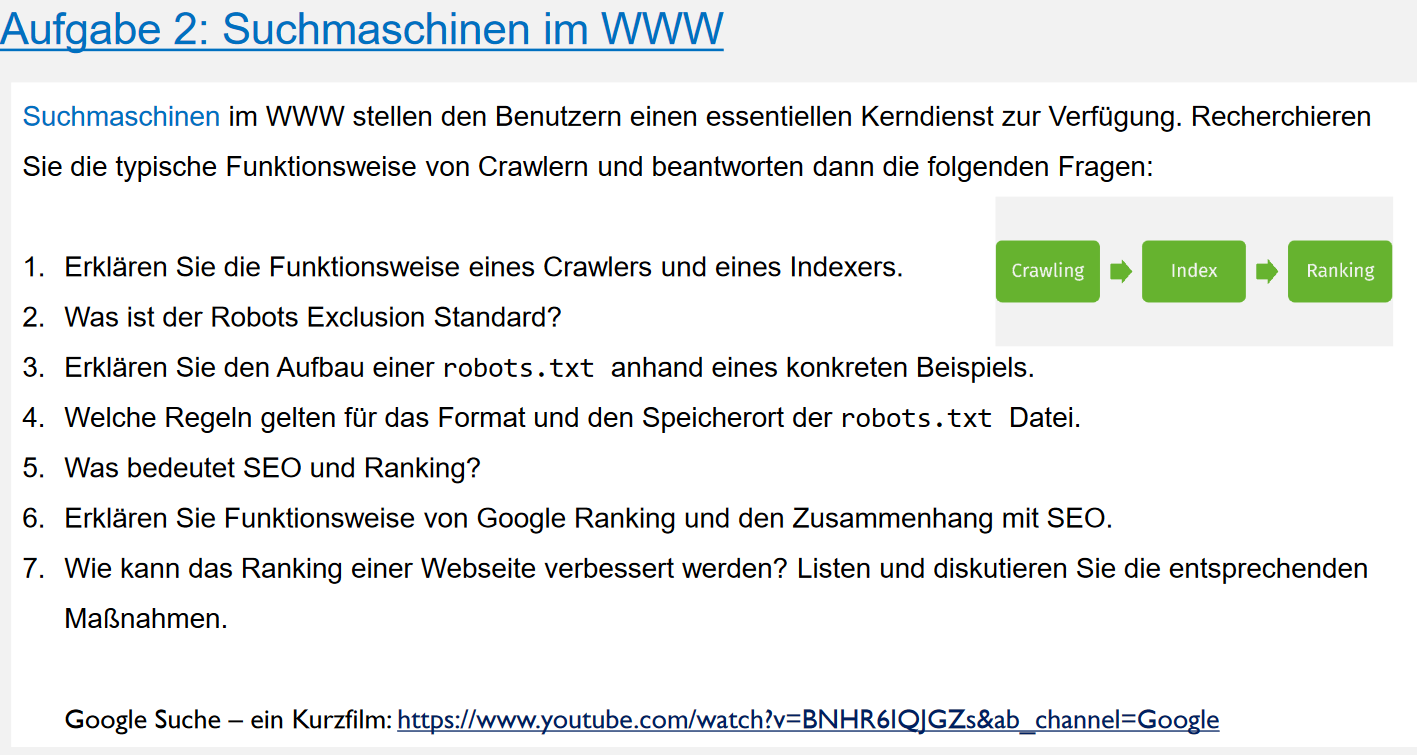
****

**Energiebedarf in Wh für die folgenden Punkte:**

1. einem Cloud-Rechenzentrum: 18 Mrd. kWh/Jahr
2. einer Google-Anfrage: 0,003 kWh
3. Berechnung eines Bitcoins: 8760 kWh
4. einer Stunde Videostreaming: 0,1-1 Wh

**Energieverbrauch in Wh:**

1. einer Tasse Kaffee: 0.03 kWh
2. einer LED-Lampe: 6-9 Wh
3. eines 3-Personen-Haushaltes: 2600-3600 kWh/Jahr
4. von Deutschland: 517 TWh (2023)

****

1. **Erklären Sie die Funktionsweise eines Crawlers und eines Indexers.**

**Crawler (auch Spider oder Robot):**

* Ein Crawler ist ein automatisiertes Programm, das das Internet durchsucht, um Webseiten zu finden und zu durchsuchen.
* Es beginnt normalerweise mit einer Liste bekannter URLs (Seed-URLs) und folgt dann den Links auf diesen Seiten, um weitere Seiten zu entdecken.
* Der Crawler ruft die Webseiten ab und speichert sie vorübergehend, um ihren Inhalt zu analysieren und Links zu anderen Seiten zu extrahieren.
* Während des Crawlings werden verschiedene Aspekte der Webseite berücksichtigt, wie z.B. die Verlinkung zu anderen Seiten, die Aktualität der Inhalte und die Meta-Informationen.
* Der Zweck des Crawlers besteht darin, eine möglichst umfassende Sammlung von Webseiten zu erstellen, die dann vom Indexer verarbeitet werden.

**Indexer:**

* Der Indexer ist für die Organisation und Strukturierung der gesammelten Informationen zuständig, um sie für die Suchmaschine zugänglich zu machen.
* Er verarbeitet die vom Crawler gesammelten Webseiten und extrahiert relevante Informationen wie Schlüsselwörter, Meta-Tags, Titel und Textinhalte.
* Die extrahierten Informationen werden dann in einer Datenbank oder einem Index gespeichert, der für schnelle Suchanfragen optimiert ist.
* Der Indexer wendet oft komplexe Algorithmen an, um die Relevanz der gespeicherten Informationen zu bewerten und sie entsprechend zu gewichten.
* Die organisierten Informationen ermöglichen es der Suchmaschine, Suchanfragen effizient zu bearbeiten und relevante Ergebnisse für die Benutzer bereitzustellen.

1. **Was ist der Robots Exclusion Standard?**

Der Robots Exclusion Standard, auch bekannt als robots.txt, ist ein Textdateiformat, das von Website-Betreibern verwendet wird, um Suchmaschinen-Crawlern Anweisungen zu geben, welche Bereiche ihrer Website gecrawlt werden dürfen und welche nicht. Dieser Standard ermöglicht es Website-Betreibern, die Indexierung bestimmter Teile ihrer Website zu steuern und so beispielsweise private oder unerwünschte Inhalte

1. **Erklären Sie den Aufbau einer robots.txt anhand eines konkreten Beispiels.**

Ein Bild, das Text, Quittung, Schrift, Algebra enthält.

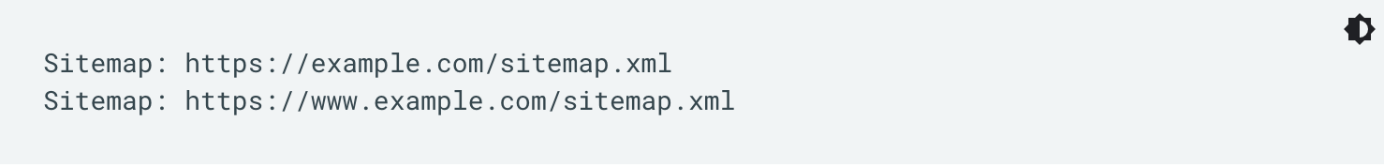
Automatisch generierte Beschreibung

Eine robots.txt-Datei besteht aus einer oder mehreren Gruppen (Regelsatz).

Jede Gruppe besteht aus mehreren Regeln (auch als Anweisungen bezeichnet), und zwar mit einer Regel pro Zeile. Jede Gruppe beginnt mit einer User-agent Zeile, die angibt, für welchen Crawler die Gruppe gelten soll (Ziel).

Eine Gruppe enthält die Informationen:

* 1. Wofür die Gruppe gilt (User-Agent)
  2. Auf welche Verzeichnisse oder Dateien der User-Agent zugreifen darf
  3. Auf welche Verzeichnisse oder Dateien der User-Agent nicht zugreifen darf
* user-agent (erforderlich, einer oder mehrere pro Gruppe): Die Regel gibt den Namen des automatischen Clients (Suchmaschinen-Crawler) an, für den die Regel gilt. Das ist die erste Zeile für jede Regelgruppe. Wenn ein Sternchen (\*) verwendet wird, gilt die Regel für alle Crawler. Ausgenommen davon sind die verschiedenen AdsBot-Crawler, die ausdrücklich benannt werden müssen.
* Disallow (pro Regel muss es mindestens einen disallow- oder allow-Eintrag geben): Gibt das Verzeichnis oder die Seite unter der Stammdomain an, die der User-Agent nicht crawlen soll. Wenn sich die Regel auf eine Seite bezieht, muss der vollständige Seitenname verwendet werden, so wie er im Browser angezeigt wird. Er muss mit dem Zeichen / beginnen und muss, sofern er sich auf ein Verzeichnis bezieht, mit dem Zeichen / enden.
* allow: (pro Regel muss es mindestens einen disallow- oder allow-Eintrag geben): Gibt das Verzeichnis oder die Seite unter der Stammdomain an, die vom zuvor genannten User-Agent gecrawlt werden darf. Eine solche Regel setzt etwaige disallow-Regeln außer Kraft und kann verwendet werden, um das Crawling eines Unterverzeichnisses oder einer Seite in einem eigentlich ausgeschlossenen Verzeichnis zu erlauben. Wenn du eine einzelne Seite zulassen möchtest, gib den vollständigen Seitennamen an, wie er im Browser angezeigt wird. Er muss mit dem Zeichen / beginnen und muss, sofern er sich auf ein Verzeichnis bezieht, mit dem Zeichen / enden.
* Sitemap (optional): Gibt den Speicherort einer Sitemap für diese Website an. Die Sitemap-URL muss eine vollständig qualifizierte URL sein. Google geht nicht von Varianten aus und prüft diese auch nicht. Sitemaps sind eine praktische Möglichkeit, Google zu zeigen, welche Inhalte gecrawlt werden sollen, im Unterschied zu den „Allow“-/„Disallow“-Anweisungen, mit denen dem Googlebot mitgeteilt wird, welche Inhalte gecrawlt werden dürfen oder eben nicht gecrawlt werden dürfen.



1. Welche Regeln gelten für das Format und den Speicherort der robots.txt Datei.

* Speicherort: Die robots.txt-Datei muss im Stammverzeichnis der Website platziert werden. Das bedeutet, dass sie unter der URL http://www.beispiel.com/robots.txt erreichbar sein sollte.
* Format: Die robots.txt-Datei verwendet ein einfaches Textformat. Jede Anweisung besteht aus einem User-Agent (der Roboter, für den die Anweisungen gelten) und einer oder mehreren Anweisungen, die angeben, welche Seiten für diesen Roboter zugänglich sind.
* User-Agent: Der User-Agent ist der Name des Robots, für den die nachfolgenden Anweisungen gelten. Zum Beispiel "Googlebot" für den Google-Crawler. Es können auch Platzhalter wie "\*" verwendet werden, um Anweisungen für alle Robots zu geben, oder spezifische User-Agents, um nur bestimmte Roboter anzusprechen.
* Anweisungen: Die Hauptanweisungen sind "Disallow" und "Allow". "Disallow" gibt an, welche Seiten für den entsprechenden User-Agent gesperrt sind, während "Allow" erlaubt, bestimmte Seiten zu crawlen, die möglicherweise von einem allgemeinen Sperrbefehl betroffen sind.
* Kommentare: Kommentare können in der robots.txt-Datei mit dem Zeichen "#" eingefügt werden. Alles, was nach "#" steht, wird von den Robots ignoriert.

1. Was bedeutet SEO und Ranking?

SEO (engl: Search Engine Optimization) oder "Suchmaschinenoptimierung" auf Deutsch bezieht sich auf den Prozess der Optimierung von Websites, um ihre Sichtbarkeit in den organischen Suchergebnissen von Suchmaschinen zu verbessern. Das Hauptziel besteht darin, die Platzierung einer Website in den Suchergebnissen für bestimmte Suchanfragen zu verbessern

Ranking bezieht sich auf die Position einer Website in den Suchmaschinen-ergebnissen für eine bestimmte Suchanfrage.

1. Erklären Sie Funktionsweise von Google Ranking und den Zusammenhang mit SEO.

Wenn eine Website beispielsweise auf der ersten Seite der Suchergebnisse für das Keyword "beste Restaurants in Berlin" erscheint, hat sie ein hohes Ranking für dieses Keyword. Das Ranking einer Website wird durch eine Vielzahl von Faktoren beeinflusst, darunter die Relevanz der Website für die Suchanfrage, die Qualität ihres Inhalts, die Autorität der Website und die Anzahl der eingehenden Links von anderen vertrauenswürdigen Websites. Die Verbesserung des Rankings ist ein zentrales Ziel der SEO-Bemühungen.

Der Zusammenhang zwischen SEO und dem Google-Ranking besteht darin, dass SEO-Praktiken darauf abzielen, diese und andere Faktoren zu optimieren, um die Sichtbarkeit und das Ranking einer Website in den Suchergebnissen zu verbessern. Durch die Implementierung von SEO-Strategien können Website-Betreiber ihre Chancen erhöhen, auf den oberen Positionen in den Suchergebnissen zu erscheinen, was wiederum zu mehr organischen Besuchern und potenziell zu mehr Conversions führen kann.

1. Wie kann das Ranking einer Webseite verbessert werden? Listen und diskutieren Sie die entsprechenden Maßnahmen.

* Keyword-Recherche und -Optimierung
* Qualitativ hochwertiger und relevanter Inhalt
* On-Page-Optimierung
* Backlink-Building
* Optimierung für lokale SEO

A screenshot of a computer

Description automatically generated

3.1.

**Browser Fingerprinting:**

Browser Fingerprinting ist eine Technik, bei der Informationen über den Browser und das Gerät eines Benutzers gesammelt werden, um ein eindeutiges Profil zu erstellen. Dabei werden Merkmale wie die Browserversion, installierte Plugins, Bildschirmauflösung und Betriebssystem verwendet. Diese Informationen können verwendet werden, um Benutzer zu identifizieren und ihr Verhalten im Internet zu verfolgen, selbst wenn sie Cookies deaktiviert haben.

**Third-Party Cookies:**

Third-Party Cookies sind kleine Textdateien, die von Websites erstellt und von Drittanbietern platziert werden, um das Nutzerverhalten über verschiedene Websites hinweg zu verfolgen. Diese Cookies werden von anderen Domains als der besuchten Website gesetzt und ermöglichen es Drittanbietern, Benutzerprofile zu erstellen und personalisierte Werbung anzuzeigen.

**Google Analytics:**

Google Analytics ist ein Webanalyse-Tool von Google, das Website-Betreibern Einblicke in das Besucherverhalten ihrer Website bietet. Es ermöglicht die Verfolgung von Seitenaufrufen, Verweisen, demografischen Daten der Besucher und vielem mehr. Website-Betreiber können diese Daten nutzen, um ihre Website zu optimieren und ihre Marketingstrategien zu verbessern.

**Fake News und Trusted Web:**

Fake News sind falsche oder irreführende Informationen, die absichtlich verbreitet werden, um die öffentliche Meinung zu manipulieren oder bestimmte Interessen zu fördern. Sie werden oft über soziale Medien, Websites oder andere Online-Plattformen verbreitet und können ernsthafte Auswirkungen auf die Gesellschaft haben, indem sie Desinformationen verbreiten und das Vertrauen in die Medien untergraben.

Trusted Web bezieht sich auf den Aufbau von Vertrauen und Glaubwürdigkeit im Internet. Dies umfasst Maßnahmen wie die Authentifizierung von Websites, die Bereitstellung von sicheren Verbindungen durch HTTPS und den Schutz der Privatsphäre der Benutzer durch transparente Datenschutzrichtlinien. Ein vertrauenswürdiges Web fördert ein positives Online-Erlebnis und trägt zur Sicherheit und Integrität des Internets bei.

**3.2 Maßnahmen um das WWW sicherer und vertrauenswürdiger gestalten:**

Verschlüsselung verwenden**:**

Websites sollten HTTPS anstelle von HTTP verwenden, um eine sichere Verbindung zwischen dem Benutzer und der Website herzustellen. Dies schützt die übertragenen Daten vor Abhören und Manipulation.

Authentifizierung implementieren**:**

Websites können Mechanismen zur Benutzer- und Datenauthentifizierung einführen, um sicherzustellen, dass nur autorisierte Benutzer auf sensible Daten zugreifen können.

Aktualisierung und Patching**:**

Alle Softwarekomponenten, einschließlich Betriebssysteme, Webserver, Frameworks und Bibliotheken, sollten regelmäßig auf Sicherheitsupdates und Patches überprüft und aktualisiert werden.

**3.3 Was sind XML-Namespaces. Erkläre im Zusammenhang von folgenden Begriffen an Beispielen:**

**XML-Namespaces**

XML-Namespaces sind eine Möglichkeit, Elemente und Attribute in XML-Dokumenten zu gruppieren, um Konflikte bei der Verwendung von gleichen Namen in verschiedenen Kontexten zu vermeiden. Sie ermöglichen es, Elemente und Attribute eindeutig zu identifizieren, auch wenn sie denselben Namen haben.

**XML-Prefix:**

Ein XML-Prefix ist eine Abkürzung oder ein Kürzel, das verwendet wird, um einen XML-Namespace zu identifizieren. Es wird normalerweise durch einen Doppelpunkt (:) vom eigentlichen Elementnamen getrennt. Zum Beispiel, wenn wir das Element `book` aus dem Namespace `http://example.com/books` verwenden möchten, könnten wir einen Prefix wie `b` definieren und das Element dann als `b:book` darstellen.

**XML-Namespace deklarieren:**

Um einen XML-Namespace in einem XML-Dokument zu deklarieren, verwenden wir das Attribut `xmlns` (XML Namespace) in einem Element. Zum Beispiel:

<root xmlns:ex="http://example.com">

<ex:element1>...</ex:element1>

<ex:element2>...</ex:element2>

</root>

Hier wird der XML-Namespace `http://example.com` mit dem Präfix `ex` deklariert. Alle Elemente, die mit diesem Präfix beginnen, gehören zu diesem Namespace.

**Default Namespace:**

Der Default Namespace ist ein Namespace, der für alle Elemente im XML-Dokument gilt, die nicht explizit einem anderen Namespace zugeordnet sind. Er wird ohne einen Prefix deklariert. Zum Beispiel:

<root xmlns="http://example.com">

<element1>...</element1> <!-- Element im Default Namespace -->

<ex:element2 xmlns:ex="http://another-example.com">...</ex:element2> <!-- Element mit explizitem Namespace -->

</root>

Hier wird `http://example.com` als Default Namespace deklariert. Das Element `element1` gehört daher implizit zu diesem Namespace, während das Element `element2` explizit dem Namespace `http://another-example.com` zugeordnet ist.

A screenshot of a computer

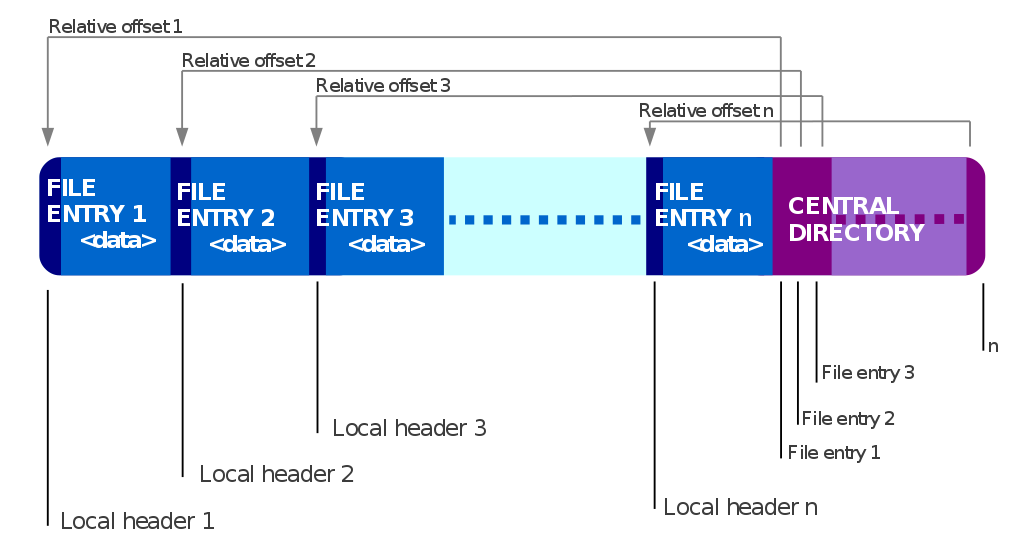
Description automatically generated

**5.1**

Ein Containerformat ist eine Dateiformat, das es ermöglicht, mehrere Datenströme in einer einzelnen Datei zu integrieren, oft zusammen mit Metadaten zur Identifizierung und weiteren Beschreibung dieser Ströme. Beispiele für Containerformate sind Archivdateien wie das ZIP-Format.

Aufbau:

|  |
| --- |



Quelle: <https://en.wikipedia.org/wiki/ZIP_(file_format)>

End of Central Directory Record: Dieser Teil befindet sich am Ende der ZIP-Datei und enthält Metadaten über die gesamte ZIP-Datei, einschließlich der Anzahl der Dateien und Verzeichnisse sowie des Offsets zum Beginn des zentralen Verzeichnisses.

Central Directory: Das zentrale Verzeichnis enthält Einträge für jede Datei oder jedes Verzeichnis in der ZIP-Datei. Jeder Eintrag enthält Metadaten wie Dateinamen, Dateigröße und den Offset zum Anfang des lokalen Dateikopfs.

Local File Header: Der lokale Dateikopf befindet sich vor den Daten einer jeden Datei in der ZIP-Datei und enthält redundante Metadaten, um schnell auf Informationen über die Datei zugreifen zu können.

Datenströme: Die eigentlichen Daten der Dateien und Verzeichnisse werden nach den lokalen Dateiköpfen gespeichert. Diese Daten können komprimiert oder unkomprimiert sein, abhängig von den Einstellungen bei der Erstellung der ZIP-Datei.

Die Kombination aus zentralem Verzeichnis, lokalem Dateikopf und Datenströmen ermöglicht eine schnelle und effiziente Verwaltung von ZIP-Dateien, einschließlich des Hinzufügens neuer Dateien und der Aktualisierung bestehender Dateien.

**5.2**

Ein MIME Media Type ist eine Kennzeichnung für den Inhalt einer Datei oder eines Datenstroms im Internet. Er gibt an, welche Art von Daten vorliegt (z. B. Text, Bild oder Anwendung) und hilft Anwendungen, den Inhalt richtig zu interpretieren und anzuzeigen.

Für Zip ist der Mime-Type: application/zip

**5.3**

„Ecma International ist eine private, internationale [Normungsorganisation](https://de.wikipedia.org/wiki/Normungsorganisation) zur [Normung](https://de.wikipedia.org/wiki/Normung) von [Informations- und Kommunikationssystemen](https://de.wikipedia.org/wiki/Computersystem) und [Unterhaltungselektronik](https://de.wikipedia.org/wiki/Unterhaltungselektronik) mit Sitz in [Genf](https://de.wikipedia.org/wiki/Genf). “

Quelle: <https://de.wikipedia.org/wiki/Ecma_International>

ECMA steht für European Computer Manufacturers Association

**5.4**

Ein OpenXML-Dokument, das im ZIP-Format gespeichert ist, enthält neben den Hauptteilen auch spezielle Beziehungs-Teile. Diese speichern die Beziehungen zwischen verschiedenen Teilen des Dokuments. Der Beziehungs-Teil für das gesamte Paket ist "/\_rels/.rels", und für einen Teil wie "/a/b/c.xml" wäre es "/a/b/\_rels/c.xml.rels". Die Anwendung muss diese Beziehungen analysieren, um das gesamte Dokument korrekt zu interpretieren. Alle anderen Teile können OpenXML, benutzerdefiniertes XML oder Inhalte beliebigen Typs enthalten.

**5.5**

**Wohl geformt** bedeutet das Objekt erfüllt die [W3C Anforderungen](http://www.w3.org/TR/REC-xml/).

**Gültig** bedeutet das eine wohl-geformte XML Datei weiteren in einem Schema deklarierten Anforderungen erfüllt.

**5.6**

Eine XML-Bomb oder auch billion laughs attack ist eine Variante einer Denial of Service Attacke. Das Ziel ist es durch Rekursion, Systemressourcen zu überlasten und zum Absturz zu bringen.

Die XML „Bombe“ nutzt rekursiven Entitäten, um eine exponentiell wachsende Anzahl von Entitäten zu erzeugen. Beim öffnen verbraucht der Parser auf einmal viel mehr Ressourcen als geplant , was zu einem Absturz führen kann.

**5.7**

Ein XML External Entity (XXE) Angriff ist eine Sicherheitslücke, bei der ein Angreifer einen XML-Parser ausnutzt, um externe Ressourcen einzubeziehen und zu lesen, die sich außerhalb des ursprünglichen XML-Dokuments befinden. Durch speziell gestaltete XML-Dokumente mit externen Entity-Deklarationen kann der Angreifer sensible Daten offenlegen oder beliebigen Code ausführen.

Beispiel:

<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE foo [

<!ELEMENT foo ANY >

<!ENTITY xxe SYSTEM "file:///dev/random" >]><foo>&xxe;</foo>

Quelle: [owasp.org/Testing\_for\_XML\_Injection](https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/07-Input_Validation_Testing/07-Testing_for_XML_Injection)

A screenshot of a computer

Description automatically generated

**6.1**

Erlaubt sind:

Buchstaben (a-Z) zwischen Groß und kleinschreibung wird nicht unterschieden

Zahlen (0-9)

Bindestrich und Punkt, jedoch darf Punkt nicht als letztes Zeichen und auch nicht direkt vor Bindestrich

**6.2**

IDNs (Internationalized Domain Names) sind Domainnamen, die Zeichen aus verschiedenen Sprachen und Schriften enthalten können, nicht nur aus dem traditionellen ASCII-Zeichensatz. Sie ermöglichen es, Domainnamen in der Muttersprache von Personen zu registrieren und anzuzeigen, was die Zugänglichkeit des Internets verbessert. Die IDNs werden durch eine Kodierung namens Punycode in eine ASCII-Zeichenfolge umgewandelt, die vom DNS-System verarbeitet werden kann.

**6.3**

Punycoding ist ein Verfahren zur Kodierung von Internationalized Domain Names (IDNs) in eine ASCII-Zeichenfolge, die vom Domain Name System (DNS) verarbeitet werden kann. Dies ermöglicht die Verwendung von Zeichen aus verschiedenen Sprachen und Schriften in Domainnamen.

Das Encoding funktioniert, indem die Unicode-Zeichen des IDNs in eine spezielle ASCII-Zeichenfolge umgewandelt werden. Dies erfolgt in mehreren Schritten:

1. Unicode-Zeichen des IDNs in ihre entsprechenden Unicode-Punkte umwandeln:

Beispiel: Das IDN "müller.de" enthält das Zeichen "ü", das dem Unicode-Punkt U+00FC entspricht.

1. Unicode-Punkte in eine ASCII-kompatible Form bringen, indem sie in das "Punycode"-Format konvertiert werden:

Beispiel: Der Unicode-Punkt U+00FC wird im Punycode-Format zu "fc" umgewandelt.

1. Punycode-Zeichen in eine ASCII-Zeichenfolge umwandeln, indem sie mit den Zeichen "xn--" beginnen:

Beispiel: Die ASCII-Zeichenfolge "xn--" wird vor dem Punycode-Teil "fc" platziert, um "xn--fc" zu bilden. Damit wird das IDN "müller.de" im DNS als "xn--mller-kva.de" dargestellt.

**6.4**

Der Browser durchgeht folgende Schritte nach Eingabe einer IDN:

1. Der Browser empfängt den eingegebenen IDN.
2. Er überprüft, ob der eingegebene Domainname ein IDN ist, indem er nach spezifischen Unicode-Zeichen sucht.
3. Wenn eine IDN erkannt wird, wird es automatisch in Punycode umgewandelt.
4. Der Browser sendet dann die Anfrage an den DNS-Server mit dem Punycode-Domainnamen.
5. Der DNS-Server erhält die Anfrage und verarbeitet den Punycode-Domainnamen, um die IP-Adresse des entsprechenden Servers zu finden.
6. Schließlich lädt der Browser die Webseite, indem er die IP-Adresse abruft, die der DNS-Server zurückgegeben hat.

**6.5**

Homografische Phishing-Angriffe nutzen die Ähnlichkeit verschiedener Alphabete aus, um gefälschte URLs zu erstellen, die echten Domainnamen ähneln. Zum Beispiel können das griechische "Ο" (Omega), das lateinische "O" und das kyrillische "О" (O) sehr ähnlich aussehen. Da diese Buchstaben unterschiedliche Unicode-Zeichen sind, ist es für den Endbenutzer oft kaum möglich, sie zu unterscheiden. Diese gefälschten URLs können dann verwendet werden, um Benutzer auf betrügerische Websites umzuleiten, die echten Websites täuschend ähnlich sind. Es ist wichtig, verdächtige URLs genau zu prüfen und auf vertrauenswürdigen Websites persönliche oder sensible Informationen einzugeben.

**6.6**

Laut [chromium.com](http://chromium.com) implementiert Chrome und Firefox folgende Schutzmaßnahmen gegen homografische Phishing-Angriffe:

1. **Schemaüberprüfung**: Der Browser überprüft das Schema der URL und erlaubt nur Navigationen mit den Schemata "http" oder "https". Dadurch werden Navigationen zu anderen Protokollen blockiert, die möglicherweise betrügerisch sind.
2. **Weiterleitungsüberprüfung**: Wenn die Navigation eine Weiterleitung ist, wird die Weiterleitungs-Kette überprüft, um sicherzustellen, dass sie sicher ist. Wenn die Weiterleitungs-Kette verdächtig erscheint, wird die Navigation blockiert.
3. **Website-Engagement-Bewertung**: Der Browser bewertet die Engagement-Bewertung der Website anhand verschiedener Kriterien wie Benutzerinteraktionen und Zuverlässigkeit. Wenn eine Website eine ausreichend hohe Engagement-Bewertung hat, wird die Navigation zugelassen.
4. **Überprüfung in der domains.list**: Der Browser überprüft, ob der Hostname der Navigation in einer sicheren Liste wie der domains.list enthalten ist. Wenn ja, wird die Navigation zugelassen.
5. **Benutzerentscheidungen**: Wenn der Benutzer zuvor den Hostnamen der Navigation als sicher markiert hat, wird die Navigation zugelassen, auch wenn sie sonst geblockt worden wäre.

Bei Erstellung eines eigenen Browsers sollte man diese Schritte auch implementieren-

Nutzer sollten folgende Maßnahmen beachten:

1. Achten Sie auf die URL der Website und überprüfen Sie sie sorgfältig, bevor Sie persönliche oder sensible Informationen eingeben.
2. Verwenden Sie Lesezeichen, um vertrauenswürdige Websites zu speichern und vermeiden Sie das Klicken auf verdächtige Links in E-Mails oder Nachrichten.
3. Aktualisieren Sie regelmäßig Ihren Browser und Ihre Sicherheitssoftware, um von den neuesten Sicherheitsfunktionen zu profitieren.
4. Seien Sie skeptisch gegenüber unerwarteten Pop-ups oder Warnmeldungen und überprüfen Sie die Echtheit der Quelle, bevor Sie auf sie reagieren.

A screenshot of a computer

Description automatically generated

**7.2**

1. Weil die Entwickler keinen neutralen Standpunkt haben können, mit dem Sie die Benutzerfreundlichkeit des Designs beurteilen könnten.

2. Es ist relativ günstig zu alternativen UX-Testing Verfahren.

3. Da das Feedback der Tester ist sehr aussagekräftig

**7.3**

Laut Nielsen reichen bereits 5 Tester aus der Zielgruppe für einen sinnvollen Test aus. Natürlich sind mehr Tester aus verschiedenen Gruppen besser.

**7.4**

Die User sollten Aufgaben bekommen, die die gewünschte Verwendung der Website nachstellen soll. Während des Tests sollten sich die Entwickler nicht einmischen, um das Ergebnis nicht zu manipulieren.

A screenshot of a computer

Description automatically generated

**8.1**

Die BEM-Konvention (Block Element Modifier) ist eine Namenskonvention für CSS-Klassen in der Webentwicklung, die dazu dient, eine klare und konsistente Struktur für den Aufbau von CSS-Regeln zu schaffen. Sie besteht aus drei Teilen:

* Block: Ein unabhängiges, wiederverwendbares Element auf der Seite, das in sich geschlossen ist.
* Element: Ein Teil eines Blocks, der innerhalb des Blocks existiert und eine funktionale oder semantische Beziehung zum Block hat.
* Modifier: Eine zusätzliche Eigenschaft oder Variation eines Blocks oder Elements

**8.2**

h1[title]:

Selektiert alle h1-Elemente, die ein title-Attribut besitzen.

span[hello="Cleveland"] [goodbye="Columbus"]:

Selektiert alle span-Elemente mit dem Attribut hello="Cleveland", die ein Kind-Element mit dem Attribut goodbye="Columbus" haben.

a[href^="http"]:

Selektiert alle a-Elemente, deren href-Attribute die mit "http" beginnt.

**8.3**

background: linear-gradient(to right, rgb(190,190,190), rgb(195,215,250));

Erstellt ein Hintergrund mit Farbüberlauf von links nach rechts mit gegebenen Farbpaletten.

**8.4**

Wenn das HTML folgenden Button mit einem Tooltip hat:

<button class="tooltip">Tooltip Button

<span class="tooltiptext">Tooltip Text</span>

</button>

Sieht die CSS für einen Hover Tooltip für 4 Sekunden folgend aus:

.tooltip .tooltiptext {

visibility: hidden;

background-color: black;

color: white;

position: absolute;

transition: visibility 0s 4s;

}

.tooltip:hover .tooltiptext {

visibility: visible;

transition-delay: 0s;

}

**8.5**  
 siehe Anhang

A screenshot of a computer script

Description automatically generated

9.1

REPL steht für "Read-Eval-Print Loop" und ist eine interaktive Umgebung, die es ermöglicht, Codezeilen einzugeben, auszuführen, das Ergebnis zu sehen und dann weitere Codezeilen einzugeben. Der Begriff "Read" bedeutet das Lesen der Eingabe des Benutzers, "Eval" steht für die Auswertung des eingegebenen Codes, "Print" für die Anzeige des Ergebnisses und "Loop" für die Wiederholung dieses Prozesses.

Ein gutes Beispiel für eine REPL-Umgebung ist die JavaScript-Entwicklungsumgebung, die in den meisten Browsern über die Konsole verfügbar ist. Hier kann der Benutzer JavaScript-Code eingeben und sofort sehen, wie dieser ausgeführt wird. Zum Beispiel:

```javascript

> var x = 5;

undefined

> var y = 10;

undefined

> x + y;

15

```

In diesem Beispiel können wir sehen, wie der Benutzer nacheinander drei Codezeilen eingibt und das Ergebnis der Ausführung jeder Zeile sofort angezeigt wird. Dies ermöglicht eine schnelle und iterative Entwicklung und Fehlersuche beim Programmieren.

9.2

Siehe Anhang

9.3

Anzahl der Dezimalstellen = 3072 / log10(2) = 1030.0336… ~1030 Stellen

9.4

Siehe Anhang