**Università degli Studi di Padova**

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica


# Progettazione e realizzazione di un'estensione per la validazione dell'accessibilità delle pagine web

*Tesi di laurea*

*Relatore*

Prof.ssa Ombretta Gaggi

*Laureando*

Riccardo Fabbian

*Matricola* 2009110

Anno Accademico 2023-2024

# Sommario

Il presente documento descrive il lavoro svolto durante uno stage di trecento ore dal laureando Riccardo Fabbian presso l'Università degli Studi di Padova. L'obiettivo principale era lo sviluppo di un plugin per l'analisi dell'accessibilità delle pagine web, in conformità con le linee guida WCAG. Questo ha comportato la progettazione e realizzazione di un'estensione per browser, l'implementazione di algoritmi avanzati per l'identificazione delle immagini, la gestione degli eventi, e la manipolazione del DOM, nonché l'uso delle API di Chrome.

Durante lo stage, ho approfondito la programmazione asincrona in JavaScript, imparando a utilizzare Promise e async/await per migliorare la reattività e l'efficienza del plugin. Il processo di verifica e validazione è stato continuo, impiegando strumenti come WAVE e HTML Validator per assicurare l'accessibilità e la conformità agli standard HTML.

Il progetto ha raggiunto con successo i suoi obiettivi. L'estensione è facilmente estendibile e modificabile, con il codice sorgente disponibile su GitHub per favorire la collaborazione e il miglioramento continuo da parte della comunità di sviluppatori.

*"The Best Way to Predict the Future is to Create it"*

— Peter Drucker

# Ringraziamenti

*Per prima cosa, vorrei esprimere la mia sincera gratitudine alla Prof.ssa Ombretta Gaggi, relatrice della mia tesi, per l'assistenza e il sostegno che mi ha fornito durante tutto il percorso di stage e la stesura della tesi.*

*Desidero ringraziare con affetto i miei genitori per il sostegno, il grande aiuto e per essermi stati vicini in ogni momento durante gli anni di studio.*

*Ho desiderio di ringraziare poi i miei amici per tutti i bellissimi anni passati insieme e le mille avventure vissute.*

*Padova, Luglio 2024* Riccardo Fabbian

# Indice

# Elenco delle figure

# Elenco delle tabelle

# Capitolo 1

# Introduzione

## 1.1 Background

1° Gennaio 1983, questo è considerato universalmente il giorno della nascita di Internet. Prima di questa data i networks non erano altro che reti locali che non possedevano i protocolli per la comunicazione verso l'esterno. Da quel giorno Internet si è rivoluzionato, unificato e modificato cambiando il mondo che conoscevamo sotto ogni aspetto. Da quel giorno, grazie a degli standard di comunicazione universali è stato finalmente possibile comunicare con chiunque in qualsiasi parte del mondo attraverso la rete.
Sebbene il raggiungimento di standard comuni ci abbia permesso di superare tutte le aspettative ed abbattare le ultime barriere della comunicazione, il Web, tra tutti il servizio Internet più comune ed utilizzato, è nato e si è inizialmente evoluto senza regole.
I primi paesi a proporre una regolamentazione nazionale per garantire l'accessibilità web furono l'Australia con il *Disability Discrimination Act* del 1992 [1], gli Stati Uniti con la *Section 508 del Rehabilitation Act* del 1998 [2] ed il Regno Unito con il *Disability Discrimination Act* del 1995 [3].
Nel 2002, la Commissione Europea ha pubblicato la Comunicazione *eEurope 2002* [4], che invita gli Stati membri a garantire l'accesso alle informazioni e ai servizi web a tutti i cittadini, indipendentemente dalle loro condizioni fisiche. L'Italia ha recepito la Comunicazione *eEurope 2002* attraverso la "Legge Stanca" [5] che ha introdotto l'obbligo di accessibilità per i siti web delle pubbliche amministrazioni.
Questi erano però solo i primi passi verso un web accessibile a tutti. Fortunamente la spinta non si è esaurita ed anzi, ha ricevuto nuovo vigore portando alla creazione di standard di accessibilità comuni a tutti i paesi, come ad esempio le *Web Content Accessibility Guidelines (WCAG)*[g]del *World Wide Web Consortium (W3C)*[g]o le *Americans with Disabilities Act (ADA)*[g]Accessibility Guidelines.

## 1.2 Idea

In questo contesto, dato anche il maggior interesse sociale verso le tematiche di inclusione e l'arrivo di nuovi pacchetti di riforme Europee in materia di accessibilità *European Accessibility Act*[6], è stato richiesto di sviluppare un'estensione per il browser Chrome che permetta di valutare l'accessibilità di un sito web in modo rapido ed efficace.
L'analisi dell'accessibilità di un sito web è un processo complesso che si compone di due macro-aree: l'analisi statica e l'analisi dinamica. L'analisi statica consiste nell'analizzare il codice sorgente di una pagina web per verificare che sia conforme agli standard di accessibilità, che rispetti le best practices e che non contenga errori quali la mancata chiusura di tag, l'uso di *tag* deprecati o l'uso di *tag* non validi.
L'analisi dinamica invece risulta più complessa in quanto richiede un processo attivo di valutazione in prima persona, da parte di un esperto di accessibilità che navighi il sito e verifichi che tutte le funzionalità siano accessibili, che non vi siano errori di usabilità e che l'interfaccia rispetti le aspettative comuni riguardanti il posizionamento di elementi quali il pulsante del menù o il pulsante di login.
Queste due fasi di analisi sono fondamentali e complementari, in quanto l'analisi statica permette di individuare velocemente errori di codifica e di struttura che altrimenti richiederebbero una considerevole quantità di tempo ad un operatore umano, mentre l'analisi dinamica permette di verificare tutti quegli aspetti che una macchina non può valutare, come ad esempio la correttezza di una descrizione o la corretta navigabilità di un sito.
L'obiettivo principale dell'estensione è eseguire un'analisi statica preventiva della pagina web e semplificare il processo di analisi dinamica raggruppando tutte le informazioni necessarie in un'unica interfaccia.

## 1.3 Organizzazione del testo

**Il secondo capitolo** introduce il progetto di stage, descrivendone gli scopi e gli obiettivi;

**Il terzo capitolo** illustra tutti gli strumenti e le tecnologie utilizzate per questo progetto;

**Il quarto capitolo** approfondisce il processo di sviluppo analizzando i requisiti identificati e le soluzioni implementate;

**Il quinto capitolo** tratta la fase di verifica e validazione del prodotto sviluppato;

**Il sesto capitolo** descrive gli obiettivi raggiunti, le conoscenze acquisite ed una valutazione personale dello stage

Riguardo la stesura del testo, relativamente al documento sono state adottate le seguenti convenzioni tipografiche:

- gli acronimi, le abbreviazioni e i termini ambigui o di uso non comune menzionati vengono definiti nel glossario, situato alla fine del presente documento;
- per la prima occorrenza dei termini riportati nel glossario viene utilizzata la seguente nomenclatura: *parola*[g];
- i termini in lingua straniera o facenti parti del gergo tecnico sono evidenziati con il carattere *corsivo*.
- i tag e gli attributi HTML, i file e le estensioni sono evidenziati come segue `tag`.

# Capitolo 2

# Descrizione del progetto di stage

## 2.1 Introduzione al progetto

L'accessibilità è una disciplina giovane e in costante evoluzione, impegnata a regolamentare un panorama vasto e diversificato come quello del web. Creare un'estensione per Chrome in grado di fornire un'analisi completa di tutti gli aspetti di accessibilità di un sito web richiede uno sforzo significativo in termini di tempo e risorse. Di conseguenza, nell'ambito di uno stage della durata di 300 ore, non era possibile affrontare un progetto di tale portata.
Per questo motivo, il progetto di stage si è concentrato su due aspetti specifici dell'accessibilità: l'analisi delle immagini e l'analisi *heading*.
L'analisi delle immagini è stata scelta poiché le immagini possono rappresentare un ostacolo significativo per gli utenti non vedenti. Spesso, queste vengono inserite all'interno delle pagine web senza un'adeguata contestualizzazione, rendendo difficile, se non impossibile, la comprensione del contenuto. La soluzione proposta consisteva nell'inserire un attributo all'interno del *tag img* che contenesse la descrizione testuale dell'immagine stessa. Tuttavia, questo attributo spesso non viene inserito, viene lasciato vuoto oppure contiene una descrizione generica che non contribuisce alla contestualizzazione dell'immagine.
Per ovviare a queste problematiche, alcune grandi aziende come Microsoft [7] hanno provato a utilizzare l'intelligenza artificiale per generare descrizioni automatiche delle immagini. Questo approccio, sebbene promettente, presenta ancora importanti limitazioni in termini di precisione e qualità delle descrizioni e richiede quindi la supervisione attiva da parte di un esperto di accessibilità.
L'analisi *heading*, invece, è stata scelta poiché gli *heading* (titoli e sottotitoli) sono elementi fondamentali per la struttura e la navigazione di un sito web. Essi non solo facilitano la lettura e la comprensione del contenuto da parte degli utenti, ma sono anche cruciali per i *crawler* dei motori di ricerca, che utilizzano questi elementi per

indicizzare e comprendere la gerarchia del contenuto.
In sintesi, concentrarsi su questi due aspetti specifici ha permesso di affrontare problematiche di accessibilità particolarmente rilevanti all'interno del tempo limitato dello stage, fornendo al contempo un contributo significativo alla creazione di contenuti web più accessibili.

## 2.2 Motivazioni

Le motivazioni alla base del progetto sono molteplici.
In primo luogo, come precedentemente accennato, negli ultimi anni c'è un crescente interesse sociale verso le tematiche di inclusione, e l'accessibilità rientra pienamente in questo contesto. L'accessibilità web è fondamentale per garantire che tutti, indipendentemente dalle loro capacità fisiche o cognitive, possano usufruire delle risorse online.
In secondo luogo, le aziende stanno riconoscendo che l'accessibilità non è solo un obbligo legale, ma anche un'opportunità di business. Rendere un sito web accessibile significa ampliarne la fruibilità per un numero maggiore di utenti, aumentando così il proprio bacino di utenza e potenziale clientela. Un sito web accessibile può migliorare l'esperienza utente, favorire il posizionamento nei motori di ricerca e aumentare la soddisfazione dei clienti, tutti fattori che contribuiscono al successo aziendale.
Infine, ultimo ma non per importanza, ad oggi non esistono strumenti in grado di fornire un'analisi completa e accurata di tutti gli aspetti di accessibilità di un sito web. Gli strumenti esistenti tendono a concentrarsi su aspetti specifici dell'accessibilità ma nessuno di essi offre una valutazione globale.

## 2.3 Analisi soluzioni esistenti

Ad oggi sono disponibili diversi strumenti per l'analisi dell'accessibilità di un sito web. Alcuni di questi sono gratuiti, altri a pagamento. Tra i più noti si possono citare:

- **WAVE**: Un'estensione gratuita disponibile nello store di Chrome, sviluppata da WebAIM. Fornisce un'analisi di base dell'accessibilità di una pagina web, evidenziando problemi come testo alternativo mancante per le immagini, errori di contrasto e problemi di struttura. Tuttavia, *Web Accessibility Evaluation Tool (WAVE)*[g]presenta alcune limitazioni significative: non offre funzionalità fondamentali come l'analisi dei *tag* o degli attributi, e non supporta la navigazione continuativa, richiedendo il riavvio del plugin ogni volta che si passa a una nuova pagina. Sebbene sia un ottimo strumento per un'analisi rapida e superficiale, non è adatto per un'analisi approfondita e accurata delle problematiche di accessibilità

di un sito web. La figura 2.1 mostra un esempio di analisi della homepage del sito unipd.it.

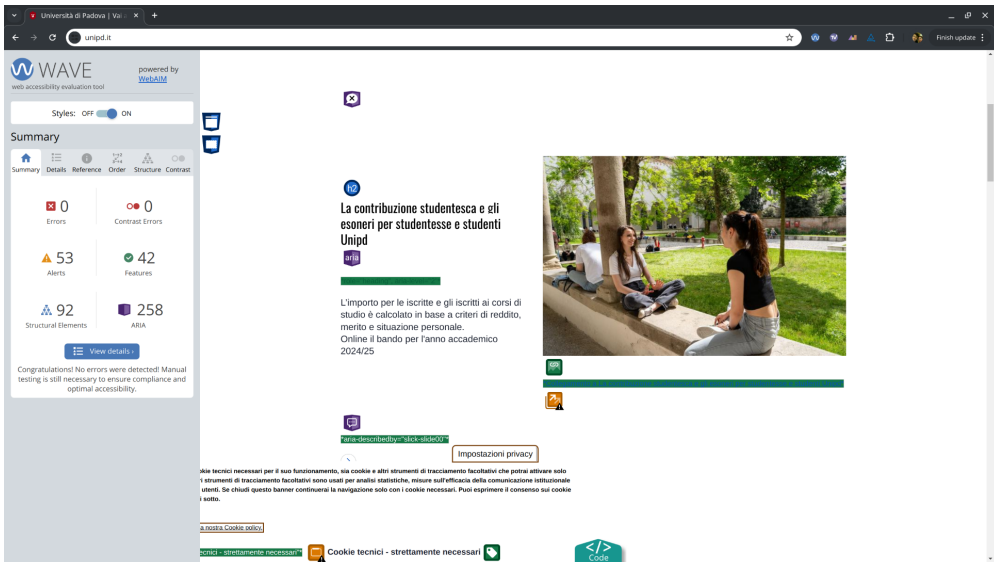


**Figura 2.1:** Analisi homepage unipd.it utilizzando l'estensione WAVE

- **Lighthouse**: Uno strumento di sviluppo integrato in Google Chrome che offre un'analisi dettagliata di diversi aspetti di una pagina web, tra cui prestazioni, accessibilità, *Search Engine Optimization (SEO)*[g]e best practices. È uno strumento potente e versatile, ampiamente utilizzato per testare la qualità complessiva dei siti web. Lighthouse esegue una serie di test automatizzati sulla pagina web, fornendo un punteggio complessivo per ciascuna delle aree analizzate. Per quanto riguarda l'accessibilità, Lighthouse identifica una serie di problemi comuni, come l'assenza di attributi *Accessible Rich Internet Applications (ARIA)*[g], link mancanti o non descrittivi, problemi di struttura *Document Object Model (DOM)*[g]e l'uso inappropriato di elementi semantici. Tuttavia, nonostante la sua utilità, Lighthouse presenta alcune limitazioni quando si tratta di un'analisi completa e approfondita dell'accessibilità:
  - **Contrasto del Colore:** sebbene Lighthouse esegua alcuni controlli di base sul contrasto del colore, non è sempre accurato nel rilevare tutti i problemi di contrasto, specialmente in situazioni complesse o in elementi dinamici della pagina.
  - **Attributi Erronei:** Lighthouse può segnalare l'assenza di attributi ARIA, ma non offre una valutazione approfondita degli attributi erronei o mal utilizzati, che possono avere un impatto significativo sull'accessibilità.

Un esempio di analisi della homepage del sito unipd.it è mostrato nella figura 2.2.

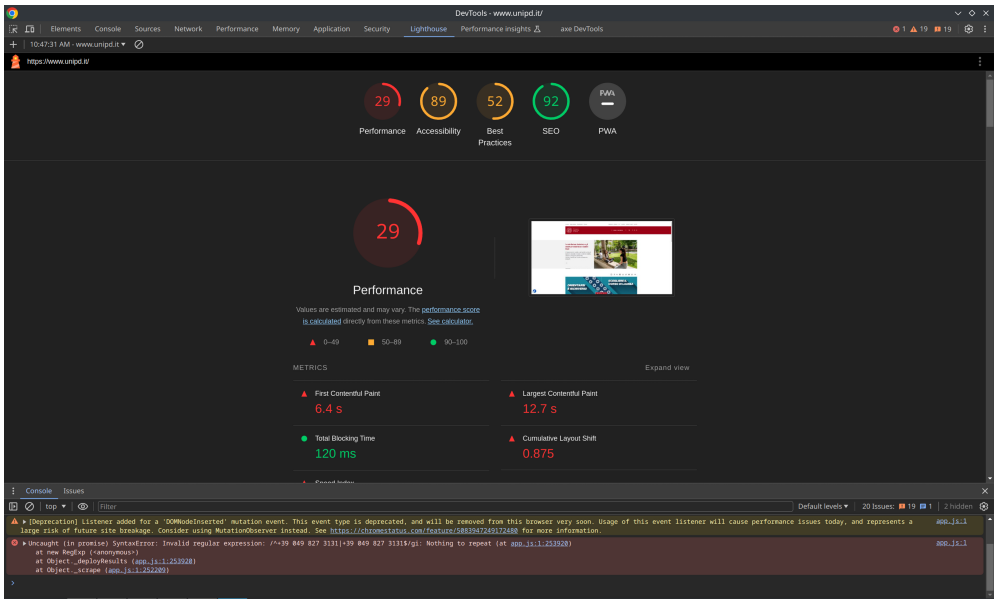


**Figura 2.2:** Analisi homepage unipd.it utilizzando lighthouse

- **axe**: Uno strumento open source sviluppato da Deque Systems che offre un'analisi dettagliata dell'accessibilità di una pagina web. Questo strumento è molto utile per la sua capacità di individuare problemi specifici di accessibilità, tra cui errori di ARIA, problemi di navigazione e contrasti inadeguati. Axe integra efficacemente i principi delle WCAG e può essere utilizzato come estensione del browser, come libreria JavaScript o integrato nei flussi di lavoro di sviluppo continuo. Tuttavia, nonostante la sua efficacia, axe presenta alcune limitazioni che impediscono una valutazione completa di tutti gli aspetti di accessibilità di un sito web.

  – **Supporto per Test Dinamici:** sebbene axe sia efficace nell'analisi di pagine statiche, può incontrare difficoltà con contenuti dinamici o interattivi, come moduli complessi o applicazioni a *Single Page Application (SPA)*[g]. L'analisi di questi elementi potrebbe richiedere configurazioni aggiuntive o integrazioni con altri strumenti di testing.

  – **Navigazione Continuativa:** axe non supporta la navigazione continuativa per l'accessibilità, richiedendo di rieseguire l'analisi ogni volta che si passa a una nuova pagina.

La figura 2.3 illustra l'analisi del sito unipd.it effettuata da axe.

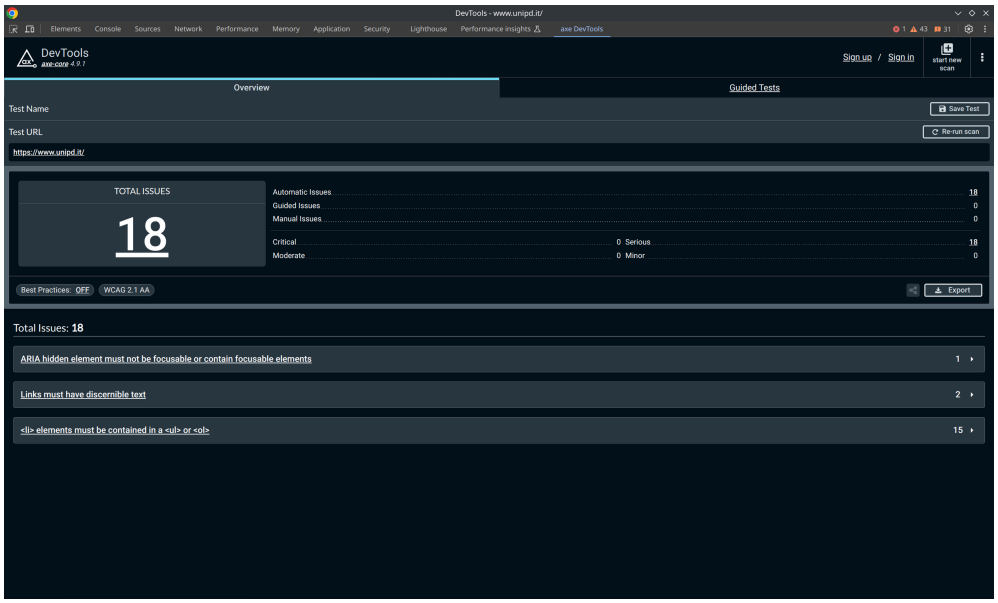


**Figura 2.3:** Analisi homepage unipd.it utilizzando axe

- **TotalValidator**: Uno strumento a pagamento che fornisce un'analisi dettagliata dell'accessibilità di un sito web, inclusa la validazione del codice *HyperText Markup Language (HTML)*[g] e la verifica dei collegamenti, consentendo così di individuare collegamenti interrotti o non funzionanti. Tuttavia, TotalValidator presenta alcune limitazioni che impediscono una valutazione complessiva e accurata di tutti gli aspetti di accessibilità di un sito web.
  - **Analisi di Accessibilità Limitata:** TotalValidator fornisce una serie di controlli di accessibilità automatizzati, ma non copre problemi di contrasto del colore in modo accurato nè l'adeguatezza delle descrizioni testuali delle immagini.
  - **Estensione incompleta:** l'estensione presente nello store di Chrome non è uno standalone, ma richiede l'installazione di un software esterno per funzionare correttamente. Al suo utilizzo, inoltre, è richiesto che il software sia in esecuzione in background.

La figura 2.4 mostra un esempio di analisi della homepage del sito unipd.it.

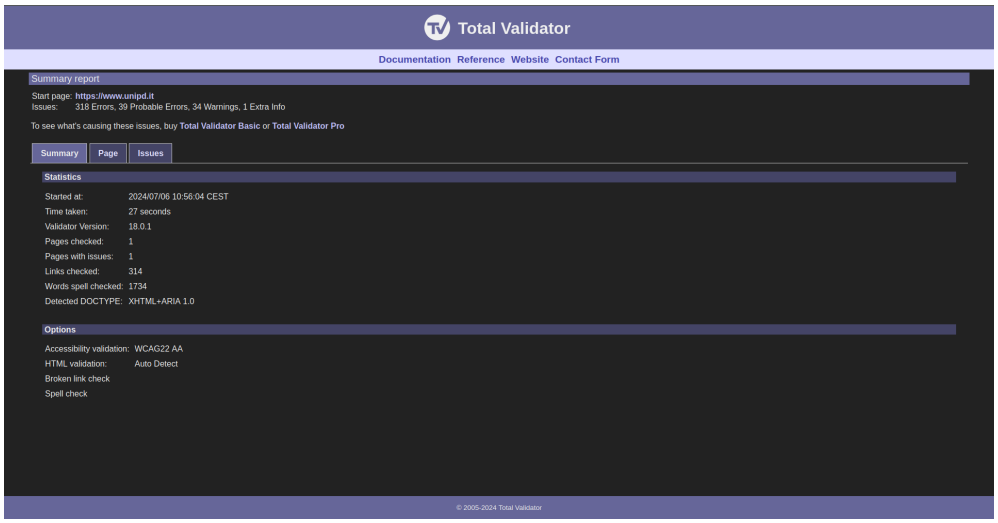


**Figura 2.4:** Analisi homepage unipd.it utilizzando totalvalidator

## 2.4 Obiettivi dello stage

Si farà riferimento ai requisiti secondo le seguenti notazioni:

- **O** per i requisiti obbligatori, vincolanti in quanto obiettivo primario richiesto dal committente;
- **D** per i requisiti desiderabili, non vincolanti o strettamente necessari, ma dal riconoscibile valore aggiunto;
- **F** per i requisiti facoltativi, rappresentanti valore aggiunto non strettamente competitivo.

Le sigle precedentemente indicate saranno seguite da una coppia sequenziale di numeri, identificativo del requisito.
Obiettivi fissati:

- **Obbligatori**
  - **O01:** Identificazione di tutte le immagini inserite tramite *tag* HTML;
  - **O02:** Identificazione di tutte le immagini di background inserite tramite *Cascading Style Sheets (CSS)*[g];
  - **O03:** Analisi attributo *alt* in tutte le immagini inserite tramite *tag* HTML;
  - **O04:** Analisi dimensione immagine;
  - **O05:** Visualizzazione immagine se presente a schermo;
  - **O06:** Visualizzazione *code snippet* del *tag* dell'immagine o del *tag* che la contiene;

- **Desiderabili**
  - **D01:** Sviluppo di un plugin che rimane attivo durante la navigazione;
  - **D02:** Sviluppo modulo di analisi *heading*;
- **Facoltativi**
  - **F01:** Pubblicazione del plugin all'interno del Chrome Web Store;

Per ogni immagine viene fornita la possibilità di creare una nota personalizzata, permettendo all'utente di annotare eventuali problemi riscontrati durante l'analisi e di consultarli in un secondo momento.
Oltre agli obiettivi prefissati, è stato possibile implementare sia la funzionalità di analisi *heading* che la navigazione verso altre pagine web.

# Capitolo 3

# Analisi degli strumenti e delle tecnologie utilizzate

## 3.1 Linguaggi

### 3.1.1 HTML

HTML è il linguaggio di markup standard utilizzato per creare e strutturare contenuti sul web. Introdotto per la prima volta da Tim Berners-Lee nel 1991, HTML è diventato uno strumento fondamentale nello sviluppo di siti web e applicazioni web. HTML permette agli sviluppatori di definire elementi come titoli, paragrafi, immagini, link e molti altri componenti fondamentali per costruire pagine web interattive e dinamiche. La struttura di un documento HTML è costituita da una serie di *tag*, ciascuno dei quali descrive un tipo di contenuto o un comportamento specifico. I *tag* sono racchiusi tra parentesi angolari e sono solitamente usati in coppie di apertura e chiusura, alcuni però, come `<img>` per le immagini, sono auto-chiudenti.
Nel 2014 l'introduzione di HTML5 (logo presente in figura 3.1), ha portato numerosi miglioramenti e nuove funzionalità, tra cui il supporto per contenuti multimediali nativi, una migliore gestione delle forme e una semantica più ricca per rendere le pagine web più accessibili e comprensibili sia per gli utenti che per i motori di ricerca.


**Figura 3.1:** Logo HTML5

### 3.1.2 CSS

CSS (logo presente in figura 3.2) è il linguaggio utilizzato per descrivere l'aspetto e la formattazione di un documento scritto in HTML. Introdotto per la prima volta da Håkon Wium Lie e Bert Bos nel 1996, CSS ha rivoluzionato il modo in cui le pagine web vengono stilizzate, separando la presentazione dal contenuto. Questo approccio consente di mantenere un codice HTML più pulito e semplificato, migliorando al contempo la flessibilità e la manutenibilità del sito web.
CSS permette agli sviluppatori di controllare vari aspetti del design di una pagina web, come *layout*, colori, *font*, spaziatura, bordi, immagini di sfondo, e molto altro. Le regole CSS sono definite in file separati con estensione `.css` o direttamente all'interno del documento HTML tramite *tag* `<style>` o attributi style. Le regole CSS seguono una sintassi specifica che include selettori, proprietà e valori.


**Figura 3.2:** Logo CSS

### 3.1.3 Javascript

JavaScript (logo presente in figura 3.3) è un linguaggio di programmazione dinamico e versatile utilizzato per creare contenuti interattivi e dinamici su pagine web. Introdotto per la prima volta nel 1995 da Brendan Eich, JavaScript è diventato uno dei pilastri fondamentali del web insieme a HTML e CSS.
Originariamente concepito per aggiungere comportamenti dinamici ai siti web, JavaScript ha evoluto in un linguaggio di programmazione completo che supporta paradigmi di programmazione orientata agli oggetti, funzionale e imperativa.
JavaScript consente agli sviluppatori di manipolare il DOM di una pagina web, rispondendo agli eventi utente come click, input da tastiera, movimenti del mouse e altro.


**Figura 3.3:** Logo Javascript

## 3.2 Tecnologie e strumenti

### 3.2.1 Chrome Extension

Le estensioni di Chrome (logo presente in figura 3.4) sono piccoli programmi software che personalizzano l'esperienza di navigazione o offrono funzionalità integrate con il browser. Il *Manifest V3* è l'ultima iterazione del formato di manifest per le estensioni di Chrome, introdotto da Google con l'obiettivo di migliorare la sicurezza, la privacy e le prestazioni delle estensioni. Questa versione introduce cambiamenti significativi rispetto al *Manifest* V2, tra cui una nuova architettura per la gestione degli script di background, un utilizzo più rigoroso delle *Application Program Interface (API)*[g]e restrizioni più severe sull'uso delle risorse. Gli elementi fondamentali di una estensione basata su *Manifest V3* sono:

- **Manifest:** il file `manifest.json` è il cuore dell'estensione, contenente le informazioni di base come il nome, la versione, i permessi richiesti e gli script da eseguire.

È essenziale per dichiarare tutte le risorse e funzionalità che l'estensione intende utilizzare. Il file `manifest.json` dichiara inoltre i permessi che l'estensione necessita per funzionare correttamente. Alcuni tra i permessi più importanti, utilizzati anche nell'estensione sviluppata, sono:

  - **activeTab:** [8] consente all'estensione di interagire con la scheda attiva.
  - **tabs:** [9] consente all'estensione di interagire con tutte le schede del browser.
  - **storage:** [10] consente all'estensione di memorizzare dati localmente.
  - **scripting:** [11] consente all'estensione di iniettare script nelle pagine web.

- **Service Worker:** [12] in *manifest V3*, il concetto di background script viene sostituito dal service worker. Un service worker è uno script che viene eseguito in background, ma a differenza dei background script di V2, non rimane sempre attivo ma solo per un periodo limitato di tempo, migliorando così le prestazioni e il consumo di risorse. Il service worker può rispondere agli eventi generati dal browser o dall'utente, come il caricamento della pagina, le richieste di rete. A differenza dei content script, non ha accesso diretto al DOM delle pagine web, ma può comunicare con i content script attraverso i messaggi.

- **Content Script:** [13] i content script vengono eseguiti direttamente all'interno delle pagine web e possono interagire con il DOM. I content script presentano un attributo *world* che fornisce o meno l'accesso al contesto della pagina. *World* può assumere due valori:

  - **Isolated World:** gli script iniettati nel contesto isolato (isolated world) sono separati dagli script della pagina per garantire la sicurezza. Questo significa che variabili e funzioni definite in questi script non possono essere accedute dagli script della pagina e viceversa. Questa separazione previene che le estensioni interferiscano con la logica della pagina e protegge le informazioni sensibili.
  - **Main World:** Il main world è il contesto di esecuzione nativo della pagina web. Gli script eseguiti in questo contesto hanno pieno accesso al DOM e alle altre risorse della pagina. Tuttavia, per questioni di sicurezza, le estensioni di Chrome non possono eseguire direttamente script nel main world a meno che non siano espressamente autorizzati tramite la dichiarazione nel manifesto.

**Figura 3.4:** Logo Chrome Extensions

### 3.2.2 ShadowRoot

Il Shadow DOM è una tecnologia del web standardizzata dal W3C che consente di creare e allegare una sottostruttura del DOM, chiamata *shadow tree*, a un elemento del DOM principale. Questa tecnologia è parte integrante dei Web Components, che includono anche *Custom Elements* e HTML *Templates*. Il Shadow DOM consente agli sviluppatori di incapsulare il markup, lo stile e il comportamento di un componente web in un'unità indipendente e riutilizzabile.
Il Shadow Root è la radice del *shadow tree*, l'elemento fondamentale che consente la creazione e la gestione di un Shadow DOM. Quando si crea un shadow root, si allega un nuovo sottodocumento (*shadow tree*) a un elemento host nel DOM principale. Questo consente di isolare il contenuto e lo stile del componente dal resto della pagina, prevenendo conflitti di stile e di comportamento. La figura 3.5 mostra un esempio di struttura ShadowRoot.

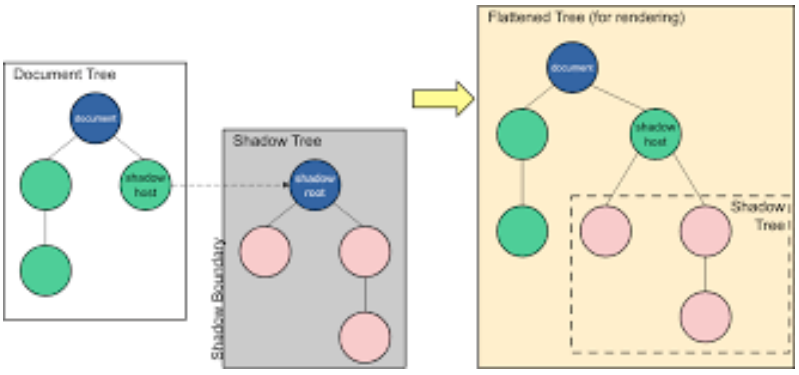


**Figura 3.5:** Esempio struttura shadowroot

### 3.2.3 Github

GitHub (logo presente in figura 3.6) è una piattaforma di hosting per lo sviluppo di software che utilizza *Git*, un sistema di controllo versione distribuito. Fondata nel 2008, GitHub è diventata rapidamente uno dei principali *hub* per la collaborazione tra sviluppatori, offrendo strumenti per il versionamento del codice, la gestione dei progetti e la collaborazione.

**Figura 3.6:** Logo Github

### 3.2.4 VSCode

Visual Studio Code (logo presente in figura 3.7), comunemente noto come VSCode, è un editor di codice sorgente sviluppato da Microsoft. Lanciato per la prima volta nel 2015, VSCode è diventato rapidamente uno degli strumenti più popolari tra gli sviluppatori grazie alla sua leggerezza, flessibilità e potenza. Disponibile gratuitamente e compatibile con Windows, macOS e Linux, VSCode offre un'ampia gamma di funzionalità che includono evidenziazione della sintassi, completamento automatico del codice, debug integrato e controllo di versione tramite *Git*.

**Figura 3.7:** Logo Visual Studio Code

### 3.2.5 Emmet

Emmet è un'estensione per Visual Studio Code che accelera significativamente il processo di scrittura di codice HTML e CSS. Originariamente noto come Zen Coding, Emmet è progettato per migliorare la produttività degli sviluppatori fornendo una serie di scorciatoie e snippet che consentono di creare rapidamente markup e stili complessi.

# Capitolo 4

# Sviluppo del progetto di stage

## 4.1 Analisi

Come precedentemente accennato, l'estensione ha il compito di analizzare la pagina web concentrandosi esclusivamente su *heading* e sulle immagini. Inizialmente, il piano di progetto prevedeva soltanto lo sviluppo dell'analisi delle immagini. Per questo motivo, comincerò introducendo questo aspetto.

### 4.1.1 Immagini

Il primo passo è stato identificare tutti gli aspetti che possono influenzare l'accessibilità di una pagina web. Secondo le direttive del W3C, le immagini in una pagina web possono essere classificate in sette tipologie differenti:

- **Immagini informative:** queste immagini rappresentano graficamente concetti e informazioni, come illustrazioni, foto e disegni. Il testo alternativo dovrebbe almeno fornire una breve descrizione che trasmetta le informazioni essenziali contenute nell'immagine. Questo è fondamentale per gli utenti con disabilità visive che utilizzano lettori di schermo.
- **Immagini decorative:** quando un'immagine ha l'unico scopo di aggiungere decorazione visiva alla pagina e non di trasmettere informazioni importanti, l'alternativa testuale dovrebbe essere nulla (`alt=""`). Questo evita di sovraccaricare i lettori di schermo con informazioni non necessarie, migliorando l'esperienza utente.
- **Immagini funzionali:** per immagini utilizzate come link o pulsanti, il testo alternativo dovrebbe descrivere la funzionalità del link o del pulsante piuttosto che l'immagine visiva. Esempi includono un'icona della stampante per rappresentare

la funzione di stampa o un pulsante per inviare un modulo. Questo aiuta gli utenti a comprendere l'azione che verrà eseguita cliccando sull'immagine.

- **Immagini testuali:** quando il testo è presentato all'interno di un'immagine, e se non si tratta di un logo, è preferibile evitare di utilizzare testo nelle immagini. Tuttavia, se si utilizzano immagini di testo, il testo alternativo dovrebbe contenere le stesse parole presenti nell'immagine. Questo assicura che l'informazione sia accessibile anche se l'immagine non viene caricata o visualizzata.
- **Immagini complesse:** per immagini che trasmettono dati o informazioni dettagliate, l'alternativa testuale dovrebbe fornire un equivalente completo dei dati o delle informazioni presenti nell'immagine. Esempi tipici sono grafici, diagrammi e mappe concettuali.
- **Gruppi di immagini:** se più immagini trasmettono un'unica informazione, il testo alternativo di una delle immagini dovrebbe trasmettere l'informazione per l'intero gruppo. Questo è utile, ad esempio, nelle gallerie fotografiche o nei collage di immagini.
- **Mappe:** per immagini contenenti più aree cliccabili, l'alternativa testuale dovrebbe fornire un contesto generale per l'insieme di link, e ogni area cliccabile dovrebbe avere un testo alternativo che descriva lo scopo o la destinazione del link. Questo è particolarmente rilevante per mappe interattive e diagrammi di navigazione.

Associare correttamente un'immagine alla categoria appropriata è un compito complesso che richiede indiscutibilmente l'intervento umano. Di conseguenza, non esiste una regola assoluta riguardo la presenza o l'assenza del testo alternativo; la decisione finale spetta a una persona. Pertanto, si è deciso di notificare attraverso un'avvertenza quando il testo alternativo è mancante, lasciando all'operatore la decisione finale.
Le linee guida del W3C descrivono unicamente le immagini inserite tramite *tag* HTML. Tuttavia, una pagina web può includere anche immagini tramite CSS. In questi casi, le immagini non hanno un attributo alt e sono sempre considerate immagini decorative. Questo avviene spesso con sfondi o elementi di design utilizzati per migliorare l'estetica della pagina senza aggiungere contenuto informativo.
Oltre al testo alternativo, l'estensione doveva analizzare anche la dimensione delle immagini e notificare quando queste fossero troppo pesanti. Analizzare la dimensione di un'immagine richiede di considerare sia la dimensione fisica che il tipo di immagine. Una soglia statica non è adeguata poiché potrebbe generare avvertenze per immagini molto grandi ma ben compresse e non generarle per immagini piccole ma con scarsa compressione. Pertanto, è stato necessario progettare un algoritmo specifico per il calcolo della soglia di allerta.
L'algoritmo doveva tenere conto di vari fattori, come la risoluzione dell'immagine,

il formato (Joint Photographic Experts Group (JPEG), Portable Network Graphics (PNG), Graphics Interchange Format (GIF), etc.) e il livello di compressione applicato. Ad esempio, un'immagine PNG non compressa può essere significativamente più pesante di una JPEG con alta compressione, pur avendo la stessa risoluzione. Il sistema notifica all'utente quando un'immagine supera la soglia calcolata.
L'immagine 4.1 mostra la *tab* dei risultati dell'analisi delle immagini identificando il numero totale di errori ed avvertenza, inoltre fornisce la possibilità di applicare filtri alla navigazione. L'immagine 4.2 mostra la *tab* della lista di immagini identificate, mentre l'immagine 4.3 mostra il focus sulle informazioni di un'immagine.

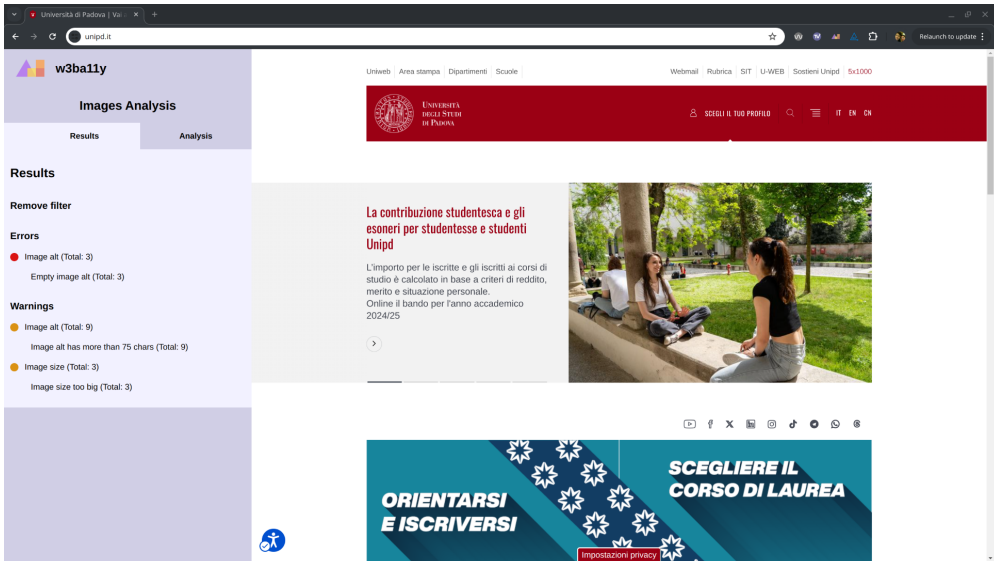


**Figura 4.1:** Estensione: tab dei risultati analisi delle immagini

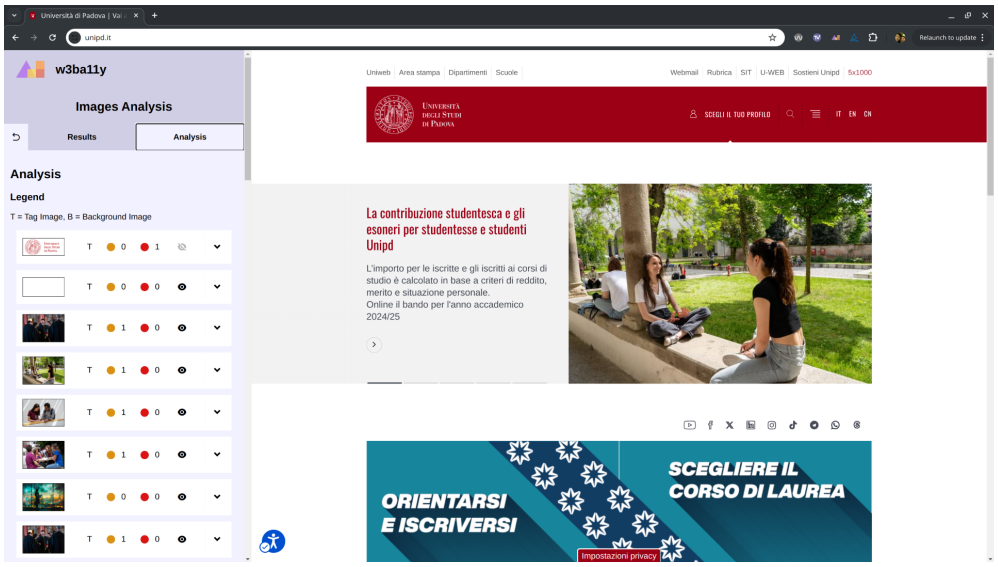


**Figura 4.2:** Estensione: tab della lista di immagini identificate

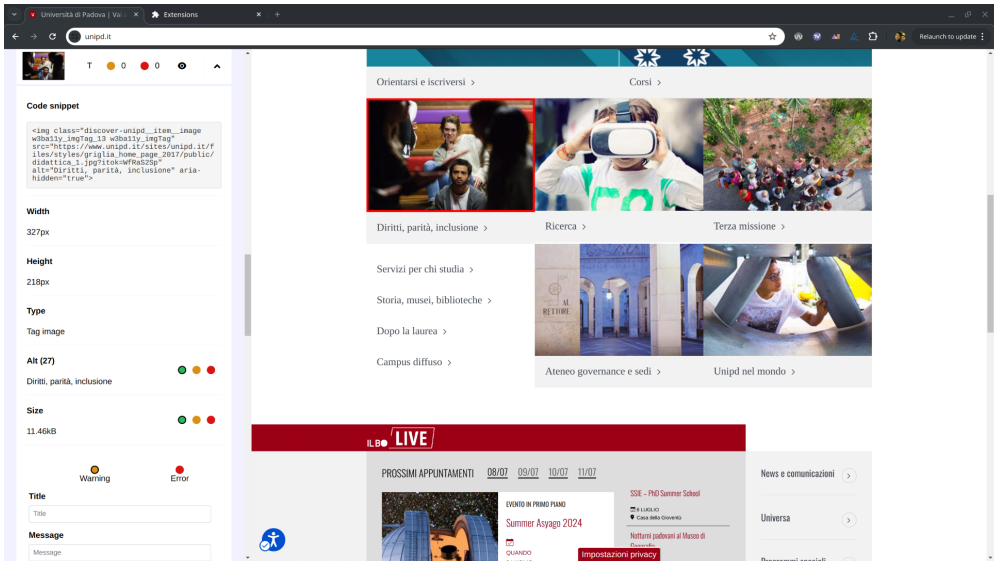


**Figura 4.3:** Estensione: focus sulle informazioni di un'immagine

### 4.1.2 Headings

Gli heading svolgono un ruolo fondamentale nell'accessibilità e nell'organizzazione delle pagine web. Permettono di strutturare il contenuto in modo gerarchico, facilitano la navigazione e la comprensione del testo sia per gli utenti umani che per i motori di ricerca e lettori di schermo.
Il loro corretto utilizzo è essenziale per diverse ragioni:

- **Accessibilità:** i lettori di schermo utilizzano gli *heading* per aiutare gli utenti non vedenti o ipovedenti a navigare attraverso i contenuti della pagina. Una struttura ben definita degli *heading* consente di saltare rapidamente tra le sezioni e trovare le informazioni desiderate.
- **SEO:** i motori di ricerca utilizzano gli *heading* per comprendere la struttura e il contenuto della pagina. Una corretta gerarchia di *heading* aiuta a migliorare il posizionamento nei risultati di ricerca.
- **Usabilità:** gli *heading* aiutano tutti gli utenti a scansionare rapidamente il contenuto della pagina, identificando le sezioni principali e le sottosezioni. Questo migliora l'esperienza utente, rendendo il contenuto più facilmente fruibile.

Secondo le linee guida del W3C, gli *heading* dovrebbero essere utilizzati in maniera semantica e gerarchica, rispettando alcune best practices:

- **Gerarchia:** utilizzare gli *heading* (da `<h1>` a `<h6>`) in maniera gerarchica, dove `<h1>` rappresenta il titolo principale della pagina e `<h2>` le sezioni principali, con livelli successivi di *heading* (`<h3>`, `<h4>`, etc.) per sottosezioni.
- **Chiarezza:** gli *heading* dovrebbero essere chiari e descrittivi, fornendo un'indicazione precisa del contenuto della sezione che introducono. Questo aiuta sia gli utenti sia i motori di ricerca a comprendere rapidamente il contenuto della pagina.
- **Coerenza:** mantenere una struttura coerente degli *heading* in tutto il sito web migliora la navigazione e l'usabilità. Gli utenti dovrebbero essere in grado di prevedere la struttura dei contenuti basandosi sull'esperienza con altre pagine del sito.

Definire la chiarezza e la coerenza degli *heading* rispetto al contesto della pagina è un compito che attualmente non può essere eseguito da una macchina. Di conseguenza, questi due aspetti devono essere gestiti manualmente da un essere umano e non sono state implementate analisi automatiche in merito.
L'unica analisi automatica implementata riguarda il controllo della gerarchia degli *heading*. Se viene identificato un heading il cui livello non rispetta la gerarchia corretta, esso verrà contrassegnato come erroneo.
La figura 4.4 mostra la *tab* dei risultati dell'analisi *heading*.

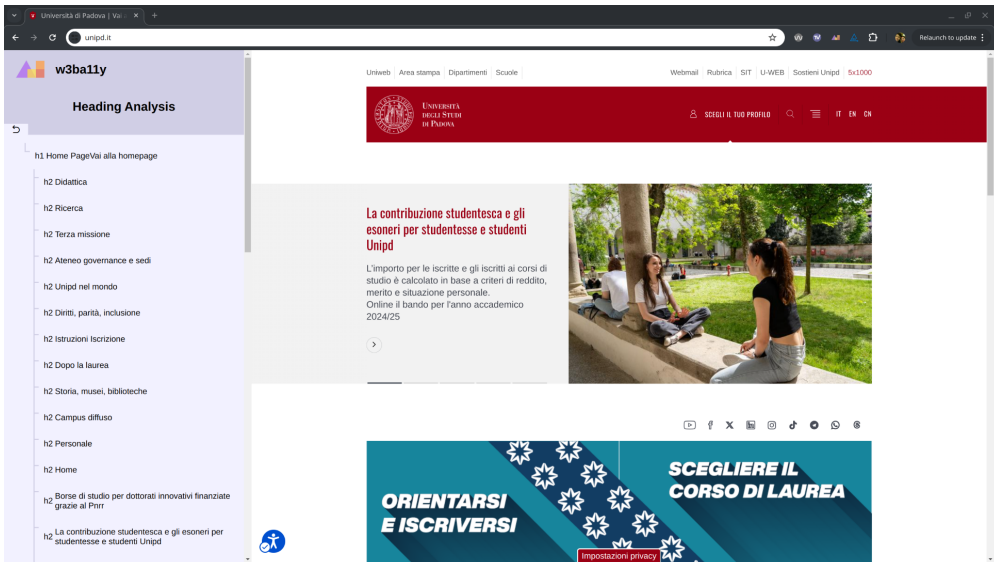


**Figura 4.4:** Estensione analisi heading

## 4.2 Casi d'uso

### Attori

Il primo passo per la definizione dei casi d'uso è l'identificazione degli attori coinvolti nell'utilizzo dell'estensione. In questo caso, considerando che l'estensione ha il compito di svolgere un'analisi statica di una pagina web, l'unico attore coinvolto è l'esperto di accessibilità. Questo attore è responsabile dell'analisi e della valutazione di siti web e desidera garantire che queste siano accessibili a tutti gli utenti, indipendentemente dalle loro capacità fisiche o cognitive.

### UC0: Navigazione sezioni principali

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore ha avviato l'estensione.

**Descrizione:** L'estensione offre la possibilità di scegliere tra l'analisi delle immagini e l'analisi *heading*.

**Postcondizioni:** L'estensione è pronto per visualizzare i risultati dell'analisi scelta.

### UC1.1: Visualizzazione dei risultati dell'analisi delle immagini

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore è nella sezione analisi delle immagini.

**Descrizione:** L'estensione elabora la visualizzazione dei risultati dell'analisi delle immagini.

**Postcondizioni:** L'estensione visualizza la pagina dei risultati.

### UC1.2.1: Visualizzazione della lista dell'analisi delle immagini

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza la scheda risultati nella sezione analisi delle immagini e naviga verso la scheda analisi immagini.

**Descrizione:** L'estensione elabora la visualizzazione della prima pagina della lista delle immagini.

**Postcondizioni:** L'estensione visualizza la scheda della lista delle immagini.

### UC1.2.2: Visualizzazione informazioni di un'immagine

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza la scheda analisi immagini nella sezione analisi delle immagini e seleziona un'immagine.

**Descrizione:** L'estensione elabora la visualizzazione delle informazioni raccolte riguardo l'immagine selezionata.

**Postcondizioni:** L'estensione visualizza le informazioni dell'immagine selezionata.

### UC1.3: Visualizzazione dei risultati dell'analisi *heading*

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore ha la sezione analisi *heading*.

**Descrizione:** L'estensione elabora la visualizzazione dei risultati dell'analisi *heading*.

**Postcondizioni:** L'estensione visualizza la pagina dei risultati.

### UC2: Applicazione filtro alla lista dell'analisi delle immagini

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza la scheda risultati nella sezione analisi delle immagini ed ha selezionato un filtro.

**Descrizione:** L'estensione elabora la visualizzazione della prima pagina della lista delle immagini applicando il filtro selezionato.

**Scenario principale:**

- L'operatore filtra le immagini per tipologia dello status;
- L'operatore filtra le immagini per titolo dello status;
- L'operatore filtra le immagini per messaggio dello status.

**Postcondizioni:** L'estensione applica il filtro e visualizza la scheda della lista delle immagini.

### UC3: Rimozione filtro dalla lista dell'analisi delle immagini

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza la scheda risultati nella sezione analisi delle immagini ed ha rimosso un filtro.

**Descrizione:** L'estensione elabora la visualizzazione della prima pagina della lista delle immagini rimuove il filtro precedentemente attivo.

**Postcondizioni:** L'estensione rimuove il filtro visualizza la scheda della lista delle immagini.

### UC4: Navigazione della lista dell'analisi delle immagini

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza la scheda analisi immagini nella sezione analisi delle immagini e naviga verso un'altra pagina.

**Descrizione:** L'estensione elabora la visualizzazione della nuova pagina della scheda analisi immagini.

**Postcondizioni:** L'estensione visualizza la nuova pagina della scheda della lista delle immagini.

### UC5: Evidenzia immagine

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza la scheda analisi immagini nella sezione analisi delle immagini e seleziona evidenzia un'immagine.

**Descrizione:** L'estensione scorre la *shadow root* fino a portare l'immagine selezionata a schermo e ne evidenzia i bordi.

**Postcondizioni:** L'estensione evidenzia l'immagine selezionata.

### UC6: Modifica informazioni di un'immagine

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza le informazioni di un'immagine e seleziona l'informazione da modificare.

**Descrizione:** L'estensione modifica l'informazione selezionata.

**Scenario principale:**

- L'operatore modifica lo status dell'alt;
- L'operatore modifica lo status della dimensione;

**Postcondizioni:** L'estensione aggiorna la visualizzazione delle informazioni dell'immagine.

### UC7: Aggiunta informazioni ad un'immagine

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza le informazioni di un'immagine ed inserisce le informazioni che desidera salvare.

**Descrizione:** L'estensione aggiunge le informazioni inserite.

**Postcondizioni:** L'estensione aggiorna la visualizzazione delle informazioni dell'immagine.

### UC7.1: Aggiunta informazioni incomplete ad un'immagine

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza le informazioni di un'immagine e non inserisce tutte le informazioni richieste.

**Descrizione:** L'estensione controlla le informazioni inserite e blocca l'aggiornamento.

**Postcondizioni:** L'estensione rimuove i campi inseriti senza inserire le informazioni richieste.

### UC8: Rimozione informazioni da un'immagine

**Attori Principali:** Esperto di accessibilità.

**Precondizioni:** L'operatore visualizza le informazioni di un'immagine ed seleziona l'informazioni che desidera rimuovere.

**Descrizione:** L'estensione rimuove l'informazioni selezionata.

**Postcondizioni:** L'estensione aggiorna la visualizzazione delle informazioni dell'immagine.

## 4.3 Tracciamento dei requisiti

Da un'attenta analisi dei requisiti e degli use case effettuata sul progetto è stata stilata la tabella che traccia i requisiti in rapporto agli use case.
Sono stati individuati diversi tipi di requisiti e si è quindi fatto utilizzo di un codice identificativo per distinguerli.
Il codice dei requisiti è così strutturato R(F/Q/V)(N/D/O) dove:

R = requisito

F = funzionale

Q = qualitativo

V = di vincolo

N = obbligatorio (necessario)

D = desiderabile

Z = opzionale

Nelle tabelle 4.1, 4.2 e 4.3 sono riassunti i requisiti e il loro tracciamento con gli use case delineati in fase di analisi.

**Tabella 4.1:** Tabella del tracciamento dei requisti funzionali

| Requisito | Descrizione | Use Case |
|---|---|---|
| RFN-1 | L'estensione deve visualizzare l'elenco delle sezioni di analisi disponibili (immagini e intestazioni) | UC0 |
| RFN-2 | L'estensione deve elaborare e visualizzare il totale di errori e avvertenze identificati nell'analisi delle immagini | UC1.1 |
| RFN-3 | L'estensione deve elaborare e visualizzare la tipologia degli errori e delle avvertenze identificati nell'analisi delle immagini | UC1.1 |
| RFN-4 | L'estensione deve elaborare e visualizzare la lista delle immagini analizzate | UC1.2.1 |
| RFN-5 | L'estensione deve visualizzare il totale degli errori e delle avvertenze per ogni immagine della lista | UC1.2.1 |
| RFN-6 | L'estensione deve visualizzare la visibilità a schermo per ogni immagine della lista | UC1.2.1 |
| RFN-7 | L'estensione deve visualizzare le informazioni specifiche di una singola immagine selezionata | UC1.2.2 |
| RFN-8 | L'estensione deve permettere di nascondere le informazioni specifiche di una singola immagine | UC1.2.2 |
| RFN-9 | L'estensione deve elaborare e visualizzare i risultati dell'analisi delle intestazioni (heading) | UC1.3 |
| RFN-10 | L'estensione deve permettere di filtrare le immagini analizzate per tipo dello status | UC2 |
| RFN-11 | L'estensione deve permettere di filtrare le immagini analizzate per titolo dello status | UC2 |
| RFN-12 | L'estensione deve permettere di filtrare le immagini analizzate per messaggio dello status | UC2 |
| RFN-13 | L'estensione deve notificare all'operatore l'applicazione del filtro e visualizzare la lista delle immagini filtrate | UC2 |
| RFN-14 | L'estensione deve permettere la rimozione del filtro applicato alla lista delle immagini analizzate | UC3 |
| RFN-15 | L'estensione deve notificare all'operatore la rimozione del filtro e visualizzare la lista delle immagini senza il filtro | UC3 |
| RFN-16 | L'estensione deve permettere la navigazione tra le varie pagine della lista delle immagini analizzate | UC4 |
| RFN-17 | L'estensione deve permettere l'evidenziazione nella pagina web dell'immagine selezionata, scorrendo fino a essa ed evidenziandone i bordi | UC4 |
| RFN-18 | L'estensione deve permettere la modifica dell'analisi dell'attributo alt effettuata su un'immagine | UC5 |
| RFN-19 | L'estensione deve permettere la modifica dell'analisi della dimensione effettuata su un'immagine | UC5 |
| RFN-20 | L'estensione deve permettere l'aggiunta di informazioni aggiuntive per un'immagine | UC6 |
| RFN-21 | L'estensione deve bloccare l'aggiunta di informazioni per un'immagine se queste risultano incomplete | UC6.1 |
| RFN-22 | L'estensione deve permettere la rimozione delle informazioni aggiunte a un'immagine | UC7 |

**Tabella 4.2:** Tabella del tracciamento dei requisiti qualitativi

| Requisito | Descrizione | Use Case |
|---|---|---|
| RQN-1 | L'estensione deve permettere la navigazione interna tramite tasto TAB | - |
| RQN-2 | L'estensione deve rispettare le linee guida WCAG riguardo il contrasto dei colori | - |
| RQD-3 | L'estensione deve permettere la navigazione verso nuove pagine senza dover essere costantemente riattivata | - |

**Tabella 4.3:** Tabella del tracciamento dei requisiti di vincolo

| Requisito | Descrizione | Use Case |
|---|---|---|
| RVZ-1 | L'estensione deve essere sviluppata utilizzando Manifest V3 per essere disponibile sullo store di Chrome | - |

## 4.4 User Story

**Tabella 4.4:** Tabella delle user story identificate

| N° | User story |
|---|---|
| 1 | Come esperto di accessibilità, voglio leggere il testo alternativo di un'immagine, in modo da valutare se la descriva correttamente. |
| 2 | Come esperto di accessibilità, voglio sapere la dimensione di un'immagine, in modo da capire se la dimensione troppo elevata possa causare rallentamenti eccessivi al caricamento della pagina. |
| 3 | Come esperto di accessibilità, voglio poter modificare gli errori o le avvertenze identificate dall'estensione, in modo da correggere dei possibili falsi positivi. |
| 4 | Come esperto di accessibilità, voglio avere accesso ad uno snippet del codice del *tag* in questione, in modo da poterlo cercare nel codice sorgente della pagina nel caso in cui l'immagine non sia presente a schermo. |
| 5 | Come esperto di accessibilità, voglio visualizzare l'immagine che sto analizzando, in modo da capire la sua pertinenza e contestualizzarla rispetto al testo presente. |
| 6 | Come esperto di accessibilità, voglio annotare problematiche particolari che ad ora non sono supportate dall'estensione, in modo da mantenere traccia di errori di accessibilità poco comuni. |
| 7 | Come esperto di accessibilità, voglio report dettagliati sulle problematiche di accessibilità riscontrate, in modo da facilitare la valutazione finale al termine dell'analisi. |
| 8 | Come esperto di accessibilità, voglio filtrare gli errori e le avvertenze, in modo da poterle analizzare in maniera più efficiente. |
| 9 | Come esperto di accessibilità, voglio che l'estensione sia accessibile, in modo che possa essere utilizzata da chiunque. |

## 4.5 Sviluppo

### 4.5.1 PoC

L'obiettivo iniziale era sviluppare un *Proof of Concept (POC)*[g] che consentisse di testare nuove tecnologie e dimostrare la fattibilità del prodotto. Questa versione iniziale non implementava *design pattern* né funzionalità avanzate, ma si limitava all'identificazione delle immagini presenti nella pagina. Durante la fase di progettazione e sviluppo del POC, non sono stati considerati esplicitamente elementi come l'estendibilità del

prodotto finale o la possibilità di integrarlo con altre tecnologie.
Il POC presentava una struttura base composta da due file principali:

- **background.js (service worker):** gestiva l'avvio dell'estensione al click dell'icona.
- **main.js (content script):** iniettava del nuovo codice HTML nel DOM della pagina ed avviava il modulo di analisi delle immagini.

A questa struttura di base si aggiungeva un modulo *img* composto da:

- **base.js:** avviava la ricerca delle immagini non appena riceveva l'ok da `main.js` e costruiva gli oggetti *Img*.
- **finder.js:** identificava tutte le immagini presenti nella pagina.
- **img.js:** definiva la classe *Img*.
- **interface.js:** iniettava il codice della nuova interfaccia al completamento dell'analisi delle immagini.

### 4.5.2 Prodotto finale

#### Architettura estensione

Come precedentemente accennato, nell'ambito di uno stage della durata di 300 ore, non era possibile sviluppare un prodotto che analizzasse tutti gli aspetti dell'accessibilità. Si è quindi deciso di concentrarsi su due campi specifici durante lo sviluppo, mantenendo però un'attenzione particolare alla progettazione di un'estensione che potesse essere facilmente estendibile.
L'architettura dell'estensione si ispira alla struttura dei microservizi, una rappresentazione grafica è mostrata in figura 4.5. L'architettura a microservizi è un modello di progettazione software che suddivide un'applicazione in una serie di servizi piccoli e indipendenti, ognuno dei quali esegue un compito specifico. Questi servizi comunicano tra loro attraverso interfacce ben definite e possono essere sviluppati, distribuiti e scalati in modo indipendente.

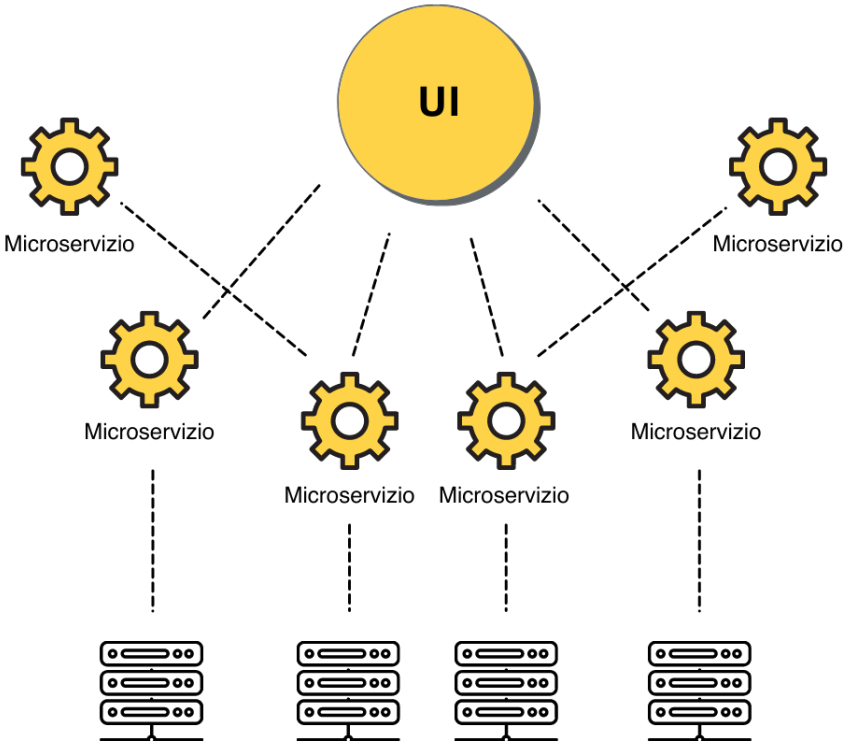


**Figura 4.5:** Esempio di architettura a microservizi

- **Modularità e indipendenza:**
  - **Estensione:** l'estensione è composta da un *core* principale e da moduli indipendenti che si occupano di compiti specifici, come l'analisi degli headings e delle immagini. Questi moduli operano in modo autonomo senza interagire direttamente tra loro.
  - **Microservizi:** nell'architettura a microservizi, ogni servizio è progettato per essere indipendente e focalizzato su una singola funzione. Questo consente di sviluppare e aggiornare ciascun servizio senza influenzare gli altri.
- **Comunicazione con il Core:**
  - **Estensione:** i moduli comunicano con il *core* principale tramite le Chrome API, notificandogli il completamento della loro esecuzione. Questo *core* funge da coordinatore, gestendo l'integrazione dei risultati provenienti dai vari moduli.
  - **Microservizi:** nei microservizi, i servizi comunicano attraverso interfacce ben definite, spesso tramite API *Representational State Transfer (REST)*[g]o messaggistica. Un servizio di orchestrazione o un gateway API può coordinare queste comunicazioni.

- **Facilità di Estensione e Scalabilità:**
  - **Estensione:** la progettazione dell'estensione permette di aggiungere facilmente nuovi moduli per analizzare altri aspetti dell'accessibilità senza necessitare di modifiche ai moduli esistenti.
  - **Microservizi:** uno dei principali vantaggi dei microservizi è la possibilità di aggiungere nuovi servizi o scalare quelli esistenti in modo indipendente, facilitando l'evoluzione e l'espansione dell'applicazione.
- **Sviluppo e Manutenzione Indipendenti:**
  - **Estensione:** ogni modulo può essere sviluppato, testato e manutenuto indipendentemente dagli altri, consentendo una gestione più efficiente del codice e facilitando il lavoro di team diversi.
  - **Microservizi:** analogamente, i microservizi permettono a diversi team di lavorare su servizi separati, accelerando lo sviluppo e semplificando la manutenzione del sistema.

Attualmente, l'estensione include un modulo per l'analisi *heading* e un modulo per l'analisi delle immagini. Questi moduli operano autonomamente, comunicando unicamente con il *core* principale per segnalare il completamento delle loro rispettive analisi. Questo approccio modulare e indipendente, ispirato all'architettura a microservizi, garantisce la facilità di estensione e manutenzione, permettendo di aggiungere nuove funzionalità in futuro senza compromettere l'integrità del sistema esistente.

### Architettura moduli

I singoli moduli sono stati sviluppati adottando il *design pattern* (in figura 4.6 la rappresentazione grafica), che separa un'applicazione in tre componenti principali, ciascuna con responsabilità ben definite. Questa suddivisione migliora la modularità e la gestione del codice, facilitando lo sviluppo, la manutenzione e l'estendibilità del sistema.

- **Modello:** è responsabile della gestione dei dati e della logica di business dell'applicazione. Il modello si occupa dell'interazione con il database o con altre fonti di dati, garantendo la coerenza e l'integrità delle informazioni gestite.
- **Vista:** è incaricata della presentazione grafica dei dati. Si occupa di visualizzare le informazioni provenienti dal modello all'utente finale. La vista si aggiorna dinamicamente in base alle modifiche dei dati, offrendo un'interfaccia utente intuitiva e reattiva.
- **Controller:** funge da intermediario tra il modello e la vista. Gestisce le interazioni dell'utente, elabora le richieste e aggiorna il modello di conseguenza.

Inoltre, il controller aggiorna la vista in base alle modifiche avvenute nel modello, garantendo una comunicazione sincronizzata tra le componenti.

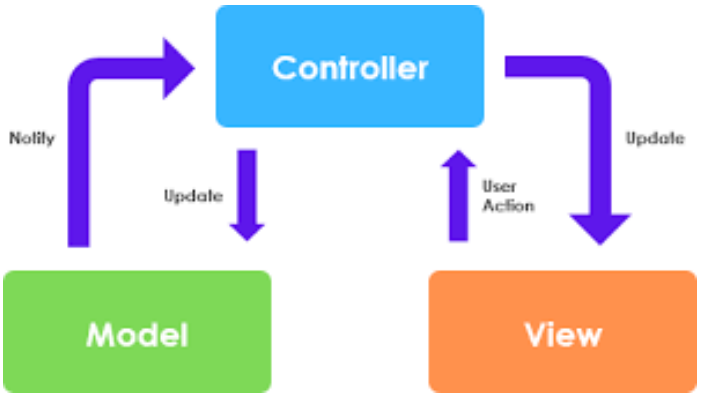


**Figura 4.6:** Esempio di MVC

### Modulo analisi immagini

Alla ricezione del messaggio di avvio, lo script `main.js` inizializza il controller. Il costruttore del controller si occupa di inizializzare la vista e di avviare l'analisi della pagina per identificare tutte le immagini presenti. Il metodo di ricerca delle immagini è asincrono e ritorna una *Promise*[g]. Alla risoluzione della Promise, ovvero quando tutte le immagini sono state trovate, il modello viene popolato con oggetti di tipo *Img*. La classe *Img*, creata appositamente, utilizza oggetti *Status*, anch'essi sviluppati ad hoc per tenere traccia di errori e avvertenze, e ha il compito di mantenere in memoria tutte le informazioni relative alle immagini.
Al termine del popolamento del modello, viene avviato il processo di *rendering* delle informazioni raccolte. Questo processo prevede l'aggiornamento dell'interfaccia base creata dal *core* dell'estensione, attraverso l'aggiunta della sezione dedicata all'analisi delle immagini. Questa sezione si compone di due tab: una di riepilogo, che tiene traccia di tutti gli errori, e un'altra per la visualizzazione di tutte le immagini identificate e analizzate.
Dopo il *rendering* della vista, il controller genera i vari *event listener*[g]e imposta le azioni da eseguire quando questi vengono attivati.

### Modulo analisi headings

Questo modulo è decisamente più semplice rispetto al precedente. Lo script `main.js` inizializza il controller, il cui costruttore si occupa di configurare la vista e di avviare la ricerca di tutti gli headings presenti nella pagina. A differenza del modulo precedente, la ricerca non restituisce una Promise, ma crea direttamente gli oggetti *H* con cui viene popolato il modello.
Una volta completato il popolamento del modello, viene avviato il processo di *rendering*

che aggiunge all'interfaccia base la sezione dedicata all'analisi *heading*. Al termine del *rendering*, il controller genera i vari event listener e configura le azioni da eseguire quando questi vengono attivati.

## 4.6 Problematiche riscontrate

Durante lo sviluppo del plugin per l'analisi dell'accessibilità di una pagina web, sono emerse due problematiche principali. La prima, già menzionata, riguarda la determinazione della soglia di dimensione delle immagini, mentre la seconda concerne il calcolo della dimensione effettiva di queste ultime.

### 4.6.1 Algoritmo di calcolo soglia limite

Le immagini presenti in una pagina web possono variare significativamente in termini di dimensioni, da pochi pixel a diverse migliaia. Utilizzare una soglia statica per valutare se un'immagine è troppo pesante può portare a risultati non ottimali. Infatti, una soglia troppo bassa penalizzerebbe l'uso di immagini di grandi dimensioni che, grazie a un'ottima compressione, risultano leggere. Al contrario, una soglia troppo alta permetterebbe l'uso di immagini di piccole dimensioni ma con livelli di compressione inadeguati.
Un altro fattore cruciale da considerare è il formato dell'immagine. Formati come `.jpg` e `.jpeg`, in generale, sono significativamente meno pesanti rispetto a formati come `.png`. La ragione di ciò risiede nel modo in cui questi formati gestiscono la compressione.
I file `.jpg` e `.jpeg` utilizzano una compressione con perdita di dati (*lossy compression*[g]), che riduce notevolmente le dimensioni del file sacrificando una parte della qualità dell'immagine. Questo tipo di compressione elimina alcune informazioni visive che l'occhio umano difficilmente nota, rendendo i file molto più leggeri.
D'altro canto, i file `.png` adottano una compressione senza perdita di dati (*lossless compression*[g]). Questo metodo mantiene intatta la qualità dell'immagine, ma comporta dimensioni di file maggiori poiché nessuna informazione viene eliminata durante il processo di compressione. Di conseguenza, i file `.png` tendono ad essere più pesanti rispetto ai file `.jpg` e `.jpeg`, rendendoli meno adatti in contesti dove la riduzione delle dimensioni è una priorità.
L'algoritmo sviluppato considera sia il formato che la dimensione dell'immagine per calcolare una soglia entro cui la dimensione dell'immagine può essere accettata. In questo modo, l'algoritmo evita di discriminare le immagini basandosi esclusivamente sulla loro dimensione. Tuttavia, prima di eseguire l'algoritmo, è stata impostata una soglia statica di 1 MB per avvisare l'utente nel caso in cui l'immagine sia eccessivamente

grande per una pagina web.
L'immagine 4.7 mostra il codice della funzione `estimatedMemorySize`.

```
109
110   estimatedMemorySize() {
111     const ext = this.src.split('.').pop().toLowerCase();
112     let bytesPerPixel;
113     switch (ext) {
114       case 'jpg':
115       case 'jpeg':
116         bytesPerPixel = 1.5;
117         break;
118       case 'png':
119         bytesPerPixel = 3;
120         break;
121       case 'gif':
122         bytesPerPixel = 1;
123         break;
124       default:
125         bytesPerPixel = 4;
126     }
127     return Math.ceil(this.width * this.height * bytesPerPixel / 1024);
128   }
129
130   addCustomStatus(status) {
131     this.customStatus.push(status);
132   }
133
```

**Figura 4.7:** Code snippet dell'algoritmo di calcolo della soglia limite

### 4.6.2 Calcolo dimensione immagine

La seconda problematica riscontrata era relativa al calcolo della dimensione delle immagini identificate. Attualmente, JavaScript non offre un supporto nativo per il calcolo della dimensione dell'immagine. In un primo momento, si è pensato di utilizzare l'algoritmo di calcolo della soglia limite per approssimare la dimensione dell'immagine. Tuttavia, senza conoscere il livello di compressione dell'immagine, il calcolo risultava eccessivamente approssimato.
La prima soluzione identificata è stata quella di utilizzare le Performance API.

#### Performance API

Le Performance API forniscono un'interfaccia per accedere ai dati di temporizzazione e prestazioni delle risorse caricate in una pagina web. Queste API permettono di misurare il tempo di caricamento delle risorse, il tempo di risposta del server, e altre metriche utili per l'analisi delle performance del sito. Utilizzando queste API, è possibile ottenere informazioni sulle dimensioni delle risorse caricate, incluse le immagini, analizzando i dati di caricamento.
Tuttavia, questa soluzione si è dimostrata limitata poiché non tutte le immagini

risultavano presenti nei dati raccolti dalle Performance API. Questo accade perché le Performance API registrano solo le risorse caricate tramite il browser durante la sessione corrente. Se un'immagine è stata caricata precedentemente e viene recuperata dalla cache del browser, potrebbe non essere inclusa nei dati di prestazione raccolti. Di conseguenza, le dimensioni di alcune immagini potrebbero non essere rilevate correttamente, limitando l'efficacia di questa soluzione.

### Fetch dell'immagine

La seconda soluzione proposta per calcolare la dimensione delle immagini consiste nell'uso della funzione `fetchImageSize`, che invia una richiesta *Hypertext Transfer Protocol (HTTP)*[g]di tipo *HEAD*[g]per ottenere solo gli header della risorsa, anziché il suo contenuto completo. In questo modo, è possibile leggere l'header "Content-Length" che indica la dimensione del file. Se la risposta non è positiva, viene inviata una richiesta *GET*[g]completa all'*Uniform Resource Locator (URL)*[g]dell'immagine. Se anche questa richiesta fallisce, viene inviato un messaggio con la dimensione impostata a 0.
L'immagine 4.8 mostra il codice della funzione `fetchImageSize`.

```
60  function fetchImageSize(src, tabId) {
61    fetch(src, {
62      method: 'HEAD'
63    })
64    .then((res) => {
65      if (!res.ok) {
66        fetch(src).then(res2 => {
67          if (!res2.ok)
68            chrome.tabs.sendMessage(tabId, {
69              action: "fetchImageSizeResponse",
70              src: src,
71              size: 0
72            });
73          else
74            chrome.tabs.sendMessage(tabId, {
75              action: "fetchImageSizeResponse",
76              src: src,
77              size: res2.headers.get("Content-Length")
78            });
79        });
80      } else
81        chrome.tabs.sendMessage(tabId, {
82          action: "fetchImageSizeResponse",
83          src: src,
84          size: res.headers.get("Content-Length")
85        });
86    })
87    .catch(function (error) {
88      chrome.tabs.sendMessage(tabId, {
89        action: "fetchImageSizeResponse",
90        src: src,
91        size: 0
92      });
93    });
94  }
95
```

**Figura 4.8:** Code snippet della funzione di fetch dell'immagine

Questa seconda soluzione introduceva nuove problematiche relative alle *Content Security Policies (CSP)*[g].
Quando la funzione fetchImageSize viene eseguita direttamente all'interno della pagina web, le richieste HEAD e GET effettuate verso domini esterni potrebbero essere bloccate dalle direttive CSP. Questo accade perché le politiche di sicurezza del sito possono impedire il caricamento di risorse da domini non autorizzati, causando il fallimento delle richieste di fetch e quindi l'impossibilità di ottenere la dimensione delle immagini.
Per ovviare a questo inconveniente, la funzione è stata spostata nel *service worker* `background.js`. Spostando la funzione fetchImageSize nel *service worker* `background.js`, si può superare il problema delle restrizioni CSP, garantendo che tutte le richieste di rete necessarie per calcolare la dimensione delle immagini vengano eseguite correttamente e senza interferenze.

### 4.6.3 Shadow Root

Durante lo sviluppo dell'estensione per Chrome, è emerso un ulteriore problema legato alle CSP. All'avvio dell'estensione, l'intera pagina web veniva clonata all'interno di uno *shadow root* per garantire una separazione netta tra la struttura del plugin e quella della pagina.
L'utilizzo di uno *shadow root* offre diversi vantaggi significativi:

- **Isolamento degli Stili:** una *shadow root* crea un ambiente isolato, impedendo alle regole CSS dell'estensione di influenzare gli stili della pagina web originale e viceversa. Questo isolamento garantisce che il design e la funzionalità della pagina non vengano alterati dall'estensione e che le modifiche apportate dal plugin non vengano sovrascritte dagli stili della pagina.
- **Miglioramento della Sicurezza:** l'isolamento fornito dallo *shadow root* riduce il rischio di conflitti tra gli script del plugin e quelli della pagina, contribuendo a mantenere un livello di sicurezza più elevato.

Tuttavia, la clonazione della struttura del DOM e il suo inserimento all'interno dello *shadow root* generava un errore CSP. Gli script inline e altre risorse JavaScript presenti nella struttura clonata violavano le direttive di sicurezza impostate dal sito. La CSP specificava che solo gli script provenienti da determinate fonti erano consentiti, escludendo gli script inline a meno che non fossero accompagnati da un *hash*[g]o un *nonce*[g]sicuro. Di conseguenza, ogni tentativo di eseguire questi script veniva bloccato, impedendo il corretto funzionamento dell'estensione.

### Iframe

Per aggirare questo problema, è stato utilizzato un `iframe` all'interno dello *shadow root* ed è stata caricata la pagina desiderata all'interno dell'`iframe`. Questa soluzione ha funzionato poiché:

- **Isolamento del Contesto di Esecuzione:** un `iframe` crea un nuovo contesto di navigazione che è isolato dalla pagina principale. Ciò significa che le direttive CSP applicate alla pagina principale non influenzano il contenuto caricato all'interno dell'`iframe`.
- **Nessuna Restrizione sugli Script Inline:** poiché il contenuto viene caricato da un URL all'interno dell'`iframe`, gli script inline e altre risorse JavaScript non sono soggetti alle stesse restrizioni della CSP della pagina principale.

Questa soluzione ha permesso di superare le limitazioni imposte dalla CSP, garantendo il corretto funzionamento dell'estensione senza generare errori di sicurezza.

# Capitolo 5

# Verifica e validazione

## 5.1 Verifica e validazione del codice

La verifica del corretto funzionamento dell'estensione si è concentrata principalmente sull'aspetto più complesso: l'identificazione delle immagini. Questo passaggio specifico ha rappresentato una delle sfide più significative per vari motivi dettagliati di seguito.

- **Tag HTML:** In HTML, i *tag* che permettono di inserire immagini sono molteplici. È stato quindi necessario analizzare la pagina alla ricerca di vari elementi, tra cui:
  - **img:** Questo è il *tag* di default per l'inserimento delle immagini ed è ampiamente utilizzato.
  - **picture:** Questo *tag* è utilizzato per inserire immagini responsive e consente di specificare diverse versioni di un'immagine per diverse condizioni di visualizzazione.
  - **source:** Utilizzato all'interno del *tag* `picture`, permette di definire le immagini da visualizzare in base alla risoluzione del dispositivo, migliorando così l'ottimizzazione e l'adattabilità delle immagini.
- **Lazy Loading:** Molti siti web utilizzano la tecnica del lazy loading per evitare che il caricamento delle immagini rallenti la visualizzazione della pagina. Questa tecnica consiste nel caricare le immagini solo quando diventano visibili all'utente, il che significa che inizialmente le immagini non sono presenti nel codice HTML della pagina ma vengono aggiunte successivamente tramite JavaScript. Questo ha complicato l'identificazione delle immagini, poiché i *tag* `img` spesso non contenevano i valori degli attributi `src` e `alt` oppure avevano placeholder che non permettevano di identificare correttamente l'immagine. La soluzione a questo problema è stata l'analisi degli attributi *data*, cercando URL all'interno di essi che indicassero la presenza di immagini.

- **CSS:** Le immagini possono essere inserite anche tramite CSS, in particolare utilizzando la proprietà `background-image`. Questo ha richiesto l'analisi del CSS della pagina per identificare le immagini definite in questo modo. L'identificazione delle immagini tramite CSS è risultata particolarmente complessa a causa della necessità di decodificare gli stili applicati agli elementi.
- **CSS Inline:** Le immagini possono essere inserite anche tramite CSS *inline*, ovvero tramite l'attributo `style` all'interno dei *tag* HTML. Anche in questo caso è stato necessario analizzare ogni singolo *tag* presente nel codice HTML alla ricerca di attributi `style` contenenti la regola `background-image`. Questo processo ha richiesto un'attenzione particolare per garantire che tutte le immagini, indipendentemente dal metodo di inserimento, fossero correttamente identificate e analizzate.

L'insieme di queste analisi ha permesso di superare le complessità legate all'identificazione delle immagini, assicurando che l'estensione funzionasse correttamente in diversi scenari di utilizzo, migliorando l'accessibilità delle pagine web esaminate.

### 5.1.1 Test effettuati

I test effettuati sono stati principalmente di due tipi: test funzionali e test di sistema. I test funzionali sono stati utilizzati per verificare il corretto funzionamento dell'estensione, in particolare per quanto riguarda l'identificazione delle immagini e la generazione dei report. I test di sistema, invece, sono stati utilizzati per verificare il corretto funzionamento dell'estensione in un ambiente reale, ovvero all'interno di un browser.
I test funzionali sono stati effettuati utilizzando un set di pagine web di test, contenenti immagini inserite in modi differenti. Questo ha permesso di verificare che l'estensione fosse in grado di identificare correttamente le immagini e generare report accurati.
I test di sistema sono stati effettuati utilizzando un browser reale, Google Chrome. Questo ha permesso di verificare che l'estensione fosse correttamente installata e funzionante all'interno del browser, garantendo che l'utente potesse utilizzarla senza inconvenienti.
Ecco alcuni dei siti utilizzati ed i risultati dell'estensione:

## 5.2 Verifica dell'accessibilità dell'estensione

### 5.2.1 Premessa

L'ironia di sviluppare un'estensione non accessibile per la validazione dell'accessibilità di una pagina web non è persa. Per questo motivo, un requisito fondamentale è stato garantire che l'estensione stessa raggiungesse i più alti standard di accessibilità.
Poiché l'estensione include un modulo specifico per l'analisi delle immagini, questa

**Tabella 5.1:** Tabella dei siti utilizzati per i test

| **Pagina** | **Risultati** |
|---|---|
| unipd.it/ | Identificate tutte le immagini presenti |
| ilbolive.unipd.it/ | Identificate tutte le immagini presenti |
| corriere.it/ | Identificate tutte le immagini presenti |
| meteo.corriere.it/italia/lombardia/milano/ | Identificate tutte le immagini presenti |
| ilgiornaledivicenza.it/ | Identificate tutte le immagini presenti |
| ferrarasummerfestival.it/ | Identificate tutte le immagini presenti |
| ferrarasummerfestival.it/convenzioni//esperienze | Identificate tutte le immagini presenti eccetto le immagini dentro la shadow root |
| tgcom24.mediaset.it/ | Identificate tutte le immagini presenti eccetto le pubblicità |
| it.tradingview.com/ | Il sito rifiuta la connessione se caricato tramite `iframe` |
| mail.google.com/mail/u/1/inbox | Il sito presenta un messaggio di errore se caricato tramite `iframe` |

sezione può risultare di difficile utilizzo per persone ipovedenti o cieche. Di conseguenza, l'attenzione è stata rivolta verso l'ottimizzazione della navigabilità dell'estensione. In particolare, sono stati verificati e migliorati aspetti come la navigazione tramite tastiera, il contrasto colore e la compatibilità con gli screen reader. Questi accorgimenti sono stati implementati per garantire che l'estensione fosse utilizzabile da tutti, indipendentemente dalle loro capacità visive e motorie.

### 5.2.2 Navigazione tramite tastiera

Durante lo sviluppo dell'estensione, si è prestata particolare attenzione alla navigazione tramite tastiera, un aspetto cruciale per garantire l'accessibilità a tutti gli utenti, indipendentemente dalle loro capacità motorie. La navigazione tramite tastiera è essenziale per persone con disabilità motorie, utenti non vedenti o ipovedenti che utilizzano screen reader, e chiunque preferisca o necessiti di utilizzare la tastiera invece del mouse.
Per migliorare l'esperienza di navigazione tramite tastiera, sono stati implementati i seguenti accorgimenti:

- **Focus visibile:** È stato garantito che il focus sia sempre chiaramente visibile. Questo accorgimento permette agli utenti di capire facilmente su quale elemento si trovano, migliorando la loro esperienza di navigazione e riducendo la probabilità di errori durante l'interazione con la pagina.
- **Ordine di tabulazione:** È stato assicurato che l'ordine di tabulazione degli elementi segua una sequenza logica e intuitiva. Questo facilita la navigazione, permettendo agli utenti di muoversi agevolmente tra i vari elementi della pagina senza confusione. L'ordine logico di tabulazione è stato progettato per riflettere la struttura e la gerarchia della pagina, migliorando l'usabilità complessiva.

### 5.2.3 WAVE

Un aspetto critico considerato nello sviluppo dell'estensione è stata la scelta dei colori, mirata a garantire un contrasto sufficiente tra il testo e lo sfondo. Il contrasto adeguato è fondamentale per assicurare che il contenuto sia leggibile da tutti gli utenti, inclusi quelli con disabilità visive come daltonismo o ipovisione.
Per verificare che il contrasto dei colori fosse adeguato, è stato utilizzato lo strumento WAVE. Questo strumento consente di analizzare la pagina web e identificare eventuali problemi di accessibilità, inclusi quelli relativi al contrasto dei colori.
L'analisi effettuata con WAVE ha confermato che il contrasto tra il testo e lo sfondo dell'estensione è sufficiente, garantendo così che l'estensione sia accessibile a tutti gli utenti, indipendentemente dalle loro capacità visive. La conformità ai requisiti di contrasto non solo migliora l'accessibilità, ma contribuisce anche a una migliore

leggibilità e usabilità generale della pagina.
I risultati dell'analisi effettuata con WAVE sono riportati nella figura 5.1.

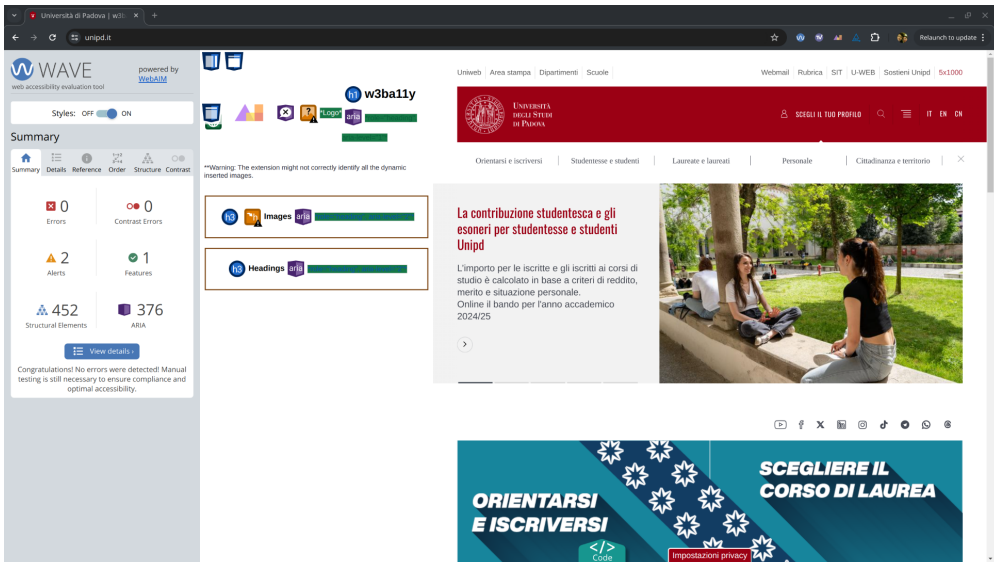


**Figura 5.1:** Analisi dell'estensione con WAVE

### 5.2.4 HTML Validator

Per assicurare che l'estensione rispettasse gli standard HTML, è stato utilizzato l'HTML Validator, uno strumento che verifica la correttezza e la buona struttura del codice HTML. L'aderenza agli standard HTML è essenziale per migliorare il livello di accessibilità dell'estensione.
L'analisi effettuata con l'HTML Validator ha confermato che il codice HTML dell'estensione è corretto e ben strutturato. Questo risultato assicura che l'estensione sia accessibile a tutti gli utenti, migliorando l'interoperabilità e riducendo i potenziali problemi di rendering che potrebbero compromettere l'esperienza utente.
In sintesi, l'attenzione dedicata alla navigazione tramite tastiera, al contrasto dei colori e alla conformità agli standard HTML ha garantito che l'estensione sviluppata sia accessibile e usabile da una vasta gamma di utenti, rispettando i principi fondamentali dell'accessibilità web.
I risultati dell'analisi effettuata con l'HTML Validator sono riportati nella figura 5.2.

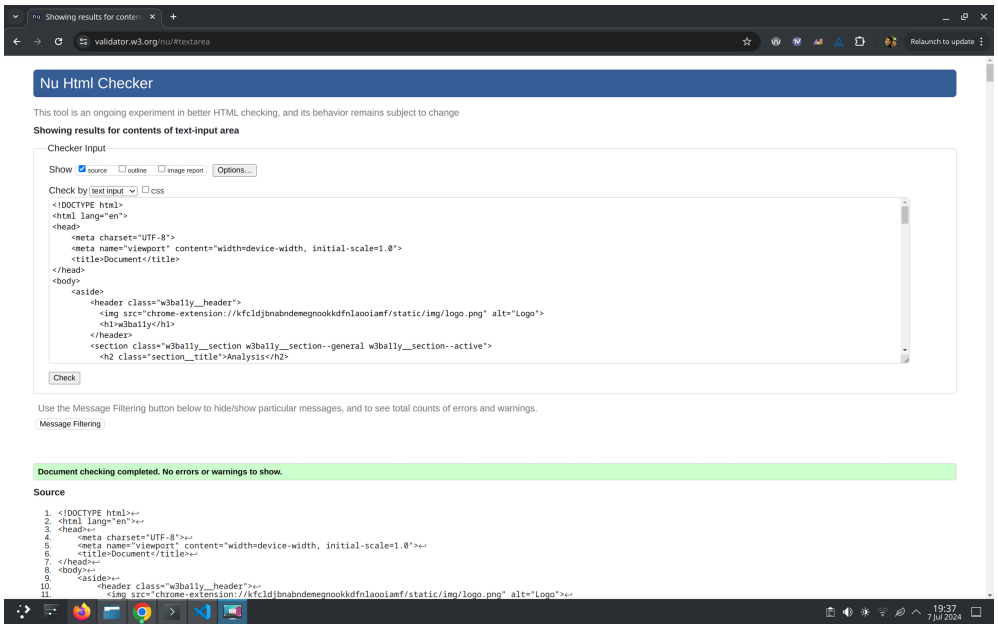


**Figura 5.2:** Analisi dell'estensione con un validator HTML

# Capitolo 6

# Conclusioni

Al termine delle otto settimane, si può affermare che l'obiettivo principale è stato raggiunto. Durante questo periodo, è stata sviluppata un'estensione capace di analizzare le pagine web e fornire all'utente un report dettagliato delle varie violazioni delle linee guida WCAG. L'attività ha coinvolto diverse fasi, tra cui la progettazione dell'architettura del software, la scrittura del codice, i test e le modifiche successive basate sui risultati ottenuti. Sono state effettuate verifiche per garantire che l'estensione fosse accurata e affidabile, e si è lavorato per ottimizzare l'algoritmo di identificazione delle immagini, migliorandone l'efficacia e la precisione.
L'estensione è stata anche sottoposta a una serie di prove pratiche su diversi siti web per assicurare la sua capacità di rilevare correttamente le violazioni delle linee guida WCAG. Questo processo ha comportato l'analisi e la risoluzione di vari problemi tecnici, affinando il funzionamento del modulo di analisi delle immagini e garantendo che l'interfaccia utente fosse intuitiva e di facile utilizzo.
L'obiettivo futuro è continuare a migliorare l'estensione, arricchendola con nuove funzionalità e ottimizzando quelle già esistenti. Grazie alle scelte di progettazione effettuate, l'estensione è facilmente estendibile e modificabile. Questo rende il processo di aggiornamento e miglioramento molto più semplice e efficiente.
Inoltre, il codice sorgente dell'estensione è pubblico e disponibile su GitHub. Questo approccio open-source permette a chiunque di contribuire al progetto, apportando miglioramenti, segnalando bug o suggerendo nuove funzionalità. La collaborazione con la comunità di sviluppatori non solo aiuta a migliorare il plugin, ma garantisce anche che rimanga aggiornato rispetto agli standard di accessibilità in continua evoluzione.

## 6.1 Consuntivo finale

Al termine del progetto, la somma totale delle ore di lavoro è stata di 302 ore, leggermente superiore rispetto al monte ore preventivato. Questo aumento è stato principalmente dovuto alla necessità di apportare modifiche all'algoritmo di identificazione delle immagini, che ha richiesto più tempo del previsto.
La tabella 6.1 mostra la ripartizione dettagliata delle ore di lavoro per ciascuna attività svolta durante il progetto. La suddivisione delle ore effettivamente impiegate si discosta solo in minima parte da quella preventivata nel piano di lavoro presentato all'inizio dello stage. Anche in questo caso, la discrepanza è stata causata dalla complessità del modulo di analisi delle immagini, che ha richiesto un'attenzione e un impegno maggiori per raggiungere gli obiettivi di precisione e affidabilità prefissati.
Durante lo sviluppo, si è reso necessario un continuo affinamento dell'algoritmo per affrontare le sfide impreviste legate alla varietà di formati e tecniche di caricamento delle immagini utilizzate nelle pagine web. Questo ha comportato ulteriori cicli di test e ottimizzazione, prolungando inevitabilmente i tempi di lavoro rispetto a quanto inizialmente stimato.

**Tabella 6.1:** Tabella di ripartizione delle ore di lavoro

| Durata | Attività |
|---|---|
| **40** | **Formazione sulle tecnologie** |
| 25 | Apprendimento Chrome Extension |
| 10 | Ripasso HTML, CSS, JavaScript |
| 5 | Ripasso linee guida WCAG |
| **30** | **Progettazione** |
| 13 | Progettazione dell'architettura dell'estensione |
| 11 | Progettazione dell'architettura dei moduli |
| 6 | Progettazione dell'interfaccia grafica |
| **192** | **Sviluppo** |
| 50 | Implementazione del core dell'estensione |
| 104 | Implementazione del modulo di analisi delle immagini |
| 42 | Implementazione del modulo di analisi heading |
| 6 | Refactoring del codice |
| **40** | **Collaudo** |
| 32 | Test dell'estensione |
| 3 | Test di usabilità |
| 5 | Test di accessibilità |

## 6.2 Conoscenze acquisite

Durante lo stage, ho avuto l'opportunità di approfondire le mie conoscenze di JavaScript, un linguaggio di programmazione estremamente versatile e flessibile, ma anche complesso e difficile da padroneggiare. L'esperienza di sviluppo di un'applicazione completa, come l'estensione per Chrome, mi ha permesso di acquisire una maggiore confidenza con il linguaggio e di esplorare alcune delle sue funzionalità più avanzate. In particolare, ho approfondito la gestione degli eventi, la manipolazione del DOM e l'utilizzo delle API di Chrome. Queste competenze sono essenziali per lo sviluppo di applicazioni web moderne e mi saranno sicuramente utili in futuro.
Un aspetto che ritengo abbia arricchito maggiormente la mia esperienza è stato l'approfondimento della programmazione asincrona in JavaScript. La gestione degli eventi e delle chiamate asincrone è cruciale per lo sviluppo di applicazioni web complesse e performanti. Durante lo stage, ho avuto l'opportunità di lavorare con le Promise e con `async/await`, due delle tecniche più avanzate per gestire il flusso di esecuzione asincrono in JavaScript. La comprensione e l'applicazione di queste tecniche sono fondamentali per garantire la reattività e l'efficienza delle applicazioni web.
Inoltre, ho potuto osservare l'importanza della scrittura di codice pulito e manutenibile, soprattutto in contesti asincroni dove la gestione delle callback può diventare rapidamente complessa. Questo mi ha insegnato l'importanza di adottare buone pratiche di programmazione e di utilizzare strumenti e librerie che facilitano la gestione del codice asincrono.

## 6.3 Valutazione personale

L'esperienza di stage è stata estremamente positiva e formativa. Ho avuto l'opportunità di lavorare su un progetto interessante e stimolante, che mi ha permesso di mettere in pratica le conoscenze acquisite durante il corso di laurea e di approfondire nuove tecnologie e metodologie di sviluppo.

# Acronimi e abbreviazioni

**ADA** Americans with Disabilities Act. 1, 50

**API** Application Program Interface. 13, 32, 36, 37, 50

**ARIA** Accessible Rich Internet Applications. 6, 7, 50

**CSP** Content Security Policies. 38, 39, 50

**CSS** Cascading Style Sheets. 9, 12, 13, 17, 19, 38, 41, 47, 50

**DOM** Document Object Model. 6, 13–15, 31, 38, 50

**GIF** Graphics Interchange Format. 20, 49

**HTML** HyperText Markup Language. 8, 9, 11–13, 15, 17, 19, 31, 40, 41, 44, 47, 51

**HTTP** Hypertext Transfer Protocol. 37, 51

**JPEG** Joint Photographic Experts Group. 20, 51

**PNG** Portable Network Graphics. 20, 52

**POC** Proof of Concept. 30, 31, 52

**REST** Representational State Transfer. 32, 52

**SEO** Search Engine Optimization. 6, 52

**SPA** Single Page Application. 7, 53

**URL** Uniform Resource Locator. 37, 39, 40, 53

**W3C** World Wide Web Consortium. 1, 15, 18, 19, 22, 53

**WAVE** Web Accessibility Evaluation Tool. 5, 43, 44

**WCAG** Web Content Accessibility Guidelines. 1, 7, 29, 46, 47, 53

# Glossario

**ADA** l'*Americans with Disabilities Act* (ADA) è una legge federale statunitense che garantisce l'accesso alle persone con disabilità a tutti i servizi pubblici e privati. La legge è stata approvata nel 1990 e ha portato a importanti cambiamenti nella società americana, garantendo l'accesso a persone con disabilità a luoghi pubblici, servizi di trasporto, servizi sanitari e servizi di comunicazione. 49

**API** in informatica con il termine *Application Programming Interface API* (ing. interfaccia di programmazione di un'applicazione) si indica ogni insieme di procedure disponibili al programmatore, di solito raggruppate a formare un set di strumenti specifici per l'espletamento di un determinato compito all'interno di un certo programma. La finalità è ottenere un'astrazione, di solito tra l'hardware e il programmatore o tra software a basso e quello ad alto livello semplificando così il lavoro di programmazione. 49

**ARIA** le *Accessible Rich Internet Applications* (ARIA) sono un insieme di specifiche sviluppate dal *Web Accessibility Initiative* (WAI) del *World Wide Web Consortium* (W3C) per migliorare l'accessibilità delle applicazioni web. Le specifiche ARIA consentono di aggiungere informazioni semantiche ai contenuti web per renderli più accessibili alle persone con disabilità. 49

**CSP** le *Content Security Policies* (CSP) sono una funzionalità di sicurezza web che permette ai proprietari di siti web di controllare le risorse che possono essere caricate ed eseguite all'interno della loro pagina. Le direttive CSP specificano da quali domini esterni possono essere caricate le risorse, incluse immagini, script, stili, ecc.. 49

**CSS** il *Cascading Style Sheets* (CSS) è il linguaggio utilizzato per descrivere l'aspetto e la formattazione di un documento scritto in HTML. 49

**DOM** il *Document Object Model* (DOM) è una rappresentazione gerarchica di un documento HTML o XML che consente di accedere e manipolare i contenuti

del documento tramite script. Il DOM è una parte fondamentale della programmazione web e viene utilizzato per creare pagine web dinamiche e interattive. 49

**event listener** un *Event Listener* è una funzione che viene eseguita quando si verifica un determinato evento. Gli *Event Listener* vengono utilizzati per gestire gli eventi del browser, come il click del mouse, il caricamento della pagina o la pressione di un tasto sulla tastiera. 34, 35

**GET** il metodo *GET* è uno dei metodi HTTP standard utilizzati per richiedere una risorsa da un server. Il metodo *GET* viene utilizzato per ottenere una risorsa specificata da un URL e viene utilizzato per recuperare informazioni dal server senza modificare lo stato del server. 37, 38

**hash** un *Hash* è una funzione crittografica che converte un input di lunghezza variabile in un output di lunghezza fissa. Gli hash sono utilizzati per proteggere i dati sensibili, come le password, e per verificare l'integrità dei dati. Gli hash sono utilizzati anche per generare firme digitali e per proteggere le comunicazioni su Internet. 38

**HEAD** l'*head* è un tipo di richiesta HTTP che viene utilizzato per ottenere le intestazioni di una risorsa senza scaricare il corpo della risorsa. La richiesta *head* viene utilizzata per ottenere informazioni sulle risorse, come la dimensione del file, la data di ultima modifica o il tipo di contenuto, senza scaricare l'intero file. 37, 38

**HTML** l'*HyperText Markup Language* (HTML) è il linguaggio di markup standard utilizzato per creare pagine web. L'HTML definisce la struttura e il contenuto di una pagina web utilizzando *tag* e attributi. L'HTML è uno dei principali linguaggi di programmazione web ed è supportato da tutti i browser moderni. 49

**HTTP** l'*Hypertext Transfer Protocol* (HTTP) è il protocollo utilizzato per trasferire i dati su Internet. L'HTTP definisce le regole per la comunicazione tra il client e il server e viene utilizzato per trasferire pagine web, immagini, video e altri contenuti su Internet. 49

**JPEG** il *Joint Photographic Experts Group* (JPEG) è un gruppo di esperti che ha sviluppato uno dei formati di file più popolari per le immagini digitali. Il formato JPEG è ampiamente utilizzato per la compressione delle immagini e supporta una vasta gamma di colori e risoluzioni. 49

**lossless compression** la *Lossless Compression* è una tecnica di compressione dei dati che consente di ridurre le dimensioni di un file senza perdita di qualità. La

compressione senza perdita è utilizzata per ridurre lo spazio di archiviazione e il tempo di trasferimento dei file senza compromettere la qualità dell'immagine o del suono. 35

**lossy compression** la *Lossy Compression* è una tecnica di compressione dei dati che consente di ridurre le dimensioni di un file sacrificando la qualità dell'immagine o del suono. La compressione con perdita è utilizzata per ridurre le dimensioni dei file e il tempo di trasferimento, ma può comportare una riduzione della qualità dell'immagine o del suono. 35

**nonce** un *Nonce* (numero usato una sola volta) è un valore casuale che viene utilizzato per proteggere le richieste HTTP da attacchi di ripetizione. Un *Nonce* viene generato dal server e inviato al client come parte di una richiesta HTTP. Il client deve includere il *Nonce* nelle richieste successive per dimostrare che la richiesta è autentica. 38

**PNG** il *Portable Network Graphics* (PNG) è un formato di file per immagini raster che supporta la trasparenza e la compressione senza perdita di dati. Il PNG è uno dei formati di file più popolari per le immagini su Internet ed è ampiamente supportato da tutti i browser moderni. 49

**POC** un *Proof of Concept* (PoC) è una dimostrazione pratica della fattibilità di un'idea o di un progetto. Un PoC viene utilizzato per dimostrare che un'idea funziona nella pratica e può essere realizzata in modo efficace. Un PoC può essere utilizzato per dimostrare la validità di un'idea a potenziali investitori o clienti. 49

**Promise** un oggetto *Promise* rappresenta il risultato di un'operazione asincrona. Una *Promise* può essere in uno dei tre stati: *pending* (in attesa), *fulfilled* (completata con successo) o *rejected* (completata con errore). Le *Promise* sono ampiamente utilizzate in JavaScript per gestire operazioni asincrone in modo più pulito e leggibile. 34, 48

**REST** *Representational State Transfer* (REST) è un'architettura software che definisce un insieme di principi per la progettazione di servizi web. I servizi RESTful sono basati su risorse e forniscono un'interfaccia uniforme per accedere e manipolare le risorse tramite operazioni standard HTTP. 49

**SEO** la *Search Engine Optimization* (SEO) è l'insieme delle tecniche e delle strategie utilizzate per migliorare la visibilità di un sito web sui motori di ricerca. L'obiettivo della SEO è quello di ottenere un posizionamento più alto nei risultati di ricerca organica, aumentando il traffico al sito web e migliorando la sua visibilità online. 49

**SPA** una *Single Page Application* (SPA) è un'applicazione web che carica una sola pagina HTML e aggiorna il contenuto della pagina dinamicamente senza dover ricaricare l'intera pagina. Le SPA sono molto popolari perché offrono un'esperienza utente più fluida e reattiva rispetto alle applicazioni web tradizionali. 49

**URL** l'*Uniform Resource Locator* (URL) è l'indirizzo di una risorsa su Internet. Un URL specifica il protocollo utilizzato per accedere alla risorsa, il nome del dominio del server che ospita la risorsa e il percorso della risorsa sul server. 49

**W3C** il *World Wide Web Consortium* (W3C) è un'organizzazione internazionale che si occupa di sviluppare standard aperti per il web. Il W3C è stato fondato nel 1994 da Tim Berners-Lee, il creatore del World Wide Web, ed è composto da un gruppo di lavoro di esperti provenienti da tutto il mondo. Il W3C sviluppa standard per garantire l'interoperabilità e l'accessibilità dei contenuti web. 49

**WCAG** le *Web Content Accessibility Guidelines* (WCAG) sono una serie di linee guida per garantire l'accessibilità dei contenuti web a persone con disabilità. Sono state sviluppate dal *Web Accessibility Initiative* (WAI) del *World Wide Web Consortium* (W3C) e sono state adottate come standard internazionale. 49

# Bibliografia

## Siti web consultati


[1] *Disability Discrimination Act 1992*. URL: https://www.legislation.gov.au/C2004A04426/2018-04-12/text.

[2] *Section 508 of the Rehabilitation Act*. URL: https://www.fcc.gov/general/section-508-rehabilitation-act.

[3] *Disability Discrimination Act 1995*. URL: https://www.legislation.gov.uk/ukpga/1995/50/contents.

[4] *eEurope 2002: Accessibilità dei siti web pubblici e del loro contenuto*. URL: https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:C:2002:086:0002:0003:IT:PDF.

[5] *Legge 9 gennaio 2004, "Legge Stanca"*. URL: https://www.senato.it/documenti/repository/eventi/dicembre2004/fscommand/Elenco%20leggi/004.pdf.

[6] *European Accessibility Act*. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32019L0882.

[7] *Failed AI image Alt*. URL: https://blogs.library.duke.edu/preservation/2023/10/13/automatic-text-generation-fail/.

[8] *Chrome Extension Permissions ActiveTab*. URL: https://developer.chrome.com/docs/extensions/develop/concepts/activeTab.

[9] *Chrome Extension Permissions Tabs*. URL: https://developer.chrome.com/docs/extensions/reference/api/tabs.

[10] *Chrome Extension Permissions Storage*. URL: https://developer.chrome.com/docs/extensions/reference/api/storage.

[11] *Chrome Extension Permissions Scripting*. URL: https://developer.chrome.com/docs/extensions/reference/api/scripting.

[12] *Chrome Extension Service Worker*. URL: https://developer.chrome.com/docs/extensions/develop/concepts/service-workers/basics.

[13] *Chrome Extension Content Script*. URL: https://developer.chrome.com/docs/extensions/reference/manifest/content-scripts.