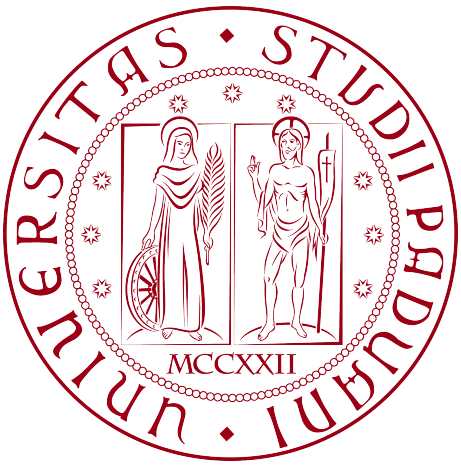
Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica



SviluppAbile: un'estensione per sviluppare pagine web accessibili con l'aiuto di un chatbot

Tesi di Laurea Triennale

Relatrice
Prof.ssa Ombretta Gaggi

Laureanda
Elena Chilese
Matricola 2008074

Anno Accademico 2024-2025

"Colui il quale ha inseguito e sconfitto i demoni Sem, che ora vagano per il mondo, domandandosi: «ma nü, chi sëm?»"

— Il grande Pdor, figlio di Kmer, della tribù di Ishtar, della terra desolata dei Kfnir, uno degli ultimi sette saggi: Pfulur, Galér, Astaparigna, Sùsar, Param, Fusus e Tarìm.

Ringraziamenti

Desidero esprimere la mia gratitudine al professor Ombretta Gaggi, mio relatore, per l'aiuto e il sostegno che mi ha dato durante la stesura dell'elaborato.

Vorrei anche ringraziare, con affetto, i miei genitori per il loro sostegno, il grande aiuto e la loro presenza in ogni momento durante gli anni di studio.

Desidero poi ringraziare i miei amici per i bellissimi anni trascorsi insieme e le mille avventure vissute.

Padova, Settembre 2025 *Elena Chilese*

Sommario

Il presente documento descrive il lavoro svolto durante il periodo di stage, della durata di trecento ore svolto presso il Dipartimento di Matematica "Tullio Levi-Civita" dell'Università di Padova.
Il prodotto finale dello stage è un'estensione web chrome-based per aiutare gli sviluppatori web nella creazione di pagine web accessibili.

Gli obiettivi del progetto sono:

- Studio dell'accessibilità web;
- Studio di Manifest V3;
- Studio delle possibili API per l'integrazione dell' IA;
- Creazione di un'estensione web che integri un chatbot AI per analizzare il codice sorgente della pagina web ed offrire risposte adeguate.

Indice

Elenco delle figure

Elenco delle tabelle

List of Listings

Capitolo 1

Introduzione

Figura 1.1: Lorem

L'accessibilità web rappresenta un aspetto fondamentale per garantire che tutte le persone, indipendentemente dalle loro abilità o disabilità, possano utilizzare i contenuti digitali in modo efficace. Le linee guida WCAG (Web Content Accessibility Guidelines) sono uno standard internazionale, esse definiscono i criteri tecnici per migliorare l'accessibilità dei siti web, intervenendo su aspetti quali la struttura semantica, la navigazione tramite tastiera, il contrasto cromatico dei colori e la compatibilità con tecnologie assistive. Un sito accessibile non solo migliora l'esperienza d'uso per persone con disabilità visive, motorie o cognitive, ma estende anche la fruibilità a un pubblico più ampio, contribuendo a un web più inclusivo e universale. Nonostante ciò, spesso la realizzazione pratica di siti accessibili incontra spesso difficoltà dovute a scarsa conoscenza delle best-practices dello sviluppo web accessibile, delle normative in questione e della scarsa diffusione di strumenti automatizzati efficaci.

Lorem Figure 1.1

1.1 L'idea del progetto

Il progetto sviluppato durante il mio stage si inserisce all'interno di un'iniziativa molto più ampia intitolata "Supporting Accessibility Auditing and HTML Validator using Large Language Models" a cui partecipano e collaborano docenti e stagisti dell'Università di Padova e dell'Università di Bologna.
Tale progetto nasce dall'esigenza di affrontare il problema dell'accessibilità web, diritto fondamentale per tutti i cittadini (in particolare per le persone con disabilità) che devono poter accedere alle informazioni sul web senza barriere.

Nonostante le normative nazionali e internazionali (come la Direttiva Europea sull'Accessibilità Web e l'European Accessibility Act) impongano requisiti chiari, la gran parte dei siti web presenta ancora numerose criticità di accessibilità. In questo contesto, il progetto mira a valutare, testare e sfruttare le capacità dei Large Language Models (LLM) per supportare l'audit dell'accessibilità e la validazione del codice HTML.
L'obiettivo finale è sviluppare strumenti innovativi che utilizzino modelli di intelligenza artificiale, come ChatGPT, per fornire un supporto interattivo agli sviluppatori, aiutandoli a identificare e correggere problematiche di accessibilità in modo più efficace e comprensibile rispetto ai metodi tradizionali. Attraverso questa integrazione, si intende migliorare la qualità e soprattutto la chiarezza dei risultati e favorire una cultura più diffusa sull'accessibilità digitale.

L'estensione "SviluppAbile" è un primo esempio di strumento per lo sviluppo guidato di pagine web accessibili. Essa nasce dall'esigenza di supportare gli sviluppatori web nel creare pagine accessibili in modo semplice e interattivo, sfruttando le potenzialità dell'intelligenza artificiale.
L'estensione chrome-based sviluppata offre due modalità principali:
una modalità di analisi assistita, che permette di ispezionare il DOM di una pagina web e di porre domande specifiche sull'accessibilità, ottenendo risposte chiare e contestualizzate;
una modalità di sviluppo guidato, in cui l'utente riceve suggerimenti di codice accessibile in tempo reale, visualizzati in una colonna centrale e pronti per essere copiati o scaricati.
Questo duplice approccio consente di integrare nel flusso di lavoro quotidiano degli sviluppatori un valido supporto basato su AI, con l'obiettivo di facilitare il rispetto delle linee guida WCAG e migliorare la qualità complessiva del codice.

1.2 Analisi delle soluzioni già presenti

Sul mercato esistono diversi strumenti e plugin dedicati alla verifica dell'accessibilità delle pagine web, come WAVE, Axe e Lighthouse, che offrono principalmente funzionalità di scansione automatica e generazione di report.
Tuttavia, questi strumenti spesso forniscono una panoramica statica e generica dei problemi riscontrati; i risultati tendono ad essere formulati in modo formale e talvolta poco comprensibile per l'utente, il quale è quindi costretto a interpretare autonomamente i dati, rischiando di applicare soluzioni non sempre pienamente conformi alle linee guida.

Alcuni software integrano suggerimenti o tutorial, ma raramente utilizzano modelli di intelligenza artificiale in grado di interagire con lo sviluppatore in linguaggio naturale, rispondendo a domande specifiche o guidandolo direttamente nella scrittura del codice.

"SviluppAbile" si distingue per l'integrazione di una chat AI che funge da assistente personale, e per la duplice modalità operativa che combina l'analisi in tempo reale con un supporto diretto allo sviluppo, colmando così una lacuna importante nelle soluzioni attualmente disponibili.

1.3 Organizzazione del testo

Il documento è diviso in xx capitoli e illustra in maniera dettagliata l'esperienza di stage svolta.

Il secondo capitolo descrive ...

Il terzo capitolo approfondisce le tecnologie e gli strumenti utilizzati per la realizzazione del progetto.

Il quarto capitolo approfondisce ...

Il quinto capitolo approfondisce ...

Il sesto capitolo approfondisce ...

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;
- i termini in lingua straniera o facenti parti del gergo tecnico sono evidenziati con il carattere *corsivo*.

int main() print("Hello, world!"); return 0;

Listing 1: Example of code

Capitolo 2

Analisi dei requisiti

2.1 Obiettivo del progetto

2.2 Casi d'uso

2.3 Tracciamento dei requisiti

Capitolo 3

Tecnologie

3.1 Linguaggi e tecnologie utilizzate

3.1.1 HTML

HTML (HyperText Markup Language) è il linguaggio di marcatura standard utilizzato per strutturare i contenuti delle pagine web. Non si tratta di un linguaggio di programmazione, ma di uno strumento che consente di definire gli elementi presenti su una pagina (come titoli, testi, immagini, link, form) assegnando loro un significato semantico tramite l'uso di tag.
Introdotto nel 1991 da Tim Berners-Lee, HTML si è evoluto nel tempo attraverso diverse versioni, fino a diventare uno standard globale supervisionato dal World Wide Web Consortium (W3C).

La struttura di un file HTML si basa su una serie di elementi, rappresentati attraverso tag, che permettono di definire la natura e la funzione dei contenuti all'interno di una pagina web. Ogni tag descrive una tipologia specifica di contenuto o un comportamento associato, contribuendo all'organizzazione semantica del documento. I tag sono delimitati da parentesi angolari (`< >`) e, nella maggior parte dei casi, sono utilizzati in coppie: un tag di apertura e uno di chiusura, quest'ultimo contraddistinto da una barra (`</tag>`). Alcuni elementi, tuttavia, non necessitano di chiusura e sono detti auto-chiudenti, come ad esempio `<img>` per l'inserimento di immagini, o `<br>` per i ritorni a capo. Questa struttura gerarchica e nidificata consente al browser di interpretare correttamente i contenuti, garantendo coerenza tra presentazione e significato.

La versione attuale, HTML5, è stata rilasciata in modo stabile nel 2014.
HTML5 rappresenta un'evoluzione del linguaggio di markup con l'obiettivo di

ridurre la dipendenza da plugin esterni, introducendo nuove funzionalità native. Tra le principali novità vi sono gli elementi `<canvas>`, `<video>` e `<audio>`, nuovi tag semantici per strutturare i contenuti (come articoli, intestazioni, sezioni) e controlli avanzati per i form (email, URL, numeri, date, ricerca).
Vengono inoltre introdotti strumenti per la memorizzazione locale dei dati sul client tramite `localStorage` e `sessionStorage`.

3.1.2 CSS

CSS (Cascading Style Sheets) è un linguaggio utilizzato per definire la presentazione dei documenti HTML e XML. Introdotto nel 1996 da Håkon Wium Lie, consente di separare la struttura dei contenuti dalla loro formattazione, migliorando la chiarezza del codice e facilitandone la manutenzione.
I fogli di stile permettono di specificare, tramite regole composte da selettori, proprietà e valori, l'aspetto degli elementi HTML: layout, colori, font, margini, spaziature, effetti visivi, animazioni e molto altro. Le regole di stile possono essere inline (tramite tag `<style>` o con attributi style) oppure riportate in file esterni .css riutilizzabili. La prima soluzione non mantiene però una netta separazione tra struttura e contenuto, non rispettando quindi le best-practice del settore.
L'attuale versione, CSS3, è modulare e introduce importanti funzionalità come Flexbox, Grid, media query per il responsive design, transizioni, trasformazioni e nuovi selettori.

3.1.3 Prism.js

Prism.js è una libreria JavaScript leggera ed estensibile progettata per evidenziare la sintassi del codice sorgente in modo chiaro e leggibile, rispettando gli standard moderni del web.
Supporta un'ampia gamma di linguaggi di programmazione e di markup, ed è facilmente personalizzabile tramite temi e plugin.
Grazie alla sua efficienza e semplicità di integrazione, viene utilizzata in milioni di siti web per migliorare la leggibilità del codice, rendendolo più comprensibile sia a sviluppatori che a utenti. E' disponibile in diverse varianti, sia per il white-mode che per il dark-mode.
Nella mia estensione, Prism.js è stato integrato per colorare il codice sorgente del DOM, facilitando l'analisi e la comprensione della struttura HTML.

3.1.4 JavaScript

JavaScript è un linguaggio di programmazione dinamico, interpretato e orientato agli oggetti, progettato originariamente per rendere le pagine web interattive. È stato sviluppato nel 1995 da Brendan Eich per Netscape con il nome LiveScript, e successivamente rinominato in JavaScript per motivi di marketing, pur non avendo legami diretti con il linguaggio Java. La sua standardizzazione è gestita da ECMA International, sotto il nome ECMAScript.

Concepito per essere eseguito direttamente all'interno del browser, JavaScript consente di scrivere script incorporati nel codice HTML, eseguibili senza compilazione al momento del caricamento della pagina. Questi script permettono di manipolare il DOM (Document Object Model), reagire agli eventi dell'utente (come click, input da tastiera o movimenti del mouse), modificare dinamicamente il contenuto della pagina e gestire funzionalità come validazioni, messaggi, animazioni, e interazioni asincrone.

Oltre alla manipolazione di elementi visivi, JavaScript permette di sfruttare numerose API messe a disposizione dal browser, dalla gestione del mouse e del touch, alla manipolazione delle immagini, fino alla gestione locale dei dati attraverso meccanismi come il LocalStorage. Questa versatilità lo rende uno strumento fondamentale per lo sviluppo di applicazioni web moderne.

JavaScript si è evoluto da semplice linguaggio client-side a tecnologia completa e versatile, oggi utilizzabile anche su server (ad esempio con Node.js) e in ambienti esterni ai browser grazie all'uso di diversi motori JavaScript i quali permettono l'esecuzione di codice JS con alte prestazioni, rendendo il linguaggio adatto anche a contesti backend, applicazioni desktop e mobile.

Oggi JavaScript rappresenta uno dei tre pilastri fondamentali del web, insieme a HTML e CSS, ed è completamente integrato con essi. Esistono anche linguaggi alternativi (come TypeScript o CoffeeScript) che vengono compilati in JavaScript, offrendo funzionalità aggiuntive mantenendo la compatibilità con l'ambiente JavaScript standard.

3.1.5 Manifest V3

Manifest V3 (MV3) è la specifica più recente per la creazione di estensioni per Chrome e altri browser basati su Chromium. Rispetto alla precedente Manifest

V2, l'ultima versione introduce diverse modifiche per migliorare la sicurezza, le prestazioni e la privacy delle estensioni.
Questa nuova versione è una risposta alle criticità emerse con il Manifest V2, e introduce cambiamenti strutturali significativi nel modo in cui le estensioni interagiscono con il browser e le pagine web.
E' importante notare che Manifest V3 non è retrocompatibile con le estensioni V2 senza modifiche sostanziali e che la versione è tuttora in evoluzione, poiché alcune funzionalità sono ancora in fase di implementazione.
Il file `manifest.json` in MV3 continua a rappresentare il cuore dell'estensione, definendo metadati fondamentali come nome, versione, permessi richiesti e script da eseguire. Tuttavia, rispetto a MV2, MV3 impone restrizioni più rigide su quali API possono essere utilizzate e introduce un modello di esecuzione più efficiente e sicuro.

Gli elementi principali di un'estensione basata su Manifest V3 sono:

- **Manifest**: come nelle versioni precedenti, il file `manifest.json` contiene tutte le informazioni necessarie per il funzionamento dell'estensione, inclusi i permessi richiesti. Tra i permessi più rilevanti vi sono:
  - downloads: permette all'estensione di avviare e gestire download di file, ad esempio per salvare contenuti generati o analizzati;
  - scripting: consente l'esecuzione di script personalizzati nelle pagine web, utile per analizzare o modificare il DOM;
  - activeTab: garantisce l'accesso temporaneo alla scheda attiva dopo un interazione dell'utente con l'estensione;
  - tabs: fornisce informazioni sulle schede del browser e permette di interagire con esse, come ottenere URL o titoli;
  - storage: consente di memorizzare e recuperare dati locali, ad esempio impostazioni o risultati di analisi.
- **Service Worker**: in Manifest V3, i tradizionali background script sono sostituiti da service worker. Questi sono script che vengono eseguiti in background ma solo quando necessario, rispondendo a eventi come le richieste di rete o azioni dell'utente. A differenza dei background script di MV2, i service worker non sono persistenti e vengono sospesi quando inattivi, riducendo così l'utilizzo di risorse e migliorando le prestazioni complessive. I service worker non possono accedere direttamente al DOM delle pagine, ma comunicano con i content script tramite messaggi.

- **Content Script**: vengono iniettati direttamente nelle pagine web e possono interagire con il DOM. In MV3, anche i content script sono soggetti a restrizioni più severe per evitare conflitti con il contenuto della pagina e migliorare la sicurezza.

Pro e Contro di Manifest V3

Pro:

- **Maggiore sicurezza**: il modello basato su service worker limita l'esecuzione continua di script in background, riducendo la superficie di attacco e il rischio di abuso delle API.
- **Migliore gestione delle risorse**: l'esecuzione temporanea dei service worker aiuta a ridurre il consumo di memoria e CPU, migliorando le prestazioni del browser.
- **Controlli più rigidi sui permessi**: le estensioni devono dichiarare con precisione quali risorse e API intendono usare, migliorando la trasparenza e la privacy per l'utente.
- **Modularità e comunicazione**: la separazione tra service worker e content script, con comunicazione tramite messaggi, aiuta a mantenere un'architettura più pulita e modulare.

Contro:

- **Limitazioni funzionali**: il passaggio ai service worker e la limitata persistenza in background possono rendere più complesso implementare funzionalità che richiedono attività di lunga durata.
- **Compatibilità ridotta**: alcune API presenti in Manifest V2 sono deprecate o rimosse in MV3, costringendo gli sviluppatori a riscrivere o adattare estensioni esistenti.
- **Curva di apprendimento**: le nuove modalità di funzionamento (service worker, restrizioni sui permessi) richiedono una maggiore comprensione da parte degli sviluppatori.
- **Controversie e opposizioni**: la comunità di sviluppatori ha espresso preoccupazioni riguardo a possibili limitazioni alle estensioni per il blocco degli annunci e altre funzionalità avanzate, potenzialmente influenzando la libertà di personalizzazione.

3.1.6 Ollama

Ollama è una piattaforma che consente di eseguire localmente modelli di intelligenza artificiale avanzati, sviluppati in collaborazione con OpenAI e altri partner, mantenendo alte prestazioni anche senza l'uso del cloud. Oltre ai modelli gpt-oss-20B e gpt-oss-120B, progettati rispettivamente per scenari a bassa latenza o per applicazioni generali ad alto livello di ragionamento, Ollama supporta numerosi altri modelli open weight, tra cui Mistral (utilizzato solo all'inizio a causa delle ridotte capacità del mio pc personale), Llama 3.1:8B e Llama 3.2:3B, che ho utilizzato nel mio progetto.
La piattaforma integra funzionalità avanzate (function calling, navigazione web integrata, esecuzione di codice Python, output strutturati), accesso completo al processo di ragionamento del modello, regolazione dello sforzo computazionale e possibilità di fine-tuning per personalizzare le prestazioni. Grazie al supporto nativo per la quantizzazione in formato MXFP4, Ollama riesce a ridurre drasticamente il consumo di memoria, permettendo l'esecuzione di modelli molto grandi anche su sistemi con risorse limitate. La licenza Apache 2.0 garantisce libertà di utilizzo e personalizzazione, mentre l'ottimizzazione per GPU NVIDIA GeForce RTX e RTX PRO assicura alte prestazioni in locale.