Sushi Nakamura

Progetto di Tecnologie Web A.A. 2019/2020

Informazioni sul gruppo

Membri	Dindinelli Alessandro - 1170457 Frison Nicolò - 1147682 Giardina Mirco - 1136663 Tommasin Alessandro - 1189293

Indice

1 Introduzione

1.1 Abstract

Il sito web **Sushi Nakamura** è stato sviluppato per permettere all'omonimo ristorante di Padova un mezzo per promuovere sè stesso ed il suo nuovo servizio di take away. Nel sito è possibile reperire tutte le informazioni riguardanti i contatti ed i prodotti che possono essere acquistati. Inoltre l'amministratore ha la possibilità di inserire, rimuovere e modificare eventuali articoli in vendita e news che possono essere visualizzate dagli utenti.

2 Analisi

2.1 Analisi dell'utenza

Questo sito possiede un bacino di utenza variegato in quanto si è ritenuto possibile che sia utenti molto giovani che utenti in età più adulta possano essere interessati ad accedere al sito e ai suoi servizi. La parte più giovane molto probabilmente è abituata all'utilizzo di servizi di take away da dispositivi mobile e probabilmente sfrutterà la parte del sito adibita all'esposizione dei prodotti disponibili e all'ordinazione take away. La controparte adulta, invece, probabilmente sfrutterà di più il sito per consultare sia i prodotti disponibili che per trovare le informazioni riguardanti l'ubicazione del ristorante ed eventualmente il contatto telefonico per prenotare.

2.2 Conclusione

Il sito dovrà quindi essere fornito di una sezione che permetta agli utenti di trovare i prodotti divisi in categorie. Dovrà inoltre gestire l'ordinazione, il pagamento e l'eventuale spedizione del prodotto, nel caso in cui si selezioni la spedizione a domicilio. Si ritiene possa essere un'aggiunta interessante avere la possibilità di salvare i metodi di pagamento e di spedizione in modo tale da non doverli reinserire ad ogni acquisto (nel caso di utente autenticato).

3 Progettazione

3.1 Obiettivi

Gli obiettivi principali perseguiti durante la progettazione del sito sono i seguenti:

- **Separazione tra struttura, presentazione e comportamento:** Obiettivo fondamentale, in quanto raggiungerlo permette di soddisfare più agevolmente anche gli altri punti. La struttura è stata realizzata con documenti in XHTML 1.0 Strict dove possibile, così da garantire una maggiore retrocompatibilità con vecchi browsers, ed HTML5 dove si sono ritenute

necessarie le funzionalità aggiuntive permesse dal linguaggio. La presentazione è stata sviluppata con fogli di stile CSS linkati, mentre il comportamento con script esterni realizzati in Javascript e PHP. In questo modo la struttura non dovrà cambiare, anche a seguito di modifiche alla presentazione del sito. Tutto il codice redatto è stato scritto secondo le raccomandazioni W3C, accertando poi che siano state rispettate, validando HTML e CSS con i rispettivi tool di W3C.

- **Accessibilità:** Il sito deve poter essere fruibile agevolmente dal maggior numero di utenti possibile, compresi quelli con differenti tipi di disabilità. Per garantire una buona accessibilità, alcune misure adottate sono:
  - Uso dei tabindex nel menù di navigazione;
  - Testo alternativo per le immagini;
  - Assenza di link circolari;
  - Testi e link con buoni livelli di contrasto;
  - Uso dell'attributo lang per testi non in italiano;
- **Fluidità:** Il sito deve poter essere consultabile tramite varie tipologie di dispositivi, tra cui PC desktop, tablet e smartphone. Bisogna quindi garantire una buona adattabilità alle differenti dimensioni di schermo.
- **Fruibilità:** Per realizzare un sito navigabile intuitivamente si sono seguite alcune linee guida comuni nel web, come ad esempio:
  - Sfruttare un layout ben strutturato, che faciliti l'individuazione del contenuto di interesse;
  - Agevolare lo scroll tramite link relativi per raggiungere diversi punti di una stessa pagina;
  - Mantenere colorazioni diverse per link visitati e non;

3.2 Layout

Si è deciso di strutturare il sito con un layout a colonna singola, come illustrato in Figura 1. Questo modello permette una buona mantenibilità, dato che anche in caso si effettuino modifiche al contenuto della pagina la sua presentazione rimane uniforme, avendo a disposizione l'intera larghezza della finestra. Dimostra di essere anche un layout molto flessibile in quanto la transizione dalla presentazione su desktop a quella su dispositivi mobili risulta fluida.

L'Header, che contiene logo e nome del ristorante, fondamentalmente varia solo in dimensione nella transizione tra i due layout.

Il Footer invece contiene informazioni generali sul ristorante, la dichiarazione del Copyright, e l'eventuale immagine di certificata validazione delle pagine XHTML 1.0 Strict. Per garantire una lettura migliore, restringendo la finestra si passa ad una visualizzazione da doppia a singola colonna.

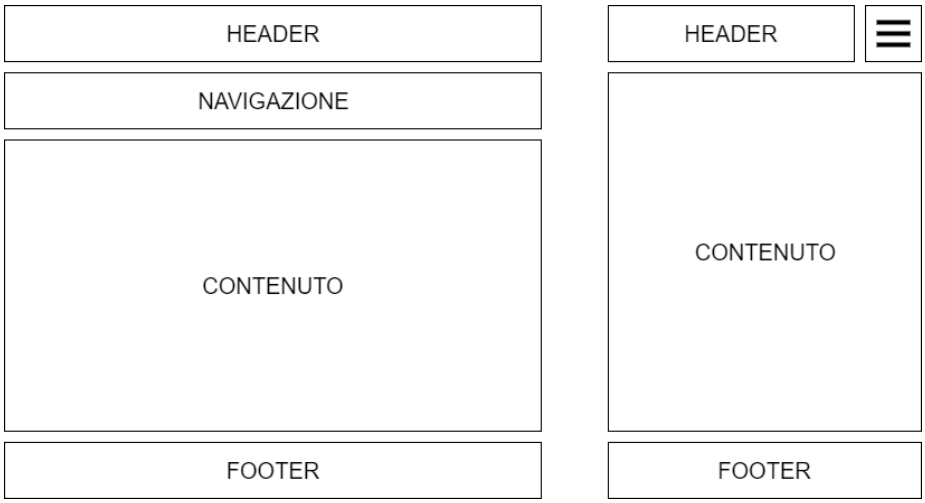


Figura 1: Schema del layout per desktop e mobile.

La principale differenza tra i layout scelti sta nella posizione e presentazione del menù di navigazione.
Nei device con finestre di visualizzazione sufficentemente larghe si è scelto di disporre le voci del menù in riga, creando quindi una barra di navigazione. I link alle varie pagine principali del sito sono disposte a partire da sinistra, mentre quelli riguardanti pagine per gli utenti e le operazioni a loro disposizione si trovano a partire da destra.
Nei dispositivi più piccoli si è deciso invece di creare un menù espandibile, raggiungibile premendo sull'icona ad hamburger nell'angolo in alto a destra, che mostra tutte le voci disponibili, per rispettare il comportamento più comune che un utente medio si aspetta utilizzando uno smartphone.

3.3 Accessibilità

3.3.1 Attributi HTML

Per rendere il sito più accessibile agli screen reader sono stati usati diversi attributi HTML.

- **Lang**: utilizzato per indicare in che lingua deve essere pronunciata una parola o frase. Facendo dei test con il software gratuito NVDA abbiamo notato che le parole identificate usando $x$ml:lang non venivano lette nella lingua corretta, si è quindi optato di usare l'attributo HTML5 $l$ang in tutte le pagine e testato che queste continuassero ad essere validate dal validatore W3C per xhtml1.0 strict.
- **Alt**: utilizzato nelle immagini per fornire una breve descrizione di quello che viene raffigurato.
- **Scope**: utilizzato per facilitare la lettura delle tabelle.
- **Tabindex**: utilizzato per assicurarsi un ordine corretto di spostamento tra i link utilizzando il tab.

É stato inoltre usato un'aiuto per gli screen reader fornendo un link prima del menu così da poterlo saltare e andare direttamente al contenuto della pagina.

3.3.2 Colori

I colori del sito sono stati selezionati apposta per passare il test WCAG AAA che richiede un rapporto di contrasto di almeno 7:1 per il testo normale e 4.5:1 per il testo in grassetto. Di seguito è riportata la tabella dei colori del testo e del loro sfondo e il loro rapporto di contrasto.

Tabella 1: Rapporto contrasti testo-sfondo

Colore testo	Colore sfondo	Rapporto
#D9D9D9	#1F1D1D	11.88:1
#E0AC00	#4A0715	7.55:1
#F5E900	#4A0715	12.41:1
#E0AC00	#1F1D1D	8.04:1
#F5E900	#1F1D1D	13.21:1
#D9D9D9	#333333	8.95:1
#D9D9D9	#4A0715	12.41:1

4 Implementazione

4.1 Linguaggi

4.1.1 XHTML 1.0 Strict e HTML5

Come già detto, per la realizzazione del sito si è optato per avere sia pagine in XHTML 1.0 Strict, che in HTML5. L'uso di XHTML 1.0 garantisce una migliore retrocompatibilità con browsers più datati ed una migliore aderenza del codice agli standard di correttezza, ma HTML5 permette di sfruttare alcune funzionalità aggiuntive che abbiamo sfruttato ad esempio in vari form per inserire input di tipo *date*, utilizzare *placeholder* o per rendere dei campi *required* in modo da chiarire e migliorare l'usabilità del sito. In entrambi i casi per ottenere codice pulito si sono seguiti i seguenti principi:

- Mantenere la separazione tra struttura e presentazione, usando file separati linkati nell'header;
- Ogni <tag> aperto deve essere poi chiuso </tag> (oppure <tag />);
- I tag di intestazione vanno usati in base all'importanza di ciò che racchiudono, e non in base all'estetica base html;

In generale, per garantire un codice corretto, sono state seguite le linee guida che sono state illustrate durante il corso di Tecnologie Web e quelle dello standard W3C. La verifica del codice è stata poi effettuata utilizzando il tool di validazione W3C Validator[1], la piattaforma SonarCloud [2] integrata con GitHub ed il software di validazione di siti Total Validator.
[3]

Si è prestata attenzione al ruolo dei metatag nell'header, il cui buon uso migliora anche l'indicizzazione del sito da parte dei motori di ricerca, aspetto fondamentale per un'attività commerciale che voglia ricevere un'adeguata visibilità nel web.
Si è deciso di fare uso anche di tabelle, rendendole più agevolmente accessibili tramite l'uso di adeguati *summary*, e di *scope* su colonne e righe di interesse.

4.1.2 CSS

CSS è stato usato per la presentazione grafica delle pagine HTML. Si è cercato di puntare ad un discreto riutilizzo del codice creando classi da poter usare su più pagine, come per esempio bottoni e form preimpostati, che poi potevano venire personalizzati dove necessario sovrascrivendo le regole necessarie. Il contenuto di ogni pagina è identificato da una classe così da avere una formattazione del testo omogenea mentre per impaginare le pagine specifiche si è optato per un identificativo univoco per ogni pagina.

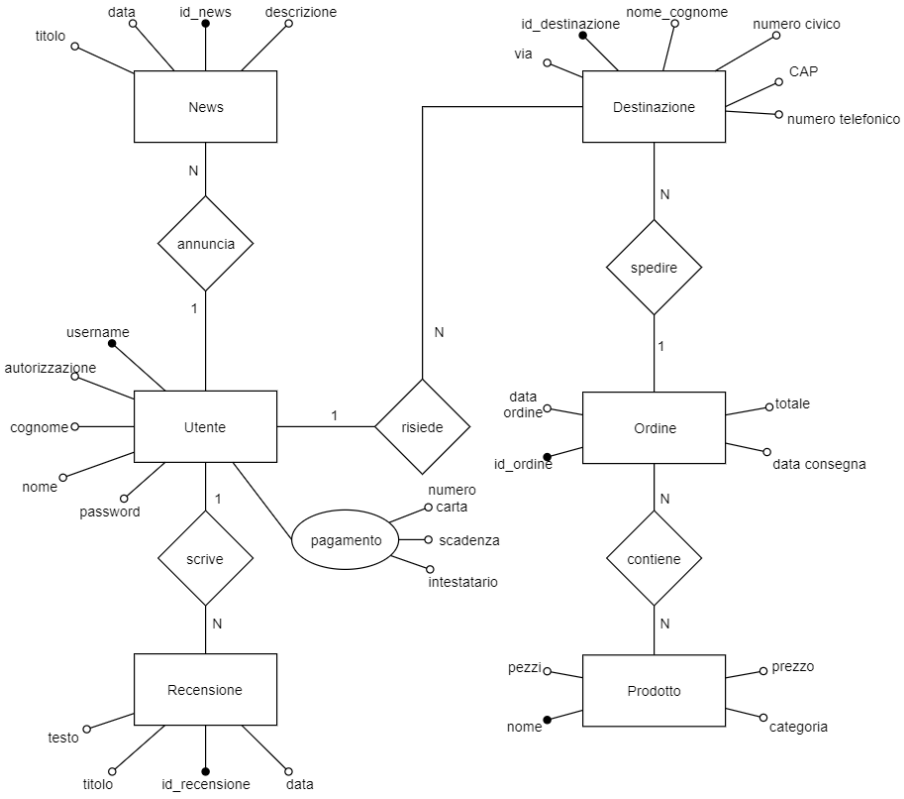
[1] Riferimento al sito di W3C Validator https://validator.w3.org/
[2] Riferimento al sito di SonarCloud https://sonarcloud.io/about
[3] Riferimento al sito di Total Validator https://www.totalvalidator.com/

4.1.3 PHP

Il linguaggio PHP è stato utilizzato per creare tutte le interazioni del sito con il database, dopo aver effettuato un controllo ed una sanificazione degli input dell'utente, al fine di evitare sia l'inserimento di contenuto malevolo e/o non corretto nel database che eventuali comportamenti anomali dello stesso. Un altro uso che è stato fatto di questo linguaggio è stato quello di creare delle pagine web dinamiche che avessero un menù personalizzato, in base al fatto che un utente fosse autenticato o meno, e dei contenuti diversi in base ai dati presenti nel database (e.g. le destinazioni salvate o le notizie pubblicate dagli amministratori). Per quanto riguarda la codifica si è cercato, ove possibile, di definire delle funzioni che realizzassero i compiti che più si sarebbero ripetuti all'interno del sito, come il controllo degli input, e generalmente si è cercato di separare la logica della pagina dalle funzioni chiamate in modo tale da diminuire il più possibile le dipendenze ed agevolare un'eventuale manutenzione del sito.

4.1.4 SQL

Sql è stato usato per codificare il database. Si rimanda al file *creazione_database.sql* nella cartella *Database* [url] della repository per il file di costruzione del database. Di seguito il diagramma ER del database:



4.1.5 JavaScript

Il linguaggio JavaScript è stato utilizzato per lo più per fare un controllo preventivo dei campi di input; in questo modo abbiamo evitato al server un compito che poteva essere effettuato in precedenza lato client. Il linguaggio è stato anche utilizzato per creare uno slideshow a scorrimento manuale nella pagina di inizio dell'utente, con delle immagini che rappresentassero il clima del ristorante, senza però scordarsi dell'accessibilità e del disturbo che potrebbe arrecare un carousel automatico. Un altro utilizzo di JavaScript è stato riguardo la creazione di un menù a scomparsa nella versione mobile del sito in modo tale da sfruttare al massimo la dimensione ridotta delle pagine ed evitare degli scroll verticali aggiuntivi che sarebbero stati richiesti nel caso di un menù verticale non a scomparsa. Infine JavaScript è stato ritenuto utile nelle pagine di takeaway, carrello e pagamento per spostare i prodotti selezionati da una pagina ad un altra o per disabilitare degli input in base alle selezioni dell'utente. Come per il linguaggio PHP si è puntato al riutilizzo del codice definendo funzioni per effettuare i controlli e riutilizzando le funzioni dove possibile.

5 Fase di test

5.1 Strumenti usati

5.1.1 W3C HTML Validator

Le pagine html sono state validate usando il validatore fornito dall'organizzazione W3C per garantire la corretta visualizzazione del contenuto della pagina senza fare entrare i browser in **Quirks Mode**. É stato usato anche per validare il risultato delle pagine php incollando il sorgente ottenuto facendo eseguire lo script php.

5.1.2 W3C CSS Validator

Tutti i file CSS sono stati validati usando lo strumento di validazione fornito da W3C per assicurarsi che fossero strutturati correttamente.

5.1.3 TotalValidator

Con questo strumento abbiamo validato tutto il codice HTML. Principalmente errori segnalati dal programma riguardano direttive sull'uso delle regole WAI-ARIA[4] relative all'utilizzo delle *aria-label*, cosa che non è necessaria in questo caso, in quanto i vari link e buttons sono facilmente indiduabili ed interpretabili.

5.1.4 SonarCloud

Servizio integrato con GitHub per la verifica del codice nella repository. Ad ogni push veniva fatta un **analisi statica** del codice alla ricerca di problemi e vulnerabilità come ad esempio un problema comune è stata la ripetizione rindondate di regole css. Questa fase di test era bloccante, ovvero perchè il codice venisse aggiunto dovevano prima essere risolti i problemi.

6 Organizzazione del lavoro

Il progetto è stato suddiviso in modo tale che ogni membro avesse la possibilità di creare sia alcune pagine HTML che il relativo CSS, facendo da verificatore nelle pagine degli altri membri. Lo stesso può essere detto per quanto concerne PHP, JavaScript e la creazione ed il popolamento del database. La sviluppo può essere seguito nella repository di GitHub utilizzata:

https://github.com/Mirco469/ProgettoSushi

[4] Web Accessibility Initiative - Accessible Rich Internet Applications - https://www.w3.org/WAI/standards-guidelines/aria/