Relazione sul progetto del generatore di tabelle CVE

Sidrit Trandafili #0013286

Versione online https://stw-trandafili.netlify.app/
Scarica sorgente https://github.com/sidis405/stw-trandafili/archive/refs/heads/main.zip
Visualizza su Github https://github.com/sidis405/stw-trandafili
Componente di interesse https://github.com/sidis405/stw-trandafili/blob/main/src/components/CveTable.vue

Descrizione progettuale

Il progetto consiste nella realizzazione di un componente Vue.js che permetta di generare una tabella HTML formattata contenente un elenco di indicatori univoci di vulnerabilità (CVE).
L'obiettivo principale è fornire un'interfaccia semplice ed efficiente per l'inserimento e la visualizzazione di un elenco di CVE, supportando la validazione, la generazione di una tabella.
Inoltre, il progetto include anche le funzionalità di l'esportazione dei dati in formato CSV e la copia in memoria.

> Rev 1:

- Implementazione: Il componente permette di visualizzare una textarea che mostra il codice della tabella generata
- Addendum: I codici duplicati vengono rimossi

Il componente Vue.js è stato implementato utilizzando le seguenti tecnologie e librerie:

- **Vue.js** per la gestione delle funzionalità del componente e l'interazione con l'utente
- **TailwindCSS** per la formattazione e lo stile dell'interfaccia utente
- **Jest** e **Vue Test Utils** per i test di unità e la copertura del codice

Motivazioni delle scelte tecniche

Vue.js è stato scelto come framework per lo sviluppo del componente, grazie alla sua semplicità, flessibilità e reattività. Vue.js permette di gestire facilmente lo stato e le

interazioni dell'utente, garantendo al contempo una struttura modulare e manutenibile. Vale la pena cennare che sono state valutate anche le opzioni di eseguire il progetto con l'ausilio di javascript puro e jQuery.

Vue fornisce un approccio più strutturato e organizzato rispetto a jQuery o JavaScript vanilla. jQuery (per il modo in cui è costruito il suo paradigma), ci avrebbe forzato a trattare il DOM come sorgente della verità, mentre con la versione vanilla avremmo sacrificato la reattività del DOM (senza implementare un gestore di shadow-dom, il che è ben oltre lo scope di questa esercitazione).
Infine, è stato considerato anche il possibile impiego di linguaggi server-side, come PHP o C#, ma le funzionalità richieste erano completamente implementabili dallo stack front-end, dunque è stato evitato il over-engineering.

TailwindCSS è stato selezionato come strumento per la formattazione dell'interfaccia utente grazie alla sua versatilità e alla vasta gamma di classi predefinite. Questo permette di creare rapidamente interfacce moderne e responsive senza dover scrivere molto codice CSS personalizzato.
Inoltre, l'approccio programmatico delle librerie utility-first, quale Tailwind, sembra risonare meglio con il modo di ragionare dei sviluppatori.
L'interfaccia utente è stata sviluppata mobile-first.

Per garantire la qualità e la robustezza del componente, sono stati scelti Jest e Vue Test Utils come strumenti per i test di unità.
Jest offre un ambiente di test completo e flessibile, mentre Vue Test Utils fornisce una serie di utility per testare i componenti Vue.js in modo efficace.

Facendo buon uso del tooling e delle pratiche dello sviluppo moderno, tutto il codice è gestito e versionato su Github.
È raggiungibile pubblicamente (fino alla valutazione del presente progetto) su https://github.com/sidis405/stw-trandafili
Inoltre, per rendere possibile un'immediata visualizzazione, il progetto è deployato automaticamente ad ogni commit di `main` su Netlify
stw-trandafili.netlify.app

Funzionalità non implementate e criticità incontrate

Durante lo sviluppo del progetto, sono state implementate tutte le funzionalità richieste, tra cui la validazione degli input, la generazione della tabella. Sono state aggiunte la copia in memoria e l'esportazione in formato CSV. Tuttavia, ci sono alcune funzionalità aggiuntive

che potrebbero essere implementate per migliorare ulteriormente il componente:

- Migliorare l'accessibilità del componente, assicurandosi che sia compatibile con gli screen reader e supporti adeguatamente la navigazione da tastiera.
- Aggiungere una funzione di ricerca o filtraggio per permettere agli utenti di trovare facilmente specifici CVE nella tabella.
- Implementare la paginazione per gestire un gran numero di CVE, migliorando così le prestazioni e l'esperienza utente.

Durante lo sviluppo del componente, non sono state riscontrate criticità significative. Tuttavia, è importante notare che i test per le funzioni copyTableToClipboard e downloadCSV sono difficili da eseguire a causa delle interazioni con il DOM e il sistema dei file, che non sono facilmente accessibili durante i test. Invece, sono stati testati i metodi generateTable e generateCSVContent che vengono utilizzati da queste funzioni.