Manuale Utente

Versione: 1.0.0 19/03/2025

Redattori	Luca Parise Maria Fuensanta Trigueros Hernandez Malik Giafar Mohamed Ion Cainareanu
Verifica	Maria Fuensanta Trigueros Hernandez Ion Cainareanu Malik Giafar Mohamed
Approvazione	Maria Fuensanta Trigueros Hernandez
Uso	Esterno

nextsoftpadova@gmail.com

Registro dei cambiamenti

Versione	Data	Autore	Descrizione	Verifica
1.0.0	12/05/2025	Ion Cainareanu	Rilascio della versione 1.0.0 e alcune correzioni finali	Malik Giafar Mohamed
0.5.2	09/05/2025	Luca Parise	Inserimento capitolo mancante sulla modifica della tracciabilità	Ion Cainareanu
0.5.1	09/05/2025	Ion Cainareanu	Correzione errori ortografici e typo sulla data del documento e sul versionamento	Luca Parise
0.5.0	04/05/2025	Luca Parise	Sostituite le immagini a seguito di cambiamenti estetici del prodotto	Ion Cainareanu
0.4.4	23/04/2025	Luca Parise	Correzione errori ortografici e inserimento/ aggiornamento delle immagini	Maria Fuensanta Trigueros Hernandez
0.4.3	23/04/2025	Luca Parise, Malik Giafar Mohamed	Miglioramento stile del testo nei vari capitoli	Maria Fuensanta Trigueros Hernandez
0.4.2	19/04/2025	Luca Parise, Malik Giafar Mohamed	Modifica capitolo per installazione immagine docker	Maria Fuensanta Trigueros Hernandez
0.4.1	16/04/2025	Malik Giafar Mohamed, Luca Parise	Modificato il path per le immagini e migliorato il testo	Maria Fuensanta Trigueros Hernandez

Versione	Data	Autore	Descrizione	Verifica
0.4.0	14/04/2025	Luca Parise, Maria Fuensanta Trigueros Hernandez	Aggiunto capitolo della configurazione	Malik Giafar Mohamed
0.3.0	12/04/2025	Luca Parise	Aggiunto capitolo per la disinstallazione del prodotto	Maria Fuensanta Trigueros Hernandez
0.2.1	11/04/2025	Luca Parise	Modifica alle immagini e continuazione stesura dei vari capitoli	Maria Fuensanta Trigueros Hernandez
0.2.0	10/04/2025	Luca Parise	Stesura dei capitoli riguardanti le funzionalità del plug-in	Maria Fuensanta Trigueros Hernandez
0.1.0	19/03/2025	Luca Parise	Creazione documento, stesura introduzione e altri capitoli	Maria Fuensanta Trigueros Hernandez

Indice

Elenco delle immagini

1 Introduzione

"**Requirement Tracker per Visual Studio Code**" è un plug-in progettato per l'omonimo editor, con l'obiettivo di supportare l'analisi dei *requisiti*[G] software. Dato un insieme di requisiti, sia tracciati che non tracciati, il plug-in esegue un'analisi approfondita e fornisce una valutazione del loro grado di implementazione all'interno del codice sorgente.

1.1 Scopo del documento

Il seguente manuale fornisce una guida dettagliata all'installazione, configurazione e utilizzo del plug-in. L'obiettivo è consentire agli utenti di comprendere il funzionamento dello strumento e di sfruttarne le funzionalità per il monitoraggio e la valutazione dei requisiti software all'interno del codice sorgente.

In particolare, il documento si propone di:

- Fornire le istruzioni passo-passo per l'installazione e la configurazione
- Descrivere le funzionalità principali del plug-in
- Spiegare come eseguire l'analisi dei requisiti e interpretare i risultati

1.2 Scopo del prodotto

Il prodotto, un plug-in per Visual Studio Code chiamato "Requirement Tracker", è progettato per automatizzare il *tracciamento dei requisiti*[G] nei progetti software complessi, con un focus particolare sull'ambito embedded. L'obiettivo principale è migliorare la qualità e la chiarezza dei requisiti, fornendo suggerimenti basati sull'analisi di un'*intelligenza artificiale*[G], riducendo al contempo i tempi e gli errori legati alla *verifica*[G] manuale dell'implementazione nel codice sorgente. Il plug-in adotta un'architettura modulare che consente un'estensibilità semplice, rendendolo facilmente adattabile a nuove funzionalità o esigenze future.

1.3 Glossario

I termini ambigui che necessitano di una spiegazione sono contrassegnati da una [G] come apice alla loro prima occorrenza nei documenti. Tutti i termini da glossario sono riportati in ordine alfabetico nell'omonimo documento.

2 Installazione

2.1 Requisiti

Sono necessarie le seguenti tecnologie installate:

- ***Node.js***[G] (versione 20.x[G] o superiore)
- ***npm***[G] (versione 10.9.0 o superiore)
- ***TypeScript***[G] (versione 5.8.2 o superiore)
- ***Visual Studio Code***[G] (versione 1.95.0 o superiore)
- ***Ollama***[G] (versione 0.6.5 o superiore)

2.1.1 Preparazione per l'installazione dell'estensione

Prima di procedere con la creazione del file *.vsix*[G], è necessario:

- **1)** Posizionarsi nella cartella del codice sorgente dell'estensione denominata "Requirement Tracker - Plugin"
- **2)** Installare i moduli `node_modules` necessari per il funzionamento dell'estensione eseguendo il comando:

```
$ npm install
```

- **3)** Convertire il codice *typescript*[G] in *javascript*[G] eseguendo il comando:

```
$ npx tsc
```

2.1.2 Come creare il file .vsix

Per creare un *file* ***.vsix***[G] (che è il pacchetto installabile di una estensione per Visual Studio Code), è necessario usare lo strumento ***vsce***[G] (Visual Studio Code Extension Manager).
Di seguito i vari passaggi per creare tale pacchetto:

- **1)** Aprire il terminale e installare vsce (Visual Studio Code Extension Manager).
  Il flag **-g** specifica che l'installazione deve essere globale, rendendo il comando vsce accessibile da qualsiasi directory del sistema. Questo è necessario perché vsce è uno strumento da riga di comando usato per creare e pubblicare estensioni per Visual Studio Code.

```
$ npm install -g vsce
```

- **2)** Navigare con il terminale fino alla cartella della tua estensione. Quindi vai nella cartella in cui si trova il file **package.json** della tua estensione. Puoi usare il seguente comando per navigare nella cartella:

```
$ cd /path/to/your/extension
```

- **3)** Effettuare la *build*[G] del file .vsix con il seguente comando, il quale genera un file .vsix, ad esempio **nome-estensione-1.0.0.vsix**:

```
$ vsce package
```

- **4)** (Facoltativo) Ignorare i file non necessari. È possibile usare un file `.vscodeignore` nella *root directory*[G] della tua estensione per escludere i file o cartelle (come node_modules, test/, ecc.) dal pacchetto finale.

2.1.3 Come installare il plug-in

Una volta creato il pacchetto .vsix come delineato nel capitolo precedente, è possibile installare il plug-in in Visual Studio Code. Per farlo, segui questi passaggi:

- **1)** Aprire Visual Studio Code
- **2)** Aprire un terminale e navigare nella cartella in cui è stato salvato il file generato in formato vsix. Potete farlo usando il seguente comando:

```
$ cd /path/to/your/extension
```

- **3)** Usare quindi il seguente comando per installare il pacchetto .vsix:

```
$ code --install-extension nome-estensione-1.0.0.vsix
```

- **4)** Se l'estensione installata non è visualizzata, provare a riavviare Visual Studio Code
- **5)** Verificare che l'estensione sia stata installata correttamente: È possibile farlo accedendo alla sezione delle estensioni di Visual Studio Code e cercando il nome del plug-in oppure, tramite terminale, utilizzando il seguente comando:

```
$ code --list-extensions
```

A questo punto, l'estensione dovrebbe risultare installata con successo. Per utilizzarla, è necessario che Ollama sia attivo e configurato con un *modello LLM*[G]. Per ulteriori dettagli, consultare il capitolo successivo.

2.1.4 Come attivare Ollama

Se Ollama non è stato ancora scaricato ed installato, si consiglia di consultare la documentazione ufficiale disponibile al seguente link "https://ollama.com/" (10/05/2025).

Per utilizzare l'estensione, è necessario che Ollama sia attivo e in ascolto sulla porta standard 11434. A tal fine, è possibile cercare e avviare l'icona di Ollama una volta installato oppure, aprire un terminale ed eseguire il comando

```
$ ollama run nome_modello
```

dove "nome_modello" rappresenta il modello utilizzato. Di default, l'estensione utilizza il modello **llama3.2:3b**, ma è possibile modificarlo in qualsiasi momento accedendo alle impostazioni dell'estensione. Per ulteriori dettagli, si rimanda al capitolo dedicato alla configurazione del modello LLM.

2.1.5 Installazione dell'*immagine docker*[G]

Per poter utilizzare correttamente il plug-in, è necessario eseguire un *server*[G] esterno che gestisca le richieste in arrivo. Questo server è realizzato come una REST *API*[G] che può essere facilmente avviata e gestita tramite *Docker*[G], una tecnologia che consente di creare ambienti isolati e replicabili chiamati *container*[G].

L'utilizzo di Docker permette di semplificare il processo di configurazione, evitando problemi legati a incompatibilità tra versioni di *librerie*[G] o ambienti operativi. Seguendo i passaggi descritti in questa sezione, sarai in grado di:

- Installare Docker sul tuo sistema (se non già presente)
- Costruire localmente l'*immagine*[G] dell'applicazione partendo da un file *Dockerfile*[G]
- Verificare che l'immagine sia stata correttamente creata
- Avviare un container funzionante, pronto a ricevere ed elaborare richieste

L'intera procedura richiede pochi comandi da terminale ed è stata progettata per essere semplice e accessibile anche a chi non ha una lunga esperienza con Docker. Assicurati di avere una connessione a internet attiva durante l'installazione e, se lavori su un sistema *Linux*[G], di avere i permessi necessari per eseguire i comandi come amministratore.

- **1)** Prima di tutto, è necessario verificare che Docker sia installato sul sistema. È possibile scaricare Docker Desktop (per Windows e macOS) oppure installare Docker Engine (per Linux) seguendo la guida ufficiale disponibile al link "https://docs.docker.com/get-docker/" (04/05/2025).

Una volta installato, verificare che Docker sia correttamente attivo eseguendo da terminale il comando:

```
$ docker --version
```

Se il comando restituisce la versione di Docker, l'installazione risulta completata con successo.

- **2)** Scaricare il Dockerfile presente all'interno della *repository*[G] del prodotto e costruire localmente l'immagine attraverso il comando (assicurarsi di essere nella directory in cui si trova il file):

```
$ cd /path/del/server/Requirement Tracker - API
```

- **3)** Una volta nella cartella del codice sorgente (all'interno della directory denominata "Requirement Tracker - API") è necessario costruire l'immagine Docker eseguendo il comando:

```
$ docker build -t requirement-tracker-api
```

- **4)** Verificare che l'immagine sia presente usando il comando

```
$ docker images
```

- **5)** Per eseguire l'immagine e creare il container utilizzare il comando:

```
$ docker run -p porta:4000 -e VAR=VAL requirement-tracker-api
```

Nel caso il nome dell'immagine sia cambiato, allora sarà necessario cambiare anche il nome nei comandi sopra. Lo stesso vale per la porta.

Un altro metodo per avviare il server tramite docker consiste nel creare un "**docker compose file**". Si tratta di un file in formato *yaml*[G] il quale permette di modificare più facilmente i *parametri*[G] di esecuzione del container con l'immagine del server. Inoltre, consente di configurare modalità di *deployment*[G] più complesse, come l'uso di più container o l'integrazione con altri servizi.

Tutto questo non è possibile con il comando `docker run` in quanto questo permette di eseguire solo un container alla volta.

2.1.5.1 Attivare il server manualmente

Per fare questo è necessario posizionarsi nella cartella Requirement Tracker - API in cui si trova il codice in *typescript*[G] che gestisce il server. Poiché viene usato node, sarà necessario convertire il codice in *javascript*[G]. Per fare ciò, aprire un terminale, posizionarsi sulla cartella del codice ed eseguire il comando:

```
$ npm install
$ npm run build
```

Questi comandi installeranno le *dipendenze*[G] (`npm install`) e convertiranno il codice typescript in codice javascript (`npx run build`) in una cartella chiamata dist.

```
$ npm run start:prod
```

Questo comando avvierà il server in modalità production che rimarrà in ascolto delle richieste e le invierà ad Ollama. Per le modalità alternative come sviluppo o *debug*[G] ci sono i comandi:

```
$ npm run start:dev
$ npm run start:debug
```

3 Configurazione

3.1 Modello LLM

L'utente può impostare il modello da utilizzare nelle varie operazioni come mostrato in figura:

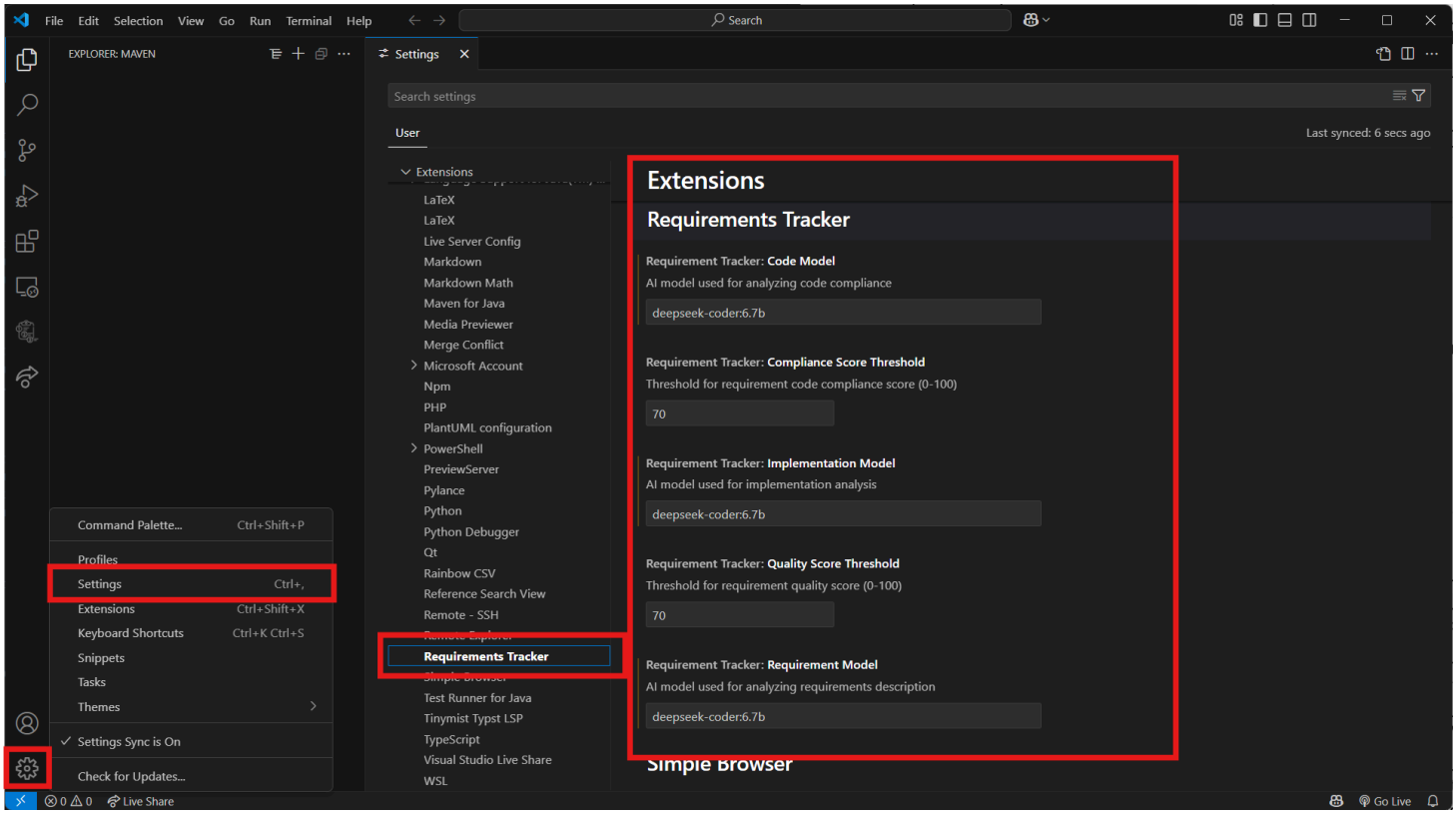

Figure 1: Configurazione del modello LLM

Il modello *LLM*[G] di Requirement Tracker per Visual Studio Code viene eseguito tramite Ollama, permettendo un'analisi locale dei requisiti software. Visto il rapido progresso e la continua uscita di nuovi modelli, il plug-in permette all'utente di poter scegliere quale modello utilizzare, purché questo sia installato all'interno della macchina.
Il plug-in permette di impostare un modello specifico per ognuna delle seguenti attività

- per analizzare la "code compliance"
- per analizzare l'implementazione il quale deve essere un modello che supporti la funzione di embedding (nomic-embed-text:latest o simili)
- per analizzare la descrizione dei requisiti

Per poter configurare il modello, l'utente deve accedere alle impostazioni della estensione installata ed inserire nei tre campi disponibili il nome del modello che vuole usare, come mostrato nelle figura di seguito :

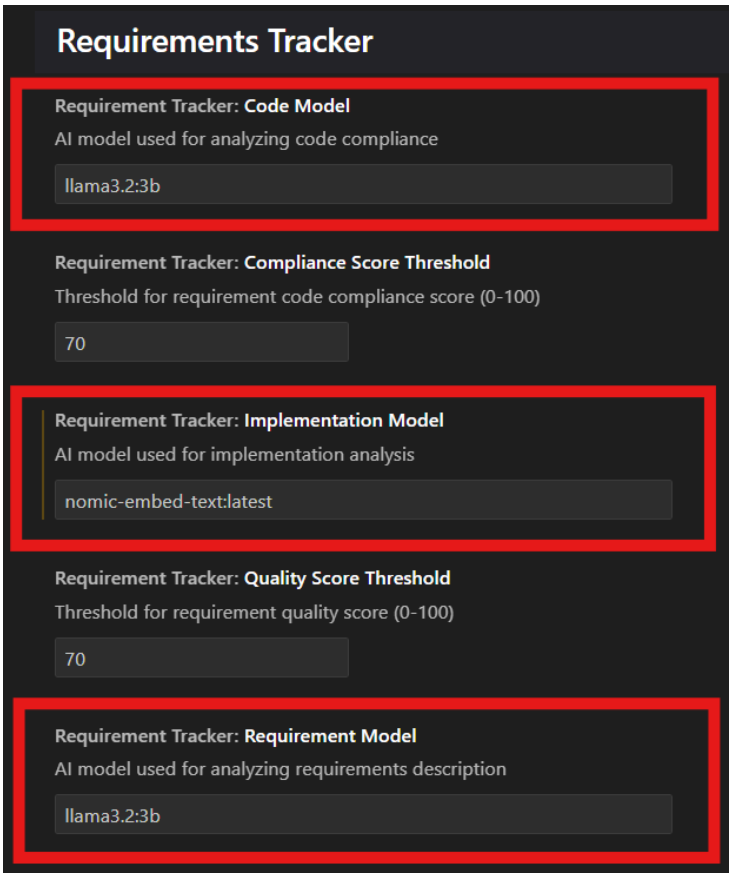

Figure 2: Configurazione dei modelli

La scelta di un modello specifico per ogni attività permette di ottimizzare le prestazioni e i risultati dell'analisi, in quanto diversi modelli possono essere più adatti a compiti specifici. Inoltre, è fondamentale la scelta del modello per l'implementazione, il quale deve fornire una funzione di embedding senza la quale il plug-in potrebbe non funzionare correttamente.

3.2 Soglia di accettazione

Un'altra configurazione permessa dal plug-in è l'impostazione della *soglia di accettazione*[G] di un requisito. Ad ogni analisi, infatti, ogni requisito riceve un punteggio da 0-100 e viene considerato "passed" o "not passed" a seconda del superamento di tale soglia. Per impostarla, andate sulle impostazioni ("Settings") del plug-in e inserite nel campo riguardante la soglia il valore che preferite come mostrato di seguito in figura :

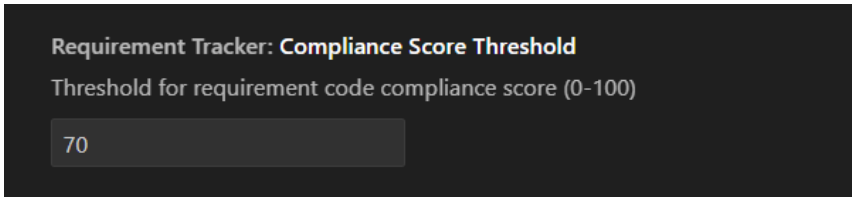

Figure 3: soglia di accettazione per l'analisi del requisito

Un altro tipo di soglia è quella riguardante la descrizione testuale del requisito, la quale viene usata per valutare la qualità del testo dello stesso. Per impostarla, andate sulle impostazioni ("Settings") del plug-in e inserite nel campo riguardante la soglia il valore che preferite come mostrato di seguito in figura :

Figure 4: Soglia di accettazione per il testo del requisito

4 Istruzioni all'uso

4.1 Prerequisiti

Se si è arrivati qui, allora il plug-in risulta installato e pronto per l'uso.
Quindi, cliccare sull'icona del plug-in presente nella colonna a sinistra della finestra di Visual Studio Code per aprire la finestra di analisi dei requisiti. L'icona è mostrata nella figura sottostante.

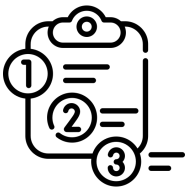

Figure 5: Icona del plug-in

4.2 Importazione dei requisiti

Per poter usare il plug-in è necessario importare i requisiti da analizzare. Per fare ciò, cliccate sull'icona denominata **"Load CSV"** mostrata in figura.

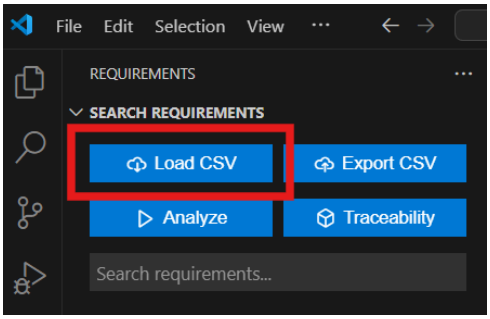

Figure 6: Icona per l'importazione dei requisiti

Una volta cliccato sull'icona verrà aperto il *file system*[G] di sistema da cui selezionare il **file *CSV***[G] contenente i requisiti. In caso venga selezionato un file di formato diverso o non valido, il plug-in restituirà un messaggio di errore.

Una volta importati, i requisiti verranno presentati in una lista nella sezione Requirements, come mostrato nella seguente figura:

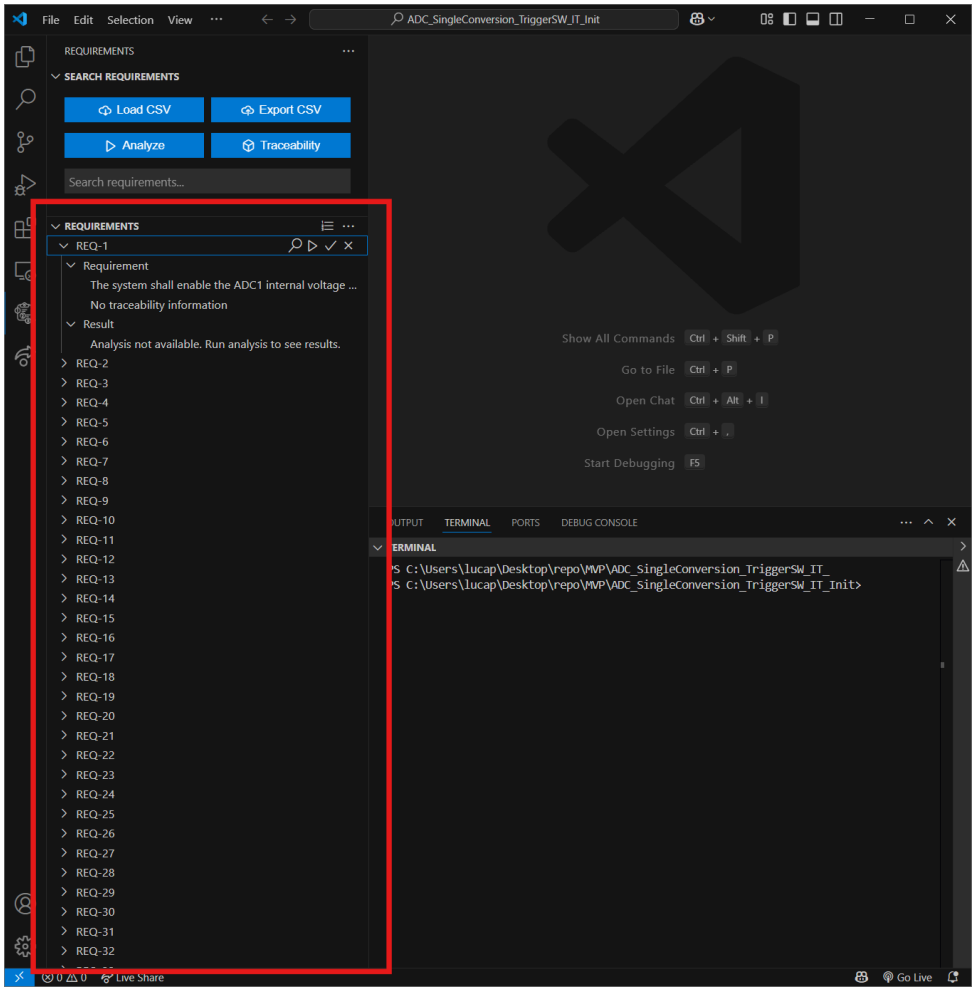

Figure 7: Requisiti importati

La lista, una volta selezionato un file CSV valido, elencherà per ogni requisito all'interno del documento:

- ID
- Descrizione
- Tracciabilità nel codice sorgente (solo se presente nel file CSV precedentemente caricato)

In caso di requisiti molto lunghi o di più righe sarà sufficiente posizionare il cursore sopra il requisito per visualizzare il testo completo.

4.3 Analisi dell'implementazione

Una volta importati i requisiti è possibile procedere con l'analisi dell'implementazione. Questa funzionalità serve a **tracciare** i requisiti nel codice sorgente qualora questi non fossero già stati tracciati (funziona anche nel caso lo fossero).
Per farlo, cliccate sull'icona **"Traceability"** mostrata in figura.

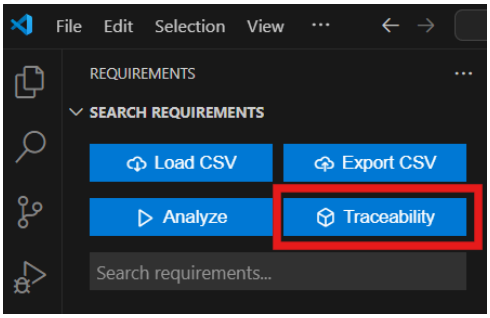

Figure 8: Bottone per l'analisi dell'implementazione (tracciamento)

Una volta eseguita e completata l'analisi, ad ogni requisito verranno associate delle linee di codice che indicano il tracciamento del requisito all'interno del codice sorgente.
In caso di requisiti non tracciati, il plug-in restituirà un messaggio di errore. La figura seguente mostra il tracciamento nella sezione "Requirement":

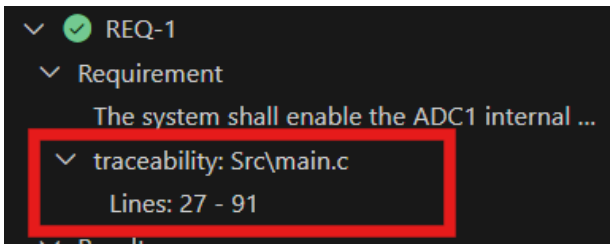

Figure 9: Tracciabilità di un requisito

4.4 Analisi dei requisiti

Questa funzionalità serve a valutare la qualità dei requisiti e il loro grado di implementazione nel codice sorgente. Una volta importati i requisiti, cliccate sull'icona **"Analyze"** mostrata in figura.

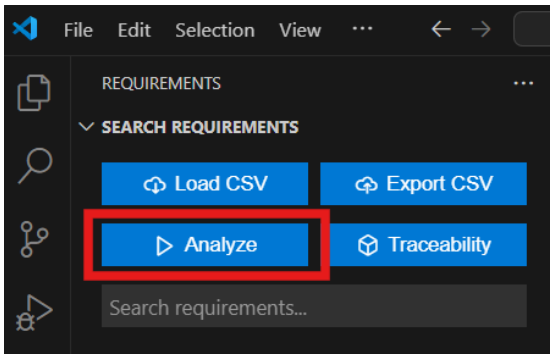

Figure 10: Bottone per l'analisi dei requisiti

In questo modo il plug-in eseguirà un'analisi approfondita di tutti i requisiti presenti nella lista. Potrete vedere in basso a destra della finestra un'icona di avanzamento che indica il progresso dell'analisi. Di seguito la figura della barra di progessione e una possibile visualizzazione dei requisiti al termine dell'analisi.

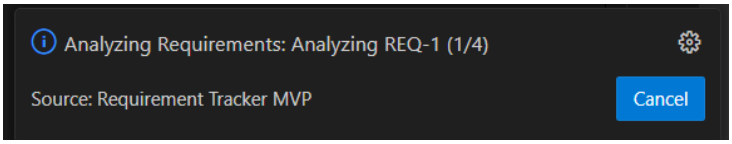

Figure 11: Stato di avanzamento analisi dei requisiti

Figure 12: Visualizzazione tipo del risultato dell'analisi

Alla fine dell'analisi ogni requisito nella lista verrà aggiornato con i seguenti campi:

- Result
  - Global Result: Passed/Not Passed
  - Requirement Quality: 0-100
  - Code Compliance: 0-100
  - Issues:
    - Issue 1
    - Issue ...
  - Suggestions:
    - Suggestion 1
    - Suggestion ...

Inoltre, ad ogni requisito sarà associato un'icona che indica se lo stesso è passato o meno all'analisi. Di seguito una figura che mostra una generica lista a seguito dell'analisi.

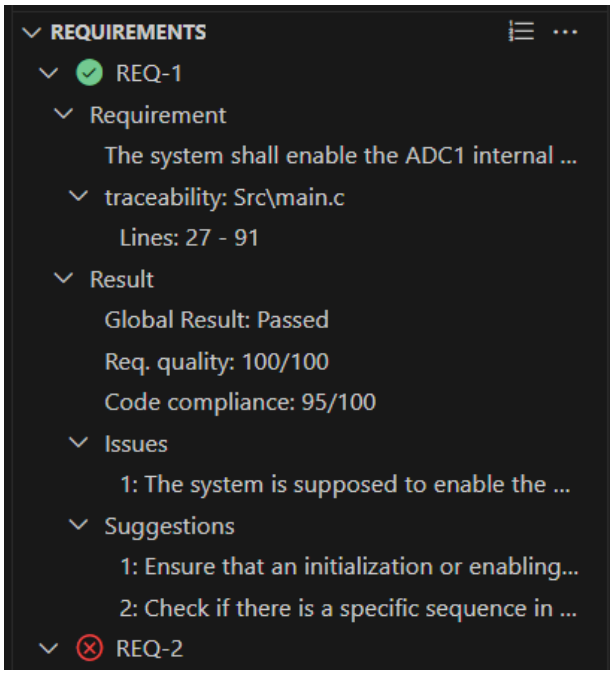

Figure 13: Risultato dell'analisi

4.5 Esportazione requisiti

Il plug-in presenta anche la funzione di esportazione, in formato CSV, dei risultati dell'analisi, qualora voleste salvarli. Per fare ciò premete sul pulsante **"Export CSV"** mostrato nella seguente figura.

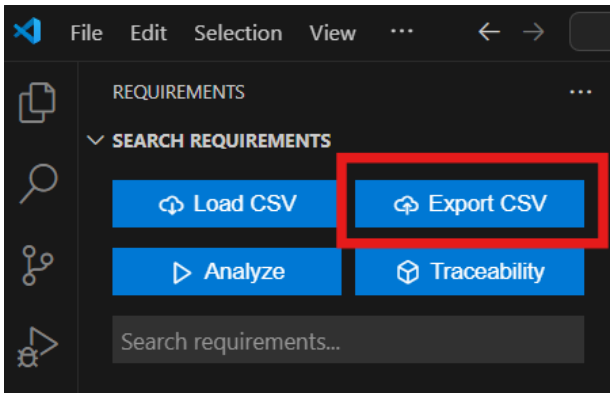

Figure 14: Bottone per l'esportazione dei requisiti

Quindi, una volta cliccato sul pulsante, verrà aperto il file system di sistema da cui selezionare la cartella in cui salvare il file CSV.

4.6 Ricerca requisiti

Il plug-in presenta anche la funzione di ricerca dei requisiti attraverso una barra di ricerca in cui inserire il codice del requisito ricercato. Per farlo, basta cliccare sul campo di input **"Search Requirements"** mostrata in figura.

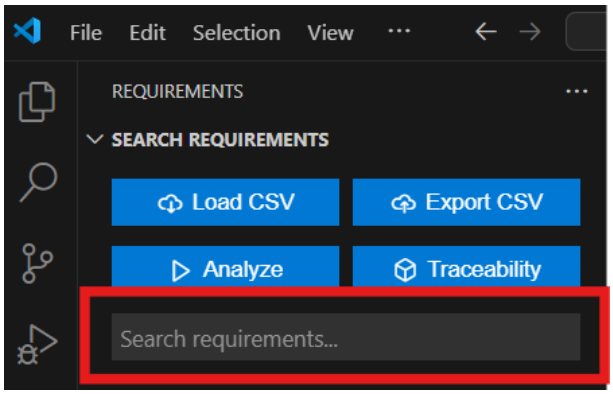

Figure 15: Campo di input per la ricerca dei requisiti

La lista **Requirements** mostrerà solo i requisiti che contengono il codice (o parte di esso) inserito nella barra di ricerca.

4.7 Analisi di un singolo requisito

Il plug-in offre la possibilità di analizzare un singolo requisito. Per farlo, basta cliccare sull'icona presente vicino al requisito che si desidera analizzare come mostrato in figura.

Figure 16: Bottone per l'analisi di un singolo requisito

4.8 Analisi dell'implementazione di un singolo requisito

Il plug-in offre la possibilità di analizzare l'implementazione (tracciamento) di un singolo requisito. Per farlo, basta cliccare sull'icona presente vicino al requisito che si desidera analizzare come mostrato in figura.

Figure 17: Bottone per l'analisi dell'implementazione di un singolo requisito

4.9 Filtraggio dei requisiti

È possibile filtrare i requisiti in base al loro codice o al loro stato di analisi.

Ci sono tre diversi modi in cui possono essere filtrati i requisiti:

- In ordine **Default** (a seconda del loro codice), premendo il pulsante **"Default Order"** mostrato in figura

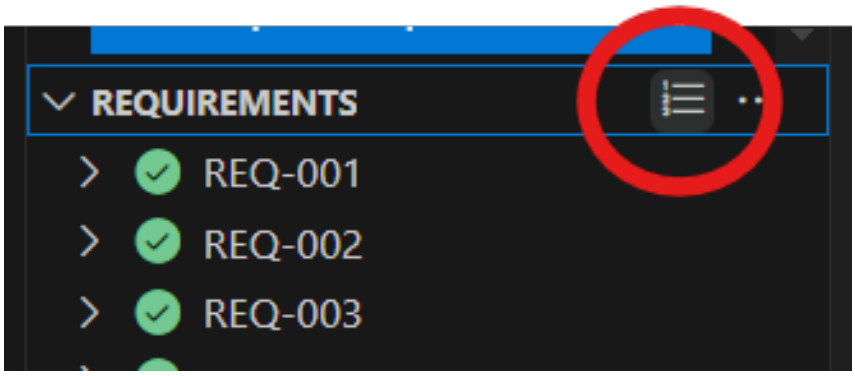

Figure 18: Bottone per organizzare i requisiti in ordine di default

- A seconda del loro **stato di analisi**:
  - **Analizzati**, usando il bottone **"Show analyzed first"** mostrato in figura

Figure 19: Bottone per mostrare prima i requisiti analizzati

  - **Non Analizzati**, usando il bottone **"Show unanalyzed first"** mostrato in figura

Figure 20: Bottone per mostrare prima i requisiti non analizzati

4.10 Approvazione/Disapprovazione di un requisito manuale

Il plug-in offre la possibilità di segnare manualmente un requisito come approvato o non approvato. Per farlo, basta cliccare sull'icona presente vicino al requisito che si desidera approvare come mostrato in figura :

Figure 21: Bottone per approvare manualmente un requisito

Figure 22: Bottone per disapprovare manualmente un requisito

Nel caso non sia ancora presente il tracciamento del requisito nel codice sorgente, il plug-in restituirà un messaggio di errore e il requisito non verrà contrassegnato come approvato (o non approvato).
L'approvazione è segnata da un icona blu mentre la disapprovazione è segnata da un icona rossa come mostrato nelle due figure seguenti:

Figure 23: Icona di requisito approvato

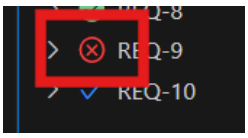

Figure 24: Icona di requisito non approvato

4.11 Modifica del tracciamento di un requisito

Il plug-in offre la possibilità di modificare il tracciamento di un requisito specifico. Per farlo, basta cliccare sull'icona a forma di matita presente vicino al requisito. Più nello specifico, all'interno della sezione **traceability** del requisito stesso, come mostrato in figura:

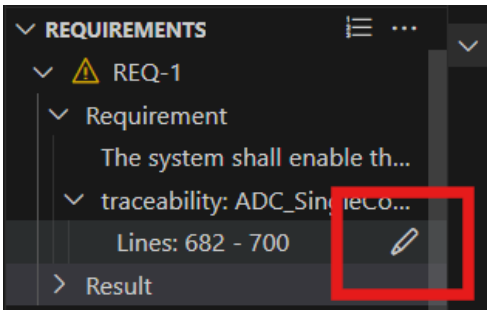

Figure 25: Bottone per modificare il tracciamento di un requisito

La funzionalità permette di modificare tre parametri:

- il percorso del file associato al requisito
- il numero di riga iniziale del requisito
- il numero di riga finale del requisito

Una volta cliccato sull'icona, il plug-in posizionerà il focus dell'utente sulla "**command palette**" di Visual Studio Code, permettendo di modificare i tre parametri. Una volta inserito il nuovo valore, premere "Invio" per confermare la modifica. Di seguito un esempio di come apparirà la command palette una volta cliccato sull'icona:

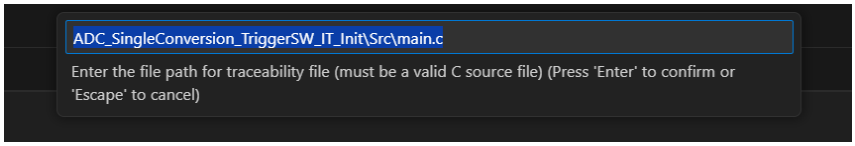

Figure 26: Command palette per la modifica del percorso del file

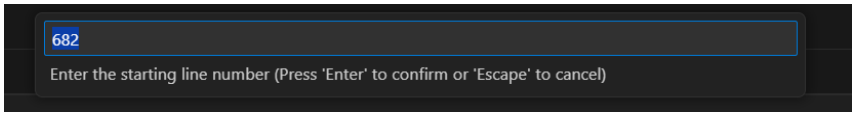

Figure 27: Command palette per la modifica della linea di inizio del requisito

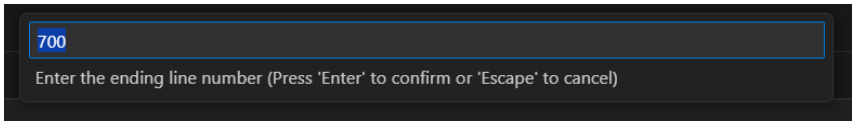

Figure 28: Command palette per la modifica della linea di fine del requisito

5 Disinstallazione del plug-in

La disinstallazione del plug-in risulta molto semplice. Per procedere, è necessario utilizzare l'interfaccia di Visual Studio Code. In basso a sinistra, cliccare sull'icona delle impostazioni a forma di ingranaggio e selezionare **"Extensions"**. A questo punto, comparirà la lista delle estensioni installate e raccomandate come mostrato in figura:

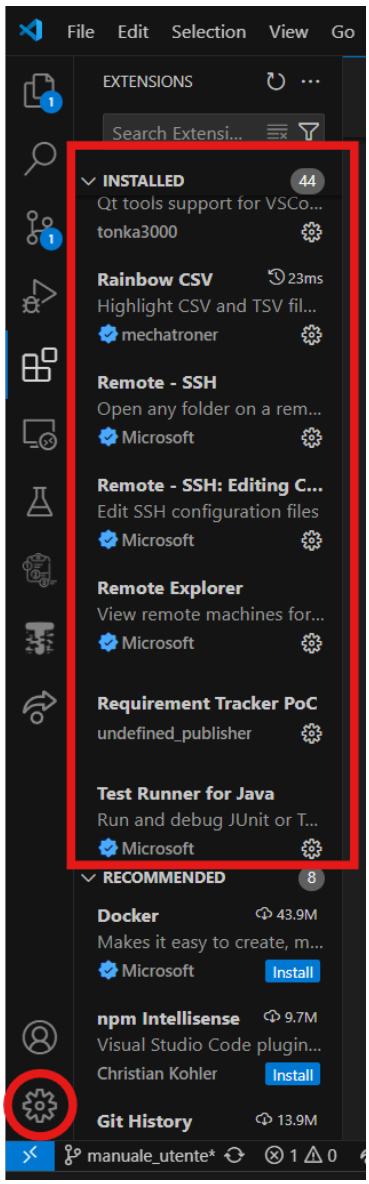

Figure 29: Icona delle estensioni

Successivamente, individuare l'estensione **Requirement Tracker** e cliccare sull'icona a forma di ingranaggio vicino ad essa. Verrà visualizzato un menù a tendina in cui selezionare **"Uninstall"** come mostrato in figura:

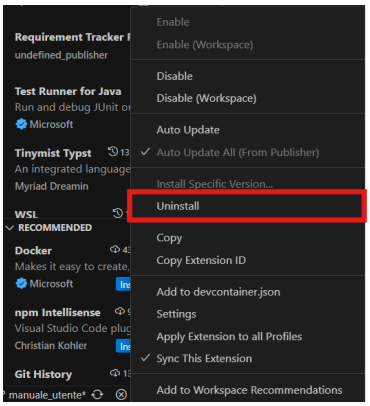

Figure 30: Icona del plug-in

Dopo aver cliccato su Uninstall, l'estensione verrà disinstallata e non sarà più visibile nella lista delle estensioni installate.