Analisi dei Requisiti

Versione: 1.0.0 23/11/2024

Redattori

Ion Cainareanu
Luca Parise
Marco Perazzolo
Malik Giafar Mohamed

Verifica

Ion Cainareanu
Luca Parise
Marco Perazzolo
Malik Giafar Mohamed
Stefano Baso
Maria Fuensanta Trigueros Hernandez

Approvazione

Uso Esterno

nextsoftpadova@gmail.com

Registro dei cambiamenti

Versione	Data	Autore	Descrizione	Verifica
1.0.0	08/03/2025	Ion Cainareanu	Aggiunti i termini del glossario e rilasciata la versione 1.0.0 del documento	
0.6.3	07/03/2025	Marco Perazzolo	Affinamento degli extension points nei diagrammi UML	Stefano Baso
0.6.2	26/02/2025	Ion Cainareanu	Outline automatico per le tabelle	Malik Giafar Mohamed
0.6.1	26/02/2025	Malik Giafar Mohamed	Aggioramento data di creazione documento	Ion Cainareanu
0.6.0	25/02/2025	Luca Parise, Ion Cainareanu, Marco Perazzolo	Modifiche post incontro con l'azienda	Malik Giafar Mohamed
0.5.0	18/02/2025	Luca Parise	Inserimento requisiti	Ion Cainareanu
0.4.0	11/02/2025	Marco Perazzolo	Inserimento dei diagrammi Use Case	Ion Cainareanu
0.3.1	06/02/2025	Marco Perazzolo	Finalizzazione Use Case testuali	Malik Giafar Mohamed
0.3.0	06/01/2025	Ion Cainareanu	Stesura iniziale degli Use Case	Marco Perazzolo, Luca Parise
0.2.0	30/12/2024	Ion Cainareanu	Stesura dell'Introduzione e Descrizione	Stefano Baso, Malik Giafar Mohamed
0.1.1	04/12/2024	Luca Parise	Aggiunta indice e creazione struttura tabella per use case	Malik Giafar Mohamed
0.1.0	23/11/2024	Malik Giafar Mohamed	Creazione Documento	Ion Cainareanu, Maria Fuensanta Trigueros Hernandez

Indice

1 Introduzione

1.1 Scopo del documento

Lo scopo del presente documento è fornire una descrizione completa e dettagliata degli obiettivi, delle funzionalità e delle caratteristiche tecniche del progetto **Requirement Tracker - Visual Studio Code Plug-in**, con particolare attenzione all'utilizzo dell'*UML*[G] per la modellazione dei *casi d'uso*[G]. Il documento funge da riferimento per tutti gli *stakeholder*[G] coinvolti, descrivendo il contesto operativo, i requisiti funzionali e non funzionali, nonché le linee guida tecnologiche necessarie per lo sviluppo del *plug-in*[G]. I casi d'uso saranno descritti utilizzando una struttura standardizzata, che includerà il nominativo del caso, gli attori principali, le *precondizioni*[G], le *postcondizioni*[G], lo *scenario principale*[G] e gli eventuali *scenari alternativi*[G] o sottocasi. Questa struttura garantisce chiarezza e coerenza, facilitando la comprensione e la tracciabilità delle funzionalità principali del sistema. Il documento intende inoltre fornire una visione condivisa del progetto, ponendo le basi per una *pianificazione*[G] e un'implementazione efficaci.

1.2 Scopo del prodotto

Lo scopo di **Requirement Tracker - Visual Studio Code Plug-in** è affrontare il problema della complessità nella gestione e nel *tracciamento dei requisiti*[G] nei progetti software di grandi dimensioni. Nei *codebase*[G] estesi, la verifica manuale della copertura e dell'implementazione dei requisiti è un processo lungo e soggetto a errori, spesso complicato dalla qualità insufficiente con cui i requisiti stessi vengono definiti. Questo può portare a malintesi e problemi durante l'implementazione, compromettendo l'allineamento tra specifiche e funzionalità sviluppate. Il plug-in mira a risolvere queste difficoltà automatizzando il tracciamento dei requisiti nel codice sorgente, migliorando la qualità della loro definizione e semplificando l'identificazione delle aree di mancata o errata implementazione. In particolare, offre strumenti per integrare requisiti tecnici derivati da manuali e datasheet di componenti hardware, fornendo analisi automatizzate e suggerimenti per rendere i requisiti più chiari, specifici e strutturati. Grazie a questo, sviluppatori potranno garantire una gestione più efficace dei requisiti, riducendo errori e aumentando la coerenza tra specifiche e implementazione.

1.3 Glossario

Per evitare ambiguità relative alle terminologie utilizzate è stato creato un documento denominato **Glossario**. Questo documento comprende tutti i termini tecnici scelti dai membri del gruppo e utilizzati nei vari documenti con le relative definizioni. Tutti i termini inclusi in questo glossario vengono segnalati all'interno del documento con l'apice [G] accanto alla parola.

2 Descrizione

2.1 Obiettivi del prodotto

L'obiettivo del progetto è realizzare un plug-in per *Visual Studio Code*[G] che consenta di tracciare e verificare l'implementazione dei requisiti di progetto, basandosi su analisi automatizzate del codice sorgente e sui requisiti tecnici espressi in documenti di riferimento, mediante l'utilizzo di tecnologie avanzate come modelli *LLM*[G] di *AI*[G]. Il plug-in sarà supportato da *API REST*[G] che si interfacciano con *Ollama*[G], fornendo un'infrastruttura flessibile e scalabile per l'integrazione di modelli di AI e garantendo un'elaborazione efficiente e sicura delle analisi richieste.

2.2 Funzionalità del prodotto

Il plug-in sarà utilizzato dal *programmatore*[G] per analizzare i requisiti implementati nel codice sorgente. Sia i requisiti che il codice saranno analizzati da vari modelli LLM reperibili attraverso la piattaforma di Ollama, grazie alle API REST che interagiscono con Ollama.
Le funzionalità implementate nell'applicazione includono:

- Importazione del file dei requisiti in formato *CSV*[G]
- Richiesta di *analisi dei requisiti*[G] tramite un modello LLM;

- Valutazione qualitativa dei requisiti;
- Visualizzazione grafica dei risultati dell'analisi;
- Filtraggio dei risultati dell'analisi;
- Possibilità di eseguire l'analisi su un *requisito*[G] specifico;
- Esportazione dei risultati dell'analisi in formato CSV;
- Ricerca dell'implementazione dei requisiti nel codice sorgente;
- *Analisi semantica*[G] dei requisiti e del codice sorgente;
- Suggerimenti per migliorare la qualità dei requisiti e del codice;
- Possibilità di modificare il modello LLM che analizza i requisiti ed il codice;
- Possibilità di modificare l'*endpoint*[G] di collegamento al server Ollama;
- Possibilità di modificare la soglia di accettazione relativa alla qualità del codice.

2.3 Utenti e caratteristiche

In seguito a un incontro con il proponente, è stato discusso come il plug-in possa essere utilizzato principalmente da un utente che ricopre il ruolo di programmatore. Di conseguenza, si è deciso di focalizzare le funzionalità del plug-in per rispondere alle esigenze di questa categoria di utenti. È stato inoltre specificato che non devono essere fatte assunzioni sulle competenze tecniche dell'utente riguardo all'uso di *Visual Studio Code*[G]. Pertanto, il plug-in deve essere progettato per essere il più intuitivo possibile, con un processo di installazione semplice e accessibile.

3 Use Case

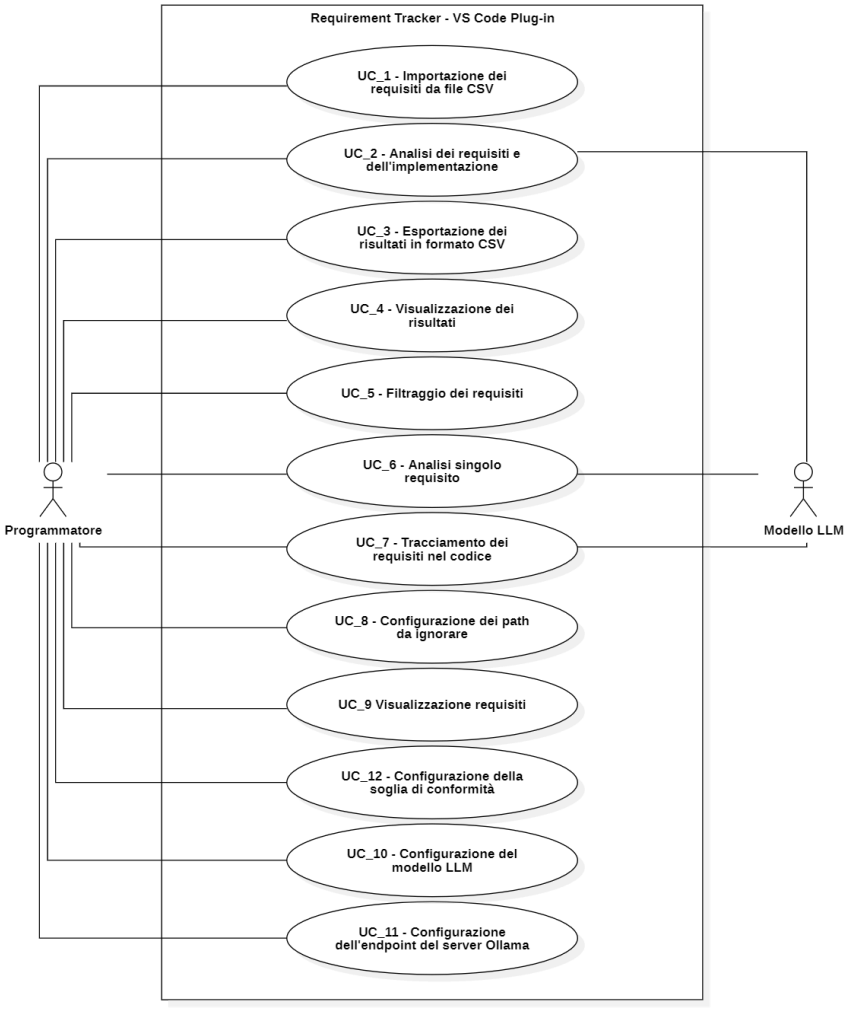

Figure 1: Panoramica delle funzionalità principali del plugin.

3.1 Obiettivi

Questa sezione si propone di identificare e descrivere i casi d'uso derivati dall'analisi del *capitolato*[G] d'appalto selezionato dal gruppo. In particolare, vengono definiti gli attori principali e le funzionalità ad essi associate.

3.2 Attori

L'applicazione è progettata con un unico *attore*[G], il **Programmatore**, esso rappresenta un utente che utilizza il plug-in *Requirement Tracker - Visual Studio Code Plug-in* per importare, analizzare e tracciare l'implementazione dei requisiti software all'interno del codice sorgente di un progetto.

3.3 UC_1 - Importazione da file CSV

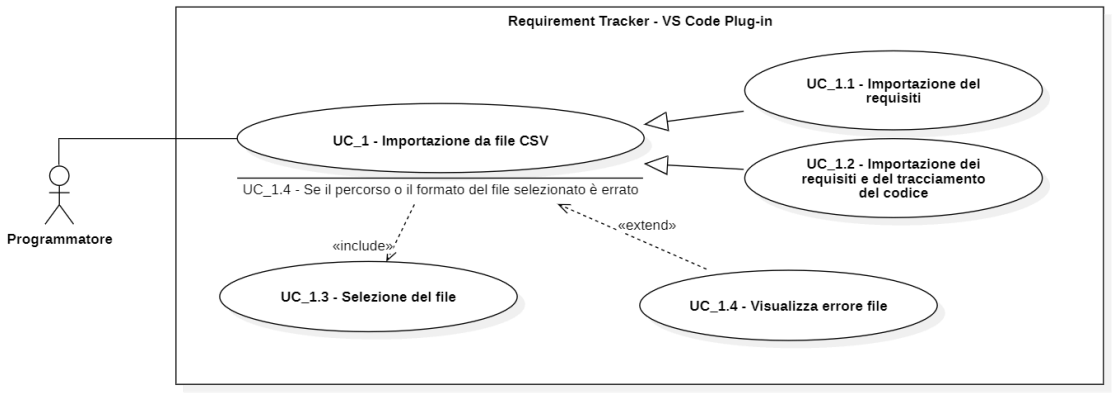

Figure 2: UC_1 - Importazione dei requisiti da file

Attori: Programmatore.

Precondizioni:
- L'utente ha aperto un progetto in Visual Studio Code.
- Il file dei requisiti è in formato CSV valido.
- Il plug-in è installato e attivo in Visual Studio Code.

Postcondizioni:
- I requisiti (e, se presente, il tracciamento) vengono importati e sono visualizzabili nel sistema.

Scenario principale:
1. In base all'opzione di importazione selezionata dall'utente, il sistema applica una delle seguenti specializzazioni:
   - Se l'utente seleziona la voce "Importa requisiti", il flusso procede con [UC_1.1].
   - Se l'utente seleziona la voce "Importa requisiti con tracciamento", il flusso procede con [UC_1.2].

Estensioni:
- **UC_1.4 - Visualizza errore file** : Se il file non rispetta il formato previsto o risulta malformato, il sistema notifica l'errore all'utente e richiede di selezionare un file corretto.

3.4 UC_1.1 - Importazione dei requisiti da file CSV

Attori: Programmatore.

Precondizioni:
- L'utente ha aperto un progetto in Visual Studio Code.
- Il file dei requisiti è in formato CSV valido.
- Il plug-in è installato e attivo in Visual Studio Code.

Postcondizioni:
- I requisiti sono importati e sono visualizzabili nel sistema.

Scenario principale:
1. L'utente seleziona l'opzione "Importa requisiti".
2. Il sistema apre un *file explorer*[G].
3. L'utente seleziona il file CSV da importare [UC_1.3]
4. Il sistema verifica la validità del file e importa i dati (ID, testo, di ogni requisito).
5. I requisiti importati vengono mostrati in una vista strutturata [UC_9].

3.5 UC_1.2 - Importazione dei requisiti e del tracciamento da file CSV

Attori: Programmatore.

Precondizioni:

- L'utente ha aperto un progetto in Visual Studio Code.
- Il file dei requisiti è in formato CSV valido, e contiene le informazioni di tracciamento.
- Il plug-in è installato e attivo in Visual Studio Code.

Postcondizioni:

- I requisiti e le relative informazioni di tracciamento sono importate e visualizzabili nel sistema.

Scenario principale:

1. L'utente seleziona l'opzione "Importa requisiti con tracciamento".
2. Il sistema apre un file explorer.
3. L'utente seleziona il file CSV da importare [UC_1.3]
4. Il sistema verifica la validità del file e importa i dati (ID, testo, file, intervallo righe, di ogni requisito).
5. I requisiti importati vengono mostrati in una vista strutturata [UC_9].

3.6 UC_1.3 - Selezione del file

Attori: Programmatore.

Precondizioni:

- Il file explorer è stato aperto dal sistema.

Postcondizioni:

- Il file CSV scelto dall'utente viene registrato per l'importazione.

Scenario principale:

1. Il sistema apre il file explorer.
2. L'utente naviga tra le cartelle e individua il file CSV desiderato.
3. L'utente seleziona il file CSV.
4. Il sistema registra la scelta e procede con l'importazione scelta [UC_1.1] oppure [UC_1.2].

3.7 UC_1.4 - Visualizza errore file importazione

Attori: Programmatore.

Precondizioni:

- L'utente ha selezionato un file CSV da importare [UC_1.3].

Postcondizioni:

- L'utente viene informato che il file non è valido.

Scenario principale:

1. Il sistema verifica il file e rileva che è malformato o non valido (per esempio: mancano le colonne ID e descrizione).
2. L'importazione del file fallisce
3. Il sistema mostra un messaggio di errore esplicativo e richiede di selezionare un file valido.

3.8 UC_2 - Analisi dei requisiti e della loro implementazione

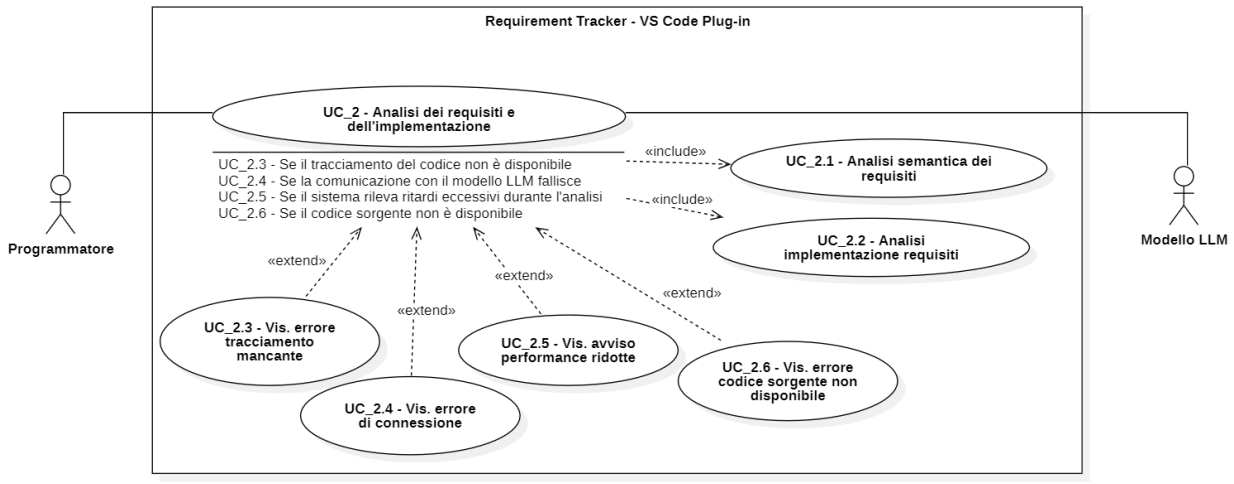

Figure 3: UC_2 - Analisi dei requisiti e dell'implementazione

Attori: Programmatore.

Attore secondario: Modello LLM.

Precondizioni:

- I requisiti sono stati importati [UC_1].
- Il tracciamento dei requisiti nel codice è disponibile, ottenuto direttamente da [UC_1.2] oppure dopo [UC_7].
- La connessione con le API REST di Ollama è attiva e disponibile.

Postcondizioni:

- Il sistema fornisce una valutazione complessiva per ciascun requisito, integrando l'analisi semantica del testo e la verifica dell'implementazione nel codice.

Scenario principale:

1. L'utente seleziona "Analisi requisiti".
2. Il sistema verifica che siano disponibili sia i requisiti che il relativo tracciamento (da [UC_1.2] o [UC_7]).
3. Il sistema esegue l'analisi semantica di ogni requisito [UC_2.1] inviandoli al modello.
4. Il sistema esegue la verifica e la valutazione dell'implementazione del requisito nel codice [UC_2.2] inviandoli al modello.
7. Il modello restituisce i risultati complessivi delle precedenti analisi.
8. Il sistema registra i dati e li rende disponibili per la visualizzazione ([UC_4]).

Estensioni:

- **UC_2.3 - Visualizzazione errore tracciamento mancante**: Se il *mapping*[G] del codice non è disponibile, il sistema visualizza un messaggio d'errore specifico.
- **UC_2.4 - Visualizzazione errore di connessione**: Se la comunicazione con il modello LLM fallisce (es. timeout o connessione interrotta), il sistema informa l'utente e consente di riprovare.
- **UC_2.5 - Visualizzazione avviso performance ridotte**: Se la risposta del modello risulta particolarmente lenta, il sistema mostra un avviso all'utente.
- **UC_2.6 - Visualizzazione errore codice sorgente non disponibile**: Se il progetto non contiene il file sorgente o non è configurato correttamente, il sistema mostra un messaggio di errore.

3.9 UC_2.1 - Analisi semantica dei requisiti

Attori: Programmatore.

Attore secondario: Modello LLM.

Precondizioni:
- I requisiti sono stati importati [UC_1].

Postcondizioni:
- Ogni requisito viene valutato semanticamente, in termini di completezza, chiarezza, correttezza e non ambiguità.
- I risultati dell'analisi semantica sono stati generati e verranno integrati nel *report*[G] finale.

Scenario principale:
1. Il sistema estrae il testo di ciascun requisito.
2. Il sistema invia il testo al modello LLM per l'analisi semantica.
3. Il Modello LLM restituisce una valutazione del requisito e dei suggerimenti.
4. Il sistema registra e integra i risultati dopo l'analisi dell'implementazione [UC_2.3].

3.10 UC_2.2 - Analisi implementazione requisiti

Attori: Programmatore.

Attore secondario: Modello LLM.

Precondizioni:
- I requisiti sono stati importati [UC_1].
- È disponibile il mapping dei requisiti nel codice (ottenuto da [UC_1.2] oppure dopo [UC_7]).

Postcondizioni:
- Il sistema verifica se le porzioni di codice associate implementano correttamente i requisiti, assegnando un punteggio, dei suggerimenti ed eventualmente dei problemi.
- I risultati della verifica sono integrati nel report finale.

Scenario principale:
1. Il sistema raccoglie le informazioni di tracciamento per ciascun requisito.
2. Il sistema estrae la porzione di codice di ogni requisito, identificata dalle informazioni di tracciamento.
3. Il sistema invia il testo del requisito e la relativa porzione di codice al modello LLM.
4. Il modello confronta il comportamento del codice con quanto richiesto dal requisito e restituisce una valutazione.
5. Il sistema registra e integra i risultati nel report finale.

3.11 UC_2.3 - Visualizza errore tracciamento mancante

Attori: Programmatore.

Precondizioni:
- Durante l'analisi [UC_2], il sistema rileva che il mapping (tracciamento del codice) non è disponibile.

Postcondizioni:
- Il sistema visualizza un messaggio d'errore che informa l'utente dell'assenza del tracciamento.

Scenario principale:
1. Durante l'esecuzione di [UC_2], il sistema verifica la presenza del mapping.
2. Se il mapping risulta mancante, il sistema mostra un messaggio d'errore specifico.

3.12 UC_2.4 - Visualizzazione errore di connessione

Attori: Programmatore.

Precondizioni:

- Durante l'analisi [UC_2], la comunicazione con il modello LLM fallisce (es. timeout, connessione interrotta).

Postcondizioni:

- Il sistema informa l'utente dell'errore di connessione.

Scenario principale:

1. Durante l'invio dei dati al modello LLM, il sistema rileva un problema di connessione.
2. Il sistema visualizza un messaggio d'errore dettagliato e consente all'utente di riprovare.

3.13 UC_2.5 - Visualizzazione avviso performance ridotte

Attori: Programmatore.

Precondizioni:

- Durante l'analisi [UC_2], il sistema rileva tempi di risposta eccessivi dal Modello LLM.

Postcondizioni:

- Il sistema mostra un avviso che informa l'utente delle prestazioni ridotte.

Scenario principale:

1. Il sistema monitora il tempo di risposta del modello LLM.
2. Se il tempo supera una soglia prestabilita, il sistema visualizza un avviso informativo.

3.14 UC_2.6 - Visualizzazione errore codice sorgente non disponibile

Attori: Programmatore.

Precondizioni:

- Il progetto non contiene file sorgente o non è configurato correttamente.

Postcondizioni:

- Il sistema informa l'utente della mancanza di codice.

Scenario principale:

1. L'utente avvia la verifica dell'implementazione dei requisiti [UC_2].
2. Il sistema verifica la presenza del codice sorgente.
3. Il sistema rileva che non è configurato correttamente o non è presente.
4. Il sistema mostra un messaggio di errore.

3.15 UC_3 - Esportazione su file CSV

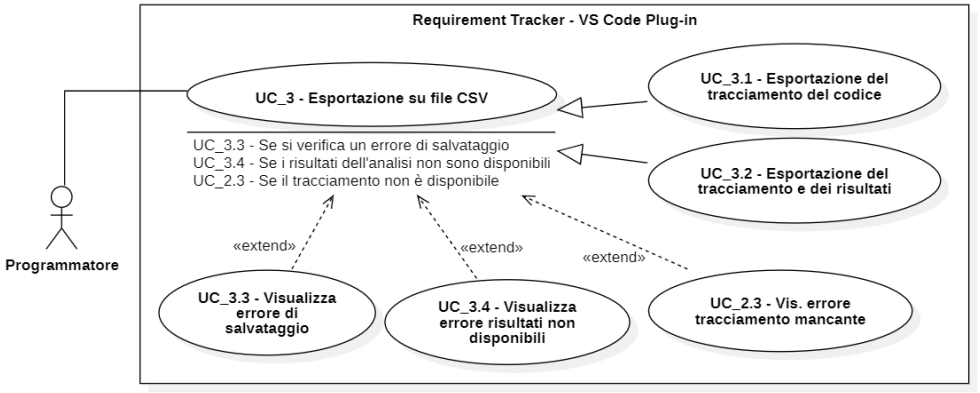

Figure 4: UC_3 - Esportazione su file CSV

Attori: Programmatore.

Precondizioni:
- È disponibile il mapping dei requisiti nel codice (ottenuto da [UC_1.2] oppure dopo [UC_7]).

Postcondizioni:
- I requisiti ed il tracciamento (e, opzionalmente, anche i risultati dell'analisi), vengono esportati correttamente in un file CSV nel percorso specificato dall'utente.

Scenario principale:
1. In base all'opzione di importazione selezionata dall'utente, il sistema applica una delle seguenti specializzazioni:
   - Se l'utente seleziona la voce "Esporta tracciamento", il flusso procede con [UC_3.1].
   - Se l'utente seleziona la voce "Esporta tracciamento e risultati", il flusso procede con [UC_3.2].

Estensioni:
- **UC_3.3 - Visualizza errore di salvataggio**: Se il salvataggio fallisce (es. permessi insufficienti o spazio esaurito), il sistema notifica l'errore all'utente e permette di riprovare.
- **UC_3.4 - Visualizza errore risultati non disponibili**: Se i risultati dell'analisi non sono disponibili, il sistema visualizza un messaggio d'errore specifico.
- **UC_2.3 - Visualizzazione errore tracciamento mancante**: Se il mapping del codice non è disponibile, il sistema visualizza un messaggio d'errore specifico.

3.16 UC_3.1 - Esportazione del tracciamento del codice

Attori: Programmatore.

Precondizioni:
- I requisiti e il loro mapping (tracciamento) sono disponibili (ottenuti da [UC_1.4] o da [UC_7]).
- Non è richiesto l'esportazione dei risultati dell'analisi.

Postcondizioni:
- Il sistema esporta in un file CSV i dati relativi al tracciamento dei requisiti (es. nome del file, intervallo di righe).

Scenario principale:
1. L'utente seleziona la voce "Esporta tracciamento" nell'interfaccia di esportazione.
2. Il sistema apre un file explorer per la scelta del percorso di salvataggio.
3. L'utente conferma il percorso e il nome del file.

4. Il sistema salva un file CSV contenente esclusivamente i requisiti ed i relativi dati di tracciamento.

Estensioni:
- **UC_3.3 - Visualizza errore di salvataggio**: Se il salvataggio fallisce (es. permessi insufficienti o spazio esaurito), il sistema notifica l'errore all'utente e permette di riprovare.

3.17 UC_3.2 - Esportazione del tracciamento e dei risultati

Attori: Programmatore.

Precondizioni:
- I requisiti sono disponibili [UC_1].
- Il mapping (tracciamento) dei requisiti è disponibile (ottenuto da [UC_1.4] o da [UC_7]).
- I risultati dell'analisi sono stati generati [UC_2].

Postcondizioni:
- Il sistema esporta in un file CSV tutti i dati: requisiti, tracciamento e risultati dell'analisi.

Scenario principale:
1. L'utente seleziona la voce "Esporta tracciamento e risultati".
2. Il sistema apre un file explorer per la scelta del percorso di salvataggio.
3. L'utente conferma il percorso e il nome del file.
4. Il sistema salva un file CSV contenente i dati relativi ai requisiti, al mapping e ai risultati.

Estensioni:
- **UC_3.4 - Visualizza errore risultati non disponibili**: Se i risultati dell'analisi non sono disponibili, il sistema mostra un messaggio d'errore specifico.
- **UC_3.1 - Visualizza errore di salvataggio**: Se il salvataggio fallisce (es. permessi insufficienti o spazio esaurito), il sistema notifica l'errore all'utente e permette di riprovare.

3.18 UC_3.3 - Visualizza errore di salvataggio

Attori: Programmatore.

Precondizioni:
- L'utente tenta di esportare i risultati, ma il salvataggio fallisce.

Postcondizioni:
- Il sistema informa l'utente dell'errore e consente di riprovare o di selezionare un percorso alternativo.

Scenario principale:
1. L'utente seleziona "Esporta risultati".
2. Il sistema tenta di salvare il file CSV.
3. Si verifica un errore durante il salvataggio.
4. Il sistema mostra un messaggio d'errore e consente di riprovare.

3.19 UC_3.4 - Visualizza errore risultati non disponibili

Attori: Programmatore.

Precondizioni:
- Durante l'esportazione del tracciamento e dei risultati [UC_3.2], il sistema rileva che i risultati non sono disponibili.

Postcondizioni:
- Il sistema visualizza un messaggio d'errore che informa l'utente dell'assenza dei risultati.

Scenario principale:

1. Durante l'esecuzione di [UC_3.2], il sistema verifica la presenza dei risultati dell'analisi.
2. Se i risultati sono mancanti, il sistema mostra un messaggio d'errore specifico.

3.20 UC_4 - Visualizzazione dei risultati

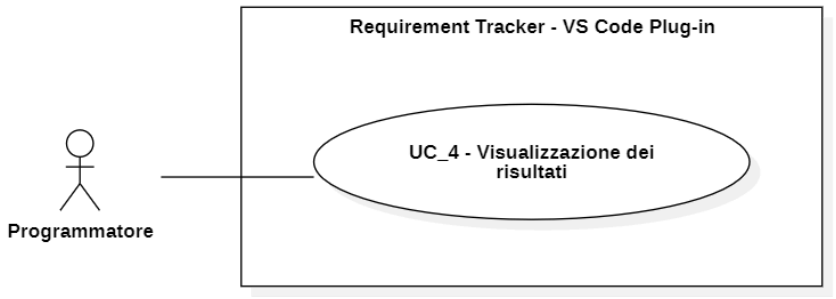

Figure 5: UC_4 - Visualizzazione dei risultati

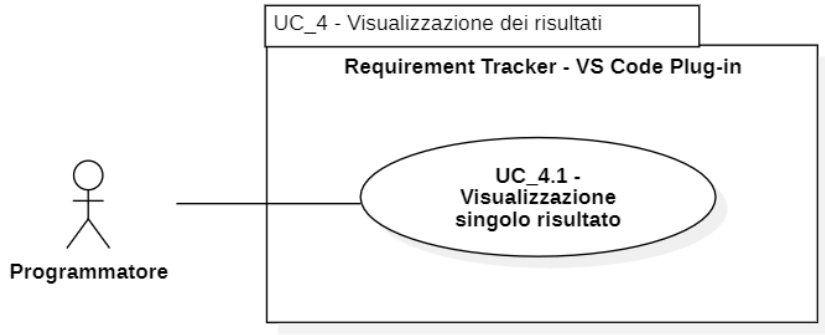

Figure 6: UC_4 - Diagramma di dettaglio sulla visualizzazione dei risultati

Attori: Programmatore.

Precondizioni:
- I risultati dell'analisi sono stati generati [UC_2].
- La sezione di visualizzazione del requisito selezionato è stata espansa [UC_9.1].

Postcondizioni:
- I risultati dell'analisi vengono integrati nel menu ad albero di ogni requisito [UC_9.1] in una sezione dedicata e sono visualizzabili.

Scenario principale:
1. L'utente seleziona un requisito dal menu ad albero [UC_9].
2. Il sistema espande il menu con la visualizzazione del requisito [UC_9.1] e la visualizzazione del risultato [UC_4.1].
3. L'utente seleziona la voce relativa alla visualizzazione del risultato [UC_4.1]
4. Il sistema espande la visualizzazione in dettaglio del risultato [UC_4.1.1] contenente i seguenti sottocasi:
   - [UC_4.1.1.1] Stato di conformità (*passed*/*not passed*).
   - [UC_4.1.1.4] Valutazione del codice in centesimi (0-100).
   - [UC_4.1.1.5] Suggerimenti generati.
   - [UC_4.1.1.6] Problemi riscontrati.

3.21 UC_4.1 - Visualizzazione singolo risultato

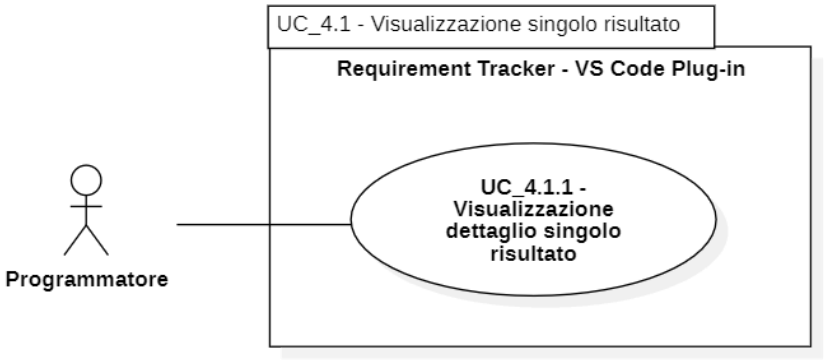

Figure 7: UC_4.1 - Visualizzazione di un singolo risultato

Attori: Programmatore.

Precondizioni:

- I risultati dell'analisi sono stati generati [UC_2].
- La sezione di visualizzazione del requisito selezionato è stata espansa [UC_9.1].

Postcondizioni:

- I risultati dell'analisi vengono integrati nel menu di dettaglio requisito [UC_4.1.1] e sono visualizzabili.

Scenario principale:

1. Il sistema espande il menu con la visualizzazione del requisito [UC_9.1] e la visualizzazione del risultato [UC_4.1].
2. L'utente seleziona la voce relativa alla visualizzazione del risultato [UC_4.1]
3. Il sistema espande la visualizzazione in dettaglio del risultato [UC_4.1.1].

3.22 UC_4.1.1 - Visualizzazione dettaglio singolo risultato

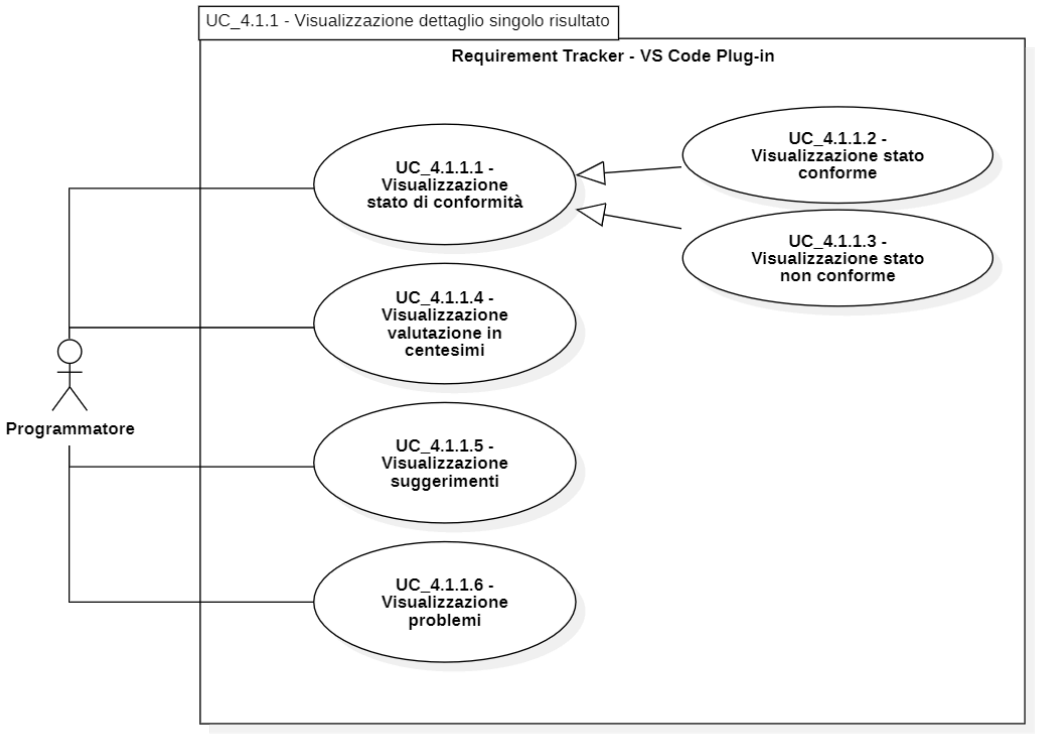

Figure 8: UC_4.1 - Visualizzazione di un singolo risultato

Attori: Programmatore.

Precondizioni:
- I risultati dell'analisi sono stati generati [UC_2].
- Il menu di visualizzazione dettagliata dei risultati è stato aperto da [UC_4.1].

Postcondizioni:
- Il sistema mostra una lista di sotto-elementi relativi al risultato dell'analisi per il requisito selezionato.

Scenario principale:
1. Il sistema visualizza i seguenti elementi nel dettaglio:
   - [UC_4.1.1.1] Stato di conformità (*passed*/*not passed*).
   - [UC_4.1.1.4] Valutazione del codice in centesimi (0-100).
   - [UC_4.1.1.5] Suggerimenti generati.
   - [UC_4.1.1.6] Problemi riscontrati.

3.23 UC_4.1.1.1 - Visualizzazione stato di conformità

Attori: Programmatore.

Precondizioni:
- I risultati dell'analisi sono stati generati [UC_2].
- Il menu ad albero relativo alla visualizzazione in dettaglio dei risultati è stato espanso [UC_4.1.1]

Postcondizioni:
- Viene visualizzato lo stato "*passed*" se il requisito è corretto semanticamente e la sua implementazione nel codice soddisfa il requisito, "*not passed*" altrimenti.

Scenario principale:

1. Il sistema mostra, per ogni requisito, lo stato di conformità basato sul punteggio ottenuto:
   - Se il risultato è *passed*, visualizza lo stato conforme [UC_4.1.1.2]
   - Se il risultato è *not-passed*, visualizza lo stato non conforme [UC_4.1.1.3]

3.24 UC_4.1.1.4 - Visualizzazione punteggio in centesimi

Attori: Programmatore.

Precondizioni:

- La valutazione del codice è disponibile [UC_2.2].
- Il menu ad albero relativo alla visualizzazione in dettaglio dei risultati è stato aperto [UC_4.1.1]

Postcondizioni:

- Il sistema mostra il punteggio numerico della valutazione dell'implementazione nel codice del requisito selezionato, espresso in centesimi.

Scenario principale:

1. Il sistema visualizza il punteggio relativo all'aderenza del codice al requisito.

3.25 UC_4.1.1.5 - Visualizzazione suggerimenti

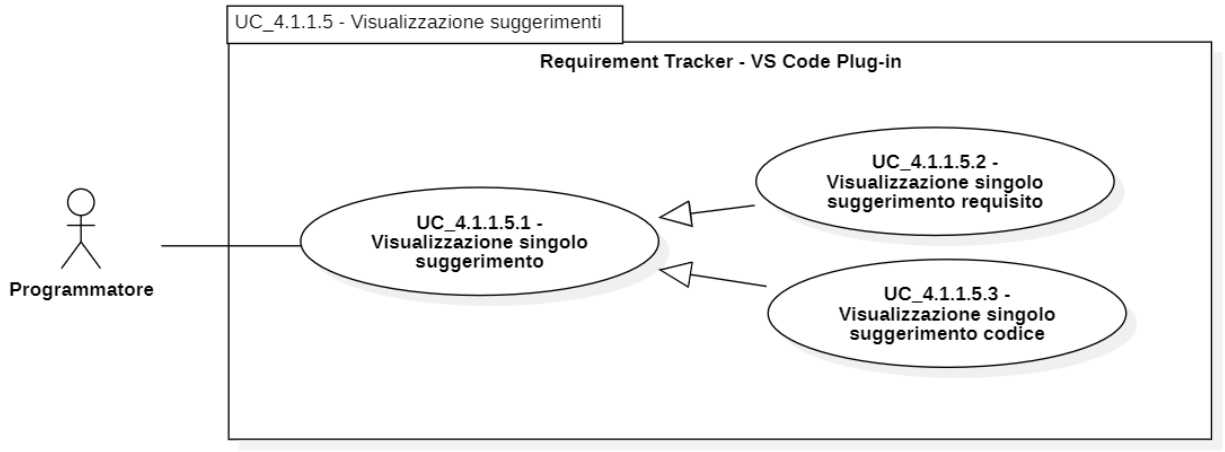

Figure 9: UC_4.1.1.5 - Visualizzazione dei suggerimenti

Attori: Programmatore.

Precondizioni:

- I suggerimenti relativi al requisito e/o al codice sono stati generati durante l'analisi [UC_2].
- Il menu ad albero relativo alla visualizzazione in dettaglio dei risultati è stato aperto [UC_4.1.1]

Postcondizioni:

- Il sistema mostra un elenco strutturato dei suggerimenti relativi al requisito [UC_4.1.1.5.2] e al codice [UC_4.1.1.5.3].

Scenario principale:

1. Il sistema raccoglie i suggerimenti generati e li visualizza in forma di elenco.

3.26 UC_4.1.1.6 - Visualizzazione problemi

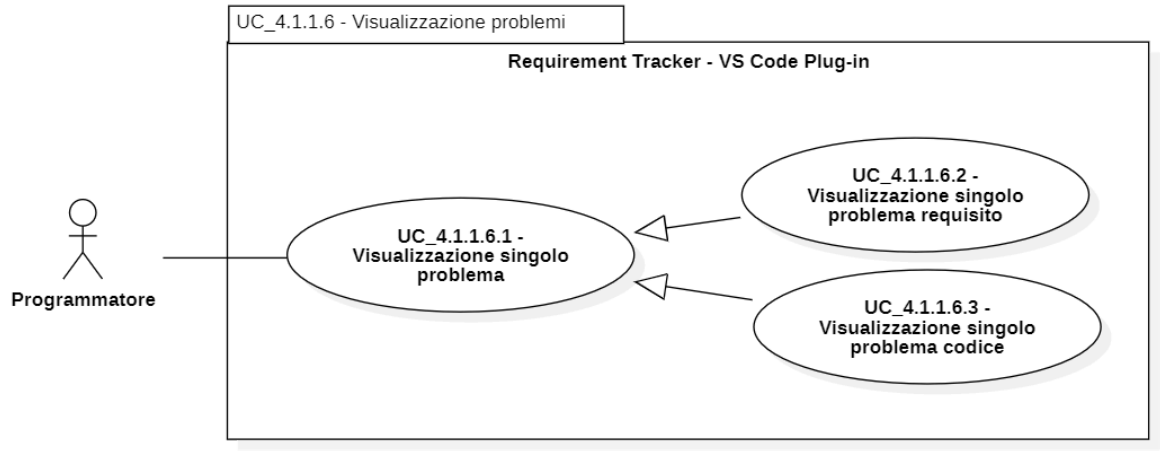

Figure 10: UC_4.1.1.6 - Visualizzazione dei problemi

Attori: Programmatore.

Precondizioni:

- I problemi relativi al requisito e/o al codice sono stati generati durante l'analisi [UC_2].
- Il menu ad albero relativo alla visualizzazione in dettaglio dei risultati è stato aperto [UC_4.1.1]

Postcondizioni:

- Il sistema mostra un elenco strutturato dei problemi relativi al requisito [UC_4.1.1.6.2] e al codice [UC_4.1.1.6.3].

Scenario principale:

1. Il sistema raccoglie i problemi generati e li visualizza in forma di elenco.

3.27 UC_5 - Filtraggio dei requisiti

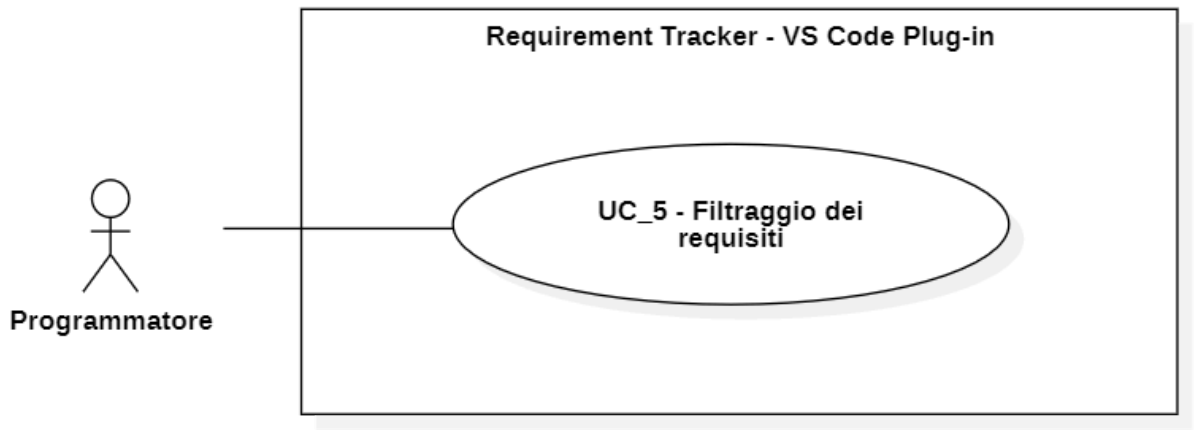

Figure 11: UC_5 - Filtraggio dei requisiti

Attori: Programmatore.

Precondizioni:

- I requisiti importati sono visualizzati [UC_9].

Postcondizioni:

- I risultati vengono filtrati in base ai campi *ID*, *descrizione*,*file sorgente* .

Scenario principale:

1. L'utente inserisce il testo da ricercare tramite la barra di ricerca.
2. Il sistema filtra la lista dei requisiti in base al campo inserito.
3. Il sistema visualizza la lista dei requisiti filtrati.

3.28 UC_6 - Analisi di un singolo requisito

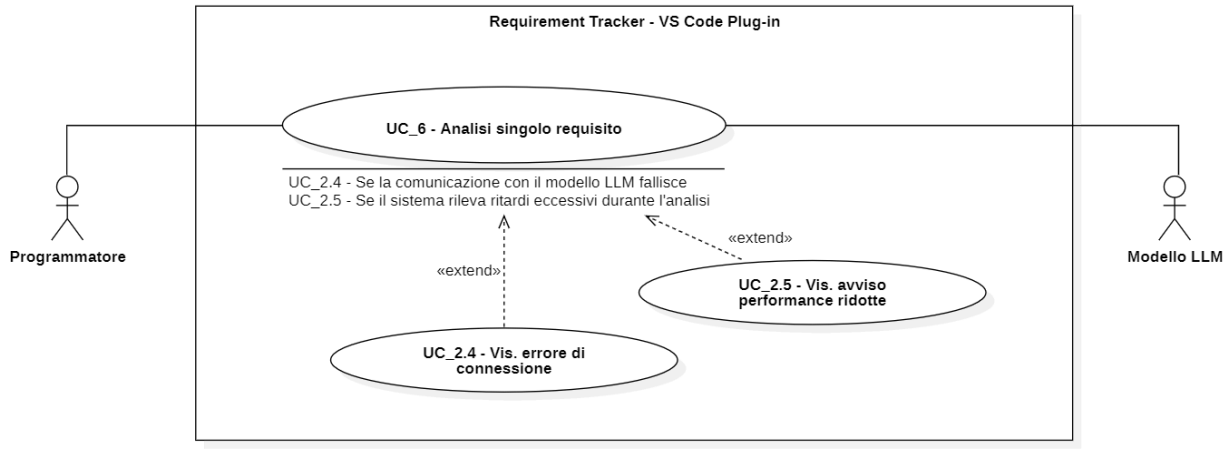

Figure 12: UC_6 - Analisi di un singolo requisito

Attori: Programmatore.

Attore secondario: Modello LLM.

Precondizioni:

- Il requisito è visualizzato nell'elenco dei requisiti [UC_9].
- È stata selezionata la funzione di ripetizione dell'analisi

Postcondizioni:

- Viene fornita una nuova valutazione per il requisito selezionato.

Scenario principale:

1. L'utente apre la visualizzazione del singolo requisito [UC_9.1].
2. L'utente clicca sull'icona relativa alla funzione di ripetizione analisi.
3. Il sistema invia il requisito ed il relativo codice al modello LLM per una nuova analisi, analogamente ad [UC_2].
4. I risultati aggiornati vengono registrati e visualizzati per il requisito selezionato [UC_4.1].

Estensioni:

- **UC_2.4 - Visualizzazione errore di connessione**: Se la comunicazione con il modello LLM fallisce (es. timeout o connessione interrotta), il sistema informa l'utente e consente di riprovare.
- **UC_2.5 - Visualizzazione avviso performance ridotte**: Se la risposta del modello risulta particolarmente lenta, il sistema mostra un avviso all'utente.

3.29 UC_7 - Tracciamento dei requisiti nel codice

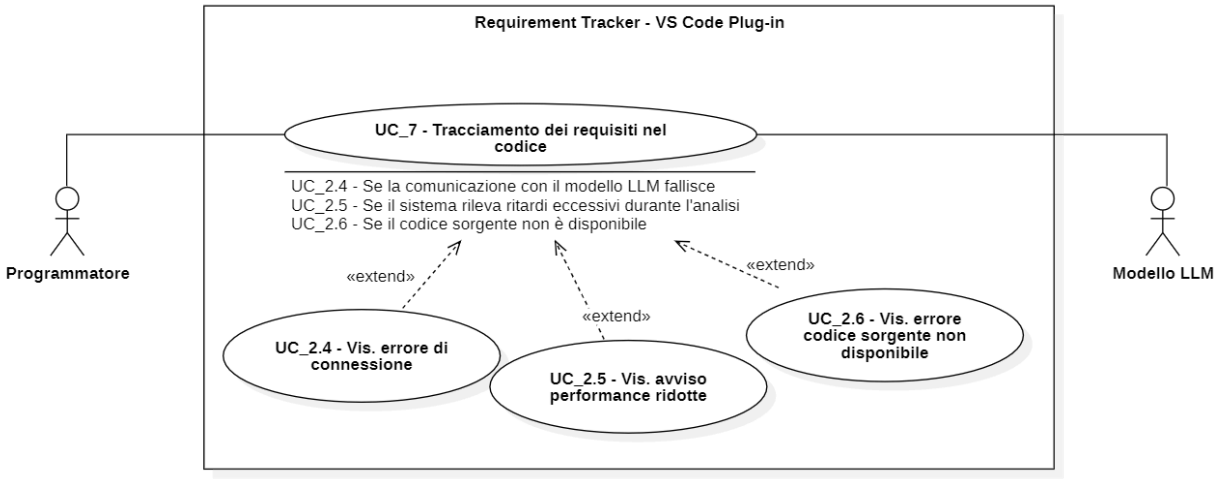

Figure 13: UC_7 - Funzione di tracciamento dei requisiti

Attori: Programmatore.

Attore secondario: Modello LLM.

Precondizioni:

- I requisiti sono stati importati senza informazioni di tracciamento [UC_1.1].
- È disponibile il codice sorgente nel progetto.

Postcondizioni:

- Il sistema esegue una ricerca nel codice sorgente per associare, a ciascun requisito, la porzione di codice che lo implementa.
- Il tracciamento ottenuto viene registrato e reso disponibile nella vista dei requisiti [UC_9.1.2].

Scenario principale:

1. L'utente, notando l'assenza del mapping, seleziona l'opzione "Tracciamento dei requisiti".
2. Il sistema invia il codice ed i requisiti al modello LLM per la mappatura.
3. Il modello confronta il contenuto del codice con i requisiti inviati e individua le porzioni che li implementano.
4. Il modello restituisce una possibile mappatura del codice, che include il nome del file e l'intervallo di righe, per ogni requisito.
5. Il mapping risultante viene registrato e visualizzato insieme ai requisiti.

Estensioni:

- **UC_2.4 - Visualizzazione errore di connessione**: Se la comunicazione con il modello LLM fallisce (es. timeout o connessione interrotta), il sistema informa l'utente e consente di riprovare.
- **UC_2.5 - Visualizzazione avviso performance ridotte**: Se la risposta del modello risulta particolarmente lenta, il sistema mostra un avviso all'utente.
- **UC_2.6 - Visualizzazione errore codice non disponibile**: Se il progetto non contiene il file sorgente o non è configurato correttamente.

3.30 UC_8 - Configurazione dei path da ignorare

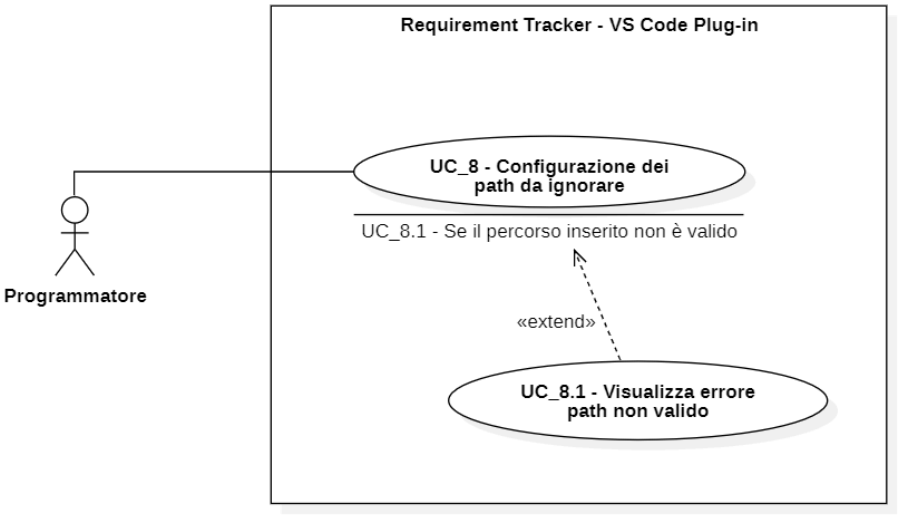

Figure 14: UC_8 - Configurazione dei path da ignorare

Attori: Programmatore.

Precondizioni:
- Il progetto è stato configurato in Visual Studio Code.
- Il plug-in è attivo e funzionante.

Postcondizioni:
- I path specificati nel file .reqignore vengono esclusi dall'analisi [UC_2] e dal tracciamento dei requisiti [UC_7].

Scenario principale:
1. L'utente crea o modifica un file .reqignore nel progetto.
2. L'utente inserisce nel file .reqignore i path o *pattern*[G] relativi ai file o directory da escludere.
3. Il sistema rileva automaticamente le modifiche apportate al file .reqignore.
4. Durante la l'analisi [UC_2] ed il tracciamento [UC_7], il sistema esclude i path specificati nel file .reqignore.
5. L'utente avvia l'analisi o il tracciamento e i path ignorati non vengono considerati.

Estensioni:
- **UC_8.1 - Visualizzazione errore path non valido**: Se il path specificato non è valido, il sistema notifica l'utente e ignora l'*entry*[G] errata mantenendo valide le altre.

3.31 UC_8.1 - Visualizzazione errore path non valido

Attori: Programmatore.

Precondizioni:
- L'utente inserisce un path o pattern non valido nel file .reqignore.

Postcondizioni:
- Il sistema ignora il path non valido e prosegue con le configurazioni valide.

Scenario principale:
1. L'utente modifica il file .reqignore e inserisce un path o pattern non valido.
2. Il sistema rileva l'entry non valida durante la verifica del file.

3. Il sistema notifica l'utente dell'errore e fornisce dettagli sul path non valido.
4. Il sistema ignora l'entry non valida e considera solo i path configurati correttamente.

3.32 UC_9 - Visualizzazione dei requisiti

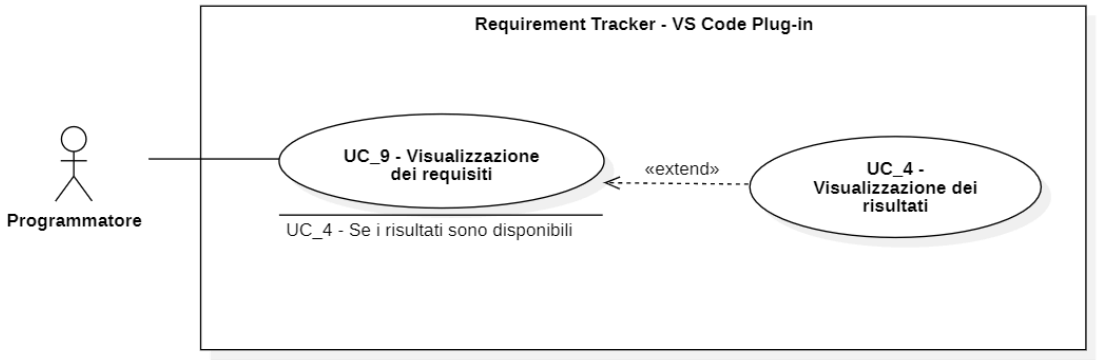

Figure 15: UC_9 - Visualizzazione dei requisiti

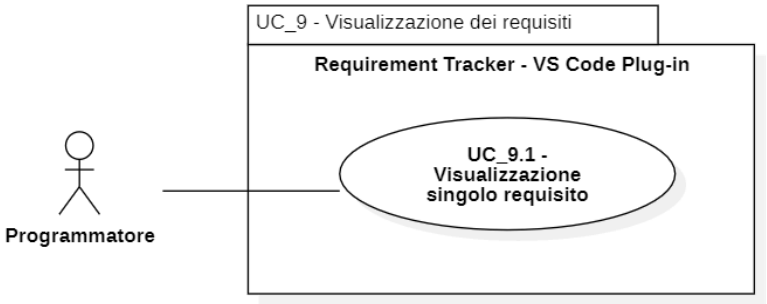

Figure 16: UC_9 - Dettaglio sulla visualizzazione dei requisiti

Attori: Programmatore.

Precondizioni:

- I requisiti sono stati importati [UC_1] e sono disponibili per la visualizzazione.

Postcondizioni:

- I requisiti vengono visualizzati in una lista ad albero, in cui ciascun requisito mostra le informazioni di base (e, se disponibili, il mapping e i risultati dell'analisi).

Scenario principale:

1. Il sistema visualizza una lista dei requisiti.
2. L'utente può selezionare un requisito per visualizzarne il dettaglio [UC_9.1.1].

3.33 UC_9.1 - Visualizzazione singolo requisito

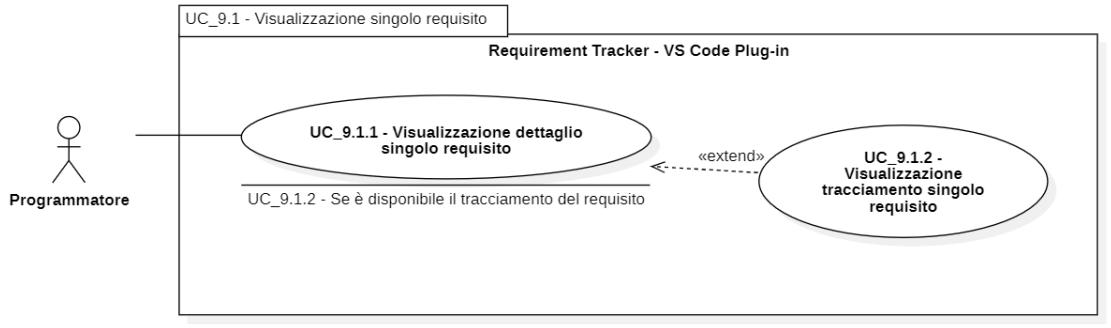

Figure 17: UC_9.1 - Visualizzazione di un singolo requisito

Attori: Programmatore.

Precondizioni:
- Un requisito è stato selezionato dalla lista [UC_9].

Postcondizioni:
- Il sistema mostra la voce “Requisito” che, se premuto, permette la visualizzazione in dettaglio di tutte le informazioni relative al requisito [UC_9.1.1].

Scenario principale:
1. L’utente seleziona un requisito dalla lista dei requisiti visualizzati in [UC_9].
2. Il sistema apre un sottomenù contenente la voce “Requisito” che, se premuto, mostra i dettagli del requisito selezionato [UC_9.1.1].

3.34 UC_9.1.1 - Visualizzazione dettaglio singolo requisito

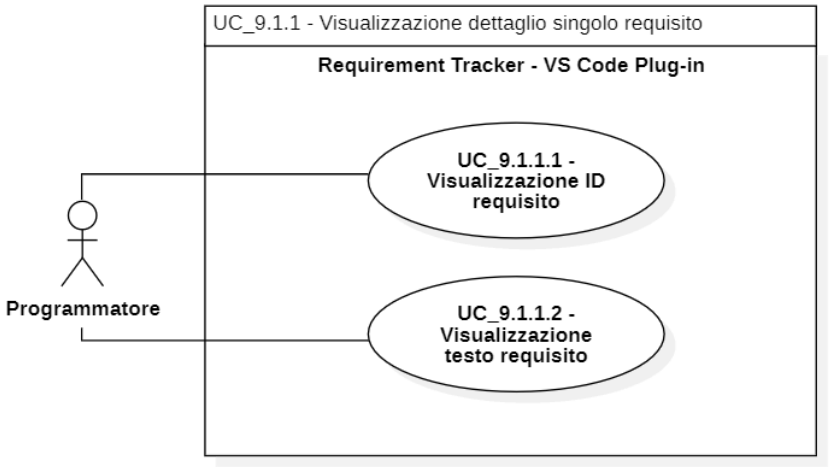

Figure 18: UC_9.1.1 - Visualizzazione in dettaglio di un singolo requisito

Attori: Programmatore.

Precondizioni:
- L’utente ha premuto sulla voce “Requisito”

Postcondizioni:
- Il sistema mostra il dettaglio completo del requisito, includendo:
  - Identificativo del requisito [UC_9.1.1.1]

- Testo descrittivo del requisito [UC_9.1.1.2]
- (Opzionale) Informazioni di tracciamento [UC_9.1.2]

Scenario principale:

1. L'utente preme sulla voce "Requisito"
2. Il sistema espande il sottomenu di dettaglio, visualizzando tutte le informazioni relative al requisito.
3. Se sono disponibili dati di tracciamento, il sistema visualizza anche il dettaglio del tracciamento [UC_9.1.2].

3.35 UC_9.1.1.1 - Visualizzazione ID requisito

Attori: Programmatore.

Precondizioni:

- Il requisito selezionato è espanso nella vista di dettaglio [UC_9.1.1].

Postcondizioni:

- Il sistema mostra il campo "ID" del requisito.

Scenario principale:

1. Il sistema visualizza l'identificativo univoco del requisito.

3.36 UC_9.1.1.2 - Visualizzazione testo requisito

Attori: Programmatore.

Precondizioni:

- Il requisito selezionato è espanso nella vista di dettaglio [UC_9.1.1].

Postcondizioni:

- Il sistema mostra il campo "Testo" completo del requisito.

Scenario principale:

1. Il sistema visualizza il testo descrittivo del requisito.

3.37 UC_9.1.2 - Visualizzazione tracciamento singolo requisito

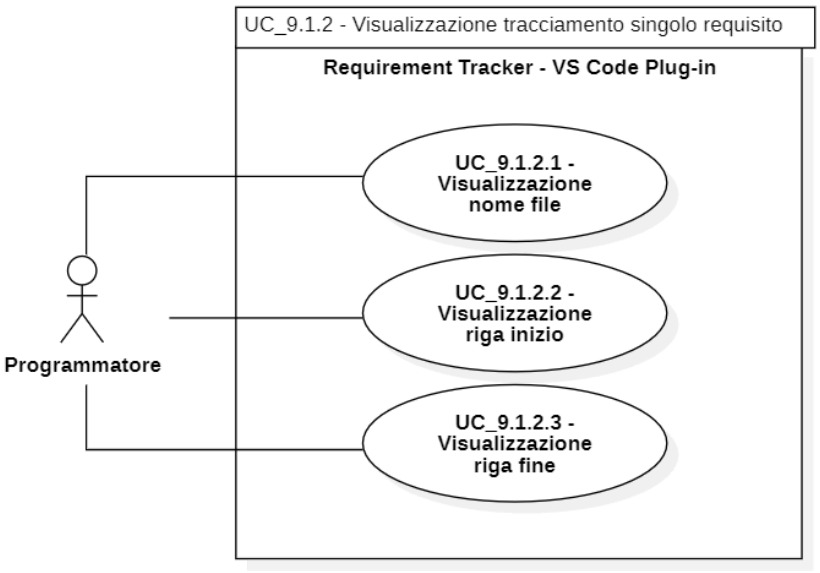

Figure 19: UC_9.1.2 - Visualizzazione delle informazioni di tracciamento di un requisito

Attori: Programmatore.

Precondizioni:
- Il requisito selezionato è espanso nella vista di dettaglio [UC_9.1.1].
- Il requisito selezionato dispone di informazioni di tracciamento, importate da [UC_1.2] oppure generate da [UC_7].

Postcondizioni:
- Il sistema visualizza una menu di tracciamento espandibile che comprende i dettagli del tracciamento.

Scenario principale:
1. Nella vista di dettaglio del requisito [UC_9.1.1], il sistema verifica la presenza di dati di tracciamento.
2. Se presenti, il sistema espande la sezione “Tracciamento” mostrando i dettagli attraverso i sottocasi [UC_9.1.2.1], [UC_9.1.2.2] e [UC_9.1.2.3].

3.38 UC_9.1.2.1 - Visualizzazione nome file

Attori: Programmatore.

Precondizioni:
- La sezione “Tracciamento” del requisito è disponibile ed espansa [UC_9.1.2].

Postcondizioni:
- Il sistema mostra il campo “Nome File” relativo al file che contiene il codice relativo all’implementazione del requisito.

Scenario principale:
1. Il sistema visualizza il nome del file associato al tracciamento del requisito.

3.39 UC_9.1.2.2 - Visualizzazione riga inizio

Attori: Programmatore.

Precondizioni:
- La sezione “Tracciamento” del requisito è disponibile ed espansa [UC_9.1.2].

Postcondizioni:
- Il sistema mostra il campo “Riga Inizio” dell’intervallo di codice nel file indicato [UC_9.1.2.1] relativo all’implementazione del requisito.

Scenario principale:
1. Il sistema visualizza la riga di inizio del tracciamento del requisito.

3.40 UC_9.1.2.3 - Visualizzazione riga fine

Attori: Programmatore.

Precondizioni:
- La sezione “Tracciamento” del requisito è disponibile ed espansa [UC_9.1.2].

Postcondizioni:
- Il sistema mostra il campo “Riga Fine” dell’intervallo di codice nel file indicato [UC_9.1.2.1] relativo all’implementazione del requisito.

Scenario principale:
1. Il sistema visualizza la riga di fine del tracciamento del requisito.

3.41 UC_10 - Configurazione del modello LLM

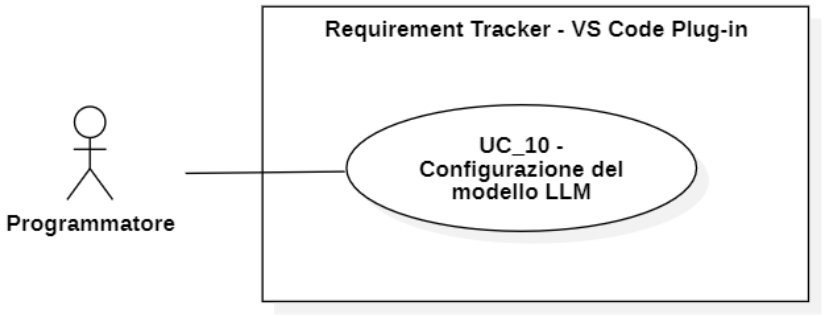

Figure 20: UC_10 - Configurazione del modello LLM per l'analisi

Attori: Programmatore.

Precondizioni:

- Il plug-in "Requirement Tracker Plug-in" è installato e attivo in Visual Studio Code.
- L'utente ha accesso alle impostazioni di Visual Studio Code nel menu "Extensions".

Postcondizioni:

- Il modello LLM configurato viene salvato e verrà utilizzato dal plug-in per le analisi dei requisiti e del codice [UC_2].

Scenario principale:

1. L'utente apre le impostazioni di Visual Studio Code e naviga nel menu "Extensions".
2. L'utente individua il plug-in "Requirement Tracker Plug-in".
3. All'interno delle impostazioni del plug-in, l'utente seleziona il campo affianco la voce "Model".
4. L'utente inserisce il nome del modello LLM desiderato (es. "llama3.2:3b" o "deepseek-coder:7b").
5. Il sistema salva la configurazione e la utilizza per le analisi successive.

3.42 UC_11 - Configurazione dell'endpoint del server Ollama

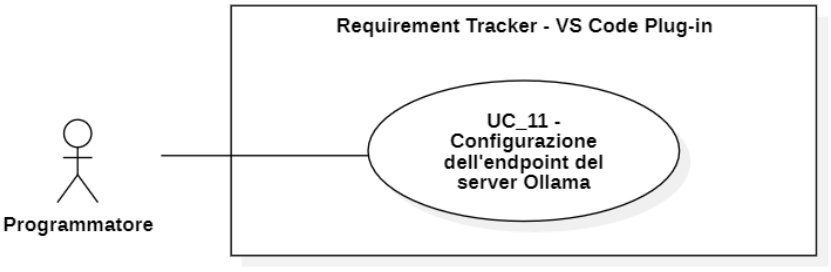

Figure 21: UC_11 - Configurazione dell'endpoint di Ollama

Attori: Programmatore.

Precondizioni:

- Il plug-in "Requirement Tracker Plug-in" è installato e attivo in Visual Studio Code.
- L'utente ha accesso alle impostazioni nel menu "Extensions" di Visual Studio Code.

Postcondizioni:

- L'endpoint del server Ollama viene salvato e utilizzato dal plug-in per le chiamate API.

Scenario principale:

1. L'utente apre le impostazioni di Visual Studio Code e naviga nel menu "Extensions".

2. L'utente individua il plug-in "Requirement Tracker Plug-in".
3. All'interno delle impostazioni, l'utente seleziona la voce "Ollama Endpoint".
4. L'utente inserisce l'indirizzo IP o il link del server Ollama.
5. Il sistema salva l'endpoint e lo utilizza per le chiamate API durante le analisi.

3.43 UC_12 - Configurazione della soglia di conformità

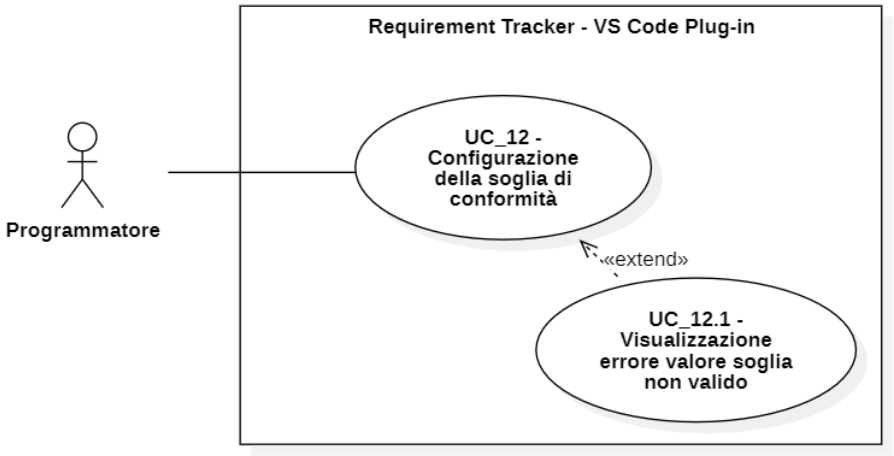

Figure 22: UC_12 - Configurazione della soglia del quality score

Attori: Programmatore.

Precondizioni:
- Il plug-in "Requirement Tracker Plug-in" è installato e attivo in Visual Studio Code.
- L'utente ha accesso alle impostazioni nel menu "Extensions" di Visual Studio Code.

Postcondizioni:
- La soglia di qualità viene salvata e utilizzata dal plug-in per determinare lo stato "passed" (true/false) dei requisiti [UC_2.2].

Scenario principale:
1. L'utente apre le impostazioni di Visual Studio Code e naviga nel menu "Extensions".
2. L'utente individua il plug-in "Requirement Tracker Plug-in".
3. All'interno delle impostazioni, l'utente seleziona la voce "Quality threshold".
4. L'utente inserisce il valore soglia desiderato (es. 80).
5. Il sistema salva la soglia e la utilizza per valutare i risultati dell'analisi dei requisiti.

Estensioni:
- **UC_12.1 - Visualizza errore valore soglia non valido**: Se il valore inserito per la soglia non è numerico o non rientra nei limiti previsti, il sistema notifica l'errore all'utente e richiede di inserire un valore corretto.

3.44 UC_12.1 - Visualizzazione errore valore soglia non valido

Attori: Programmatore.

Precondizioni:
- Durante la configurazione della soglia di conformità in [UC_12], l'utente inserisce un valore non numerico o un valore che non rientra nei limiti previsti.

Postcondizioni:

- Il sistema visualizza un messaggio d'errore che informa l'utente dell'inserimento non valido e richiede la correzione del valore.

Scenario principale:

1. L'utente inserisce il valore per la soglia di conformità nella sezione "Quality threshold" delle impostazioni del plug-in.
2. Il sistema verifica il valore inserito.
3. Se il valore non è numerico o non rientra nei limiti previsti, il sistema visualizza un messaggio d'errore specifico.
4. Il sistema richiede all'utente di inserire un valore corretto.

- **UC_1.4 - Visualizza errore file importazione** : Se il file non rispetta il formato previsto o risulta malformato, il sistema notifica l'errore all'utente e richiede di selezionare un file corretto.

4 Requisiti

4.1 Introduzione

Il gruppo NextSoft, a seguita di una attenta analisi dichiara che i requisiti che il prodotto finale andrà a soddisfare sono i seguenti. Questi vengono mostrati di seguito in forma tabellare, seguendo quanto detto all'interno del documento *Norme di Progetto*

4.2 Requisiti Funzionali

Questi requisiti descrivono cosa il sistema deve fare

Codice	Classificazione	Descrizione	Fonti
RFO001	*Obbligatorio*[G]	Il sistema deve essere in grado di caricare il file dei requisiti in formato CSV dal *filesystem*[G]	Capitolato, UC_1, UC_1.1, UC_1.3, Proponente
RFO002	Obbligatorio	Il sistema deve visualizzare i requisiti caricati in una vista strutturata ad albero	UC_9, UC_9.1, UC_9.1.1, UC_9.1.1.1, UC_9.1.1.2, UC_9.1.2, UC_9.1.2.1, UC_9.1.2.2, UC_9.1.2.3
RFO003	Obbligatorio	Il sistema deve validare i requisiti inseriti all'interno del file CSV	UC_1, UC_1.4
RFO004	Obbligatorio	Il sistema deve informare l'utente se il file CSV caricato non è valido in caso di colonne id e descrizione mancanti	UC_1.4
RFO005	Obbligatorio	Il sistema deve fornire una valutazione dei requisiti in termini di completezza, coerenza e adderenza al codice	UC_2, UC_2.1, UC_2.2, Capitolato
RFO006	Obbligatorio	L'utente deve ricevere suggerimenti su come rendere i requisiti più specifici, misurabili, realizzabili e pertinenti	Capitolato, UC_4.1.1.5
RFO007	Obbligatorio	Il sistema deve comunicare con un modello LLM tramite una REST API per ottenere delle valutazioni	UC_2
RFO008	Obbligatorio	Il sistema deve tracciare l'implementazione dei requisiti nel codice e verificarne la copertura	Capitolato, UC_2.2, UC_7

Codice	Classificazione	Descrizione	Fonti
RFO009	Obbligatorio	Il sistema deve consentire l'esportazione dei dati in formato CSV	UC_3, UC_3.1, UC_3.2
RFO010	Obbligatorio	Il sistema deve visualizzare graficamente i risultati delle analisi integrandoli nella lista dei requisiti	UC_4, UC_4.1, UC_4.1.1, UC_4.1.1.1, UC_4.1.1.4, UC_4.1.1.5, UC_4.1.1.6
RFO011	Obbligatorio	Il sistema deve filtrare i risultati delle analisi in base ai criteri specificati dall'utente (ID, descrizione, file di implementazione)	UC_5
RFO012	Obbligatorio	Il sistema deve consentire l'analisi di un singolo requisito	UC_6
RFO013	Obbligatorio	Il sistema deve informare l'utente nel caso, a seguito di un analisi, non ci siano risultati	UC_4.1.1.6
RFF014	*Facoltativo*[G]	Il codice relativo ad un requisito da analizzare può essere presente in file diversi	Proponente
RFO015	Obbligatorio	L'utente deve essere in grado di scegliere il modello da utilizzare prima dell'analisi	UC_10, Proponente
RFF016	Facoltativo	L'utente deve essere in grado di configurare il modello utilizzato per l'analisi	Proponente
RFO017	Obbligatorio	L'utente deve essere in grado di configurare l'endpoint di ollama	UC_11, Proponente
RFO018	Obbligatorio	L'utente deve essere in grado di configurare la soglia del quality score accettabile	UC_12, Proponente
RVF019	Facoltativo	Il sistema deve informare l'utente in caso di mancanza del codice sorgente con un messaggio di errore	UC_2.6
RFO020	Obbligatorio	Il sistema deve permettere l'esclusione dall'analisi di alcuni file, indicati all'interno di un documento (file .reqignore)	UC_8, Proponente

Table 1: Requisiti Funzionali

4.3 Requisiti di qualità

Questi requisiti riguardano le caratteristiche qualitative del sistema

Codice	Classificazione	Descrizione	Fonti
RQO001	Obbligatorio	Il plug-in deve essere modulare per consentire e facilitare l'aggiunta di nuove *feature*[G] in base a esigenze o aggiornamenti futuri del progetto	Capitolato
RQO002	Obbligatorio	Il prodotto deve essere sviluppato secondo quanto detto all'interno del file *Norme di Progetto*	*Norme di Progetto*
RQO003	Obbligatorio	Il prodotto deve essere sviluppato secondo quanto detto all'interno del file *Piano di Progetto*	*Piano di Progetto*

Table 2: Requisiti di Qualità

4.4 Requisiti di vincolo

Questi requisiti specificano limiti tecnici o di conformità

Codice	Classificazione	Descrizione	Fonti
RVO001	Obbligatorio	Deve supportare i linguaggi C/C++	Capitolato
RVF002	Facoltativo	Deve supportare altri linguaggi oltre a C/C++	Capitolato
RVF003	Facoltativo	Il sistema deve fornire valutazioni conformi alle normative sulla sicurezza funzionale (ISO 26262 o IEC 61508)	Capitolato
RVO004	Obbligatorio	Il sistema deve comunicare con un modello LLM attraverso una REST API	UC_2
RVO005	Obbligatorio	L'estensione di Visual Studio Code deve essere in inglese	Proponente

Table 3: Requisiti di Vincolo

4.5 Requisiti Prestazionali

Questi requisiti descrivono aspetti legati alla velocità e alle prestazioni del sistema.

Codice	Classificazione	Descrizione	Fonti
RPD001	*Desiderabile*[G]	Il sistema deve informare l'utente in caso di rallentamenti dovuti ad una connessione lenta (risposte con tempo di attesa >20s) o a un modello troppo grande (*prompt*[G] maggiore di 6000 *token*[G] e/o velocità di risposta < 20 token/s)	UC_2.4, UC_2.5
RPO002	Obbligatorio	Il sistema deve informare l'utente in caso di errore di connessione e consentire di riprovare	UC_2.4

Table 4: Requisiti Prestazionali

4.6 Tracciamento dei Requisiti

Fonte	Requisiti
UC_1	RFO001, RFO003
UC_1.3	RFO001
UC_1.1	RFO001
UC_1.2	
UC_1.4	RFO003, RFO004
UC_2	RFO005, RVO004
UC_2.1	RFO005
UC_2.2	RFO005
UC_2.3	
UC_2.4	RPD001,RPO002
UC_2.5	RPD001
UC_2.6	RVF019
UC_3	RFO009

Fonte	Requisiti
UC_3.1	RFO009
UC_3.2	RFO009
UC_3.3	
UC_4	RFO010
UC_4.1	RF010
UC_4.1.1	RFO010
UC_4.1.1.1	RFO010
UC_4.1.1.2	
UC_4.1.1.3	
UC_4.1.1.4	RFO010
UC_4.1.1.5	RFO006, RFO010
UC_4.1.1.6	RFO010, RFO013
UC_4.1.1.5.1	
UC_4.1.1.5.2	
UC_4.1.1.5.3	
UC_4.1.1.6.1	
UC_4.1.1.6.2	
UC_4.1.1.6.3	
UC_5	RFO011
UC_6	RFO012
UC_7	RFO008
UC_8	RFO020
UC_8.1	
UC_9	RFO002
UC_9.1	RFO002
UC_9.1.1	RFO002
UC_9.1.1.1	RFO002
UC_9.1.1.2	RFO002
UC_9.1.2	RFO002
UC_9.1.2.1	RFO002
UC_9.1.2.2	RFO002
UC_9.1.2.3	RFO003
UC_10	RFO015
UC_11	RFO017
UC_12	RFO018
UC_12.1	

Table 5: Tracciamento dei Requisiti

4.7 Riepilogo

Tipologia	Obbligatorio	Desiderabile	Facoltativo	Totale
Funzionale[G]	18		2	20
Di Qualità[G]	3			3
Di Vincolo[G]	3		3	6
Prestazionale	1	1		2

Table 6: Repilogo

Elenco delle immagini

Elenco delle tabelle