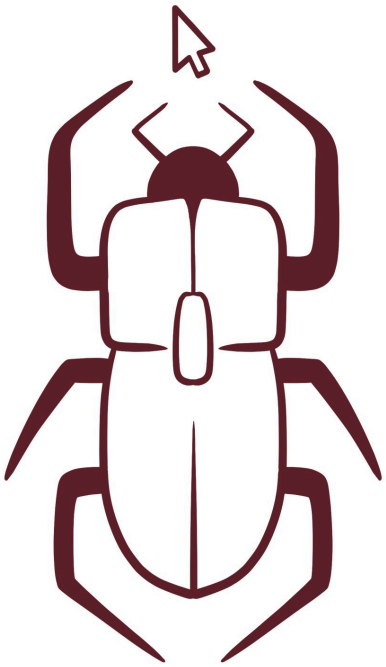


Piano di Qualifica

v2.0.0

Registro delle Modifiche

Data	Versione	Descrizione	Redattore	Verificatore
2026/04/27	2.0.0	Revisione per PB		Sandu Antonio
2026/04/18	1.3.0	Aggiunta e correzione Test di Unità e Sistema per microservizio di gestione account	Berengan Riccardo	Zago Alice
2026/04/19	1.2.0	Aggiornamento Cruscotto di Valutazione	Zago Alice	Suar Alberto
2026/03/11	1.1.0	Correzione numerazione test dopo modifiche AdR per errori segnalati per la RTB	Sgreva Andrea	Zago Alice
2026/03/02	1.0.0	Ufficializzazione per RTB		Suar Alberto
2026/02/26	0.11.0	Aggiunta grafici mancanti	Zago Alice	Suar Alberto
2026/02/25	0.10.0	Aggiunta Test di Unità mancanti	Zago Alice	Suar Alberto
2026/02/06	0.9.0	Test di Accettazione e rielaborazione Test di Sistema	Zago Alice	Suar Alberto
2026/01/27	0.8.1	Aggiunta grafico BV-SV, revisione Metodi di Testing e aggiunta test, modifiche minori al documento	Zago Alice	Suar Alberto
2026/01/22	0.7.0	Grafici CPI-SPI, EAC, RSI e SGA	Zago Alice	Suar Alberto
2026/01/21	0.6.0	Cruscotto di valutazione, grafico PV-AC-EV	Zago Alice	Suar Alberto
2026/01/17	0.5.2	Revisione Automiglioramento	Suar Alberto	Zago Alice
2026/01/13	0.5.1	Rielaborazione introduzione	Suar Alberto	Zago Alice

Data	Versione	Descrizione	Redattore	Verificatore
		documento e qualità di processo		
2025/12/02	0.5.0	Modifica tabelle qualità di processo, inserimento tabelle qualità di prodotto	Zago Alice	Suar Alberto
2025/12/30	0.4.0	Iniziati metodi di testing, inserimento tabelle test	Berengan Riccardo	Suar Alberto
2025/12/28	0.3.0	Processi secondari e processi organizzativi con tabelle soglie metriche, iniziata sezione automiglioramento e qualità di prodotto	Zago Alice	Suar Alberto
2025/12/27	0.2.0	Qualità di processo, processi primari	Zago Alice	Suar Alberto
2025/12/26	0.1.0	Inizio stesura documento, introduzione, scopo e riferimenti	Zago Alice	Suar Alberto
2025/12/23	0.0.0	Creazione documento	Zago Alice	Suar Alberto

Indice

Indice tabelle

1 Introduzione

1.1 Contesto del Progetto

Il presente documento descrive il Piano di Qualifica [G] relativo al progetto Code Guardian [G], commissionato dall'azienda Var Group [G] e realizzato dal team di sviluppo Skarab Group [G] nell'ambito del corso di Ingegneria del Software presso l'Università degli Studi di Padova.

Il progetto ha come obiettivo la realizzazione di un sistema per l'automazione dei processi di audit [G] e remediation [G] delle vulnerabilità del software. L'architettura si basa sul paradigma degli agenti [G] software intelligenti, operanti su repository di codice sorgente. La conformità del sistema è vincolata ai requisiti definiti nel Capitolato **C2**.

La piattaforma supporta attività di analisi statica del codice sorgente e di individuazione delle principali criticità di sicurezza, fornendo suggerimenti di correzione attraverso meccanismi automatizzati basati su modelli di linguaggio di grandi dimensioni (LLM [G]).

1.2 Finalità del Documento

Il Piano di Qualifica definisce l'impostazione metodologica per la gestione della qualità, specificando come il gruppo intenda prevenire, rilevare e correggere i difetti.

Il documento costituisce il riferimento primario per il Responsabile [G] e per i Verificatori [G], strutturando gli obiettivi nelle seguenti macro-aree:

- **Piano della Qualità (Quality Assurance)**: definizione della strategia di gestione della qualità, identificando gli standard di riferimento (in particolare ISO/IEC 25010 [G]), le metriche di misurazione e le relative soglie di accettazione/ottimalità.
- **Controllo di Qualità (Quality Control)**: pianificazione operativa delle attività di Verifica [G] (analisi statica, test dinamici) per garantire la correttezza tecnica degli artefatti prodotti.
- **Validazione di Prodotto**: definizione delle procedure necessarie per accertare che il sistema soddisfi i bisogni effettivi degli Stakeholder [G] e i requisiti del capitolato.
- **Miglioramento Continuo**: applicazione di meccanismi retroattivi (basati sul ciclo Plan-Do-Check-Act [G]) che utilizzano i risultati delle misurazioni per ottimizzare i processi e il **Way of Working [G]** in corso d'opera.

1.3 Traguardi Qualitativi

L'assicurazione della qualità segue l'approccio incrementale del progetto, fissando obiettivi specifici per le due principali milestone:

1.3.1 Revisione dei Requisiti e della Tecnologia (RTB)

Per la milestone RTB (**25/02/2026**), le attività di qualità si concentrano sulla correttezza formale e sulla fattibilità tecnica:

- **Qualità dei Documenti**: Verifica approfondita della documentazione (Analisi dei Requisiti, PdP, NdP) tramite analisi statica e walkthrough, per garantire assenza di ambiguità e coerenza interna (Indice di Gulpease).
- **Qualità del Prototipo (PoC [G])**: L'attività di verifica è focalizzata esclusivamente sulla **dimostrazione della fattibilità tecnica** (Technology Baseline), con particolare attenzione all'interazione Agenti-LLM. Il testing in questa fase ha valore *sperimentale e propedeutico*: esso funge da caso di studio per calibrare le metriche e validare le strategie di verifica che saranno poi applicate in modo sistematico ed estensivo sul MVP [G].

1.3.2 Revisione di Accettazione (Product Baseline – PB)

Per il rilascio finale (**27/04/2026**), il focus si sposta sulla robustezza, sulla copertura e sulla soddisfazione dei requisiti:

- **Qualità del Prodotto (MVP)**: Esecuzione completa dei Test di Unità (UT), Test di Integrazione (IT) e Test di Sistema (ST). Validazione finale rispetto ai requisiti funzionali e prestazionali del capitolato.
- **Qualità del Codice**: Rispetto dei vincoli di stile, assenza di **code smells** [G] e raggiungimento delle soglie di copertura del codice Code Coverage [G] definite nel presente piano.
- **Validazione Utente**: Verifica dell'usabilità tramite Test di Accettazione (TA) basati sui casi d'uso principali.

1.4 Glossario

Al fine di prevenire ambiguità interpretative, è stato redatto un glossario che definisce in modo univoco la terminologia tecnica, gli acronimi e i concetti di dominio utilizzati all'interno della documentazione.

Nel testo, **ogni termine evidenziato tramite una G come apice**, rimanda alla voce corrispondente del Glossario pubblicato sul sito ufficiale del gruppo, consentendo al lettore di accedere direttamente alla definizione associata.

La versione più recente del Glossario è disponibile al seguente link: Link al Glossario (v2.0.0).

1.5 Riferimenti

1.5.1 Riferimenti Normativi

I seguenti documenti hanno valore vincolante per la definizione delle strategie di qualità e per le attività di verifica:

- **Capitolato C2**: Piattaforma ad agenti per l'audit e la remediation dei repository software.
  https://www.math.unipd.it/~tullio/IS-1/2025/Progetto/C2.pdf
  (ultimo accesso: **27/04/2026**)
- **Norme di Progetto**: Il documento definisce il "Way of Working", stabilendo gli strumenti e le procedure che questo Piano si occupa di misurare.
  https://skarabgroup.github.io/DocumentazioneProgetto/PB/NdP.pdf
  (versione: **v2.0.0**)

1.5.2 Riferimenti Informativi

- **ISO/IEC 25010:2011**: Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE).
  https://iso25000.com/index.php/en/iso-25000-standards/iso-25010
  (ultimo accesso: **27/04/2026**)
- **ISO/IEC 12207:2008**: Systems and software engineering – Software life cycle processes.
  https://ieeexplore.ieee.org/document/4475826
  (ultimo accesso: **27/04/2026**)
- **Dispense del corso di Ingegneria del Software – Qualità del software**
  https://www.math.unipd.it/~tullio/IS-1/2025/Dispense/T07.pdf
  (ultimo accesso: **27/04/2026**)

2 Qualità di Processo

La garanzia della qualità del prodotto finale è intrinsecamente legata alla qualità dei processi produttivi che lo generano. Per il progetto *Code Guardian*, la gestione dei processi mira a rendere il Way of Working [G] sostenibile, tracciabile e soggetto a miglioramento continuo attraverso l'applicazione del ciclo PDCA.

2.1 Centralizzazione delle Metriche e Obiettivi

Il presente documento costituisce il riferimento unico e autoritativo per la gestione della qualità del progetto Code Guardian. Il Piano di Qualifica centralizza la 'scienza della misurazione' del gruppo, definendo rigorosamente i criteri analitici, le metriche e le soglie necessarie per garantire la conformità degli artefatti agli standard prefissati.

In particolare, ogni metrica qui esposta è corredata da:

- **Identificativo univoco:** (MPC per il processo, MPD per il prodotto);
- **Formulazione matematica:** Per garantire l'oggettività del calcolo;
- **Soglie di Valutazione:** Distinte in "Accettabilità" (requisito minimo per la validazione) e "Ottimalità" (target di eccellenza desiderato).

Ogni scostamento rilevato tra i valori misurati e le soglie qui definite viene analizzato durante le retrospettive di fine Sprint [G]. Tali evidenze costituiscono la base oggettiva per l'attivazione di azioni correttive o per la ricalibrazione delle soglie stesse, garantendo che il processo di qualità evolva insieme alla maturità del team.

2.1.1 Processi Primari: Fornitura e Sviluppo

Questi processi definiscono le attività core per la realizzazione del software. Il monitoraggio si focalizza sul rispetto dei vincoli di tempo e budget (tramite la metodologia EVM) e sulla gestione rigorosa dell'ambito di progetto.

ID	Nome	Formula	V. Accettabile	V. Ottimo
MPC01	Budget at Completion (BAC)	Preventivo	Preventivo	Preventivo
MPC02	Planned Value (PV)	PV	≥ 0	Da Piano
MPC03	Actual Cost (AC)	AC	$\leq EAC$	$\leq EV$
MPC04	Earned Value (EV)	EV	$\geq 90\% PV$	$\geq PV$
MPC05	Budget Variance (BV)	$BV = BAC - EAC$	≥ 0	> 0
MPC06	Schedule Variance (SV)	$SV = EV - PV$	$> -10\%$ BAC	≥ 0
MPC07	Cost Performance Index (CPI)	$CPI = \frac{EV}{AC}$	$0.90 \leq v \leq 1.10$	1.00
MPC08	Schedule Performance Index (SPI)	$SPI = \frac{EV}{PV}$	$0.90 \leq v \leq 1.10$	1.00
MPC09	Estimate at Completion (EAC)	$EAC = \frac{BAC}{CPI}$	$\leq BAC + 5\%$	$\leq BAC$

Table 1: Soglie metriche per il processo di Fornitura (EVM)

Il monitoraggio della stabilità dei requisiti è cruciale per prevenire lo **scope creep**, specialmente a seguito delle revisioni correttive post-S2.

ID	Nome	Formula	V. Accettabile	V. Ottimo
MPC10	Requirements Stability Index	$RSI = \frac{R_{\text{tot}} - \Delta R}{R_{\text{tot}}} \times 100$	$\geq 75\%$	100%

Table 2: Soglie metriche per il processo di Sviluppo

2.1.2 Processi di Supporto

I processi di supporto garantiscono l'integrità e la verificabilità degli artefatti. La leggibilità della documentazione (Indice di Gulpease) e la copertura dei test sono i parametri cardine per assicurare la manutenibilità futura.

ID	Nome	Formula	V. Accettabile	V. Ottimo
MPC11	Gulpease Index	$89 + \frac{300(L_f) - 10(L_p)}{F_p}$	≥ 40	≥ 60
MPC12	Correttezza Ortografica	Errori segnalati	0	0
MPC13	Code Coverage	$\frac{\text{Linee coperte}}{\text{Linee totali}} \times 100$	$\geq 70\%$	$\geq 80\%$
MPC14	Test Success Rate	$\frac{\text{Passati}}{\text{Eseguiti}} \times 100$	100%	100%

Table 3: Soglie metriche Documentazione e Verifica

2.1.3 Processi Organizzativi

Misurano l'efficienza interna del team Skarab Group nell'auto-organizzarsi e nel rispettare gli impegni presi durante gli Sprint.

ID	Nome	Formula	V. Accettabile	V. Ottimo
MPC15	Metrics Satisfaction	$\frac{\text{Metriche OK}}{\text{Metriche Tot}} \times 100$	$\geq 90\%$	100%
MPC16	Sprint Goal Achievement	$\frac{\text{Completati}}{\text{Pianificati}} \times 100$	$\geq 80\%$	100%

Table 4: Soglie metriche Organizzative

3 Qualità di Prodotto

La qualità di prodotto valuta il software consegnato rispetto ai requisiti e alle caratteristiche intrinseche definite dallo standard ISO/IEC 25010.

3.1 Adeguatezza Funzionale e Affidabilità

Si misura la capacità del sistema di svolgere i compiti richiesti e di rimanere operativo senza guasti critici, parametro fondamentale per un tool di audit.

ID	Nome	Formula	V. Accettabile	V. Ottimo
MPD01	Copertura Req. Obbligatori	$\frac{\text{Soddisfatti}}{\text{Totale Obbl.}} \times 100$	100%	100%
MPD02	Failure Density	$\frac{\text{N. guasti}}{\text{KLOC}}$	≤ 0.5	0
MPD03	Availability	$\frac{\text{Tempo Up}}{\text{Tempo Tot}} \times 100$	$\geq 98\%$	$\geq 99.9\%$

Table 5: Metriche Adeguatezza e Affidabilità

3.2 Manutenibilità e Sicurezza

Data la natura del progetto Code Guardian, queste metriche rappresentano il valore distintivo del prodotto. Un codice manutenibile e privo di vulnerabilità è condizione necessaria per l'accettazione.

ID	Nome	Formula	V. Accettabile	V. Ottimo
MPD04	Comment Density	$\frac{\text{Linee commento}}{\text{Linee codice}} \times 100$	$\geq 15\%$	$20\% - 25\%$
MPD05	Cyclomatic Complexity	$V(G)$	≤ 15	≤ 10
MPD06	Coupling (Fan-out)	Dipendenze esterne	≤ 6	≤ 3
MPD07	Vulnerability Detection	N. vulnerabilità critiche	0	0

Table 6: Metriche Manutenibilità e Sicurezza

4 Strategie di Testing

Il processo di testing rappresenta una fase cruciale nello sviluppo del prodotto *CodeGuardian*.

Skarab Group ha adottato un approccio di testing multilivello che copre:

- **Test di Sistema (ST)**.
- **Test di Unità (UT)**.
- **Test di Accettazione (TA)**.
- **Test di Regressione (RT)**.
- **Test di Integrazione (IT)**.

La definizione dei test e la nomenclatura utilizzata sono presenti all'interno delle **Norme di Progetto**, alla sezione **2.2.1.5.1** (Procedura **PR-SVIL-06**).
I **Test di Integrazione (IT)** sono stati formalizzati e implementati durante lo svolgimento delle attività per la *Product Baseline* (PB), come riportato di seguito. I **Test di Regressione (RT)**, invece, non sono stati implementati nel presente documento in quanto la stabilità delle funzionalità core è stata garantita dalla copertura estensiva offerta dai Test di Unità e dai Test di Integrazione già presenti.

4.1 Test di Sistema (ST)

ID Test	Descrizione	UC	Stato
ST-001	Verificare la corretta creazione di un account CodeGuardian a seguito dell'inserimento di dati validi.	UC1	**Superato**
ST-002	Verificare la validazione dello username rispetto ai vincoli di formato (alfanumerico, 4-20 caratteri).	UC1.1.1	**Superato**
ST-003	Verificare l'inibizione della registrazione in caso di username già associato a un account esistente.	UC1.1.2	**NI**
ST-004	Verificare la validazione sintattica dell'indirizzo email secondo gli standard previsti.	UC1.2.1	**Superato**
ST-005	Verificare l'inibizione della registrazione in caso di email già associata a un account esistente.	UC1.2.2	**Superato**
ST-006	Verificare la validazione della password rispetto ai criteri di complessità (sicurezza).	UC1.3.1	**Superato**
ST-007	Verificare la segnalazione di errore in caso di invio del modulo con campi obbligatori vuoti.	UC1.0.1	**Superato**
ST-008	Verificare l'accesso alle funzionalità riservate tramite inserimento di credenziali corrette.	UC2	**Superato**

ID Test	Descrizione	UC	Stato
ST-009	Verificare la validazione del formato delle credenziali in fase di login.	UC2	**Superato**
ST-010	Verificare la segnalazione di errore per identificativo non presente a sistema.	UC2.0.2	**Superato**
ST-011	Verificare la segnalazione di errore in caso di password non corrispondente all'identificativo fornito.	UC2.0.2	**Superato**
ST-012	Verificare l'inibizione dell'accesso in caso di modulo di login incompleto.	UC2.0.1	**Superato**
ST-013	Verificare la corretta presa in carico del sistema di una richiesta di analisi per un repository GitHub.	UC4	**Superato**
ST-014	Verificare la segnalazione del messaggio informativo se l'analisi risulta già aggiornata rispetto ai dati remoti.	UC4.0.1	**Superato**
ST-015	Verificare l'impossibilità di non selezionare almeno un'area di interesse.	UC4.1.1	**Superato**
ST-016	Verificare la navigazione e la corretta visualizzazione dell'elenco dei repository analizzati.	UC5	**Superato**
ST-017	Verificare l'inibizione del rendering e la notifica di errore qualora i servizi di persistenza non siano raggiungibili.	UC5.0.2	**Superato**
ST-018	Verificare il caricamento della dashboard di dettaglio a seguito della selezione di un report.	UC6	**Superato**
ST-019	Verificare l'aggiornamento dinamico delle sezioni visibili tramite i filtri (Codice, Sicurezza, Doc.).	UC6.1	**Superato**
ST-020	Verificare l'impossibilità di non selezionare alcuna area da visualizzare.	UC6.1.1	**Superato**
ST-021	Verificare l'esposizione corretta dei metadati di audit (timestamp, hash commit, richiedente).	UC6.2.1, UC6.2.2	**Superato**

ID Test	Descrizione	UC	Stato
ST-022	Verificare la visualizzazione del messaggio di assenza di criticità se non vi sono remediation.	UC6.3.1.1	**Superato**
ST-023	Verificare la generazione della vista comparativa previo inserimento di un intervallo valido.	UC7	**Superato**
ST-024	Verificare la segnalazione di errore in caso di invio con campi temporali incompleti.	UC7.0.1	**NI**
ST-025	Verificare la segnalazione di assenza dati se non vi sono report nel periodo scelto.	UC7.0.2	**NI**
ST-026	Verificare la segnalazione di errore in caso di data inizio successiva alla data fine.	UC7.0.3	**NI**
ST-027	Verificare l'inibizione della richiesta se l'intervallo supera l'ampiezza massima (12 mesi).	UC7.0.4	**NI**
ST-028	Verificare la corretta generazione dei grafici di andamento e della tabella comparativa.	UC8	**Superato**
ST-029	Verificare l'esposizione dei dati puntuali all'interazione (click/hover) con il grafico.	UC8	**NI**
ST-030	Verificare il calcolo e la visualizzazione degli indicatori di trend in tabella.	UC8	**NI**
ST-031	Verificare l'esposizione dei rilievi di analisi statica (bug, smell, vulnerabilità).	UC9.1	**Superato**
ST-032	Verificare la visualizzazione delle metriche di copertura dei Test di Unità (UT).	UC9.2	**Superato**
ST-033	Verificare la visualizzazione dell'informativa di esito positivo per l'area codice.	UC9.3.1	**Superato**
ST-034	Verificare l'esposizione delle vulnerabilità delle librerie e conformità OWASP.	UC10.1, UC10.2	**Superato**
ST-035	Verificare la visualizzazione dell'informativa di assenza criticità di sicurezza.	UC10.3.1	**Superato**
ST-036	Verificare la visualizzazione degli errori sintattici e della completezza documentale.	UC11.1, UC11.2	**Superato**

ID Test	Descrizione	UC	Stato
ST-037	Verificare la visualizzazione dell'informativa di assenza criticità documentali.	UC11.3.1	**Superato**
ST-038	Verificare la generazione della graduatoria ordinata per punteggio di qualità globale.	UC12	**Superato**
ST-039	Verificare la segnalazione di assenza dati se l'utente non ha mai effettuato analisi.	UC12.1	**Superato**
ST-040	Verificare il corretto download del report nel formato selezionato (PDF/JSON).	UC14, UC14.2	**Superato**
ST-041	Verificare l'impossibilità per l'utente di non selezionare alcun formato.	UC14.1.1	**Superato**
ST-042	Verificare la segnalazione di errore in caso di password corrente omessa.	UC15.1.1	**Superato**
ST-043	Verificare la segnalazione di errore in caso di password corrente errata.	UC15.1.2	**Superato**
ST-044	Verificare la segnalazione di errore se la nuova password è assente o non conforme.	UC15.2.1, UC15.2.2	**Superato**
ST-045	Verificare la segnalazione di errore se la nuova password coincide con la precedente.	UC15.2.3	**Superato**
ST-046	Verificare la corretta persistenza e la notifica di successo post-modifica.	UC15.3	**Superato**
ST-047	Verificare la corretta visualizzazione dei dettagli di una remediation selezionata.	UC16	**NI**
ST-048	Verificare che il Sistema verifichi con successo l'accessibilità di un repository pubblico tramite le API GitHub.	UC17	**Superato**
ST-049	Verificare la gestione dell'errore di comunicazione con GitHub.	UC17.0.1	**Superato**
ST-050	Verificare che il Sistema tenti l'accesso tramite credenziali in caso di repository privato.	UC17.1	**Superato**
ST-051	Verificare che il Sistema annulli l'audit se tutti i metodi di accesso falliscono.	UC17.1.1	**Superato**

ID Test	Descrizione	UC	Stato
ST-052	Verificare che l'Utente Avanzato possa accettare una singola remediation.	UC18	**NI**
ST-053	Verificare che l'Utente Avanzato possa rifiutare una singola remediation.	UC19	**NI**
ST-054	Verificare la corretta creazione di una raccolta di report a seguito dell'inserimento di nome e URL validi.	UC20	**Superato**
ST-055	Verificare la segnalazione di errore in caso di tentativo di conferma con campi obbligatori non popolati.	UC20.0.1	**Superato**
ST-056	Verificare la corretta acquisizione del nome identificativo della raccolta nel campo dedicato.	UC20.1	**Superato**
ST-057	Verificare la segnalazione di errore in caso di nome raccolta non conforme ai vincoli alfanumerici.	UC20.1.1	**Superato**
ST-058	Verificare la corretta acquisizione dell'URL del repository GitHub nel campo dedicato.	UC20.2	**Superato**
ST-059	Verificare la segnalazione di errore in caso di URL sintatticamente non valido.	UC20.2.1	**Superato**
ST-060	Verificare la segnalazione di errore in caso di repository non accessibile.	UC20.2.2	**Superato**
ST-061	Verificare la segnalazione di errore in caso di campo URL non popolato al momento della conferma.	UC20.2.3	**Superato**
ST-062	Verificare la corretta acquisizione della descrizione della raccolta.	UC20.3	**Superato**
ST-063	Verificare che l'Orchestratore avvii le richieste verso tutti gli strumenti esterni.	UC21	**Superato**
ST-064	Verificare la corretta clonazione del repository nell'ambiente AWS.	UC21.1	**Superato**
ST-065	Verificare che l'Orchestratore interrompa il processo in caso di errore durante la clonazione.	UC21.1.1	**Superato**

ID Test	Descrizione	UC	Stato
ST-066	Verificare che l'Orchestratore inoltri i file allo strumento di analisi del codice.	UC21.2	**Superato**
ST-067	Verificare che l'Orchestratore inoltri i file allo strumento di analisi documentale.	UC21.3	**Superato**
ST-068	Verificare che l'Orchestratore inoltri la codebase allo strumento di analisi della sicurezza.	UC21.4	**Superato**
ST-069	Verificare che lo stato dell'analisi venga registrato correttamente come "pending" nella persistenza.	UC22	**Superato**
ST-070	Verificare che, in caso di errore critico nella scrittura dello stato, l'Orchestratore notifichi l'utente.	UC22.0.1	**Superato**
ST-071	Verificare che l'Orchestratore recuperi correttamente i risultati al completamento delle analisi.	UC23	**Superato**
ST-072	Verificare che il sistema proceda con i soli dati disponibili in caso di risultati parziali.	UC23.0.1	**Superato**
ST-073	Verificare il corretto controllo periodico dello stato delle attività degli strumenti.	UC23.1	**Superato**
ST-074	Verificare che i file dei risultati vengano acquisiti e validati.	UC23.2	**Superato**
ST-075	Verificare la corretta aggregazione dei dati provenienti dai diversi strumenti in un unico report.	UC24	**Superato**
ST-076	Verificare che il report venga validato prima del salvataggio.	UC24	**Superato**
ST-077	Verificare che il report finale venga archiviato permanentemente.	UC25	**Superato**
ST-078	Verificare la notifica di errore all'utente in caso di fallimento del salvataggio del report.	UC25.0.1	**Superato**
ST-079	Verificare l'invio della notifica di completamento dell'analisi del repository.	UC26	**Superato**

ID Test	Descrizione	UC	Stato
ST-080	Verificare che il fallimento della notifica venga registrato nei log interni.	UC26.0.1	**Superato**
ST-081	Verificare l'esposizione delle informazioni identificative del repository selezionato.	UC27	**Superato**
ST-082	Verificare la corretta cancellazione del profilo a seguito della verifica dell'identità tramite password.	UC28, UC28.1	**Superato**
ST-083	Verificare che a seguito della cancellazione vengano rimossi i dati personali e credenziali vengano invalidate.	UC28.1	**Superato**
ST-084	Verificare la corretta visualizzazione del dettaglio di una singola remediation dell'area codice.	UC30	**NI**
ST-085	Verificare la corretta visualizzazione del dettaglio di una singola remediation dell'area sicurezza.	UC31	**NI**
ST-086	Verificare la corretta visualizzazione del dettaglio di una singola remediation dell'area documentazione.	UC32	**NI**
ST-087	Verificare l'applicazione della remediation del codice e l'aggiornamento dello stato a "eseguita".	UC33	**NI**
ST-088	Verificare la notifica di fallimento all'utente in caso di errore durante l'applicazione.	UC33.0.1	**NI**
ST-089	Verificare che il rifiuto di una remediation del codice aggiorni lo stato a "rifiutata".	UC34	**NI**
ST-090	Verificare l'applicazione delle patch di sicurezza e l'aggiornamento dello stato a "eseguita".	UC35	**NI**
ST-091	Verificare la notifica di fallimento all'utente in caso di errore durante l'applicazione (sicurezza).	UC35.0.1	**NI**
ST-092	Verificare che il rifiuto di una remediation di sicurezza aggiorni lo stato a "rifiutata".	UC36	**NI**

ID Test	Descrizione	UC	Stato
ST-093	Verificare l'applicazione delle modifiche documentali e l'aggiornamento dello stato a "eseguita".	UC37	**NI**
ST-094	Verificare la notifica di fallimento all'utente in caso di errore durante l'applicazione (doc).	UC37.0.1	**NI**
ST-095	Verificare che il rifiuto di una remediation documentale aggiorni lo stato a "rifiutata".	UC38	**NI**
ST-096	Verificare la corretta presa in carico di una richiesta di analisi per un repository privato.	UC39	**Superato**
ST-097	Verificare il corretto inserimento di un repository privato nel catalogo personale.	UC40	**Superato**
ST-098	Verificare la segnalazione di duplicazione in caso di inserimento di un URL già presente.	UC40.0.1	**Superato**
ST-099	Verificare la corretta visualizzazione del catalogo dei repository privati inseriti.	UC41	**Superato**
ST-100	Verificare la visualizzazione dell'informativa specifica quando il catalogo privato risulta vuoto.	UC41.0.1	**Superato**
ST-101	Verificare la corretta rimozione di un repository dal catalogo privato previa conferma esplicita.	UC42, UC42.1	**Superato**
ST-102	Verificare che l'annullamento della rimozione mantenga intatto il catalogo privato.	UC42.1.1	**Superato**
ST-103	Verificare la corretta visualizzazione dell'elenco dei profili autorizzati per un repository privato.	UC43	**NI**
ST-104	Verificare la visualizzazione dell'informativa di assenza utenti autorizzati quando la lista è vuota.	UC43.0.1	**NI**
ST-105	Verificare la corretta aggiunta di un utente autorizzato tramite username o email.	UC44, UC44.1	**NI**
ST-106	Verificare la segnalazione di errore per formato non valido, utente inesistente o campo vuoto.	UC44.1.1, UC44.1.2	**Superato**

ID Test	Descrizione	UC	Stato
ST-107	Verificare la corretta revoca dei permessi di consultazione per un utente precedentemente autorizzato.	UC45, UC45.1	**NI**
ST-108	Verificare la corretta rimozione di una raccolta di report senza che i singoli report vengano eliminati.	UC46, UC46.1	**Superato**
ST-109	Verificare che l'annullamento dell'operazione mantenga intatta la raccolta nel profilo utente.	UC46.1.1	**Superato**

Table 7: Tabella dei Test di Sistema (ST)

4.2 Test di Unità (UT)

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-001	FRObl	Verifica rendering del componente di creazione account.	Caricamento corretto del modulo di registrazione.	**Superato**
UT-002	FRObl2	Verifica predisposizione comando di conferma nel modulo.	Pulsante di conferma presente e funzionante.	**Superato**
UT-003	FRObl3	Verifica esecuzione della validazione completa di tutti i campi obbligatori all'invio.	Tutti i controlli vengono eseguiti prima di procedere.	**Superato**
UT-004	FRObl4	Verifica che la finalizzazione sia consentita solo dopo validazione positiva.	Blocco della procedura in caso di parametri non validi.	**Superato**
UT-005	FRObl5	Verifica logica di persistenza dei dati utente nel DB.	Le credenziali vengono scritte correttamente nel database.	**Superato**
UT-006	FRObl6	Verifica della funzione di hashing.	La password non è leggibile; l'hash prodotto è coerente.	**Superato**
UT-007	FRObl7	Verifica atomicità della registrazione.	Nessun record parziale viene mantenuto nel database.	**Superato**
UT-008	FRObl8	Verifica visualizzazione messaggio di conferma avvenuta creazione account.	Messaggio di conferma mostrato.	**Superato**
UT-009	FRObl9	Controllo rilevamento campi obbligatori vuoti (null check).	Rilevamento campi vuoti nel modulo di registrazione.	**Superato**
UT-010	FRObl10	Verifica logica di inibizione e notifica per campi mancanti.	Impossibilità di procedere; messaggio specifico per campo.	**Superato**
UT-011	FRObl11	Verifica input username: vincoli alfanumerici e lunghezza (4-20).	Rifiuto di stringhe < 4 o > 20 caratteri.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-012	FROp12	Query di verifica unicità dello username nel database.	Identificazione di collisioni con account già esistenti.	**NI**
UT-013	FROp13	Verifica vincolo di unicità lato persistenza su username.	Il sistema rileva la duplicazione e annulla la registrazione.	**NI**
UT-014	FROp14	Verifica notifica username già in uso a seguito di violazione unicità.	Feedback visivo immediato per username non disponibile.	**NI**
UT-015	FRObl5	Verifica inibizione e trigger notifica errore formato username non conforme.	Comparsa del messaggio di errore; procedura inibita.	**Superato**
UT-016	FRDe16	Verifica input email e validazione sintattica secondo standard RFC.	Accettazione di formati standard (user@domain.ext).	**Superato**
UT-017	FRObl7	Verifica rifiuto di email con spazi o prive del carattere “@”.	Email malformate rifiutate con messaggio di errore.	**Superato**
UT-018	FRObl8	Query di verifica unicità email nel database.	Identificazione di email già associate ad altri profili.	**Superato**
UT-019	FRObl9	Verifica vincolo di unicità lato persistenza su email.	Il sistema impedisce registrazioni duplicate.	**Superato**
UT-020	FRObl20	Verifica trigger notifica errore email non valida o già registrata.	Messaggio di errore per email duplicata o malformata.	**Superato**
UT-021	FRObl21	Verifica requisito lunghezza password (minimo 8 caratteri).	Password con meno di 8 caratteri rifiutate.	**Superato**
UT-022	FRObl22	Verifica requisiti complessità password.	Validazione positiva solo se tutti i criteri sono soddisfatti.	**Superato**
UT-023	FRObl23	Verifica rifiuto password coincidente o contenente lo username.	Password che contengono lo username come sottostringa rifiutate.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-024	FRObi24	Verifica trigger notifica errore password non conforme ai requisiti.	Elenco puntuale dei criteri non rispettati.	**Superato**
UT-025	FRObi25	Verifica rendering pagina di Login.	Visualizzazione corretto del form di autenticazione.	**Superato**
UT-026	FRObi26	Verifica predisposizione comando di conferma per il login.	Pulsante di conferma presente e funzionante.	**Superato**
UT-027	FRObi27	Verifica validazione completa credenziali all'invio.	Procedura inibita se uno dei controlli fallisce.	**Superato**
UT-028	FRObi28	Verifica autorizzazione post-validazione credenziali.	Rilascio della sessione solo con dati corretti.	**Superato**
UT-029	FRObi29	Verifica reindirizzamento verso dashboard a seguito di autenticazione.	L'utente viene reindirizzato correttamente.	**Superato**
UT-030	FRObi30	Verifica protocollo di trasmissione credenziali (HTTPS).	Dati cifrati durante il transito verso il server.	**Superato**
UT-031	FRDe31	Verifica utilizzo username per fetch del record account dalla persistenza.	Lo username viene utilizzato per recuperare il record DB.	**NI**
UT-032	FRObi32	Verifica confronto hash password fornita con hash memorizzato.	Accesso concesso solo se coincidono.	**Superato**
UT-033	FRObi33	Test del meccanismo di rate limiting / lockout temporaneo.	Blocco dell'account dopo N tentativi falliti.	**Superato**
UT-034	FRObi34	Verifica visualizzazione spinner durante validazione credenziali.	Indicatore mostrato; bottone disabilitato.	**Superato**
UT-035	FRObi35	Verifica rilevamento campi mancanti e inibizione del login.	Trigger errore per campi vuoti; accesso negato.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-036	FRObb36	Verifica notifica formato email non conforme in fase di login.	Messaggio di errore su formato email errato.	**Superato**
UT-037	FRObb36	Verifica notifica email non esistente nel sistema.	Feedback specifico per email non censito.	**Superato**
UT-038	FRObb36	Verifica notifica formato password errato in login.	Feedback su errore sintattico password.	**Superato**
UT-039	FRObb36	Verifica notifica password errata (hash non corrisponde).	Feedback specifico per credenziali non corrispondenti.	**Superato**
UT-040	FRObb47	Verifica predisposizione campo URL nel modulo di richiesta analisi.	Campo URL presente e funzionante nel modulo.	**Superato**
UT-041	FRObb48	Verifica vincoli sintattici URL: protocollo HTTPS e dominio GitHub.	Validazione positiva solo per URL conformi.	**Superato**
UT-042	FRObb49	Verifica dimensione repository e inibire analisi qualora il limite venisse superato	Analisi bloccata in caso di limite superato	**Superato**
UT-043	FRObb50	Verifica disabilitazione comando di conferma dopo prima pressione.	Bottone disabilitato dopo il click per prevenire duplicati.	**Superato**
UT-044	FROp51	Verifica consegna notifica di fine analisi tramite canali scelti.	Notifica recapitata sul canale configurato.	**NI**
UT-045	FRDe52	Verifica inclusione dettagli analisi (nome, ora) nell'avviso ricevuto.	Messaggio contiene nome progetto e timestamp.	**Superato**
UT-046	FRObb53	Verifica invio avviso immediato con causa errore in caso interruzione.	Notifica di errore con breve spiegazione tempestiva.	**Superato**
UT-047	FRObb54	Verifica restituzione immediata del report esistente.	Notifica all'utente dei dati già aggiornati.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-048	FRObs56	Verifica inibizione richiesta in assenza di selezione aree.	Messaggio di errore se nessuna area è selezionata.	**Superato**
UT-049	FRObs57	Verifica ordinamento lista repository per data (decrescente).	Repository visualizzati in ordine dall'analisi più recente.	**Superato**
UT-050	FRObs61	Validazione logica per visualizzazione informativa lista vuota.	Messaggio informativo mostrato.	**Superato**
UT-051	FRObs62	Verifica inibizione rendering lista in caso di errore persistenza.	Messaggio di errore tecnico mostrato.	**Superato**
UT-052	FRObs63	Verifica presenza e funzionamento comando di aggiornamento (Refresh).	Nuovo tentativo di caricamento avviato al click.	**Superato**
UT-053	FRObs58	Verifica consultabilità risultati	Il report è accessibile dalla dashboard in ogni caso.	**Superato**
UT-054	FRObs59	Verifica contrassegno analisi come "Fallita" nella lista progetti.	Stato "Fallita" visibile nella dashboard.	**Superato**
UT-055	FRObs60	Verifica avviso fallimento con cause nella dashboard.	Avviso mostrato correttamente nella dashboard indipendentemente dalla ricezione dell'avviso di errore.	**Superato**
UT-056	FRObs64	Verifica selezione e caricamento report da lista.	Caricamento riuscito dei dati del report selezionato.	**Superato**
UT-057	FRObs65	Verifica validazione server: report appartiene al repository utente.	Rendering inibito con errore per report non associati.	**Superato**
UT-058	FRObs66	Verifica inibizione rendering per report non autorizzati.	Errore di autorizzazione mostrato.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-059	FRObi67	Verifica gestione timeout nel recupero dati analitici con notifica.	Notifica di indisponibilità temporanea mostrata.	**Superato**
UT-060	FRObi68	Verifica stato dei filtri aree (toggle on/off).	Le aree analitiche sono correttamente filtrate.	**Superato**
UT-061	FRObi69	Verifica aggiornamento dinamico contenuto in base ai filtri.	Il report si aggiorna alla variazione dei filtri.	**Superato**
UT-062	FRObi70	Controllo validazione: almeno un'area attiva nei filtri.	Almeno un'area rimane sempre selezionata.	**Superato**
UT-063	FRObi71	Verifica esposizione metadati identificativi del report.	Metadati caricati correttamente.	**Superato**
UT-064	FRObi72	Verifica correttezza timestamp generazione audit (formato ISO 8601).	Data e ora corrispondono al record del database.	**Superato**
UT-065	FRObi73	Sistema deve visualizzare l'identificativo SHA del commit.	Link al commit fornito.	**Superato**
UT-066	FRObi74	Controllo visualizzazione username richiedente report.	Lo username corrisponde all'autore della richiesta.	**Superato**
UT-067	FRObi75	Verifica integrità metriche tecniche aggregate per aree attive.	Dati numerici visualizzati correttamente.	**Superato**
UT-068	FRObi76	Verifica caricamento e visualizzazione lista azioni correttive.	Lista remediation caricata e associata alle criticità.	**Superato**
UT-069	FRObi77	Verifica espansione dettaglio singola remediation.	Dettaglio tecnico della proposta di risoluzione visibile.	**Superato**
UT-070	FRObi78	Controllo messaggio esito positivo in assenza di criticità.	Badge di conformità mostrato per aree sicure.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-071	FRObi79	Verifica selezione intervallo temporale tramite input di data.	Intervallo acquisito correttamente.	**Superato**
UT-072	FRObi80	Verifica predisposizione e invio comando aggiornamento confronto.	Trigger di ricalcolo attivato correttamente.	**Superato**
UT-073	FRObi81	Controllo campi obbligatori temporali: inibizione se non popolati.	Avviso mostrato e confronto inibito.	**Superato**
UT-074	FRObi82	Validazione coerenza: data inizio precedente alla data fine.	Blocco e segnalazione errore.	**Superato**
UT-075	FRObi83	Controllo ampiezza massima intervallo.	Errore restituito e richiesta inibita.	**Superato**
UT-076	FRObi84	Verifica query di ricerca report in intervallo.	Messaggio “Nessun report trovato” se range vuoto.	**Superato**
UT-077	FRDe85	Verifica logica di generazione dataset per grafici dinamici.	Dati trasformati in serie storiche.	**Superato**
UT-078	FRDe86	Verifica tooltip informativi all’hover sui punti dati del grafico.	Valore esatto mostrato.	**Superato**
UT-079	FRObi87	Verifica allineamento dati tra vista grafica e tabellare.	Dati coerenti tra grafico e tabella.	**Superato**
UT-080	FRObi88	Verifica popolamento righe tabella comparativa in ordine cronologico.	Ordinamento cronologico corretto.	**Superato**
UT-081	FRObi89	Validazione algoritmo calcolo indicatori di variazione.	Calcolo variazione score eseguito correttamente.	**Superato**
UT-082	FRObi90	Verifica fallback visualizzazione dati grezzi in tabella in caso di errore.	Dati mostrati in formato tabellare.	**Superato**
UT-083	FRObi91	Verifica caricamento sezione “Codice” solo se area attiva.	Modulo renderizzato correttamente.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-084	FRObz92	Verifica esposizione risultati analisi statica con gravità e posizione.	Bug, code smell e vulnerabilità mostrati.	**Superato**
UT-085	FRObz93	Controllo calcolo percentuale copertura test (Code Coverage).	Valore normalizzato e conteggio calcolati correttamente.	**Superato**
UT-086	FRObz94	Verifica presentazione lista specifica per codice.	Lista esposta con titolo, file, riga per ogni remediation.	**Superato**
UT-087	FRObz95	Controllo informativa “Codice Conforme” in assenza di bug.	Esito positivo mostrato.	**Superato**
UT-088	FRObz96	Verifica caricamento asincrono sezione sicurezza.	I dati di sicurezza vengono caricati in modo indipendente.	**Superato**
UT-089	FRObz97	Verifica esposizione dipendenze vulnerabili con CVE e CVSS.	Librerie elencate con campi informativi.	**Superato**
UT-090	FRObz98	Verifica del mappatore di conformità OWASP Top 10.	Associazione corretta vulnerabilità-categoria.	**Superato**
UT-091	FRObz99	Verifica presentazione remediation di sicurezza ordinate per criticità.	Remediation esposte in ordine decrescente.	**Superato**
UT-092	FRObz100	Validazione logica “Repository Sicuro” in assenza di vulnerabilità.	Restituisce stato “Safe” se contatore zero.	**Superato**
UT-093	FRObz97	Verifica parser dipendenze vulnerabili da scanner esterni (Trivy/Grype) in `analysis/security`.	Le finding critiche vengono normalizzate e rese disponibili al report.	**Superato**
UT-094	FRObz98	Verifica parser OWASP (Semgrep) con filtro risultati non conformi e gestione errori tool.	Le categorie OWASP vengono mappate correttamente e gli errori non bloccano il flusso.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-095	FRObi01	Verifica caricamento e visualizzazione sezione documentazione.	Sezione renderizzata correttamente.	**Superato**
UT-096	FRObi02	Verifica rilevamento errori sintattici e link interrotti.	Identifica URL malformati o errori testuali.	**Superato**
UT-097	FRObi03	Calcolo indice di completezza documentale su interfacce pubbliche.	Rapporto coerente interfacce-blocchi documentazione.	**Superato**
UT-098	FRObi04	Verifica esposizione suggerimenti per integrazione documentazione.	Suggerimenti visualizzati per lacune rilevate.	**Superato**
UT-099	FRObi05	Controllo informativa “Documentazione Completa” se nessuna criticità.	Esito positivo mostrato.	**Superato**
UT-100	FRObi06	Verifica del calcolo del punteggio di qualità globale pesato.	Media pesata dei punteggi delle tre aree.	**Superato**
UT-101	FRObi07	Algoritmo di generazione graduatoria con ordinamento decrescente.	Lista ordinata dal punteggio più alto al basso.	**Superato**
UT-102	FRObi08	Verifica esposizione dati per riga: posizione, nome, punteggio, trend.	Tutti i campi popolati correttamente.	**Superato**
UT-103	FRObi09	Verifica inibizione rendering ranking in assenza di analisi completate.	Messaggio suggerimento prima analisi mostrato.	**Superato**
UT-104	FRObi14	Verifica disponibilità link di download del file generato.	Link di download presente e funzionante.	**Superato**
UT-105	FRObi15	Verifica supporto formati di esportazione PDF e JSON.	Accetta esclusivamente PDF o JSON.	**Superato**
UT-106	FRObi16	Verifica inibizione invio richiesta in assenza di formato selezionato.	Impossibilità di non selezionare il formato.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-107	FRObi17	Verifica modulo generazione file: mapping dati e metadati.	I dati vengono mappati senza perdite.	**Superato**
UT-108	FRObi18	Verifica processo di generazione file asincrono senza blocco interfaccia.	L'UI rimane responsiva durante il parsing.	**Superato**
UT-109	FRObi19	Verifica accesso alla sezione dedicata alla modifica password.	Rendering corretto del modulo nel profilo.	**Superato**
UT-110	FRObi20	Verifica inserimento obbligatorio della password attualmente in uso durante la modifica password.	Password corrente richiesta e validata.	**Superato**
UT-111	FRObi21	Verifica inibizione della procedura per password non corretta.	Procedura bloccata per password non inserita o errata.	**Superato**
UT-112	FRObi22	Validazione sulla nuova password.	Controllo dei vincoli di complessità sulla nuova password.	**Superato**
UT-113	FRObi23	Verifica inibizione procedura se hash password coincidono.	Impedita la modifica se i valori coincidono.	**Superato**
UT-114	FRObi24	Verifica Adattatori per Modifica Password ('PostgresAdapter' - update).	Le query UPDATE SQL vengono eseguite correttamente garantendo la transazionalità.	**Superato**
UT-115	FRObi25	Verifica invio notifica email automatica a seguito di modifica.	Email di notifica inviata post-cambio.	**Superato**
UT-116	FRObi26	Verifica invalidazione di tutte le sessioni attive post-cambio password.	Sessioni parallele invalidate; corrente attiva.	**Superato**
UT-117	FRObi27	Verifica visualizzazione dettaglio tecnico singola remediation.	Dettaglio correttamente caricato.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-118	FRObi28	Verifica esposizione campi obbligatori remediation (descrizione, snippet).	Tutti i campi previsti presenti.	**Superato**
UT-119	FRObi29	Verifica inclusione riferimenti esterni per vulnerabilità note.	Link a documentazione esterna mostrati.	**Superato**
UT-120	FRObi30	Verifica gestione ciclo verifica accessibilità tramite API GitHub.	Chiamate asincrone eseguite e tracciate.	**Superato**
UT-121	FRObi31	Verifica meccanismo “Exponential Backoff” per errori di rete.	Ritardi crescenti; interruzione su max tentativi.	**Superato**
UT-122	FRObi32	Verifica validazione raggiungibilità endpoint tramite “Heartbeat”.	Verifica operatività servizio remoto.	**Superato**
UT-123	FRObi33	Verifica tentativo accesso pubblico prima dell’uso di credenziali.	Richiesta senza intestazioni di autorizzazione.	**Superato**
UT-124	FRObi34	Verifica accesso privato via token su errore 403/404 della risorsa.	Seconda richiesta con token iniettato.	**Superato**
UT-125	FRObi35	Verifica controllo “scopes” del token: permessi minimi di lettura.	Token insufficiente viene rifiutato.	**Superato**
UT-126	FRObi36	Verifica aggiornamento stato analisi a “FAILED_ACCESS”.	Stato impostato a “FAILED_ACCESS”.	**Superato**
UT-127	FRDe137	Verifica applicazione automatica modifiche tramite integrazione GitHub.	Commit inviato al repository remoto.	**NI**
UT-128	FRDe138	Verifica validazione di integrità della proposta correttiva prima del commit.	Proposta validata; commit bloccato se fallisce.	**NI**
UT-129	FRDe139	Verifica aggiornamento stato remediation in “Applied” nel DB.	Stato correttamente aggiornato.	**NI**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-130	FRDe140	Verifica notifica all'utente in caso di fallimento del commit.	Messaggio errore; codebase invariata.	**NI**
UT-131	FRDe141	Verifica definizione nome univoco raccolta.	Nomi non conformi rifiutati.	**Superato**
UT-132	FRDe142	Verifica validazione sintattica URL GitHub.	URL non conformi vengono rifiutati.	**Superato**
UT-133	FRDe143	Verifica interrogazione API GitHub per conferma esistenza repository.	Repository inesistenti bloccano la raccolta.	**Superato**
UT-134	FRDe144	Verifica gestione repository inaccessibile con notifica utente.	Avviso specifico mostrato.	**Superato**
UT-135	FRDe145	Verifica impedimento creazione raccolta duplicata per stesso utente.	Errore di duplicazione gestito.	**Superato**
UT-136	FRDe146	Verifica memorizzazione descrizione facoltativa con supporto UTF-8.	Caratteri speciali memorizzati correttamente.	**Superato**
UT-137	FRDe147	Verifica parallelizzazione richieste verso gli strumenti esterni.	Richieste inviate in parallelo.	**Superato**
UT-138	FRDe148	Verifica inclusione parametri di configurazione utente nelle richieste.	Parametri trasmessi correttamente.	**Superato**
UT-139	FRDe149	Verifica trasmissione sicura credenziali al servizio AWS per clonazione.	Credenziali non esposte in chiaro.	**Superato**
UT-140	FRDe150	Verifica monitoraggio completamento clonazione e gestione timeout.	Procedura interrotta con segnalazione su errori.	**Superato**
UT-141	FRDe151	Verifica inibizione inoltro strumenti su errore clonazione.	Nessuna richiesta inoltrata agli strumenti.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-142	FRObr152	Verifica inoltro codebase agli strumenti di analisi via protocolli sicuri.	File trasmessi tramite canali cifrati.	**Superato**
UT-143	FRObr147	Verifica orchestrazione processi agenti locali (code/ docs/security) e corretta raccolta stream output.	I processi vengono avviati, monitorati e chiusi senza deadlock.	**Superato**
UT-144	FRObr153	Verifica registrazione stato analisi come “PENDING”.	Stato scritto correttamente nel DB.	**Superato**
UT-145	FRObr154	Verifica associazione univoca ID analisi a repository e utente.	ID associato; impossibile creare duplicati.	**Superato**
UT-146	FRObr155	Verifica persistenza metadati di avvio (hash commit, timestamp).	Dati registrati all’avvio.	**Superato**
UT-147	FRObr156	Verifica Rollback e segnalazione su errore scrittura stato.	Analisi annullata su fallimento persistenza.	**Superato**
UT-148	FRObr157	Verifica registrazione log di audit su fallimento persistenza.	Log di errore completi scritti.	**Superato**
UT-149	FRObr158	Verifica polling o ricezione segnale completamento strumenti.	Rilevata disponibilità risultati.	**Superato**
UT-150	FRObr159	Verifica download risultati non appena disponibili.	File scaricati correttamente.	**Superato**
UT-151	FRObr160	Verifica controllo integrità file ricevuti.	File corrotti rilevati e segnalati.	**Superato**
UT-152	FRObr161	Verifica prosecuzione report con dati parziali su fallimento strumento.	Report generato con i dati disponibili.	**Superato**
UT-153	FRObr162	Verifica impostazione timeout massimo per strumento ritardatario.	Attesa interrotta post timeout.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-154	FRObi63	Verifica segnalazione nel DB di report con dati parziali.	Flag “parziale” impostato.	**Superato**
UT-155	FRObi64	Verifica unificazione dati in documento di sintesi unico.	Output aggregati correttamente.	**Superato**
UT-156	FRObi65	Verifica conversione formati eterogenei in modello standard.	Formati normalizzati senza perdite.	**Superato**
UT-157	FRObi66	Verifica validazione completezza report prima del salvataggio.	Report incompleto non inoltrato al DB.	**Superato**
UT-158	FRObi67	Verifica calcolo punteggi di riepilogo per aree.	Punteggi calcolati coerentemente.	**Superato**
UT-159	FRObi66	Verifica costruzione entità report (code/documentation/ security) e validazione dei value object di supporto.	Le entità risultano consistenti e serializzabili nel modello di dominio.	**Superato**
UT-160	FRObi68	Verifica archiviazione permanente report con collegamento repository.	Report salvato e associato a profilo e repo.	**Superato**
UT-161	FRObi69	Verifica aggiornamento stato analisi a “Completato”.	Stato impostato a seguito di conferma scrittura.	**Superato**
UT-162	FRObi70	Verifica notifica utente in caso impossibilità salvataggio.	Messaggio di errore generato per notifica.	**Superato**
UT-163	FRObi71	Verifica tracciamento fallimento salvataggio per audit.	Log errore scritto dettagliatamente.	**Superato**
UT-164	FRObi72	Verifica copia temporanea report su errore salvataggio definitivo.	Copia disponibile per recupero.	**Superato**
UT-165	FRObi68	Verifica persistenza Mongo dei report e collegamento corretto tra analisi, report e utente.	Le associazioni sono mantenute correttamente nel database.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-166	FRObi73	Verifica generazione automatica notifica utente a report salvato.	Notifica generata dopo archiviazione.	**Superato**
UT-167	FRObi74	Verifica inclusione link accesso diretto nella notifica.	Notifica contiene riferimento report.	**Superato**
UT-168	FRObi75	Verifica inclusione informazioni identificative nella notifica.	Nome repo e data presenti nel messaggio.	**Superato**
UT-169	FRObi76	Verifica indipendenza stato report dall'esito invio notifica.	Report disponibile anche se notifica fallisce.	**Superato**
UT-170	FRObi77	Verifica registrazione anomalia su errore invio notifica.	Log aggiornato con causa fallimento.	**Superato**
UT-171	FRObi78	Verifica meccanismo di retry invio notifica per problemi di rete.	Tentativi ripetuti prima di rinunciare.	**Superato**
UT-172	FRObi79	Controllo mapping metadati: Nome, URL, Data ultima analisi.	Oggetti popolati correttamente.	**Superato**
UT-173	FRObi80	Verifica Servizi e Controller di Cancellazione Account ('DeleteService', 'DeleteUserController').	Il flusso di rimozione account invalida i token ed elimina i dati utente senza errori.	**Superato**
UT-174	FRObi81	Verifica della corretto avviso di azione irreversibile.	Messaggio di conferma con possibilità di tornare indietro mostrato.	**Superato**
UT-175	FRObi82	Verifica Adattatori per Cancellazione Account ('PostgresAdapter' - deleteUser).	La query DELETE SQL rimuove coerentemente il profilo dal database.	**Superato**
UT-176	FRDe183	Verifica scambio codice OAuth GitHub in token persistente e gestione risposte non valide.	Il token viene ottenuto solo con codice valido.	**NI**
UT-177	FRDe184	Verifica cifratura del token GitHub prima del salvataggio su persistenza.	Il token non viene mai salvato in chiaro.	**NI**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-178	FRDe185	Verifica associazione univoca token OAuth al profilo utente autorizzante.	Ogni token resta vincolato all'utente proprietario.	**NI**
UT-179	FRDe186	Verifica annullamento collegamento su codice OAuth scaduto/non valido con richiesta nuova procedura.	Il sistema invalida la procedura corrente e richiede nuova autorizzazione.	**NI**
UT-180	FRDb187	Verifica caricamento dettaglio singola remediation codice.	Dettaglio caricato correttamente.	**Superato**
UT-181	FRDb188	Verifica presenza campi obbligatori nella remediation codice.	Campi previsti presenti.	**Superato**
UT-182	FRDb189	Verifica caricamento dettaglio singola remediation sicurezza.	Dettaglio caricato correttamente.	**Superato**
UT-183	FRDb190	Verifica presenza campi obbligatori nella remediation sicurezza.	Campi previsti presenti.	**Superato**
UT-184	FRDb191	Verifica caricamento dettaglio singola remediation documentazione.	Dettaglio caricato correttamente.	**Superato**
UT-185	FRDb192	Verifica presenza campi obbligatori nella remediation documentale.	Campi previsti presenti.	**Superato**
UT-186	FRDe193	Verifica abilitazione comando accettazione remediation codice (Utente Avanzato).	Solo Utente Avanzato vede comando.	**NI**
UT-187	FRDe194	Verifica applicazione modifiche codebase a seguito accettazione.	Commit inviato al repository.	**NI**
UT-188	FRDe195	Verifica aggiornamento stato remediation codice a "eseguita".	Stato aggiornato correttamente.	**NI**
UT-189	FRDe196	Verifica notifica fallimento e invarianza codebase in caso errore.	Nessuna modifica in caso di fallimento.	**NI**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-190	FRDe197	Verifica abilitazione comando di rifiuto remediation codice.	Comando disponibile nell'interfaccia.	**NI**
UT-191	FRDe198	Verifica aggiornamento stato a "rifiutata" senza modifiche.	Stato aggiornato; codebase invariata.	**NI**
UT-192	FRDe199	Verifica abilitazione comando di accettazione remediation sicurezza.	Comando disponibile per Utente Avanzato.	**NI**
UT-193	FRDe200	Verifica applicazione patch/ configurazioni di sicurezza.	Modifiche applicate al repository.	**NI**
UT-194	FRDe201	Verifica aggiornamento stato remediation sicurezza a "eseguita".	Stato aggiornato nella dashboard.	**NI**
UT-195	FRDe202	Verifica notifica fallimento applicazione remediation sicurezza.	Vulnerabilità non mitigata in caso insuccesso.	**NI**
UT-196	FRDe203	Verifica abilitazione comando di rifiuto remediation sicurezza.	Comando disponibile nell'area.	**NI**
UT-197	FRDe204	Verifica aggiornamento stato a "rifiutata" senza modifiche.	Repository invariato.	**NI**
UT-198	FRDe205	Verifica abilitazione comando accettazione remediation documentale.	Comando disponibile per Utente Avanzato.	**NI**
UT-199	FRDe206	Verifica applicazione modifiche ai file documentali a seguito accettazione.	File aggiornati nel repository.	**NI**
UT-200	FRDe207	Verifica aggiornamento stato remediation documentale a "eseguita".	Stato aggiornato nella dashboard.	**NI**
UT-201	FRDe208	Verifica notifica errore e invarianza documentazione in caso di fallimento I/O.	File documentazione rimangono invariati.	**NI**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-202	FRDe209	Verifica accesso al dettaglio per procedura di rifiuto documentale.	Dettaglio visualizzato prima del rifiuto.	**NI**
UT-203	FRDe210	Verifica abilitazione comando di rifiuto remediation documentale.	Comando disponibile nell'area.	**NI**
UT-204	FRDe211	Verifica aggiornamento stato remediation documentale a "rifiutata".	Stato aggiornato a seguito del rifiuto.	**NI**
UT-205	FRDe212	Verifica che il rifiuto non comporti modifiche ai file.	Nessuna modifica al repository.	**NI**
UT-206	FRDe213	Verifica rimozione/marcatura visiva remediation rifiutata.	Remediation non più pendente.	**NI**
UT-207	FRDe214	Verifica visualizzazione conferma visiva avvenuto rifiuto.	Messaggio di conferma rifiuto mostrato.	**NI**
UT-208	FRDe215	Verifica abilitazione richiesta repository privato solo dopo aver inserito PAT token del repositoy.	Funzionalità inibita per utenti che non l'hanno inserito.	**Superato**
UT-209	FRDe216	Verifica validazione integrazione GitHub attiva per risorse private.	Richiesta bloccata senza token valido.	**Superato**
UT-210	FRDe217	Verifica inibizione richiesta analisi in assenza selezione aree.	Errore mostrato se nessuna area attiva.	**Superato**
UT-211	FRDe218	Verifica inserimento URL repository privato nel catalogo.	Repository aggiunto correttamente.	**Superato**
UT-212	FRDe219	Verifica impedimento inserimento URL duplicato con notifica.	Catalogo invariato su duplicati.	**Superato**
UT-213	FRDe220	Verifica ordinamento elenco repository privati in ordine decrescente.	Lista ordinata per data inserimento.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-214	FRObi221	Verifica visualizzazione informativa catalogo vuoto.	Messaggio suggerimento mostrato.	**Superato**
UT-215	FRObi222	Verifica esposizione lista repository privati con nome e URL.	Lista popolata correttamente.	**Superato**
UT-216	FRObi223	Verifica rimozione repository previa conferma esplicita utente.	Repository rimosso dopo conferma.	**Superato**
UT-217	FRObi224	Verifica integrità catalogo in caso di annullamento rimozione.	Repository mantenuto nel catalogo.	**Superato**
UT-218	FRObi225	Verifica avvio procedura di rimozione di un repository.	Caricamento corretto del dialog.	**Superato**
UT-219	FRObi226	Verifica visualizzazione elenco profili autorizzati per repository.	Lista caricata correttamente.	**Superato**
UT-220	FRObi227	Verifica informativa per accesso limitato esclusivamente al proprietario.	Messaggio mostrato se lista è vuota.	**Superato**
UT-221	FRObi228	Verifica esposizione informazioni identificative del profilo autorizzato.	Username/email visibili.	**Superato**
UT-222	FRObi229	Verifica aggiunta utente autorizzato con validazione profilo.	Profilo inesistente rifiutato con errore.	**Superato**
UT-223	FRObi230	Verifica validazione corrispondenza identificativo in piattaforma.	Identificativo non trovato genera avviso.	**Superato**
UT-224	FRObi231	Verifica impedimento autorizzazione multipla medesimo profilo.	Avviso duplicazione; lista invariata.	**Superato**
UT-225	FRObi232	Verifica predisposizione comando conferma per aggiunta utente.	Pulsante presente e funzionante.	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
UT-226	FRObi233	Verifica notifica errore sintattico per identificativo non valido.	Messaggio di errore mostrato.	**Superato**
UT-227	FRObi234	Verifica inibizione form autorizzazione con identificativo vuoto.	Procedura inibita su campo vuoto.	**Superato**
UT-228	FRObi235	Verifica revoca permessi utente autorizzato previa conferma.	Profilo rimosso; accesso revocato.	**Superato**
UT-229	FRObi236	Verifica selezione utente e avvio procedura revoca.	Azione di revoca avviata.	**Superato**
UT-230	FRObi237	Verifica rimozione raccolta senza eliminazione singoli report.	Raccolta rimossa; report consultabili.	**Superato**
UT-231	FRObi238	Verifica presenza dialog di conferma esplicita eliminazione raccolta.	Richiesta conferma mostrata.	**Superato**
UT-232	FRObi239	Verifica ripristino stato su annullamento rimozione raccolta.	Raccolta mantenuta inalterata.	**Superato**
UT-233	FRObi163	Verifica ‘DocumentationReport Entity’ (Creation, Equality, Real JSON report).	Report istanziati correttamente e regole di uguaglianza per ID rispettate.	**Superato**
UT-234	FRObi163	Verifica ‘GitHubAnalysis Entity’ (Creation, StateMachine transitions, Equality).	Stati evolvono correttamente da PENDING a COMPLETED/FAILED.	**Superato**
UT-235	FRObi163	Verifica ‘AnalysisId’, ‘ReportId’, ‘CommitHash’, ‘UserId’ e ‘RepoURL’ (Success e Failure casi).	Impedita l’istanziazione per formati UUID o URL non validi; conformità formale garantita.	**Superato**
UT-236	FRObi93	Verifica costrutti di Coverage (CoveragePercentage, FileCoverage,	Controllo del range (0-100) per percentuali;	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
		CoverageFinding, CoverageEvaluation).	fail-fast per dati fuori limite.	
UT-237	FRObb97	Verifica costrutti Vulnerabilità (OWASPFinding, SecretFinding, DependencyFinding, SeverityFinding).	Controllo logica di equivalenza tra finding e validazione del mapping della Severity.	**Superato**
UT-238	FRObb102	Verifica Value Object Documentazione (DocsDiscrepancy, MissingFile, APIViolation, ReadmeDependency).	Regole sintattiche dei VO rispettate in costruzione e controllo formale del JSON.	**Superato**
UT-239	FRObb183	Verifica 'NewPatService' (instantiation, success path, failure path - port/thorws).	Invio credenziali a token salvati; gestione dell'exit status o failure del db.	**Superato**
UT-240	FRObb186	Verifica 'DeletePatService' (execute con success e port/thorws paths).	Risoluzione della cancellazione ed eventuali eccezioni gestite.	**Superato**
UT-241	FRObb184	Verifica 'UpdatePatService' e 'PATPasswordProvider' (Validation Errors, Constructor e Integrity).	Logica hashing applicata correttamente per update e integrità del token verificata.	**Superato**
UT-242	FRObb129	Verifica 'GitValidatorService' (Commit / Branch / Default Validation Strategy, Edge Cases).	Cambiamento dinamico strategia risolutiva in base agli argomenti immessi (Branch vs Hash).	**Superato**
UT-243	FRObb132	Verifica 'GitAuthorizerService' (Private Strategy, Public Strategy, Strategy Switching).	La chiamata scala al token privato solo su rifiuto dal path non autenticato.	**Superato**
UT-244	FRObb146	Verifica 'AnalysisOrchestratorService'	Flussi agenti microservizi chiamati assieme, ed errori	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
			recuperabili loggati bypassando fault.	
UT-245	FROb64	Verifica 'GetAnalysisService' (execute, getAllAnalysesForUser).	Restituzione DTO esatti degli storage e filtri corretti a livello DB.	**Superato**
UT-246	FROb57	Verifica servizi Collections ('GitHubCollectionChecker', 'GitHubCollectionDeleter', 'GitHubCollectionGetter').	Check dei duplicati (executeAll), prelievi e delezione controllati a livello Use Case.	**Superato**
UT-247	FROb164	Verifica 'ReportEntitiesProvider' (mappe DocsAgentResponse, CodeAgentResponse e SecurityAgentResponse).	La serializzazione dei payload remoti dai container viene idratata nel Dominio TS in sicurezza.	**Superato**
UT-248	FROb149	Verifica 'GitHubAdapter' (check Commit/Branch, URL building, Auth header) e 'GitHubAdapter.clone'.	Protocolli endpoint, fallback rami e costruzione command line di clonazione.	**Superato**
UT-249	FROb151	Verifica runAnalysis Streams di 'LocalCodeAnalysisAdapter', 'LocalSecurityAnalysisAdapter' e 'DocumentationAnalysisAdapter'.	Container lifecycle catturato nei buffer stdio per success_path, invalid_output e container_error handling.	**Superato**
UT-250	FROb167	Verifica 'MongoDBAdapter' (save, authorize, update/deletePAT, checkDuplicate, addCollection, save*Report).	Test integrità repository di persistenza Mongoose e delle collezioni custom via adapter di storage.	**Superato**
UT-251	FROb97	Verifica parser di vulnerabilita grype (parse_not_found, invalid_json, missing_locations, filters_critical, fix_all).	Estrazione vulnerabilità critical senza false positive da file strutturati difettosi.	**Superato**
UT-252	FROb97	Verifica CLI wrapper syft/ grype (success, timeout,	Sottoprocessi chiamati gestendo retry o	**Superato**

ID Test	Req.	Descrizione Test	Risultato Atteso	Stato
		db_outdated, generic/catalog/ unsupported_sbom failure).	interruzioni causate da sbom bloccanti.	
UT-253	FRObs98	Verifica tool SAST semgrep (parser_valid, skips_non_error, invalid_json, tool_success/ timeout).	OWASP categorizzati; i warning non di sicurezza o low tier scartati in mapping (no_owasp).	**Superato**
UT-254	FRObs97	Verifica finding di segreti in trivy (parser_valid, no_secrets, low_severity_filtered, tool_error).	Risultati limitati a leakage di credenziali/ segreti in source_code con scarto anomalie trivial.	**Superato**

Table 8: Tabella dei Test di Unità (UT) (Completa)

4.3 Test di Accettazione (TA)

ID Test	Descrizione	Stato
TA-001	Verificare che l'utente possa completare con successo la procedura di registrazione e la successiva autenticazione al sistema.	**Superato**
TA-002	Verificare che le chiavi di accesso non siano mai salvate o trasmesse in chiaro, garantendo l'integrità del sistema di hashing.	**Superato**
TA-003	Verificare che il sistema gestisca correttamente il reindirizzamento e il ritorno dalla piattaforma esterna GitHub, associando correttamente l'identificativo OAuth e cifrando il token ottenuto.	**NI**
TA-004	Verificare che l'utente possa configurare e avviare una richiesta di analisi fornendo un URL valido e selezionando le aree di interesse.	**Superato**
TA-005	Verificare che il sistema inibisca l'avvio di analisi ridondanti qualora il repository non abbia subito modifiche dall'ultimo report.	**NI**
TA-006	Verificare che il sistema impedisca l'avvio di analisi concorrenti sul medesimo repository, notificando correttamente lo stato di "Analisi in corso".	**NI**
TA-007	Verificare che l'accesso all'analisi di repository privati sia interdetto agli utenti che non hanno completato l'integrazione con GitHub, e che la richiesta di analisi (pubblica o privata) sia inibita in assenza di selezione di almeno un'area di interesse.	**NI**
TA-008	Verificare che il sistema protegga i dati di sessione e i token GitHub tramite cifratura e protocolli di comunicazione sicuri (HTTPS).	**Superato**
TA-009	Verificare che l'utente possa visualizzare correttamente lo storico globale dei repository analizzati, identificando chiaramente i progetti tramite i metadati esposti.	**Superato**
TA-010	Verificare che l'utente possa navigare nel dettaglio di un singolo report, filtrando le sezioni di interesse (Codice, Sicurezza, Documentazione) e visualizzando i relativi metadati di audit.	**Superato**

ID Test	Descrizione	Stato
TA-011	Verificare che il sistema presenti chiaramente le criticità rilevate e le relative remediation suggerite, distinguendo i casi di conformità (esito positivo).	**Superato**
TA-012	Verificare che l'utente possa impostare un intervallo temporale valido per generare un confronto storico tra le metriche di diversi report.	**NI**
TA-013	Verificare che il sistema generi visualizzazioni grafiche o tabelle comparative coerenti, evidenziando i trend di miglioramento o peggioramento delle metriche del codice.	**Superato**
TA-014	Verificare che l'analisi della qualità del codice esponga correttamente i dati di analisi statica (bug/smell) e le percentuali di copertura dei Test di Unità.	**Superato**
TA-015	Verificare che l'analisi della sicurezza esponga correttamente le vulnerabilità delle librerie (CVE) e i rilievi di conformità agli standard OWASP.	**Superato**
TA-016	Verificare che l'analisi della documentazione identifichi correttamente gli errori di sintassi e il grado di completezza rispetto al codice sorgente.	**Superato**
TA-017	Verificare che l'utente possa consultare il ranking dei repository ordinati per punteggio di qualità globale, ricevendo un'informativa corretta in assenza di dati.	**Superato**
TA-018	Verificare che l'utente possa disconnettere l'account GitHub dal profilo CodeGuardian, con conseguente revoca delle autorizzazioni e dei token.	**NI**
TA-019	Verificare che l'utente possa esportare i report di analisi in formati standard (PDF/JSON), garantendo la selezione obbligatoria del formato.	**Superato**
TA-020	Verificare che l'utente possa modificare la propria password di accesso previa validazione della credenziale attuale e rispetto dei criteri di sicurezza.	**Superato**
TA-021	Verificare che l'utente possa creare una raccolta di report associata a un repository GitHub, fornendo nome e URL validi, con eventuale descrizione facoltativa.	**Superato**
TA-022	Verificare che il Servizio di Analisi verifichi correttamente l'accessibilità del repository prima di avviare l'effettiva	**Superato**

ID Test	Descrizione	Stato
	analisi, distinguendo tra risorse pubbliche e private e gestendo i fallimenti di accesso.	
TA-023	Verificare che l'Orchestratore avvii correttamente il processo di analisi, clonando il repository e distribuendo la codebase agli strumenti di analisi per le aree selezionate.	**Superato**
TA-024	Verificare che il sistema aggreghi i risultati degli strumenti di analisi in un report strutturato, lo archivi correttamente nel sistema di persistenza e aggiorni lo stato dell'analisi a "completato".	**Superato**
TA-025	Verificare che l'utente riceva una notifica al completamento dell'analisi e che il report risulti consultabile nella propria area personale anche in assenza di ricezione della notifica.	**Superato**
TA-026	Verificare che l'utente venga notificato in caso di errore critico durante l'analisi e che lo stato di fallimento sia visibile nella dashboard indipendentemente dalla ricezione della notifica.	**NI**
TA-027	Verificare che l'Utente Avanzato possa accettare una remediation proposta, con conseguente applicazione delle modifiche al repository e aggiornamento dello stato nella dashboard.	**NI**
TA-028	Verificare che l'Utente Avanzato possa rifiutare una remediation proposta, con conseguente scarto della proposta e invarianza del repository.	**NI**
TA-029	Verificare che l'Utente Avanzato possa avviare con successo l'analisi di un repository GitHub privato presente nel proprio catalogo, selezionando le aree di interesse.	**Superato**
TA-030	Verificare che l'Utente Avanzato possa gestire il proprio catalogo di repository privati, inserendo, visualizzando e rimuovendo risorse, con corretta gestione dei duplicati.	**NI**
TA-031	Verificare che il proprietario di un repository privato possa gestire i permessi di accesso ai report, aggiungendo e revocando le autorizzazioni per altri utenti della piattaforma.	**NI**
TA-032	Verificare che l'utente possa rimuovere una raccolta di report dal proprio profilo senza che i singoli report in essa contenuti vengano eliminati.	**Superato**

ID Test	Descrizione	Stato
TA-033	Verificare che l'utente possa cancellare definitivamente il proprio profilo CodeGuardian, con conseguente rimozione dei dati personali e invalidazione delle credenziali precedenti.	**Superato**

Table 9: Tabella dei Test di Accettazione (TA)

4.4 Test di Integrazione (IT)

ID Test	Componenti Coinvolti	Descrizione	Stato
IT-001	Frontend `authApi` ⟶ `POST /account/auth/register` `RegistrationController`	Verificare che il frontend trasmetta correttamente le credenziali di registrazione all'endpoint del microservizio account e che la risposta contenga i token JWT e i dati utente attesi.	**Superato**
IT-002	Frontend `authApi` ⟶ `POST /account/auth/login` `LoginController`	Verificare che il frontend invii le credenziali di login, riceva i token di accesso e refresh, e li persista nel `tokenStorage` per le richieste successive.	**Superato**
IT-003	Frontend `gateway` (interceptor) ⟶ `POST /account/auth/refresh`	Verificare che l'interceptor Axios rilevi la scadenza del token di accesso, esegua automaticamente la richiesta di refresh verso il microservizio account, e reinstradi le richieste in coda con il nuovo token.	**Superato**
IT-004	Frontend `authApi` ⟶ `POST /account/auth/logout` `LogoutService`	Verificare che la chiamata di logout trasmetta correttamente il refresh token al microservizio account per l'invalidazione della sessione e che il `tokenStorage` venga svuotato lato frontend.	**Superato**
IT-005	Frontend `authApi` ⟶ `PATCH /account/auth/update` `UpdateController`	Verificare che la richiesta di modifica password includa il Bearer token nell'header Authorization, che il microservizio account lo validi correttamente, e che la nuova password venga persistita.	**Superato**
IT-006	Frontend `AuthContext` ⟶ `RegistrationController` ⟶ `RegistrationService` ⟶ PostgreSQL	Verificare l'intero flusso di registrazione end-to-end: dal form frontend alla scrittura del record utente su PostgreSQL, includendo l'hashing della	**Superato**

ID Test	Componenti Coinvolti	Descrizione	Stato
		password e la generazione dei token JWT.	
IT-007	Frontend `repositoriesApi.startAnalysis` → `POST /analysis/start` `AnalysisController`	Verificare che il frontend trasmetta correttamente i parametri di avvio analisi (URL repository, aree richieste, branch/commit opzionali) e che la risposta contenga l'identificativo univoco dell'analisi avviata.	**Superato**
IT-008	Frontend `analysisApi.getHistory` → `GET /analysis/all` `RepositoriesController`	Verificare che la richiesta autenticata di storico analisi restituisca la lista delle analisi associate all'utente corrente, con i metadati attesi (ID, stato, URL, timestamp).	**Superato**
IT-009	Frontend `analysisApi.getById` → `GET /analysis/one` `GetAnalysisService`	Verificare che il recupero di una singola analisi per ID restituisca correttamente i dati completi del report, inclusi i risultati dei sotto-agenti (codice, sicurezza, documentazione).	**Superato**
IT-010	Frontend `patApi.add` → `POST /analysis/pat` `NewPatService`	Verificare che il frontend trasmetta correttamente il Personal Access Token e l'URL del repository privato, e che il microservizio analisi persista le credenziali cifrate su MongoDB.	**Superato**
IT-011	Frontend `patApi.update` → `PUT /analysis/pat` `UpdatePatService`	Verificare che la richiesta di aggiornamento PAT sostituisca correttamente le credenziali esistenti per il repository specificato, garantendo la continuità dell'accesso ai repository privati.	**Superato**

ID Test	Componenti Coinvolti	Descrizione	Stato
IT-012	Frontend `patApi.delete` → `DELETE /analysis/pat` `DeletePatService`	Verificare che la richiesta di eliminazione PAT rimuova correttamente le credenziali associate all'URL del repository, inibendo i futuri accessi privati non autorizzati.	**Superato**
IT-013	Frontend `repositoriesApi.create` → `POST /analysis/repositories` `AddRepositoryCollectionUseCase`	Verificare che la creazione di una raccolta repository trasmetta nome, URL e descrizione al microservizio analisi e che la raccolta venga correttamente persistita e associata all'utente.	**Superato**
IT-014	Frontend `repositoriesApi.list` → `GET /analysis/repositories` `GetRepositoryCollectionUseCase`	Verificare che la lista delle raccolte repository venga recuperata dal microservizio analisi e restituita al frontend con i metadati corretti (nome, URL, data creazione).	**Superato**
IT-015	Frontend `repositoriesApi.delete` → `DELETE /analysis/repositories/:id` `DeleteRepositoryCollectionUseCase`	Verificare che l'eliminazione di una raccolta repository rimuova la raccolta senza cancellare i singoli report ad essa associati, mantenendo la loro consultabilità.	**Superato**
IT-016	Frontend `gateway` (normalizer) → Risposta `AnalysisController`	Verificare che il normalizzatore di risposta del gateway frontend converta correttamente i valori di stato del backend (`PENDING`, `IN_PROGRESS`, `COMPLETED`, `FAILED`) nei valori attesi dal frontend (`pending`, `in-progress`, `completed`, `failed`).	**Superato**
IT-017	Frontend `useAnalysisSocket` → Socket.io Gateway `SocketContext`	Verificare che il frontend si connetta correttamente al canale WebSocket del microservizio analisi, riceva le notifiche di aggiornamento dello stato in tempo reale e	**NI**

ID Test	Componenti Coinvolti	Descrizione	Stato
		aggiorni l'interfaccia utente in modo reattivo.	
IT-018	`GitAuthorizerService` (strategia pubblica) ⟶ GitHub API `GET /repos/{owner}/{repo}/branches/{branch}`	Verificare che il servizio di autorizzazione esegua correttamente la verifica di accessibilità pubblica tramite l'API GitHub, senza header di autorizzazione, e che la risposta positiva permetta di procedere con la clonazione.	**Superato**
IT-019	`GitAuthorizerService` (strategia privata) ⟶ MongoDB `GitCredential` ⟶ GitHub API	Verificare che per i repository privati il servizio recuperi le credenziali cifrate da MongoDB, decifri il PAT, e lo utilizzi nell'intestazione Authorization della chiamata GitHub, consentendo l'accesso alla risorsa protetta.	**Superato**
IT-020	`GitValidatorService` ⟶ GitHub API (branch/commit strategy)	Verificare che il servizio di validazione selezioni dinamicamente la strategia corretta (branch vs. commit hash) in base ai parametri ricevuti, e che la risposta dell'API GitHub venga correttamente interpretata per ricavare il commit SHA da usare come riferimento dell'analisi.	**Superato**
IT-021	`GitHubAdapter.clone` ⟶ `git clone` (subprocess) ⟶ Volume Docker condiviso	Verificare che l'adattatore GitHub costruisca correttamente il comando `git clone` con le credenziali e i parametri di branch/commit, che il processo figlio cloni il repository nel volume Docker condiviso (`analysis_tmp_data`), e che il percorso risultante sia accessibile agli agenti di analisi.	**Superato**

ID Test	Componenti Coinvolti	Descrizione	Stato
IT-022	`GitHubAdapter` (exponential backoff) ⟶ GitHub API (errori di rete)	Verificare che in presenza di errori di rete transitori l'adattatore GitHub rispetti il meccanismo di retry con backoff esponenziale, e che dopo il numero massimo di tentativi l'analisi venga correttamente marcata come fallita.	**Superato**
IT-023	`AnalysisOrchestratorService` ⟶ `LocalCodeAnalysisAdapter` ⟶ Container `strands-code-analyzer`	Verificare che l'orchestratore avvii il container Docker dell'agente codice con i parametri corretti (percorso volume, variabili d'ambiente), catturi l'output JSON dallo stdout e lo deserializzi nel modello di dominio `CodeAgentReport`.	**Superato**
IT-024	`AnalysisOrchestratorService` ⟶ `LocalSecurityAnalysisAdapter` ⟶ Container `strands-security-analyzer`	Verificare che l'orchestratore avvii il container Docker dell'agente sicurezza, catturi correttamente l'output strutturato contenente le vulnerability findings (Grype, Trivy, Semgrep) e le mappi al modello di dominio `SecurityReport`.	**Superato**
IT-025	`AnalysisOrchestratorService` ⟶ `DocumentationAnalysisAdapter` ⟶ Container agente documentazione	Verificare che l'orchestratore avvii il container Docker dell'agente documentazione, catturi il JSON di output con le discrepanze e le violazioni rilevate, e lo deserializzi nella entità `DocumentationReport`.	**Superato**
IT-026	`AnalysisOrchestratorService` ⟶ Tutti e tre gli agenti (parallelizzazione)	Verificare che i tre container degli agenti (codice, sicurezza, documentazione) vengano avviati in parallelo dall'orchestratore, e che un fallimento parziale di uno degli agenti non blocchi il	**Superato**

ID Test	Componenti Coinvolti	Descrizione	Stato
		completamento del report con i dati degli altri agenti disponibili.	
IT-027	`LocalCodeAnalysisAdapter` → Volume Docker `analysis_tmp_data` (lifecycle)	Verificare che il volume Docker condiviso tra il microservizio di analisi e i container degli agenti venga montato correttamente, che i file del repository clonato siano accessibili agli agenti in lettura, e che i risultati vengano resi disponibili al microservizio al termine dell'esecuzione.	**Superato**
IT-028	`MongoDBAdapter.saveGitHubAnalysis` → Schema `GitHubAnalysis` → MongoDB	Verificare che il salvataggio di un'analisi in stato `PENDING` su MongoDB avvenga correttamente, con la corretta associazione tra ID analisi, ID utente e URL repository, e che lo schema Mongoose rispecchi il modello di dominio.	**Superato**
IT-029	`MongoDBAdapter.saveCodeReport` → Schema `CodeReport` → MongoDB	Verificare che il report dell'agente codice, una volta aggregato dall'orchestratore, venga persistito correttamente su MongoDB con tutti i campi previsti (static analysis findings, coverage data, AI interpretations).	**Superato**
IT-030	`MongoDBAdapter.saveDocsReport` → Schema `DocsReport` → MongoDB	Verificare che il report dell'agente documentazione venga persistito correttamente su MongoDB con i campi relativi a discrepanze, file mancanti, violazioni API e audit delle dipendenze.	**Superato**
IT-031	`MongoDBAdapter.saveSecurityReport` → Schema `SecurityReport` → MongoDB	Verificare che il report dell'agente sicurezza venga persistito correttamente su MongoDB con le vulnerability	**Superato**

ID Test	Componenti Coinvolti	Descrizione	Stato
		findings normalizzate (OWASP, CVE, segreti esposti).	
IT-032	`MongoDBAdapter.getAnalysisFromId` → Schema `GitHubAnalysis` → `GetAnalysisService`	Verificare che il recupero di un'analisi per ID dall'adattatore MongoDB restituisca l'entità completa con i report annessi, e che il mapping verso i DTO di risposta avvenga senza perdita di informazioni.	**Superato**
IT-033	`MongoDBAdapter.getAllAnalysesForUser` → Schema `GitHubAnalysis` → Frontend	Verificare che la query di recupero di tutte le analisi per utente filtri correttamente per `userId`, restituisca i risultati in ordine cronologico decrescente e che il mapping verso i DTO sia coerente con quanto atteso dal frontend.	**Superato**
IT-034	`MongoDBAdapter` (PAT CRUD) → Schema `GitCredential` → `GitAuthorizerService`	Verificare il ciclo completo di gestione delle credenziali PAT: salvataggio cifrato, lettura e decifratura per autorizzazione, aggiornamento e cancellazione, garantendo che in nessun momento il token sia accessibile in chiaro al di fuori del servizio di cifratura.	**Superato**
IT-035	`MongoDBAdapter` (collections) → Schema `RepositoryCollection` → Frontend	Verificare che le operazioni CRUD sulle raccolte repository (aggiunta con controllo duplicati, recupero lista, cancellazione) siano correttamente mediate dall'adattatore MongoDB e che i risultati siano coerenti con lo stato atteso dal frontend.	**Superato**
IT-036	`RegistrationService` → `IUserSavePort` → `PostgresAdapter` → PostgreSQL	Verificare che la registrazione di un nuovo utente risulti nella corretta scrittura del record su PostgreSQL, con password hashata tramite bcrypt e	**Superato**

ID Test	Componenti Coinvolti	Descrizione	Stato
		vincolo di unicità su email rispettato.	
IT-037	`LoginService` ⟶ `IUserFindPort` ⟶ `PostgresAdapter` ⟶ PostgreSQL ⟶ `BcryptAdapter`	Verificare che il flusso di autenticazione recuperi correttamente il record utente da PostgreSQL tramite email, confronti l'hash della password fornita con quello memorizzato, e generi i token JWT in caso di corrispondenza.	**Superato**
IT-038	`UpdateService` ⟶ `IUserUpdatePort` ⟶ `PostgresAdapter` ⟶ PostgreSQL	Verificare che l'aggiornamento della password esegua correttamente la query UPDATE su PostgreSQL in modo transazionale, e che la vecchia password non sia più valida per autenticazioni successive.	**Superato**
IT-039	`DeleteService` ⟶ `IUserDeletePort` ⟶ `PostgresAdapter` ⟶ PostgreSQL	Verificare che la cancellazione dell'account esegua la query DELETE su PostgreSQL rimuovendo il profilo utente e tutti i dati associati, e che l'invalidazione delle sessioni attive avvenga contestualmente.	**Superato**
IT-040	`JwtAdapter` ⟶ `ITokenProviderPort` ⟶ `RegistrationService`/ `LoginService`	Verificare che il JWT adapter generi correttamente i token di accesso e refresh con il payload atteso (sub, email, scadenza), e che il meccanismo di verifica li validi correttamente in fase di autenticazione delle richieste successive.	**Superato**
IT-041	`LogoutService` ⟶ `ISessionDeletePort` ⟶ Redis/ SessionStore ⟶ PostgreSQL	Verificare che il logout invalidi correttamente il refresh token nello store di sessione, impedendo il suo riutilizzo per ottenere nuovi token di accesso anche se non ancora scaduto.	**Superato**

ID Test	Componenti Coinvolti	Descrizione	Stato
IT-042	Frontend ⟶ Account MS ⟶ Analisi MS ⟶ GitHub API ⟶ Agenti Docker ⟶ MongoDB ⟶ Frontend	Verificare il flusso completo di analisi di un repository pubblico: dalla richiesta autenticata del frontend, all'autorizzazione su GitHub, alla clonazione, all'orchestrazione degli agenti, all'aggregazione del report su MongoDB, fino alla notifica di completamento ricevuta dal frontend.	**Superato**
IT-043	Frontend ⟶ Analisi MS (PAT) ⟶ MongoDB ⟶ GitHub API (privato) ⟶ Agenti Docker ⟶ MongoDB ⟶ Frontend	Verificare il flusso completo di analisi di un repository privato: dalla registrazione del PAT, al suo recupero cifrato per la clonazione autenticata, all'esecuzione degli agenti, fino alla disponibilità del report nel frontend.	**Superato**
IT-044	Frontend ⟶ `authApi.register` ⟶ Account MS ⟶ JWT ⟶ Analisi MS (richiesta autenticata)	Verificare che un utente appena registrato sul microservizio account possa immediatamente utilizzare i token JWT ricevuti per effettuare richieste autenticate al microservizio analisi, validando la consistenza del sistema di autorizzazione cross-microservizio.	**Superato**
IT-045	`AnalysisOrchestratorService` ⟶ `ReportEntitiesProvider` ⟶ `MongoDBAdapter` ⟶ `GetAnalysisService` ⟶ Frontend	Verificare che i dati prodotti dagli agenti Docker, dopo la trasformazione in entità di dominio tramite `ReportEntitiesProvider` e la persistenza su MongoDB, siano correttamente recuperati e serializzati dal `GetAnalysisService` nella forma attesa dal frontend per la visualizzazione del report.	**Superato**

Table 10: Tabella dei Test di Integrazione (IT)

4.5 Riepilogo Quantitativo Test (PB)

A completamento della specifica dei test, la seguente tabella riassume i risultati quantitativi conseguiti durante la campagna di test per la Product Baseline. Tali valori riflettono lo stato di avanzamento della verifica rispetto ai requisiti implementati e ai test pianificati.

Parametro	Valore	Descrizione
Test Pianificati (Specifica)	441	Totale casi di test formalizzati.
Test Eseguiti	368	Test effettuati sulle funzionalità stabili.
Test Superati	368	Esiti positivi (Passed).
Success Rate	100%	Rapporto Superati / Eseguiti.
Copertura Requisiti	83,4%	Rapporto Eseguiti / Pianificati.

Table 11: Sintesi quantitativa della Campagna di Test

5 Cruscotto di Valutazione

Il presente cruscotto costituisce il sistema di monitoraggio attraverso il quale Skarab Group valuta oggettivamente l'andamento del progetto. Le metriche qui raccolte rappresentano l'evidenza empirica necessaria per attivare il ciclo *PDCA* (Plan-Do-Check-Act), trasformando i dati grezzi in informazioni per il miglioramento continuo.

In questa sezione vengono presentati i risultati delle misurazioni effettuate nel periodo di riferimento. L'analisi dei dati non è fine a se stessa, ma è orientata a fornire una visione oggettiva ("Data-Driven") dello stato di salute del progetto e della qualità del software rilasciato. I dati sono organizzati per area di processo e per qualità di prodotto, permettendo una rapida identificazione delle aree critiche e il confronto con le soglie di accettabilità definite nel Piano di Qualifica.

5.1 Processi Primari: Fornitura (EVM)

Questa sezione monitora l'andamento economico e temporale del progetto utilizzando lo standard **Earned Value Management**. L'obiettivo è evidenziare scostamenti tra quanto pianificato (Baseline) e quanto effettivamente realizzato.

5.1.1 Trend di Progetto (PV, AC, EV)

Metriche: MPC02, MPC03, MPC04

Viene visualizzato l'andamento cumulativo del valore pianificato (**Planned Value**), del costo reale sostenuto (**Actual Cost**) e del valore guadagnato (**Earned Value**). La sovrapposizione delle curve indica un progetto in linea con le aspettative; divergenze significative segnalano la necessità di interventi correttivi su budget o scadenze.

5.1.2 Indici di Efficienza (CPI, SPI)

Metriche: MPC07, MPC08

Vengono riportati gli indici di performance puntuali per ogni Sprint. Questi valori normalizzati permettono di capire immediatamente l'efficienza di costo (**CPI**) e di schedulazione (**SPI**), dove un valore pari o superiore a 1.00 rappresenta lo stato ottimale.

5.1.3 Varianze e Previsioni (CV, SV, EAC)

Metriche: MPC05, MPC06, MPC09

Questa sezione quantifica in termini monetari l'eventuale risparmio o perdita (**Cost Variance**) e l'anticipo o ritardo (**Schedule Variance**). Viene inoltre proiettata la stima del costo finale a finire (**Estimate At Completion**) confrontandola con il budget iniziale.

5.2 Processi Primari: Sviluppo

Questa sezione analizza la stabilità dell'ambito tecnico del progetto.

5.2.1 Requirements Stability Index (RSI)

Metrica: MPC10

Grafico che traccia la volatilità dei requisiti nel tempo. Un indice stabile e alto garantisce che il team stia lavorando su obiettivi consolidati, mentre fluttuazioni frequenti possono indicare incertezze nell'analisi o richieste di modifica eccessive (**Scope Creep**).

5.3 Processi di Supporto: Documentazione

Monitoraggio della qualità formale e della fruibilità della documentazione prodotta.

5.3.1 Indice di Gulpease e Correttezza

Metriche: MPC11, MPC12

Viene riportato il livello di leggibilità linguistica (**Gulpease Index**) calcolato sui documenti

principali, unitamente al numero di errori ortografici rilevati, per garantire che la documentazione sia accessibile e professionale.

5.4 Processi di Supporto: Verifica

Monitoraggio dell'efficacia delle attività di testing dinamico.

5.4.1 Code Coverage e Test Success

Metriche: MPC13, MPC14

Cruscotto tecnico che visualizza la copertura del codice raggiunta dai test automatizzati e il tasso di successo dei test eseguiti.

5.5 Processi di Supporto: Gestione della Qualità

Visione d'insieme sull'efficacia del Piano di Qualifica stesso.

5.5.1 Soddisfazione delle Metriche

Metrica: MPC15

Indicatore sintetico (KPI) che mostra la percentuale totale delle metriche di progetto che rispettano le soglie di accettabilità definite. Fornisce un'indicazione immediata sulla conformità complessiva dei processi.

5.6 Processi Organizzativi: Gestione dei Processi

Analisi dell'efficienza del metodo di lavoro Agile adottato dal team.

5.6.1 Sprint Goal Achievement

Metrica: MPC16

Viene illustrata la capacità del team di completare gli obiettivi pianificati all'inizio di ogni iterazione (Sprint Planning). Questo dato è essenziale per calibrare la **Velocity** del team e migliorare la precisione delle pianificazioni future.

5.7 Qualità di Prodotto

In questa sezione si verifica la conformità del software rilasciato rispetto ai requisiti e agli standard di qualità ISO/IEC 25010.

5.7.1 Copertura Funzionale

Metriche: MPD01, MPD02, MPD03

Visualizzazione dello stato di implementazione dei requisiti, suddivisi per priorità (Obbligatori, Desiderabili, Opzionali), per confermare l'adeguatezza funzionale del rilascio corrente.

5.7.2 Affidabilità e Manutenibilità

Metriche: MPD04, MPD05, MPD08, MPD09, MPD10

Analisi tecnica che combina indicatori di affidabilità (densità guasti, disponibilità) e metriche statiche del codice (complessità ciclomatica, densità commenti) per valutare la salute tecnica del prodotto.

5.7.3 Usabilità e Sicurezza

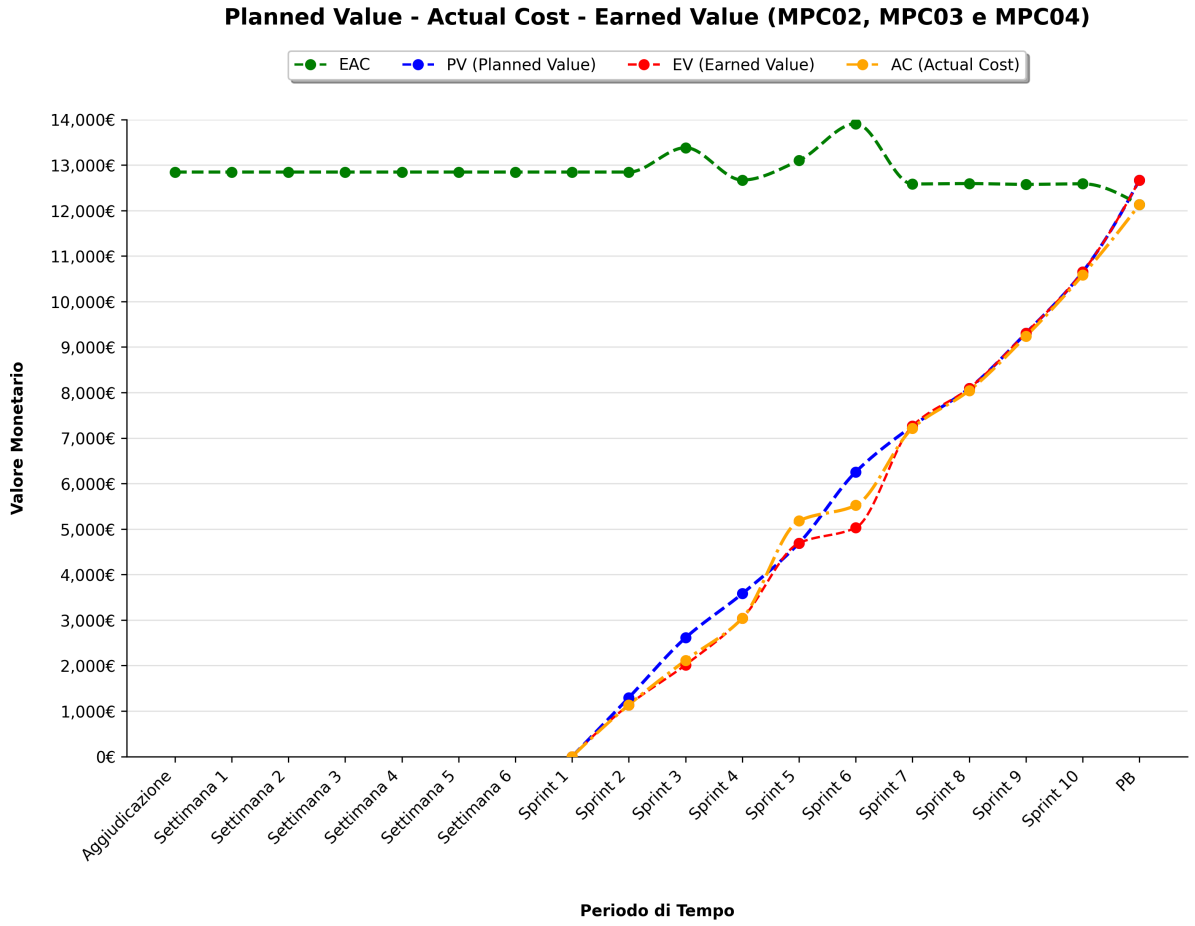
Metriche: MPD06, MPD07, MPD11

Report sugli esiti delle verifiche di usabilità (comprensibilità, prevenzione errori) e scansione delle vulnerabilità di sicurezza, garantendo che il prodotto sia sicuro e utilizzabile dall'utente finale.

È importante evidenziare che il periodo iniziale, dall'aggiudicazione fino all'avvio formale delle attività di progetto (*Sprint 1*), ha rappresentato una fase di "palestra" durante la quale il gruppo si è dedicato allo studio approfondito delle tecnologie necessarie, partecipando anche a sessioni di formazione organizzate dall'azienda proponente Var Group.

5.8 Processi Primari: Fornitura e Sviluppo

5.8.1 Planned Value - Actual Cost - Earned Value (MPC02, MPC03 e MPC04)



5.8.1.1 Requirements and Technology Baseline (RTB)

Dopo la fase iniziale, in cui le attività di formazione e setup sono state gestite come investimento interno senza gravare sul budget, il progetto è entrato nella fase operativa con l'avvio dello *Sprint 1*. In questa prima iterazione Skarab Group ha mostrato un buon equilibrio economico, completando il lavoro con un dispendio di risorse coerente con il valore prodotto, pur registrando un lieve ritardo rispetto alla pianificazione ideale.

Tuttavia, la situazione ha subito una variazione significativa durante lo *Sprint 2*: a fronte di un incremento del *Planned Value* (PV) e dell'*Actual Cost* (AC), l'*Earned Value* (EV) ha subito una flessione. Questo testimonia l'insorgere di inefficienze produttive e debito tecnico, legati alla necessità di ricalibrare task qualitativamente insufficienti che hanno rallentato la produzione.

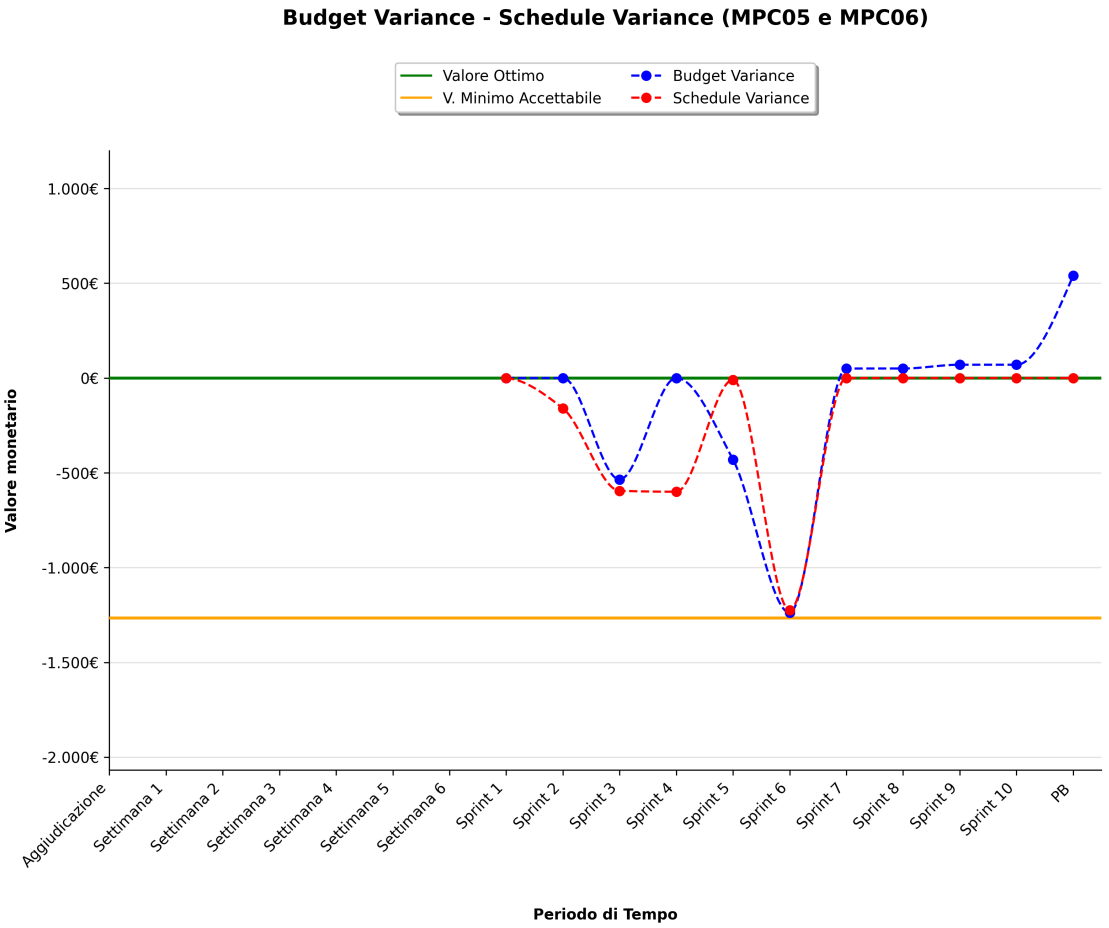
Nello *Sprint 3*, nonostante una parziale ripresa, l'*Earned Value* rimane ancora al di sotto del *Planned Value*, segnalando che il ritardo accumulato non è ancora stato recuperato. Nello *Sprint 4* EV e PV iniziano ad allinearsi, a fronte però di un aumento dell'AC.

5.8.1.2 Product Baseline (PB)

All'inizio della Product Baseline si è registrata una temporanea divergenza tra le curve: nello *Sprint 5*, l'*Actual Cost* è risultato superiore all'*Earned Value*, con quest'ultimo che è rimasto al di sotto del valore pianificato. Tale scostamento è indicativo di un rallentamento produttivo. Tuttavia, nel corso dello *Sprint 6*, il team è riuscito a riallineare le metriche, mantenendo un equilibrio ottimale tra

lavoro prodotto, costi e pianificazione fino alla conclusione del progetto, con un Actual Cost finale inferiore al budget preventivato.

5.8.2 Budget Variance - Schedule Variance (MPC05 e MPC06)



5.8.2.1 Requirements and Technology Baseline (RTB)

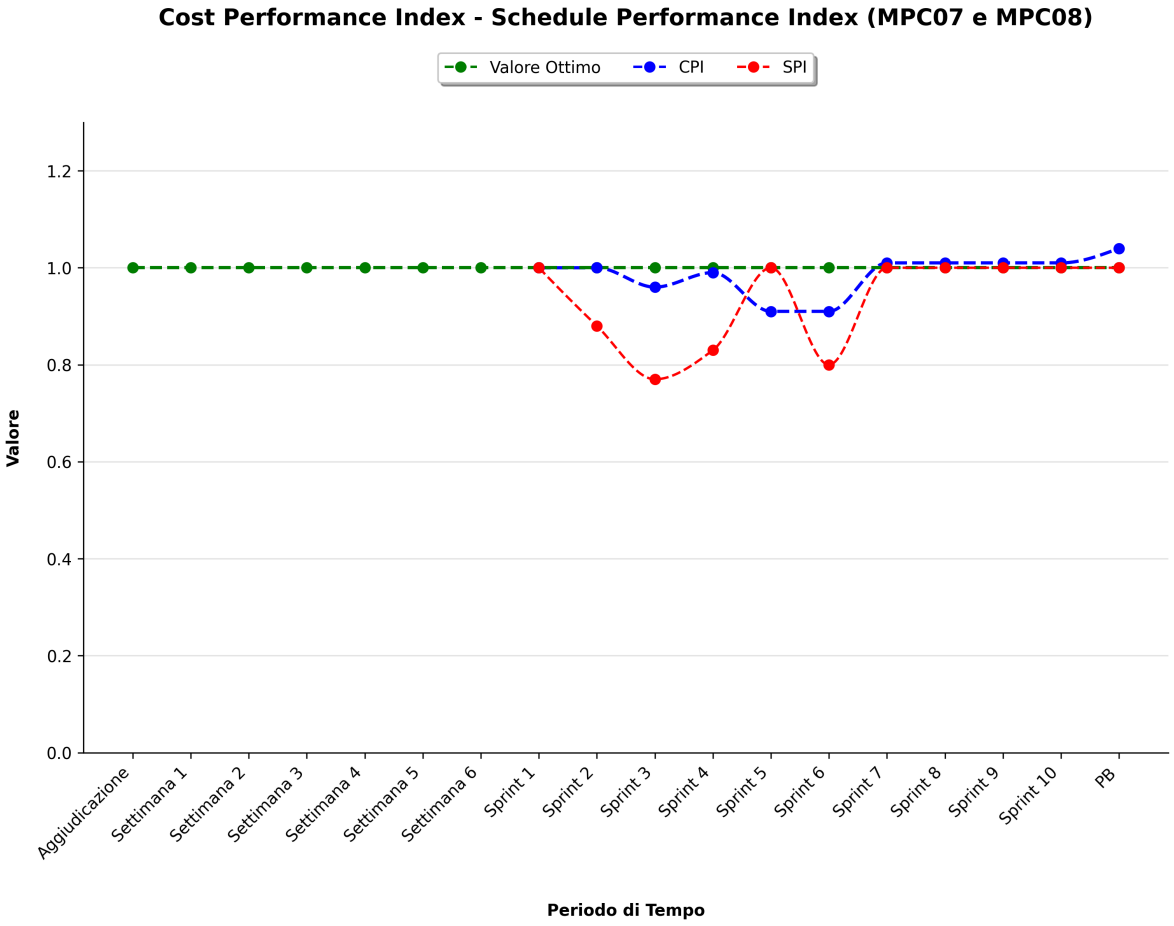
Il grafico monitora la salute economica e temporale del progetto a partire dallo *Sprint 1* durante il quale la *Schedule Variance* (SV) mostra una leggera flessione. Quest'ultima si è accentuata nello *Sprint 2*, riflettendosi anche sulla *Budget Variance* (BV).

Successivamente, il ritardo accumulato negli sprint precedenti ha continuato a pesare sulla metrica. Nel corso dello *Sprint 4*, però, la *Schedule Variance* torna quasi a zero.

5.8.2.2 Product Baseline (PB)

Questa baseline è stata caratterizzata da una marcata criticità nello *Sprint 5*, dove sia la *Budget Variance* che la *Schedule Variance* hanno subito una brusca flessione negativa. Questo scostamento è stato prontamente analizzato e corretto dal team: a partire dallo *Sprint 7*, entrambe le metriche sono tornate stabilmente sopra lo zero, confermando una gestione economica sana (con una Budget Variance finale di 540€) e una puntualità ottimale.

5.8.3 Cost Performance Index - Schedule Performance Index (MPC07 e MPC08)



5.8.3.1 Requirements and Technology Baseline (RTB)

Dal grafico è possibile notare come, inizialmente, lo *Schedule Performance Index* (SPI) sia inferiore a 1, indicando un leggero ritardo fisiologico. La buona gestione dei costi è invece documentata dal *Cost Performance Index* (CPI).

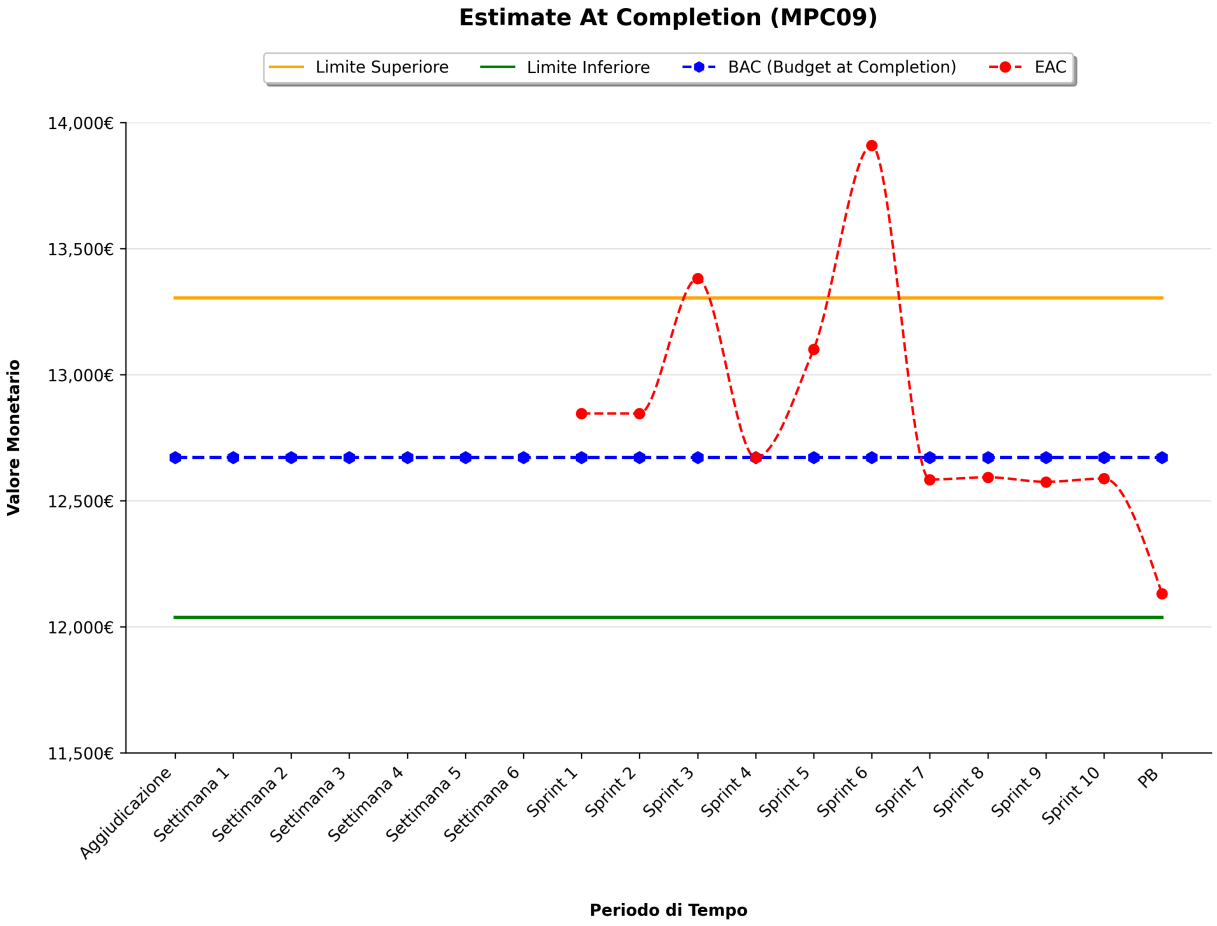
La situazione è peggiorata nel corso dello *Sprint 2*, durante il quale si è verificato un crollo dello *Schedule Performance Index* (SPI) che segnala un ritardo critico rispetto alla pianificazione.

A partire dallo *Sprint 3* lo SPI inizia a recuperare progressivamente. Nello Sprint 4 lo SPI raggiunge il valore ottimale, mentre il CPI scende a causa dello sforamento orario.

5.8.3.2 Product Baseline (PB)

Il *CPI* ha oscillato inizialmente tra 0.91 e 1.01, stabilizzandosi sul valore ottimale a partire dallo *Sprint 7* e raggiungendo un indice finale di 1.04. Lo *SPI* ha mostrato un calo significativo nello *Sprint 5*, per poi risalire e mantenersi costantemente a 1.00 per il resto della baseline. Questa progressione testimonia la capacità del team di assorbire i ritardi e ottimizzare l'efficienza di costo dopo le difficoltà incontrate.

5.8.4 Estimate at Completion (MPC09)



5.8.4.1 Requirements and Technology Baseline (RTB)

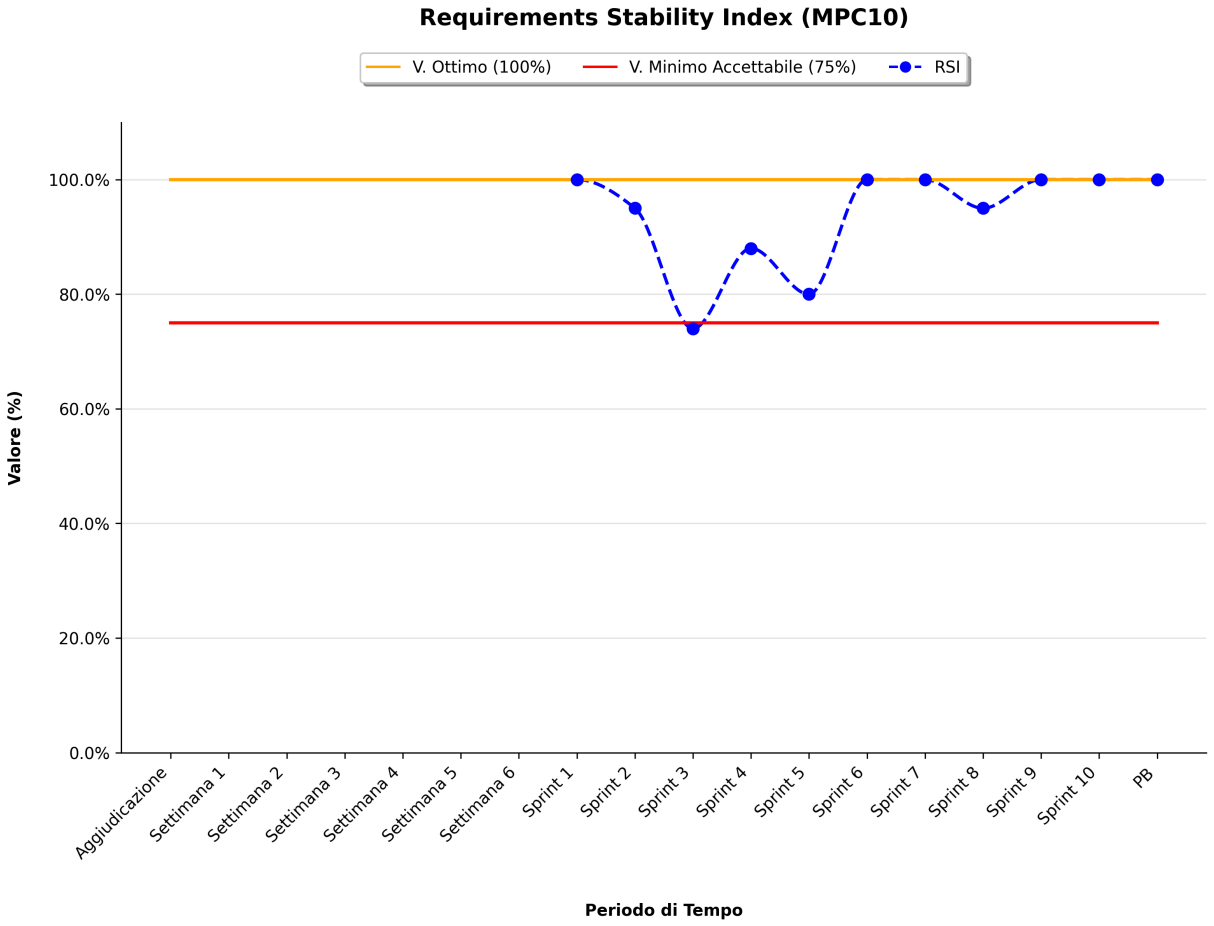
Dopo una fase iniziale di stabilità coincidente con il budget originale, è possibile notare come una gestione inefficiente delle risorse abbia spinto la previsione di spesa verso il limite massimo.

Nello *Sprint 3* la situazione è migliorata, con l'EAC che è rientrato entro i limiti accettabili. Tuttavia, nello *Sprint 4*, lo sforamento orario dovuto alle revisioni dell'Analisi dei Requisiti ha spinto nuovamente l'EAC a 13.100€, superando il BAC ma rimanendo al di sotto del limite superiore. Il team si impegna ad adottare azioni correttive nella fase successiva per ricondurre la previsione di spesa entro i parametri ottimali.

5.8.4.2 Product Baseline (PB)

L'*EAC* ha subito una fluttuazione marcata nello *Sprint 5*, raggiungendo un picco di spesa prevista di 13.908€. Tale incremento è stato causato dal dispendio di ore superiore al previsto per la progettazione. Grazie alle azioni correttive intraprese, a partire dallo *Sprint 7* le stime a finire sono rientrate stabilmente sotto il budget preventivato, attestandosi definitivamente a 12.130€.

5.8.5 Requirements Stability Index (MPC10)



5.8.5.1 Requirements and Technology Baseline (RTB)

Il *Requirements Stability Index* (RSI) registra un peggioramento nel corso dello *Sprint 2*. Tale flessione è riconducibile a una sottostima iniziale dei requisiti impliciti e all'emersione di ulteriori requisiti in seguito al colloquio con il Prof. Cardin: il team ha dovuto apportare modifiche significative per aggiungere i requisiti non tracciati in precedenza dagli Analisti.

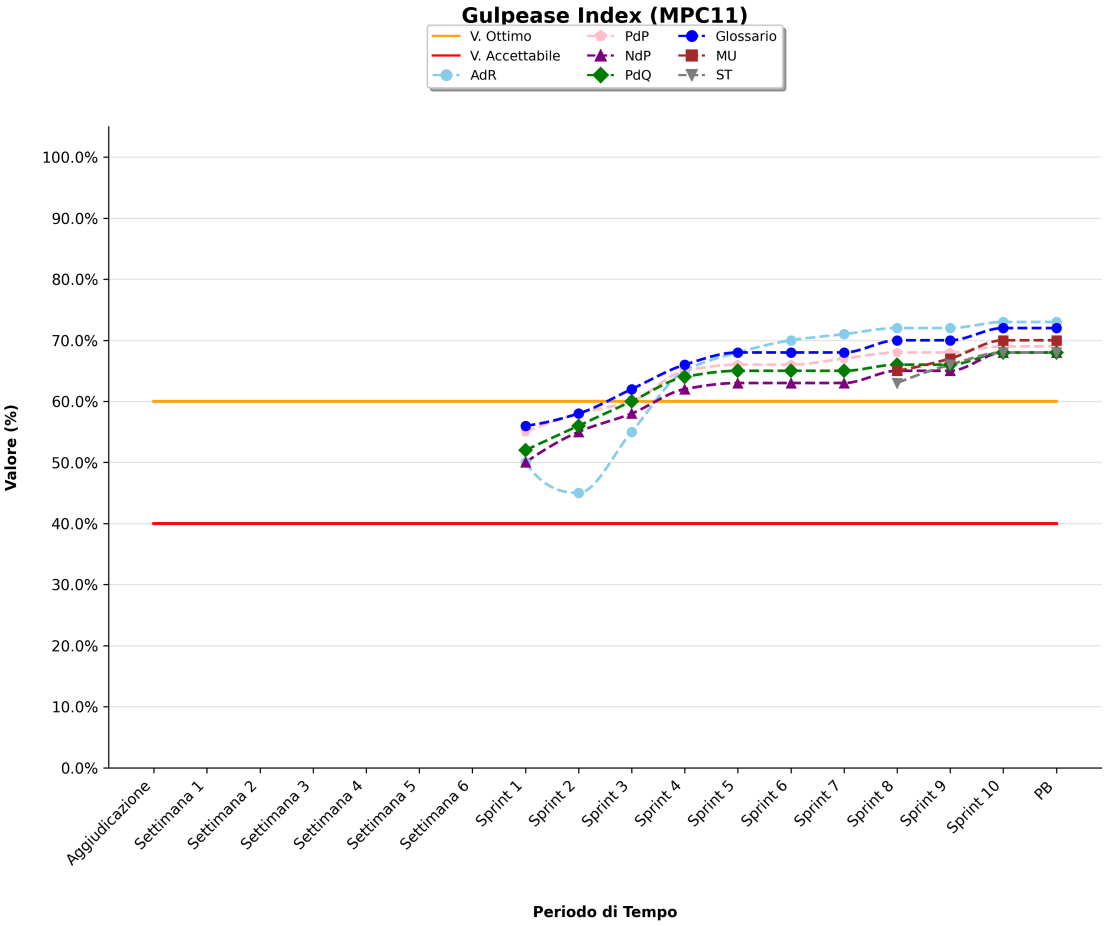
Nello *Sprint 4*, la riscrittura di casi d'uso malposti e l'inserimento di requisiti mancanti hanno causato un ulteriore calo dell'indice, mantenendosi comunque al di sopra della soglia minima accettabile.

5.8.5.2 Product Baseline (PB)

Il *Requirements Stability Index* ha raggiunto il valore ottimale del 100% per quasi tutta la durata del periodo, con un'unica flessione fisiologica nello *Sprint 8* (95%). Il team ha garantito una gestione rigorosa dell'ambito tecnico, consolidando i requisiti in vista del rilascio finale e limitando al minimo le variazioni tardive.

5.9 Processi di Supporto

5.9.1 Gulpease Index (MPC11)



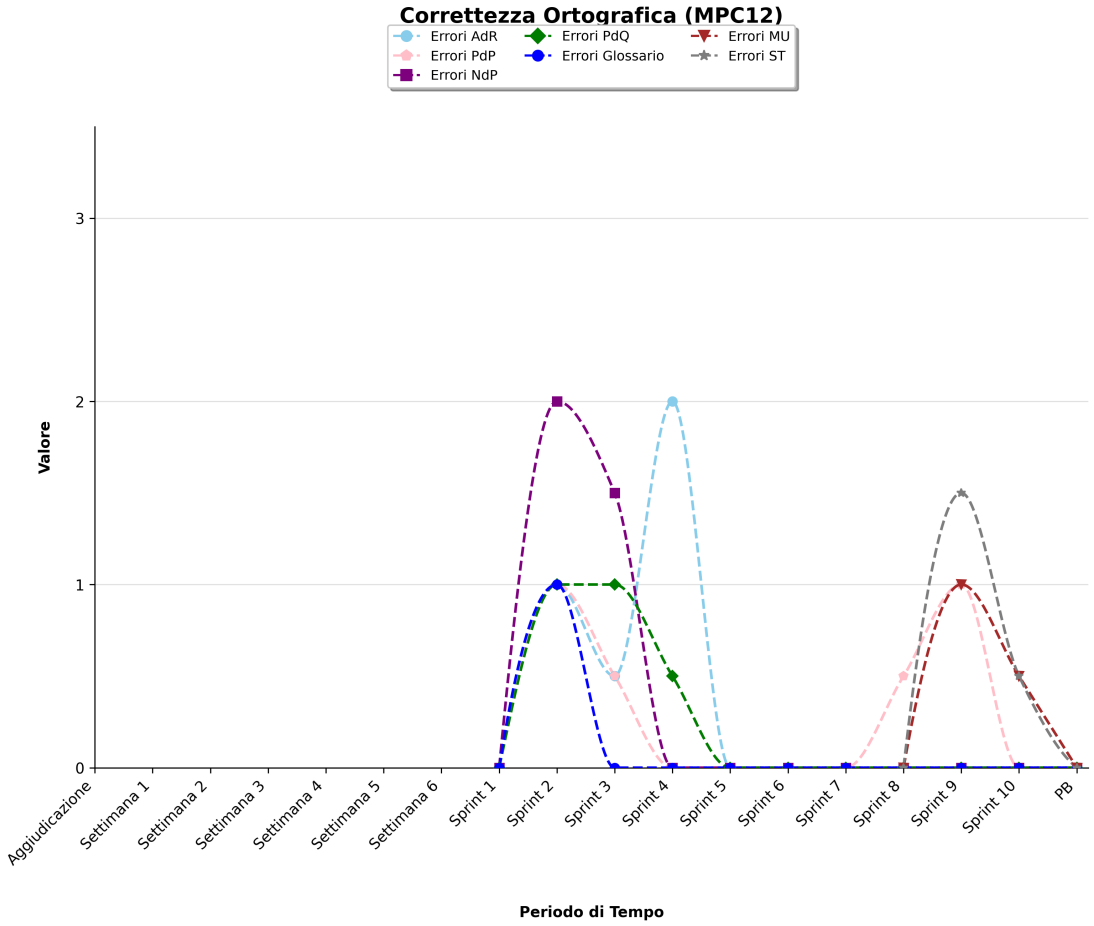
5.9.1.1 Requirements and Technology Baseline (RTB)

Il grafico mostra un andamento complessivamente positivo dei documenti monitorati: a partire dallo *Sprint 1*, i valori si attestano al di sopra della soglia minima accettabile. L'*Analisi dei Requisiti* presenta una lieve flessione nello *Sprint 2*, riconducibile alla necessità di introdurre terminologia tecnica più densa e alla fase di revisione intensiva dei requisiti. In generale, Skarab Group si impegna a mantenere nel tempo una buona leggibilità dei documenti.

5.9.1.2 Product Baseline (PB)

Durante la PB, l'*Indice di Gulpease* per tutti i documenti monitorati si è attestato stabilmente sopra la soglia ottimale di 60. L'introduzione del *Manuale Utente* e della *Specifica Tecnica* a partire dallo *Sprint 8* ha mostrato indici inizialmente vicini alla soglia accettabile, che sono poi progressivamente cresciuti.

5.9.2 Correttezza Ortografica (MPC12)



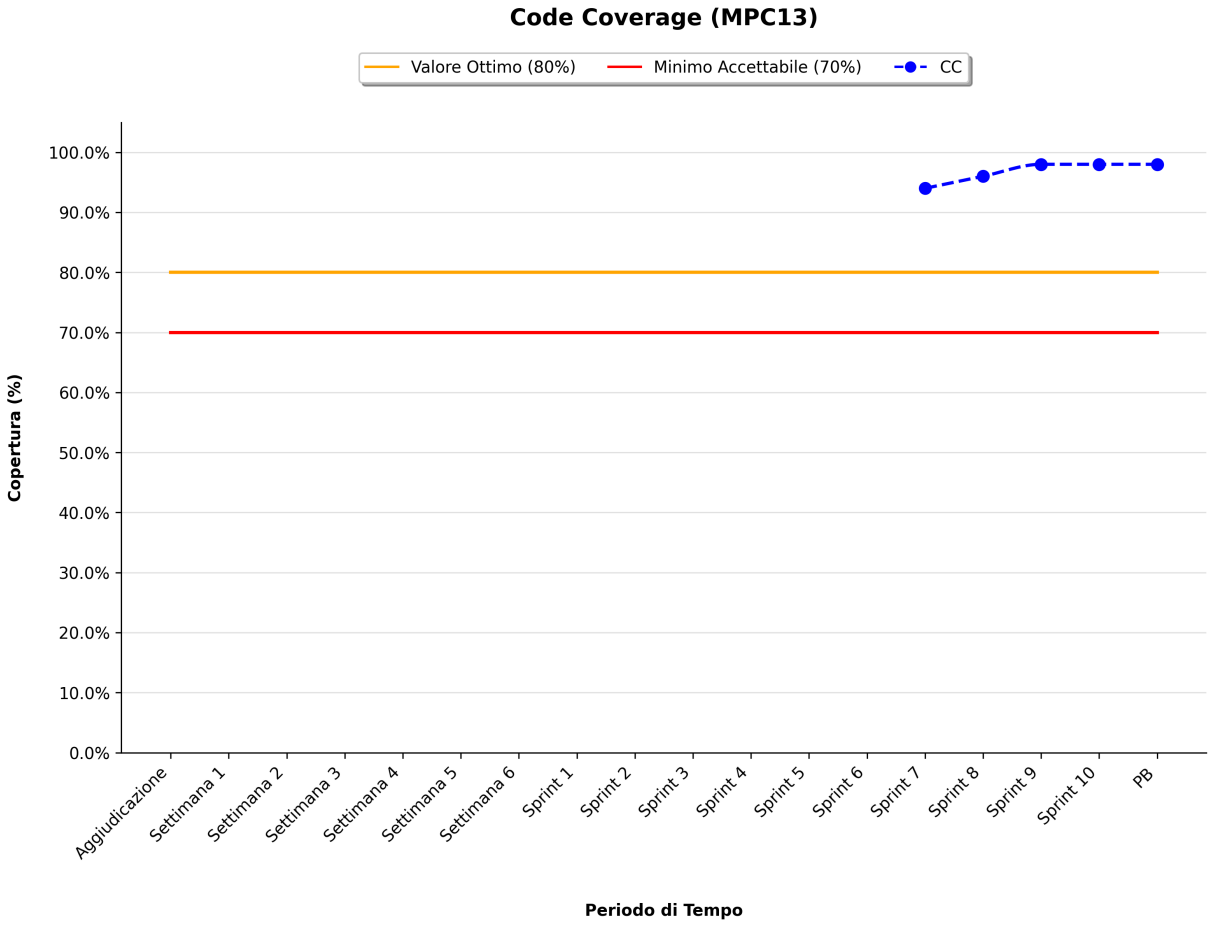
5.9.2.1 Requirements and Technology Baseline (RTB)

Durante i primi sprint, il grafico evidenzia la presenza di alcuni errori ortografici nella documentazione, con un picco registrato nello *Sprint 2*. Il team ha prontamente identificato la criticità e adottato misure correttive, portando il conteggio degli errori a zero entro lo *Sprint 3* per la maggior parte dei documenti.

5.9.2.2 Product Baseline (PB)

La correttezza ortografica è stata mantenuta a zero errori per quasi tutti i documenti consolidati nella RTB, eccezion fatta per il *Piano di Progetto* che ha registrato alcune imperfezioni tra lo *Sprint 7* e lo *Sprint 8* a causa di revisioni interne. L'introduzione del *Manuale Utente* e della *Specifica Tecnica* ha comportato la fisiologica comparsa di alcuni errori ortografici durante gli *Sprint 9* e *10*, legati alla stesura di questa nuova documentazione. Il team ha prontamente intensificato le procedure di revisione incrociata, correggendo le criticità e riportando il conteggio a zero errori per l'intero corredo documentale.

5.9.3 Code Coverage (MPC13)



5.9.3.1 Requirements and Technology Baseline (RTB)

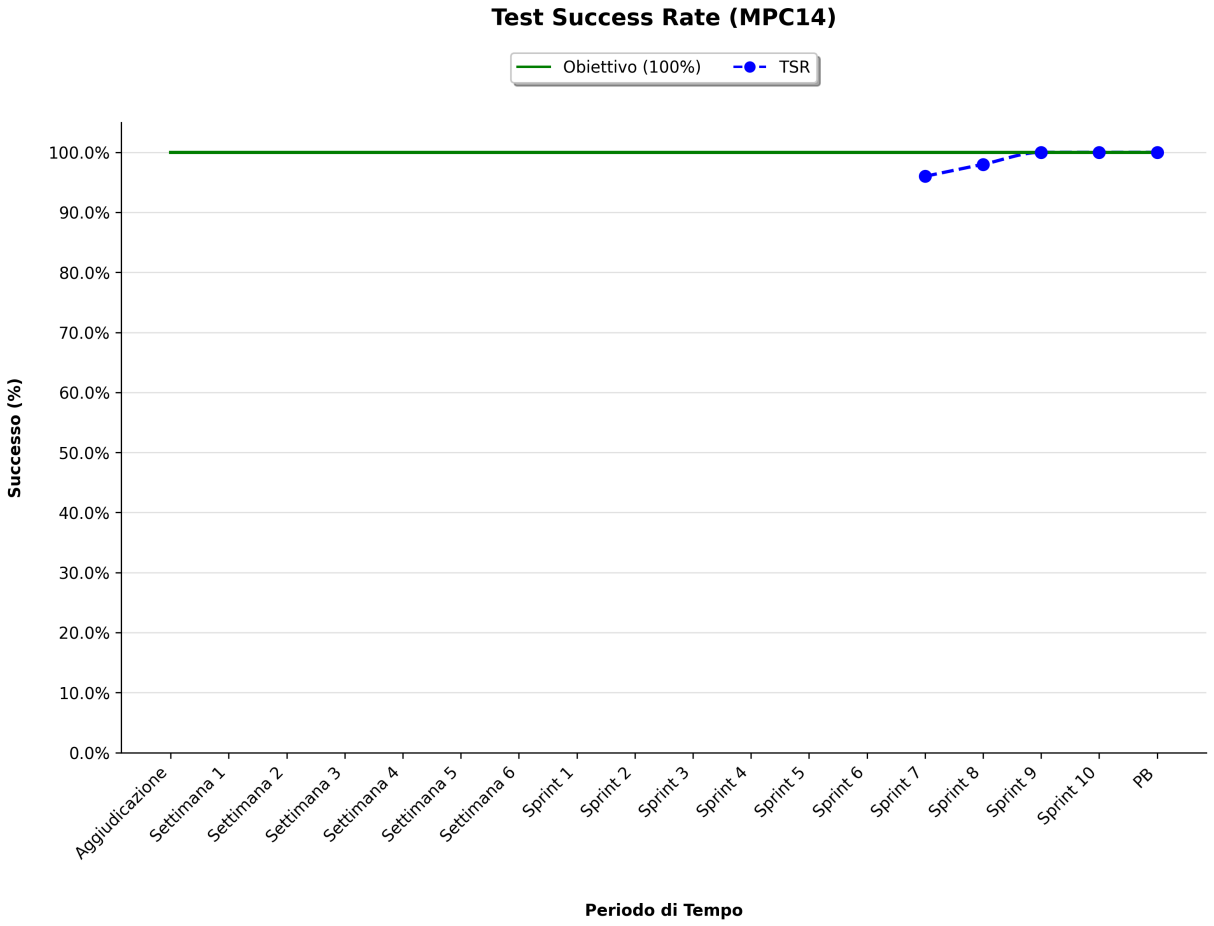
La metrica *Code Coverage* non è stata rilevata durante la baseline RTB.

5.9.3.2 Product Baseline (PB)

La metrica non è stata rilevata durante i primi sprint della PB, in quanto il team era focalizzato sulla progettazione. Il valore iniziale rilevato nello *Sprint 7* si è attestato al 94%, superando la soglia minima del 70%. Negli Sprint successivi la copertura ha mostrato un trend di crescita costante fino a raggiungere il 98%, superando ampiamente la soglia di qualità del 90% imposta nelle configurazioni di test del progetto.

Per cogliere tempestivamente le necessità di miglioramento, il team ha reso questa metrica “attiva” e direttamente vincolante: qualora la *Code Coverage* scenda sotto la soglia ottimale durante una nuova Pull Request, l’indicatore innesca il blocco automatico del merge tramite le pipeline di GitHub Actions, forzando il programmatore a un’azione correttiva tempestiva prima dell’integrazione del codice.

5.9.4 Test Success Rate (MPC14)



5.9.4.1 Requirements and Technology Baseline (RTB)

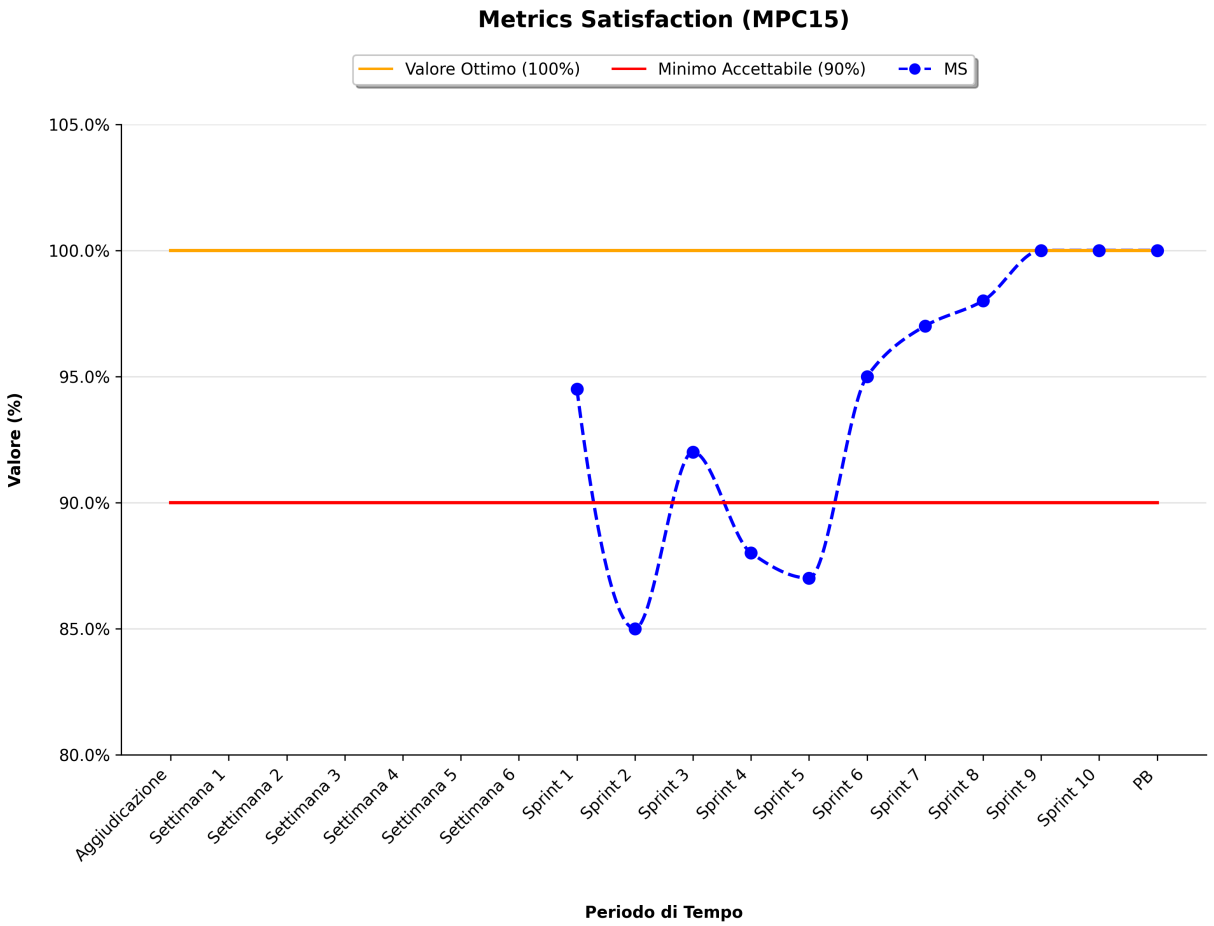
La metrica *Test Success Rate* non è stata rilevata durante la baseline RTB.

5.9.4.2 Product Baseline (PB)

A partire dallo *Sprint 7*, il *Test Success Rate* ha mostrato un trend di crescita costante, passando dal 96% fino a raggiungere il valore ottimale del 100% nello *Sprint 9*. Il team è riuscito a mantenere stabilmente il valore obiettivo per tutta la fase conclusiva della PB.

5.10 Processi Organizzativi

5.10.1 Metrics Satisfaction (MPC15)



5.10.1.1 Requirements and Technology Baseline (RTB)

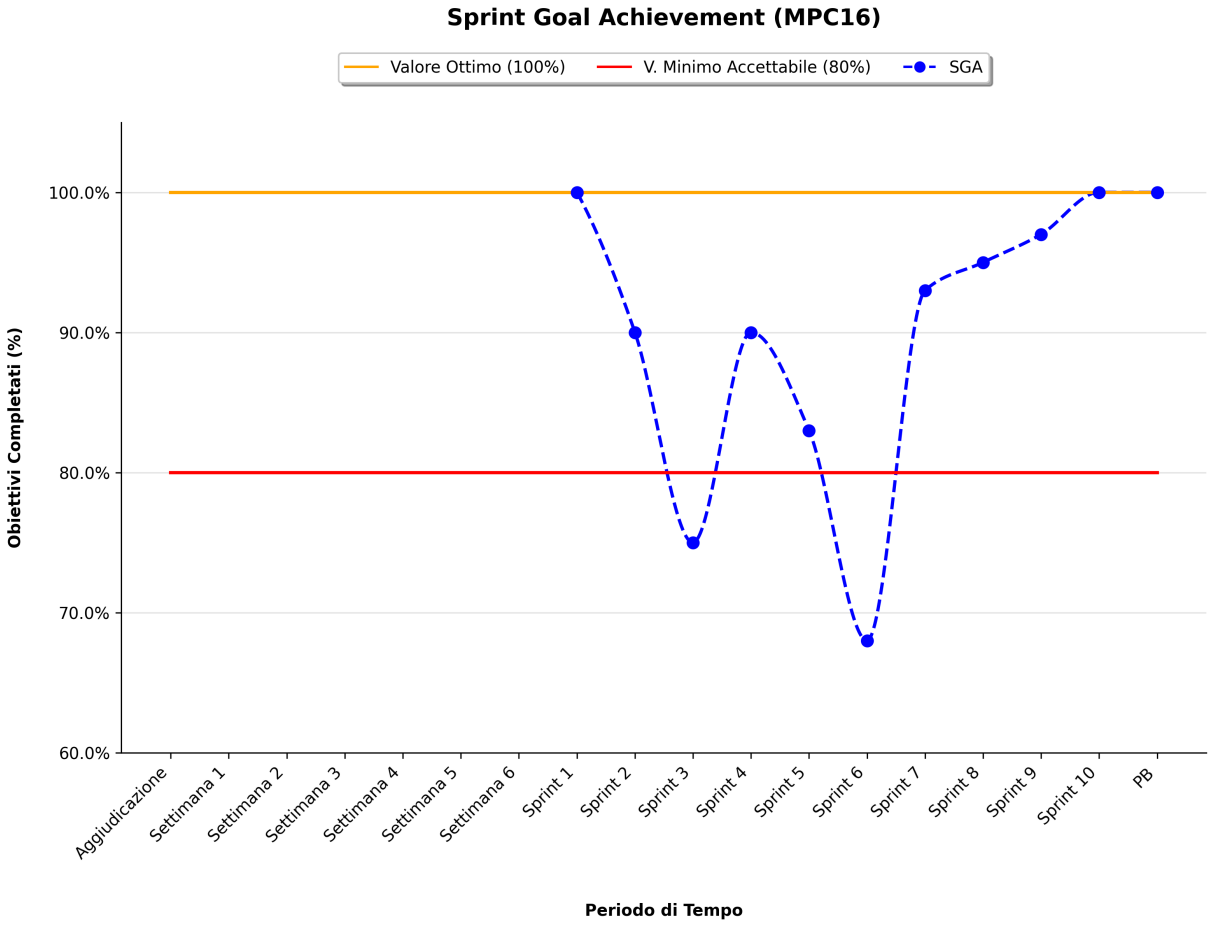
Durante lo *Sprint 1* il valore si è attestato al di sopra della soglia minima accettabile, indicando una buona ma non ancora ottimale conformità ai criteri di qualità definiti. Nello *Sprint 2* si è registrata una flessione al di sotto della soglia minima accettabile.

Nello *Sprint 4*, lo sforamento orario e le difficoltà incontrate hanno causato un nuovo calo. Il team prende atto della criticità e si impegna ad adottare misure correttive nella fase successiva per garantire un maggiore rispetto delle metriche definite.

5.10.1.2 Product Baseline (PB)

Nello *Sprint 5* si è registrata una lieve flessione al di sotto della soglia minima a causa del mancato raggiungimento di alcuni obiettivi. A partire dallo *Sprint 6*, tuttavia, il valore ha intrapreso un trend di crescita costante (dal 95% al 98%), raggiungendo e mantenendo il valore ottimale del 100% a partire dallo *Sprint 9*. Questo andamento progressivo testimonia il consolidamento dei processi qualitativi nel corso della fase conclusiva del progetto.

5.10.2 Sprint Goal Achievement (MPC16)



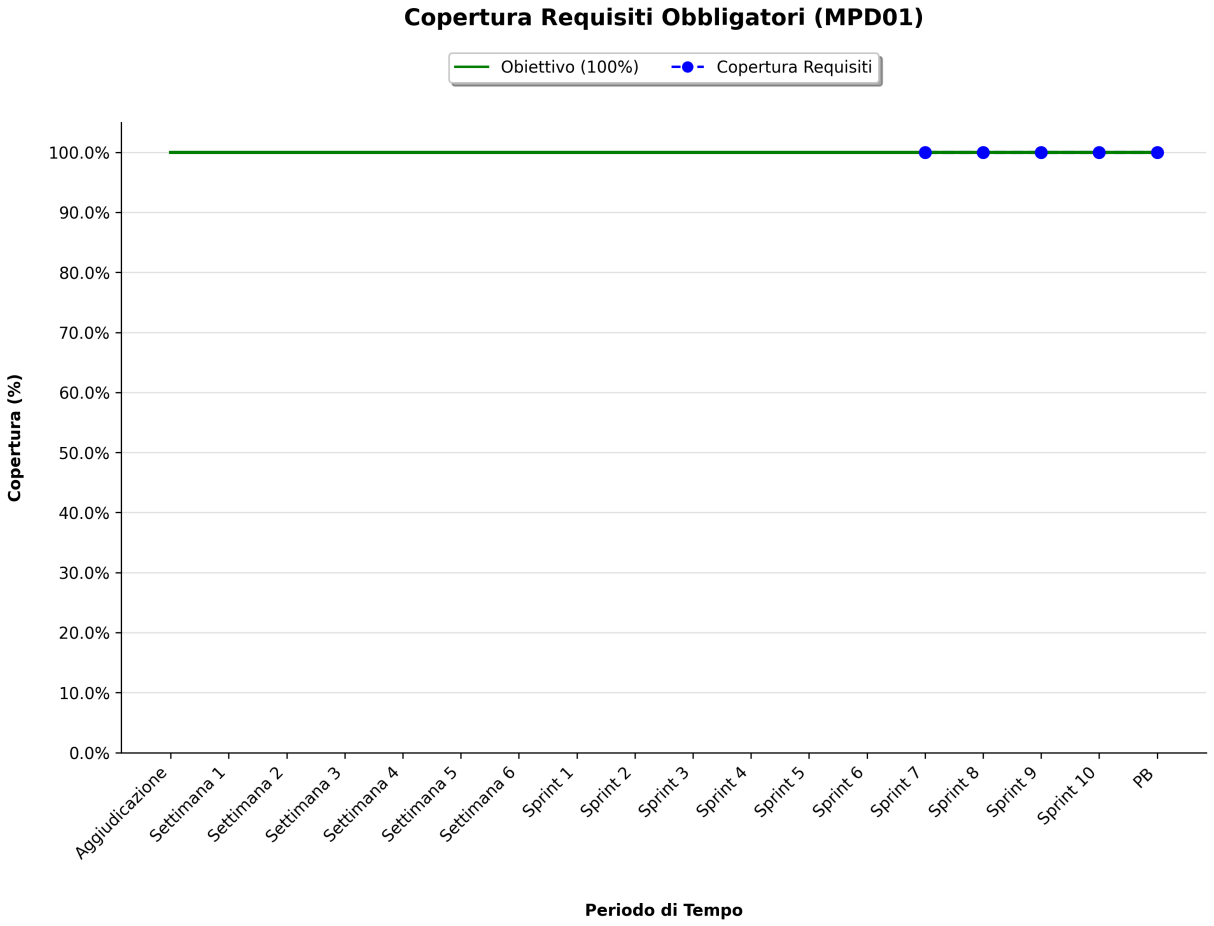
5.10.2.1 Requirements and Technology Baseline (RTB)

Dal grafico è possibile osservare la buona efficacia operativa dimostrata durante lo *Sprint 1* dal team, che è riuscito a completare gli obiettivi prefissati quasi nella loro interezza. Durante lo *Sprint 2*, invece, la metrica ha subito una flessione poiché gli obiettivi prefissati non sono stati pienamente raggiunti.

5.10.2.2 Product Baseline (PB)

La PB è iniziata con una criticità nello *Sprint 5*, dove lo *Sprint Goal Achievement* è sceso al 68%, al di sotto della soglia minima accettabile dell'80%. Il team ha reagito tempestivamente rivedendo, in particolare, la granularità dei task: tale intervento ha innescato un recupero progressivo e costante dell'efficacia negli Sprint successivi (con valori dal 93% al 97%), fino al pieno raggiungimento degli obiettivi (100%).

5.10.3 Copertura Requisiti Obbligatori (MPD01)



5.10.3.1 Requirements and Technology Baseline (RTB)

La metrica *Copertura dei Requisiti Obbligatori* non è stata rilevata durante la baseline RTB.

5.10.3.2 Product Baseline (PB)

La copertura dei requisiti obbligatori ha raggiunto il valore ottimale del 100%. Alcuni requisiti che inizialmente erano stati classificati come obbligatori sono stati declassati a opzionali e non sono stati effettivamente implementati, ma tutti i requisiti rimasti obbligatori sono stati pienamente soddisfatti.

5.10.4 Failure Density - Availability (MPD02 e MPD03)

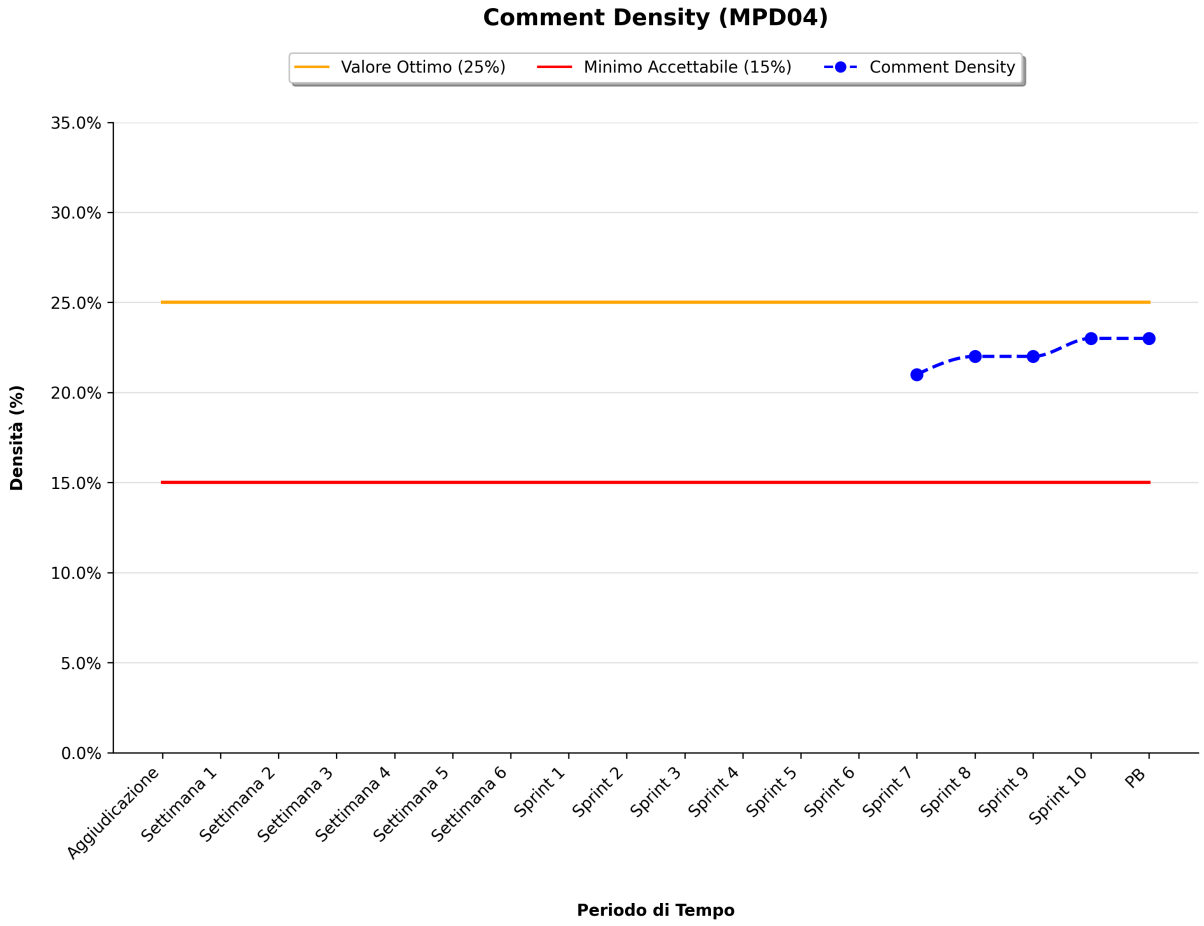
5.10.4.1 Requirements and Technology Baseline (RTB)

Le metriche *Failure Density* e *Availability* non sono state rilevate durante la baseline RTB.

5.10.4.2 Product Baseline (PB)

La *Failure Density* e l'*Availability* sono metriche che richiedono un periodo di esercizio prolungato del sistema in produzione per poter essere misurate in modo affidabile. Il prodotto *Code Guardian*, non essendo ancora in esercizio continuo, non dispone di un campione temporale sufficiente per determinare valori statisticamente significativi.

5.10.5 Comment Density (MPD04)



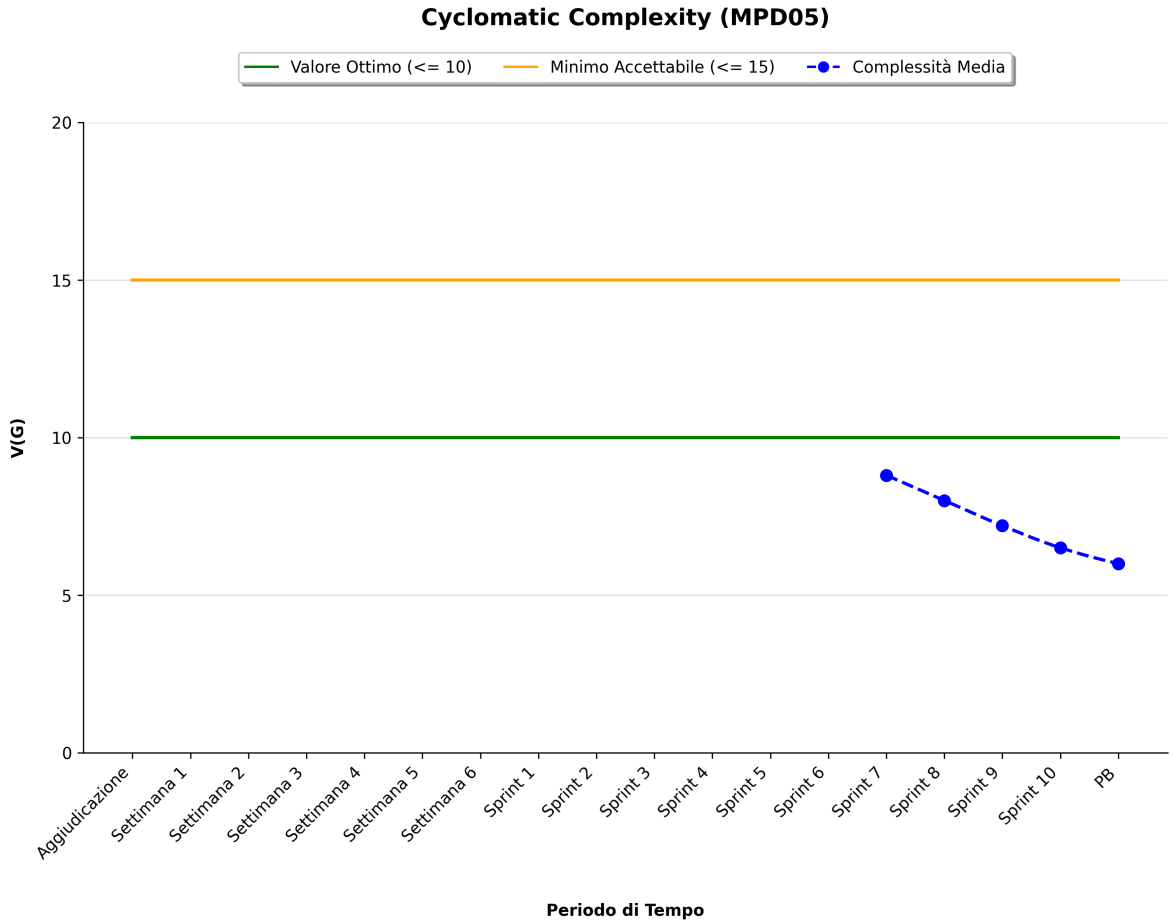
5.10.5.1 Requirements and Technology Baseline (RTB)

La metrica *Comment Density* non è stata rilevata durante la baseline RTB.

5.10.5.2 Product Baseline (PB)

A partire dallo *Sprint 7*, la *Comment Density* si è attestata tra il 21% e il 23%, rientrando stabilmente nel range ottimale definito (15% - 25%).

5.10.6 Cyclomatic Complexity (MPD05)



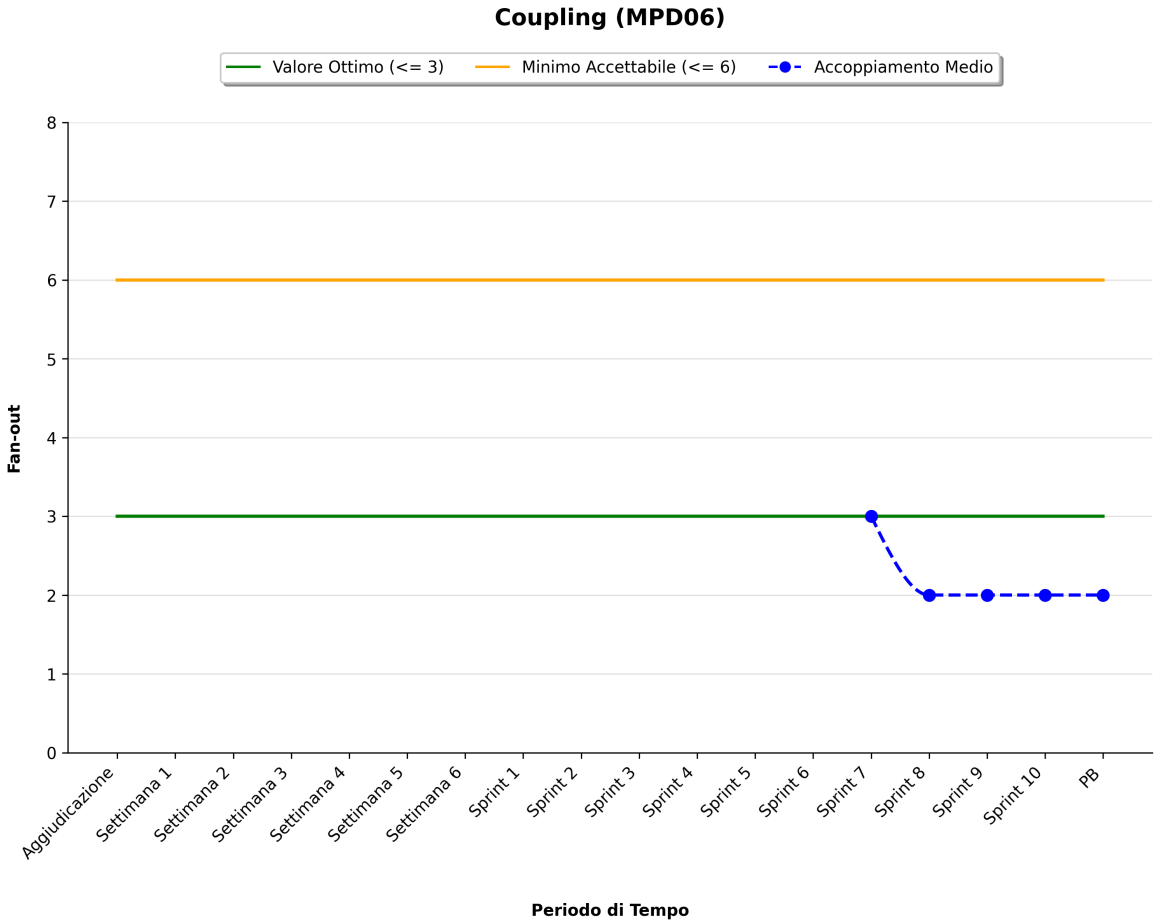
5.10.6.1 Requirements and Technology Baseline (RTB)

La metrica *Cyclomatic Complexity* non è stata rilevata durante la baseline RTB.

5.10.6.2 Product Baseline (PB)

A partire dallo *Sprint 7*, la *Cyclomatic Complexity* media si è attestata su valori compresi tra 9 e 6, rimanendo ampiamente entro la soglia ottimale di V(G) ≤ 10. Il progressivo abbassamento del valore testimonia l'efficacia delle attività di refactoring e la scomposizione del codice in metodi granulari e focalizzati.

5.10.7 Coupling (MPD06)



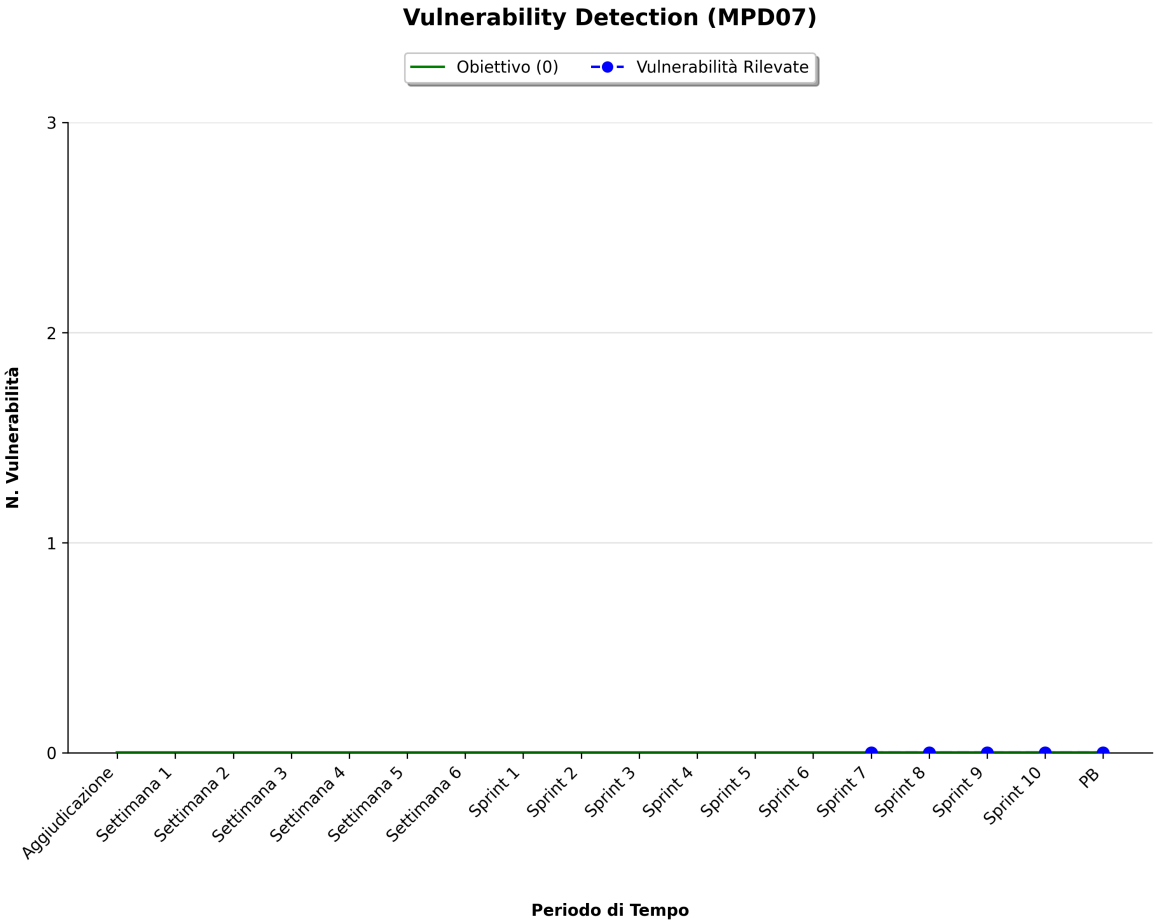
5.10.7.1 Requirements and Technology Baseline (RTB)

La metrica *Coupling* non è stata rilevata durante la baseline RTB.

5.10.7.2 Product Baseline (PB)

Dallo *Sprint 7*, il valore di *Coupling* si è attestato tra 3 e 2, in pieno rispetto della soglia ottimale (≤ 3). Il rigoroso utilizzo dell'*Architettura Esagonale* e l'Inversione delle Dipendenze tramite *Porte* hanno permesso di mantenere un basso grado di accoppiamento tra i moduli.

5.10.8 Vulnerability Detection (MPD07)



5.10.8.1 Requirements and Technology Baseline (RTB)

La metrica *Vulnerability Detection* non è stata rilevata durante la baseline RTB.

5.10.8.2 Product Baseline (PB)

La scansione delle vulnerabilità ha registrato costantemente zero vulnerabilità critiche rilevate, rispettando la soglia ottimale definita nel piano. Il risultato è attribuibile all'adozione di pratiche di secure coding e all'utilizzo di librerie aggiornate.

6 Miglioramento Continuo

Il processo di miglioramento continuo rappresenta il motore evolutivo del **Way of Working** del gruppo Skarab Group. Non ci si limita a correggere gli errori nel codice, ma si punta a ottimizzare sistematicamente i processi organizzativi e di supporto per prevenire la ricorrenza delle anomalie.

La strategia adottata implementa rigorosamente il ciclo di Deming (PDCA), integrandosi con le iterazioni previste dalla metodologia Agile [G]:

- **Plan (Pianificazione):** Definizione degli obiettivi di qualità e delle metriche e pianificazione delle attività.
- **Do (Esecuzione):** Svolgimento delle attività di sviluppo e gestione durante lo Sprint.
- **Check (Controllo):** Al termine di ogni iterazione si misurano i valori delle metriche e si confrontano con le soglie attese.
- **Act (Azione):** Qualora si rilevino scostamenti negativi, vengono definite **Azioni Correttive** che modificano le Norme di Progetto, diventando operative dallo Sprint successivo.

6.1 Azioni di Miglioramento Intraprese

Di seguito sono riportate le criticità emerse e le relative azioni correttive, suddivise per ambito di intervento.

ID	Problema / Causa	Azione Correttiva
Area: Comunicazione		
AM01	**Inefficienza nella Comunicazione Interna** Si sono verificati rallentamenti operativi causati da una comunicazione asincrona poco reattiva. Il team ha preso piena consapevolezza della scarsa efficacia delle modalità iniziali, che generavano colli di bottiglia.	**Ristrutturazione dei Flussi Informativi** I membri coinvolti hanno analizzato le cause (Root Cause Analysis) e compreso la lezione. Sono stati istituiti canali dedicati alle urgenze e aumentata la frequenza dei micro-allineamenti per sbloccare i task pendenti.
Area: Ruoli e Pianificazione		
AM02	**Pressione sulle Scadenze (Time-to-Result)** La necessità di produrre risultati tangibili (PoC) in tempi brevi per la revisione RTB rischiava di non essere soddisfatta con la pianificazione lineare iniziale.	**Ridistribuzione del Budget Orario** È stata effettuata una rimodulazione delle ore pianificate, allocando maggiori risorse sulle attività critiche di sviluppo e riducendo temporaneamente quelle a basso valore aggiunto, per garantire il rilascio puntuale.
Area: Strumenti e Tecnologie		
AM03	**Disomogeneità nella Documentazione** La stesura parallela dei documenti da parte di più persone ha	**Adozione di Template e Funzioni Comuni** Per garantire coerenza, sono state ingegnerizzate le funzioni di typesetting (in

ID	Problema / Causa	Azione Correttiva
	inizialmente generato incoerenze stilistiche e ripetizioni ridondanti o formattate diversamente.	Typst) e creati template condivisi. Questo forza l'uniformità visiva e strutturale indipendentemente dall'autore della sezione.
AM04	**Overhead Nuovi Strumenti** L'adozione contemporanea di nuovi strumenti (Jira, GitHub, Typst) ha comportato un rallentamento iniziale dovuto alla curva di apprendimento.	**Consolidamento della Toolchain** Dopo la fase di rodaggio, l'uso degli strumenti è stato standardizzato nelle Norme di Progetto. La corretta rendicontazione è ora integrata nel flusso di lavoro quotidiano, trasformando l'overhead iniziale in un guadagno di efficienza.

Table 12: Storico delle azioni di miglioramento (Periodo RTB)

ID	Problema / Causa	Azione Correttiva
Area: Tecnologie e Architettura		
AM05	**Complessità Architettura Serverless** Durante la prima fase di progettazione, l'adozione di AWS Step Functions e Lambda per l'orchestrazione degli agenti si è rivelata eccessivamente complessa.	**Cambio di Architettura** Il team ha deciso tempestivamente di scartare la soluzione Serverless, internalizzando la logica di orchestrazione direttamente nell'esagono del microservizio di analisi.
AM06	**Inconsistenze d'Integrazione del Codice** Con l'avvio della codifica distribuita, sono emersi rischi legati a disomogeneità implementative e build fallite al momento del merge sul repository condiviso.	**Adozione di Pipeline CI/CD** Sono state introdotte e rese vincolanti pipeline di Continuous Integration tramite GitHub Actions, automatizzando linting, formattazione e test.
AM07	**Difficoltà nel Deployment** L'inesperienza pregressa del team con i servizi AWS ha reso le configurazioni di deployment più dispendiose in termini di tempo rispetto a quanto stimato inizialmente.	**Studio Mirato e Containerizzazione** Per superare questo ostacolo logistico, il team ha dedicato sessioni di studio alla documentazione AWS e ha standardizzato gli ambienti tramite l'uso di container (Docker).
Area: Organizzazione del Team		

ID	Problema / Causa	Azione Correttiva
AM08	**Colli di Bottiglia nel Coordinamento Globale** La gestione contemporanea dell'intera codebase da parte di tutto il team generava confusione sui task, sovrapposizioni e rallentamenti nelle decisioni di basso livello.	**Divisione in Sottogruppi Specializzati** Il team si è diviso in tre sottogruppi (Frontend, Microservizio Account, Microservizio Analisi). Questo ha snellito le decisioni interne, aumentando il parallelismo e la responsabilità individuale.

Table 13: Storico delle azioni di miglioramento (Periodo PB)

7 Conclusioni

L'attività di miglioramento continuo per il progetto *Code Guardian* si è rivelata non solo una pratica formale, ma il vero motore per garantire la qualità finale del prodotto. Durante la RTB, l'analisi delle metriche ha evidenziato come l'avvio del progetto abbia scontato il "prezzo d'ingresso" della curva di apprendimento dei nuovi strumenti e del necessario assestamento delle dinamiche comunicative. Le azioni correttive iniziali (**AM01**, **AM03**) hanno permesso di superare quella prima frammentazione operativa.

Con l'ingresso nella PB, la complessità tecnica ed organizzativa è aumentata. Le nuove criticità non riguardavano più l'assestamento iniziale, ma l'implementazione pratica. In questo contesto, il ciclo di *Plan-Do-Check-Act* ha dimostrato la sua reale efficacia: la decisione di cambiare tempestivamente l'architettura scartando il Serverless (**AM05**) ha salvaguardato le scadenze, l'imposizione di rigorose pipeline CI/CD (**AM06**) ha arginato il debito tecnico e la standardizzazione del deployment (**AM07**) ha mitigato le complessità del cloud.

Dal punto di vista organizzativo, la transizione verso sottogruppi specializzati (**AM08**) ha trasformato il gruppo in un team di sviluppo maturo ed efficiente.