Norme di Progetto

Autore	Nicolò Lattanzio, Giulia Romanato, Lorenzo Grolla, Alessandro Frison
Verificatore	Alessandro Morabito, Damiano Berti, Giacomo Nalotto
Approvazione	Tutti i membri del gruppo

1 Registro delle versioni

Versione	Data	Autore	Verificatore	Descrizione delle modifiche
1.0.0	22/02/2026	Tutti i membri del gruppo	Tutti i membri del gruppo	Versione finale del documento, con tutte le sezioni completate e revisionate.
0.11.3	12/02/2026	Nicolò Lattanzio		Correzione ortografiche varie e modifiche minori.
0.11.2	15/01/2026	Lorenzo Grolla		Sistemata sezione Metriche di Qualità del Processo
0.11.1	15/01/2026	Alessandro Frison		Sistemata sezione Metriche e standard di Qualità e Metriche di Qualità del Prodotto
0.11.0	13/01/2026	Giulia Romanato	Damiano Berti	Aggiunta sotto sezione 5.1.1 Metodologia, aggiornamento 5.4 Formazione e aggiornamento sezione 4.1.2 Verbali (definizione struttura interna dei verbali, sezione tabella decisioni e azioni diventa sottosezione 4.1.2.1)
0.10.0	12/01/2026	Giulia Romanato	Damiano Berti	Aggiunta sotto sezione 5.7 Formazione e 5.5 Gestione delle infrastutture e degli strumenti
0.9.0	12/01/2026	Nicolò Lattanzio	Damiano Berti	Aggiunta sezione Metriche di Qualità del processo e prodotto, e completamento sezione 3.2 Processi di Sviluppo.
0.8.0	11/01/2026	Giulia Romanato	Alessandro Morabito	Aggiunta sotto sezione 3.4 rapporto con proponente e 3.5 rapporto col committente e aggiunta del norme di progetto, verbale, dichiarazione di impegni e dichiarazione degli impegni nella sezione 3.1.1 Documenti Principali

Versione	Data	Autore	Verificatore	Descrizione (continua)
0.7.0	08/01/2026	Nicolò Lattanzio	Alessandro Morabito	Aggiunta sotto sezione diari di bordo e sezione Gestione della configurazione
0.6.0	05/01/2026	Giulia Romanato	Alessandro Morabito	Aggiunta sezione processi organizzativi
0.5.0	22/12/2025	Nicolò Lattanzio	Giacomo Nalotto	Aggiunta sezione Strumenti di supporto e sezione Produzione e archiviazione.
0.4.0	15/12/2025	Nicolò Lattanzio	Giacomo Nalotto	Aggiunta sezione Documenti Principali, insieme ai primi documenti chiave del progetto.
0.3.0	07/12/2025	Nicolò Lattanzio	Alessandro Morabito	Aggiunta sottosezione 4.1.1 per la Struttura dei documenti, 4.1.2 per il loro versionamento e 4.1.3 per la tabella Decisioni e Azioni.
0.2.0	06/12/2025	Nicolò Lattanzio	Alessandro Morabito	Aggiunta sezione Processi Primari, con relative sottosezioni, e prima redazione della sezione Processi di Supporto.
0.1.0	30/11/2025	Nicolò Lattanzio	Alessandro Morabito	Creazione e prima redazione del documento.

Indice

2 Informazioni introduttive

2.1 Scopo del documento

Questo documento definisce le norme di progetto adottate dal gruppo ***BYTE HOLDERS*** per la gestione e lo sviluppo del progetto. Esso stabilisce le linee guida per la documentazione, la comunicazione, la gestione delle versioni e le responsabilità dei membri del team.
Per la realizzazione del documento, e' stato deciso di prendere come riferimento lo standard ISO/IEC 12207, che definisce i processi di ciclo di vita del software e fornisce una struttura per la gestione del progetto. Le tipologie di processi sono le seguenti:

- **Processi Primari** : Consiste nelle attività principali che portano alla creazione, consegna e mantenimento del prodotto finale.
- **Processi di Supporto**: Processi che forniscono supporto alle attività principali, possono essere utilizzato da qualsiasi processo e ne garantiscono la qualità e la correttezza
- **Processi Organizzativi**: Processi necessari a creare l'ambiente e le risorse necessarie affinché i processi primari possano esistere.

2.2 Scopo del prodotto

Il prodotto sviluppato dal gruppo ***BYTE HOLDERS*** è un'applicazione web che mira a fornire una piattaforma intuitiva per la gestione delle repository GitHub, fornendo informazioni utili in base alle varie necessità. Tra queste funzionalità vi sono:

- Analisi statica della qualità del codice
- Analisi della qualità della documentazione
- Analisi OWASP per la verifica delle vulnerabilità di sicurezza tramite l'integrazione di strumenti di terze parti
- Integrazione con API di GitHub per fornire statistiche dettagliate sulle repository
- Proposte di remediation per migliorare la qualità del codice e della documentazione

Il prodotto segue la proposta del **capitolato C2 - Code Guardian** fornito dal proponente ***Var-Group S.r.l.***
Il gruppo si è posto l'obiettivo di completare l'MVP di questo progetto entro il **20 Marzo 2026**, con un costo totale di **13.290 Euro.**

2.3 Riferimenti

- Capitolato C2 - Code Guardian
- Documentazione del progetto

3 Processi Primari

Questa sezione descrive i processi del ciclo di vita primari, definiti dallo standard ISO/IEC 12207, che sono direttamente coinvolti nella creazione, fornitura e manutenzione del prodotto software. Essi costituiscono il cuore operativo del progetto, governando le attività principali

dall'acquisizione delle esigenze alla consegna.
Possiamo distinguere due macro processi:

- **Processo di Fornitura:** Si occupa della gestione delle relazioni con il committente e il proponente, dalla risposta alla richiesta iniziale fino alla consegna del prodotto.
- **Processo di Sviluppo:** Copre tutte le attività ingegneristiche necessarie per trasformare i requisiti in un prodotto software funzionante, includendo analisi, progettazione, implementazione, testing e rilascio.

3.1 Processo di Fornitura

Questo processo definisce le attività che il gruppo Byte Holders, in qualità di fornitore, implementa per garantire la corretta fornitura del prodotto software a VarGroup S.r.l.,proponente del progetto, ai docenti Vardanega e Cardin, committenti del progetto. Esso disciplina l'intero ciclo, dall'aggiudicazione del capitolato alla consegna finale.
Vediamo ora le attività principali coinvolte in questo processo:

1. **Risposta al Bando:** Analisi del capitolato presentato dal proponente e preparazione dell'offerta (documento *Candidatura*).
2. **Negoziazione Requisiti:** Realizzazione di una controproposta per il proponente, definendo i requisiti funzionali e non funzionali che il gruppo Byte Holders, come fornitore, prevede di riuscire a realizzare.
3. **Pianificazione della Fornitura:** Creazione del *Piano di Progetto* che dettaglia come verranno eseguiti i processi di sviluppo e gestione.
4. **Esecuzione e Controllo:** Realizzazione del progetto conforme al piano, con reporting periodico allo stakeholder.
5. **Revisione e Valutazione:** Condotta di revisioni tecniche e attività di verifica e validazione.
6. **Consegna e Completamento:** Consegna del prodotto finale insieme alla relativa documentazione.

3.1.1 Documenti Principali:

Per questo progetto saranno prodotti i seguenti documenti chiave:

Valutazione Capitolati

Autore: Responsabile

Destinatari: Professore Tullio Vardanega

Scopo:
Nella Valutazione Capitolati il gruppo Byte Holders presenta l'obiettivo, i pregi, i difetti e un resoconto dell'eventuale incontro con l'azienda proponente di ogni capitolato proposto.

Dichiarazioni Impegni

Autore: Responsabile

Destinatari: Professore Tullio Vardanega

Scopo:
Nella Dichiarazione degli Impegni il gruppo Byte Holders presenta una stima dei costi del progetto, a seguito di un approfondito ragionamento riguardo le responsabilità dei vari ruoli e all'impegno orario di ogni componente del gruppo. Inoltre viene proposta una data di cosegna finale del progetto.

Lettera di Candidatura

Autore: Responsabile

Destinatari: VarGroup S.r.l., Professori Tullio Vardanega e Riccardo Cardin

Scopo:
Con la lettera di candidatura il gruppo Byte Holders si propone come fornitore per la realizzazione del progetto descritto nel capitolato C2 - Code Guardian, presentando le proprie competenze, esperienze e motivazioni per intraprendere questa collaborazione con il committente VarGroup S.r.l.

Analisi Dei Requisiti

Autore: Analista

Destinatari: VarGroup S.r.l., ByteHolders, Professori Tullio Vardanega e Riccardo Cardin

Scopo:
Nell'analisi dei requisiti il gruppo Byte Holders si impegna a raccogliere, analizzare e documentare i requisiti funzionali e non funzionali del sistema richiesto dal committente VarGroup S.r.l. In particolare saranno descritti i vari casi d'uso, le specifiche di interfaccia e i vincoli tecnici.

Glossario

Autore: Analista

Destinatari: VarGroup S.r.l., ByteHolders, Professori Tullio Vardanega e Riccardo Cardin

Scopo:
Nel glossario il gruppo Byte Holders si impegna a definire e spiegare i termini tecnici, acronimi e abbreviazioni utilizzati nel contesto del progetto richiesto dal committente VarGroup S.r.l., al fine di garantire una comprensione comune tra tutti gli stakeholder coinvolti.

Piano di Progetto

Autore: Responsabile

Destinatari: VarGroup S.r.l., Professori Tullio Vardanega e Riccardo Cardin

Scopo:
Nel Piano di Progetto il gruppo Byte Holders si impegna a definire le strategie, le risorse e le tempistiche necessarie per la realizzazione del progetto richiesto dal committente VarGroup S.r.l. garantendo il rispetto degli standard qualitativi e dei vincoli contrattuali.

Piano di Qualifica

Autore: Amministratore

Destinatari: VarGroup S.r.l., Professori Tullio Vardanega e Riccardo Cardin

Scopo:
Nel Piano di Qualifica il gruppo Byte Holders si impegna a definire le strategie, i criteri e le procedure di verifica e validazione che garantiranno che il prodotto software sviluppato soddisfi i requisiti specificati dal committente VarGroup S.r.l. e rispetti gli standard di qualità concordati.

Norme di Progetto

Autore: Amministratore

Destinatari: VarGroup S.r.l., Professore Tullio Vardanega, ByteHolders

Scopo:
Nelle Norme di Progetto il gruppo Byte Holders si impegna a definire il way of working e le procedute implementate dal gruppo per lo svolgimento del progetto.

Verbali

Autore: Amministratore

Destinatari: VarGroup S.r.l., Professori Tullio Vardanega e Riccardo Cardin,

Scopo:
Nei verbali, che possono essere interni (solo tra i membri del gruppo) o esterni (tra il gruppo e l'azienda proponente), il gruppo Byte Holders si impegna a documentare le riunioni svolte in modo da tracciare le decisioni e azioni prese e le discussioni intraprese.

3.1.2 Strumenti a supporto

Per il processo di fornitura il gruppo ***BYTE HOLDERS*** ha deciso di utilizzare i seguenti strumenti, suddivisi in due parti, una per la comunicazione interna al gruppo:

- **Discord**: Per riunioni principalmente vocali tra i membri del gruppo. Grazie alla possibilità di creare canali testuali specifici per ogni attività, viene utilizzato per tenere traccia di file e link utili allo sviluppo del progetto.

- **WhatsApp**: Per comunicazioni rapide e asincrone tra i membri del gruppo, utilizzato in oltre per tenere traccia degli appuntamenti previsti e per inviare promemoria.

La seconda sezione degli strumenti a supporto riguarda la comunicazione verso l'azienda proponente:

- **Teams**: Per riunioni vocali e video con il proponente VarGroup S.r.l., grazie alla sua integrazione con Outlook si ha anche la possibilità di pianificare riunioni direttamente dal calendario.
- **Slack**: Per la comunicazione rapida con l'azienda proponente, utilizzato principalmente per concordare riunioni.

3.2 Processo di Sviluppo

Il processo di sviluppo definisce il ciclo di vita ingegneristico attraverso il quale i requisiti del committente vengono trasformati nel prodotto software finale. Questo processo organizza in fasi strutturate le attività di analisi, progettazione, implementazione e validazione, garantendo un approccio sistematico e tracciabile alla realizzazione del sistema.

- **Attività Chiave (Fasi del Ciclo di Vita):**
  1. **Analisi dei Requisiti:** Raccolta, analisi, specifica e validazione dei requisiti con gli stakeholder. Prodotto: *Documento di Analisi dei Requisiti*.
  2. **Progettazione dell'Architettura:** Definizione dell'architettura di sistema e software ad alto livello.
  3. **Progettazione di Dettaglio:** Definizione dettagliata dei componenti e delle interfacce.
  4. **Costruzione (Implementazione e Codifica):** Scrittura del codice secondo gli standard di progetto e utilizzo di un sistema di versioning (Git). Prodotto: Codice sorgente e repository.
  5. **Integrazione e Testing:**
     - Testing delle unità (sviluppatori)
     - Testing d'integrazione (team di integrazione)
     - Testing di sistema (team di qualità)
- **Metodologia:** Adotteremo pratiche ibride Agile, con sprint per lo sviluppo e milestone formali per le revisioni.

3.2.1 Analisi dei Requisiti

L'analisi dei requisiti è una fase cruciale del processo di sviluppo, in cui il gruppo ***BYTE HOLDERS*** si impegna a comprendere e documentare le esigenze del committente VarGroup S.r.l. Tale analisi e' reperibile nel documento *Analisi Dei Requisiti*, che include:

- Requisiti funzionali e non funzionali
- Casi d'uso dettagliati
- Specifiche di interfaccia

- Vincoli tecnici e di progetto

3.2.2 Nomenclatura Casi d'Uso

Per garantire l'univocità e la facile identificazione delle funzionalità offerte dal sistema, ogni caso d'uso è identificato da un codice alfanumerico strutturato secondo la seguente sintassi:

UC[CodicePadre].[CodiceFiglio]

Dove:

- **UC**: Acronimo fisso che sta per *Use Case*;
- **CodicePadre**: Numero intero progressivo che identifica il caso d'uso principale (o padre);
- **CodiceFiglio** (Opzionale): Numero intero progressivo che identifica un sotto-caso d'uso (o figlio) incluso o esteso dal padre.

Esempi:

- *UC1*: Login (Caso d'uso principale);
- *UC1.1*: Recupero Password (Sotto-caso d'uso del Login).

3.2.3 Nomenclatura Requisiti

In conformità con la necessità di tracciabilità dei requisiti (processo di sviluppo software ISO/IEC 12207), ogni requisito è classificato e identificato tramite un codice univoco che ne esplicita la tipologia e la priorità.

La sintassi adottata è la seguente:

R[Tipologia][Importanza][Codice]

Dove le singole voci assumono i seguenti significati:

3.2.3.1 Tipologia

Indica la natura del requisito:

- **F**: *Funzionale* - Descrive una funzionalità o un comportamento del sistema;
- **Q**: *Qualitativo* (o Non Funzionale) - Descrive attributi di qualità come performance, affidabilità o sicurezza;
- **V**: *Vincolo* - Descrive limitazioni tecniche, normative o progettuali imposte dal committente o dall'ambiente;
- **P**: *Prestazionale* - Sottocategoria specifica per requisiti legati alle performance (opzionale, spesso accorpato in Q).

3.2.3.2 Importanza

Definisce la priorità di implementazione del requisito:

- **O**: *Obbligatorio* (Must have) - Requisito indispensabile per il funzionamento del sistema o richiesto esplicitamente dal vincolo contrattuale;
- **D**: *Desiderabile* (Should have) - Requisito che porta valore aggiunto ma la cui assenza non compromette le funzionalità base;
- **F**: *Facoltativo* (Could have/Nice to have) - Requisito opzionale da implementare solo se le risorse e le tempistiche lo permettono.

3.2.3.3 Codice

Un numero intero progressivo univoco che identifica il requisito (spesso gerarchico).
Esempi:

- *RFO1*: Requisito Funzionale Obbligatorio numero 1;
- *RQD12*: Requisito Qualitativo Desiderabile numero 12.

3.2.4 Codifica del codice sorgente

Questa sezione ha l'obiettivo di definire le linee guida per la scrittura di codice sorgente in maniera chiara, mantenibile e coerente all'interno del progetto sviluppato dal gruppo ***BYTE HOLDERS***.
Le seguenti convenzioni di codifica sono adottate per garantire uniformità e qualità del codice:

- **Lingua**: I nomi delle variabili, funzioni ed eventuali classi devono essere scritti in inglese.
- **Funzioni**: Realizzare funzioni con una singola responsabilità, evitando funzioni troppo lunghe o complesse, l'utilizzo di funzioni specifiche per ogni compito le rende riutilizzabili e facilmente testabili.
- **Commenti**: Utilizzare commenti per spiegare il funzionamento del codice, specialmente in sezioni complesse o non intuitive. I commenti devono essere chiari e concisi, evitando di commentare cose facilmente interpretabili dal codice.
- **Indentazione e Spaziatura**: Utilizzare un'indentazione coerente (ad esempio 2 o 4 spazi) per migliorare la leggibilità del codice. Mantenere una spaziatura adeguata tra le sezioni di codice per separare logicamente i blocchi.

Relativamente al posizionamento dei file all'interno della repository GitHub, il gruppo ***BYTE HOLDERS*** ha deciso di adottare la seguente struttura:

- **Cartelle**: Ogni macro funzionalità del progetto avrà una propria cartella dedicata, contenente tutti i file correlati a quella funzionalità (es. cartelle separate per *frontend* e *backend*). All'interno di queste, il codice sarà ulteriormente suddiviso in base al pattern architetturale scelto (es. `components`, `services`, `controllers`).
- **Nomenclatura dei file**: I nomi dei file devono essere esplicativi e riflettere chiaramente il loro contenuto o la loro responsabilità. Si adotterà una convenzione coerente con il framework utilizzato: *PascalCase* per i componenti React (es. `UserProfile.tsx`) e *kebab-case* per i file di backend o utility (es. `user-controller.ts`).

- **File di configurazione**: Tutti i file di configurazione globale dell'ambiente o dei tool di sviluppo (es. `.gitignore`, `docker-compose.yml`, file per pipeline CI/CD) devono risiedere nella directory principale (root) della repository o nella root del rispettivo modulo.
- **Codice di test**: I test automatizzati devono essere facilmente distinguibili dal codice di produzione. Saranno posizionati in cartelle dedicate (es. `__tests__`) oppure affiancati al file sorgente che testano, utilizzando suffissi standard e riconoscibili (es. `nome-file.spec.ts` o `nome-file.test.ts`).

3.2.5 Strumenti di supporto

Per il processo di sviluppo il gruppo ***BYTE HOLDERS*** ha deciso di utilizzare i seguenti strumenti:

- **Visual Studio Code**: per la scrittura del codice sorgente, scelto per la sua versatilità e supporto a numerose estensioni.
- **Draw.io**: per la realizzazione dei diagrammi UML e altre rappresentazioni grafiche necessarie per la realizzazione degli use-case pubblicati nel documento di Analisi Dei Requisiti e per progettazione del software finale.

3.3 Rapporti col proponente

Il proponente mette a disposizione:

- un canale slack per comunicare in modo rapido con i membri del gruppo ByteHolders, per rispondere a eventuali dubbi e domande, organizzare incontri e condividere materiale
- la possibilità di organizzare degli incontri, principalmente da remoto, su richiesta del fornitore a seguito del cui il gruppo rederrà un verbale esterno
- degli incontri di formazione sulle tecnologie consigliate (backend, AWS, frontend, introduzione su AI)
- un incontro in presenza di design thinking all'avvio del progetto

Il fornitore si impegnerà ad ascoltare i feedback della proponente e a presentare un MVP (Minimum Viable Product) nei tempi e costi previsti.

3.4 Rapporti col committente

Il committente mette a diposizione:

- la possibilità di richiedere ricevimenti o/e scrivere mail per chiarire dubbi riguardo i documenti da consegnare o per duscutere problemi riguardanti i rapporti con la proponente o tra i vari membri del gruppo
- incontri periodici, detti Diari di Bordo, in cui il gruppo presenta i dubbi e le difficoltà incotrate

Il fornitore si impegna a partecipare a due momenti di revisione di avanzamento:

- **RTB (Requirements and Tecnology Baseline):** in cui si prevede la consegna del documento di Analisi dei Requisiti, la stesura del Paino di Progetto, Piano di Qualifica, Norme

di Progetto e Glossario e la presentazione di un PoC (proof of Concept), una demo eseguibile che motivi la scelta delle tecnologie, framework, librerie adottate e dimotri la loro adeguatezza e interoperatività

- **PB (Product Baseline):** in cui si prevede la consegna del documento di Specifica Tecnica e la presentazione di un MVP (Minimum Viable Product)

4 Processi Di Supporto

Questa sezione descrive i processi di supporto secondo lo standard ISO/IEC 12207, che forniscono l'infrastruttura tecnica e metodologica necessaria per garantire qualità e coerenza nell'esecuzione dei processi primari. Tali processi, di natura trasversale, abilitano e migliorano le attività di sviluppo, gestione e verifica attraverso strumenti, procedure e controlli dedicati.

4.1 Documentazione

La documentazione del progetto viene gestita utilizzando **Latex** e viene archiviata in un repository dedicato su **GitHub**.
Ogni documento segue una struttura standardizzata per garantire coerenza e facilità di lettura.

4.1.1 Struttura dei documenti

Tutti i documenti formali prodotti dal gruppo seguono una struttura standardizzata per garantire coerenza, professionalità e facilità di consultazione. Il template **Latex** utilizzato definisce chiaramente gli elementi che compongono ciascun documento:

- **Frontespizio:**
  - Logo del gruppo Byte Holders e titolo del progetto
  - Nome del documento (es. "Norme di Progetto", "Verbale Interno")
  - Informazioni di identificazione: data, autore, verificatore e approvatore
- **Registro delle Versioni:**
  - Tabella posizionata dopo l'indice
  - Tracciamento cronologico inverso (LIFO) di tutte le modifiche
  - Colonne per versione, data, autore e descrizione delle modifiche
- **Indice:**
  - Elenca tutte le sezioni, sottosezioni e relative pagine
  - Fornisce una mappa navigabile del documento
- **Contenuto Principale:**
  - Organizzato gerarchicamente in sezioni, sottosezioni e sotto-sottosezioni
- **Stile Grafico Uniforme:**
  - Palette di colori definita dal gruppo (blu-viola in diverse tonalità)

  - Font sans-serif (Helvetica) per migliorare la leggibilità
  - Margini standardizzati

Questa struttura si applica a tutti i tipi di documenti prodotti, tra cui: Verbali(interni ed esterni), Norme di Progetto, Analisi dei Requisiti...

4.1.2 Verbali

Oltre alle caratteristiche generali applicate a qualsiasi documento di progetto, i verbali presentano una struttura interna standardizzata, articolata nelle seguenti sezioni:

- **Informazioni introduttive**: suddivise nelle seguenti sottosezioni:
  - **Durata e luogo**: indicazione dell'orario di inizio e di fine e del luogo e modalità con cui si è tenuta (in presenza oppure da remoto tramite strumenti quali Discord o Microsoft Teams)
  - **Partecipanti**: tabella riportante l'elenco dei membri del gruppo presenti alla riunione
- **Contenuto della riunione**: suddiviso nelle seguenti sottosezioni:
  - **Ordine del giorno**: elenco degli argomenti da discutere e trattati durante la riunione
  - **Resoconto della discussione**: sintesi delle principali discussioni e considerazioni emerse durante la riunione; tale sezione può essere ulteriormente articolata in sottosezioni, a discrezione dell'autore del verbale
- **Tabella delle decisioni e delle azioni**: riepilogo strutturato delle decisioni prese e delle azioni concordate, con eventuale indicazione dei responsabili.

4.1.2.1 Tabella Decisioni e Azioni

Ogni verbale relativo al progetto conterrà una tabella decisioni e azioni, che riassume le decisioni prese durante la riunione e le azioni assegnate ai membri del team. Anche in questo caso, per garantire la tracciabilità, ogni decisione o azione avrà un proprio codice di identificazione, composto da un prefisso che ne indica la natura (DEC per decisione, AZ per azione); seguito dalla fase del progetto in cui è stata presa (RTB o PB); e infine un numero progressivo che identifica univocamente la decisione o l'azione all'interno di quella fase.
Tutto ciò ci permette di tenere traccia delle decisioni e delle azioni in modo chiaro e organizzato, facilitando il monitoraggio del progresso del progetto e la responsabilizzazione dei membri del team.
Qui sotto è riportato un esempio di tabella decisioni e azioni:

Codice	Descrizione	Assegnatario
DEC-RTB-001	Scelta del capitolato	Tutti
DEC-PB-003	Delineamento dell'analisi dei requisiti	Tutti
AZ-RTB-001	Brainstorming Analisi dei requisiti	Tutti
AZ-PB-004	Miglioramento struttura dei documenti	Tutti

4.1.3 Diari di bordo

Nell'ambito delle lezioni di **Ingegneria del Software**, il prof. Tullio Vardanega ha organizzato, tramite il calendario delle lezioni, delle date per l'attività denominata "Diario di bordo". Questa attività prevede che per ogni gruppo ci sia un membro designato che rediga un breve resoconto dei progressi fatti fino alla data di presentazione del diario di bordo. Il diario di bordo deve includere:

- Risultati raggiunti e confronto con previsioni all'inizio dello sprint.
- Obiettivi e attività previste per il periodo successivo, fino al diario di bordo seguente.
- Difficoltà riscontrate e problemi ancora da risolvere.

4.1.4 Strumenti a supporto

Per la documentazione del progetto il gruppo ***BYTE HOLDERS*** utilizza i seguenti strumenti:

- **Linguaggio LaTeX**: adottato come standard per la stesura di documentazione tecnica; garantisce un'alta qualità tipografica, uniformità stilistica e una gestione avanzata di riferimenti incrociati e bibliografia, separando il contenuto dalla formattazione.
- **Visual Studio Code**: IDE (Integrated Development Environment) utilizzato come editor principale per la scrittura del codice sorgente LaTeX, scelto per la sua leggerezza, estensibilità e integrazione nativa con i sistemi di controllo versione.
- **LaTeX Workshop**: estensione per Visual Studio Code che automatizza il processo di compilazione e build dei documenti; fornisce funzionalità di linting (controllo sintattico), intellisense e anteprima PDF sincronizzata (SyncTeX) per un feedback visivo immediato durante la stesura.

4.2 Gestione della configurazione

L'obiettivo di questa sezione è definire l'approccio del gruppo ***BYTE HOLDERS*** per la gestione della configurazione della documentazione di progetto, al fine di garantire coerenza, tracciabilità e controllo delle modifiche. In questo caso, come specificato nella sezione degli strumenti di supporto, il gruppo utilizza **GitHub** come piattaforma principale per il versionamento e la gestione della configurazione dei documenti.
In particolare e' stato deciso di realizzare due repository distinte per l'RTB:

- **Repository Documentazione:** Contiene tutti i documenti formali del progetto, come Norme di Progetto, Verbali, Analisi dei Requisiti, Piano di Progetto e altri documenti chiave. Ogni documento è organizzato in cartelle specifiche per facilitarne la navigazione.
- **Repository Codice Sorgente PoC:** Contiene il codice sorgente della Proof of Concept (PoC) sviluppata durante l'RTB. Questa repository include il codice, gli script di build, i file di configurazione e la documentazione tecnica correlata alla PoC.

4.2.1 Produzione e archiviazione

Il gruppo ***BYTE HOLDERS*** adotta una strategia ben definita per la produzione e l'archiviazione della documentazione di progetto, al fine di garantire accessibilità, tracciabilità e sicurezza delle informazioni. La produzione di un documento segue questi passaggi:

1. **Creazione Issue e Branch dedicato:** A seguito dell'apertura di una Issue su GitHub, viene creato un branch di lavorazione dedicato che si distacca dal ramo principale (*main*). La nomenclatura del branch deve seguire rigorosamente la convenzione: `azione_nome_documento_data` (es. `aggiunta_verbale_interno_11_12_2025`).
2. **Assegnazione e Stesura:** Il membro assegnato alla Issue lavora localmente sul file `.tex`. Durante questa fase vengono effettuati commit frequenti sul branch dedicato per salvare i progressi, utilizzando Visual Studio Code e LaTeX Workshop per la scrittura e il controllo sintattico in tempo reale.
3. **Verifica (Pull Request):** Completata la stesura, viene aperta una *Pull Request* su GitHub. In questa fase, i verificatori controllano il contenuto (accuratezza e completezza) e la forma (corretta compilazione LaTeX), segnalando eventuali correzioni direttamente nel codice o nei commenti.
4. **Approvazione e Merge:** Superata la fase di verifica, il Responsabile o un membro designato approva il documento e di seguito la Pull Request ed esegue il *merge* (unione) del branch dedicato nel ramo *main*. A questo punto, il documento aggiornato è considerato ufficiale e viene compilata la versione PDF definitiva per il rilascio.

4.2.2 Versionamento della documentazione

Il versionamento dei documenti è un processo fondamentale per garantire la tracciabilità delle modifiche, la collaborazione ordinata tra i membri del team e la chiara identificazione dello stato corrente di ciascun documento.
Per questo motivo, il gruppo **Byte Holders** adotta un sistema di numerazione semantica e un registro strutturato delle modifiche.

Per la gestione del versionamento, abbiamo deciso di usare uno schema a tre livelli, X.Y.Z, dove:

- **X (Approvazione):** Incrementato per cambiamenti significativi che richiedono l'approvazione formale. E quindi il raggiungimento di un documento stabile e definitivo.
- **Y (Major Update):** Incrementato per l'aggiunta di nuove sezioni o modifiche sostanziali che alterano la struttura complessiva del documento. Il documento non è ancora definitivo.

- **Z (Minor Update):** Incrementato per correzioni minori, come errori di battitura o aggiornamenti di dettaglio.

4.2.3 Visualizzazione della documentazione

Per facilitare la consultazione e la fruizione della documentazione di progetto, il gruppo ***BYTE HOLDERS*** ha implementato un sistema di visualizzazione basato su GitHub Pages. Questo sistema, inserito nella repository dedicata alla documentazione, consente di visualizzare tutta la documentazione presente nel branch main della repository, sotto forma di pdf. Il sito web è suddiviso in 4 sezioni principali:

- **Candidatura:** Contiene tutti i documenti relativi alla fase di candidatura e risposta al capitolato.
- **RTB:** Contiene tutti i documenti prodotti durante la fase di RTB.
- **PB:** Contiene tutti i documenti prodotti durante la fase di PB, alcuni di questi documenti sono versioni aggiornate di documenti gia presenti durante la fase di RTB.
- **Diari di bordo:** Contiene tutti i diari di bordo prodotti durante il corso del progetto.

Oltre alle 4 sezioni elencate sono presenti dettagli relativi ai membri del gruppo e email per eventuali comunicazioni.

4.2.4 Strumenti a supporto

Per la documentazione del progetto il gruppo ***BYTE HOLDERS*** utilizza i seguenti strumenti:

- **GitHub**: Piattaforma per il versionamento e collaborazione, utilizzata per la gestione del codice sorgente e della documentazione..
- **GitHub Pages**: Servizio che permette di pubblicare il contenuto della repository su un sito web statico, utilizzato per visualizzare i documenti in formato PDF, come descritto nel punto 4.2.3.

4.3 Garanzia della Qualità (Quality Assurance)

Il gruppo **Byte Holders** implementa il processo di Garanzia della Qualità in conformità con il processo 6.3 dello standard ISO/IEC 12207:1995. Questo processo ha l'obiettivo di fornire un'assicurazione oggettiva che i prodotti software e i processi del ciclo di vita siano conformi ai requisiti specificati e ai piani stabiliti. Le attività includono:

- **Controllo di Processo:** Verifica periodica che le attività svolte dai membri del team siano conformi alle procedure descritte nel Piano di Progetto e nel Piano di Qualifica.
- **Verifica degli Standard:** Ogni documento e artefatto viene controllato per assicurare il rispetto delle convenzioni di formattazione, nomenclatura e versionamento.
- **Indipendenza della valutazione:** Per garantire l'oggettività, l'attività di **Quality Assurance** è svolta, se possibile, da membri non direttamente coinvolti nella creazione del documento o sezione di codice.

4.4 Verifica

In accordo con il processo 6.4 dello standard ISO/IEC 12207:1995, il processo di Verifica ha lo scopo di determinare se i prodotti di una determinata fase dello sviluppo soddisfano le condizioni imposte all'inizio di quella fase.

Le attività di verifica adottate dal gruppo **Byte Holders** comprendono:

- **Walkthrough e Inspection:** Revisioni interne approfondite sui documenti (Analisi dei Requisiti, Specifica Tecnica) e sul codice sorgente per identificare anomalie, discrepanze o errori logici prima di procedere alla fase successiva.
- **Analisi Statica:** Utilizzo di strumenti automatici per la verifica della qualità del codice (es. controllo dello stile, rilevamento di potenziali bug) senza l'esecuzione del programma.
- **Tracciabilità:** Verifica che ogni requisito software sia correttamente tracciato e coperto nelle fasi di progettazione e implementazione.

4.5 Validazione

Il processo di Validazione, conforme al processo 6.5 dello standard ISO/IEC 12207:1995, assicura che il prodotto finale soddisfi l'uso specifico previsto e le esigenze dell'utente.

Le attività di validazione includono:

- **Test di Sistema:** Esecuzione del software completo per verificare che rispetti i requisiti funzionali e non funzionali specificati nel documento di Analisi dei Requisiti.
- **Test di Accettazione (UAT):** Coinvolgimento del committente per dimostrare che il sistema è pronto per l'uso operativo. Questa fase corrisponde all'acquisizione del feedback formale da parte del proponente.
- **Validazione Documentale:** Conferma che la documentazione prodotta (Manuali Utente, Manuali Sviluppatore) sia comprensibile, completa e utilizzabile dal pubblico di destinazione.

5 Processi Organizzativi

Questa sezione descrive i processi organizzativi, definiti dallo standard ISO/IEC 12207, che regolano le modalità e gli strumenti di coordinamento utilizzati dal gruppo, ovvero come vengono gestite le persone, i ruoli, le comunicazioni e attraverso quali strumenti ciò avviene. Inoltre lo scopo di questa sezione è anche quello di stabilire procedure e metodologie che il team seguirà durante il ciclo di vita del progetto e il processo di sezione è anche quello di stabilire procedure e metodologie che il team seguirà durante il ciclo di vita del progetto e il processo di miglioramento continuo dei processi.

5.1 Gestione dei processi

La Gestione dei Processi ha l'obiettivo di definire le attività da svolgere e i ruoli responsabili della loro esecuzione, favorendo una comunicazione interna ed esterna efficace e garantendo lo svolgimento ordinato ed efficiente delle attività attraverso un'adeguata pianificazione.

5.1.1 Metodologia

Il gruppo ha deciso di adottare un metodo di sviluppo Agile per l'organizzazione e la pianificazione delle attività progettuali.
Le attività vengono organizzate secondo le seguenti modalità operative:

- **Suddivisione in sprint:** il lavoro è diviso in cicli di durata pari a due settimane, ciascuno caratterizzato da obiettivi definiti e misurabili decisi all'inizio dello sprit
- **Gestione delle attività:** i task vengono definiti, pianificati e assegnati tramite le issue di GitHub, organizzate all'interno di una dashboard Kanban, che funge da sistema di issue tracking, supporta la distribuzione delle responsabilità tra i membri del gruppo e aiuta a monitorare l'avanzamento delle attività
- **Revisione iterativa e migliorativa:** al termine di ogni sprint viene svolta un'attività di retrospettiva finalizzata all'analisi dei risultati, all'individuazione di possibili miglioramenti dei processi adottati e alla pianificazione di eventuali azioni correttive o migliorative da applicare nel sprint successivo

L'adozione di tale modalità di gestione consente di assicurare un elevato grado di adattabilità ai cambiamenti dei requisiti e contribuisce al miglioramento continuo dei processi organizzativi del progetto.

5.1.2 Gestione dei ruoli

Ogni membro del gruppo in ogni Sprint può assumere diversi ruoli, in base alle attività che dovrà svolgere durante quel periodo. L'unico ruolo unico all'interno di uno Sprint è quello di Responsabile, che viene assegnato ad un membro diverso ad ogni Sprint. Gli altri ruoli possono essere assegnati a più membri contemporaneamente, in base alle necessità del progetto. I ruoli sono ruotati in modo che per fine del progetto ogni membro abbia ricoperto tutti i ruoli almeno una volta.

5.1.2.1 Responsabile

Il responsabile analizzando lo stato di avanzamento del progetto ha il compito di pianificare ed individuare le attività da svolgere per raggiungere gli obiettivi di progetto e organizzare le risorse in modo da rispettare tempi e costi preventivati.
Ha il compito di verificare che tutti i componenti del gruppo portino a termine le attività a loro assegnate nei tempi previsti.
Deve analizzare i possibili rischi e gestire problemi ed imprevisti in modo da raggiungere gli obiettivi del progetto.
Si occupa dei rapporti e delle comunicazioni con l'azienda proponente.
Le attività che il responsabile svolge sono:

- **Ad inizio sprint:** definire gli obiettivi dello sprint, individuare i possibili rischi, suddividere le attività e assegnarle ai membri del gruppo, controllando che ad ogni membro sia assegnato un nummero coerente di ore per ogni ruolo in base ai compiti che dovrà svolgere

- **Durante lo sprint:** dovrà assicurarsi che le attività vengano svolte nei tempi previsti, monitorare i rischi individuati, cercando di risolvere eventuali problemi che dovessero sorgere, e gestire le comunicazioni con l'azienda proponente
- **A fine sprint:** guidare l'attività di retrospettiva dello sprint, capendo le difficoltà riscontrati e cercando di trovare una possibile soluzione, inoltre verifica che gli obiettivi siano stati raggiunti
- **Redazione piano di progetto:** dovrà redigere e aggiornare il piano di progetto con gli obbiettivi, le ore preventivate, i rischi attesi, le ore effettive, i rischi incontrati e le osservazioni uscite durante la retrospettiva dello sprint, inoltre si occupa della gestione dei rischi
- **Approvazione documenti:** ha il compito di approvare i documenti prodotti dal gruppo prima della loro pubblicazione o di approvare i verbali e verificare il glossario e le norme di progetto

5.1.2.2 Amministratore

L'Amministratore si occupa di gestire e migliorare l'ambiente di lavoro, overro si occupa tenere aggiornati e migliorare l'infrastruttura tecnologica e organizzativa (gli strumenti di sviluppo, verifica e validazione, i sistemi di archiviazione e versionamento della documentazione e del codice).
Le attività che l'amministratore svolge sono:

- **Redazione norme di progetto:** si occupa di redigere e aggiornare le norme di progetto, in modo da garantire che tutti i membri del gruppo seguano le stesse regole e procedure.
- **Redazione piano di qualifica:** si occupa di redigere e aggiornare il piano di qualifica, in modo da garantire che tutti i membri del gruppo seguano le stesse regole e procedure per la verifica e validazione del prodotto
- **Redazione piano dei verbali:** si occupa di redigere i verbali delle riunioni interne ed esterne, in modo da garantire che tutte le decisioni prese durante le riunioni siano documentate e tracciabili
- **Automazione e Miglioramento dei Processi:** si occupa di automatizzare i processi, individuare punti di miglioramento nei processi e mettere in opera risorse per migliorare l'ambiente di lavoro
- **Gestione delle Infrastrutture:** si occupa di gestire l'infrastruttura e gli strumenti utilizzati, mantenere efficiente l'ambiente di sviluppo e fornire strumenti adeguati ai membri del gruppo e gestire errori e segnalazioni di malfunzionamenti con gli strumenti tecnologici

5.1.2.3 Analista

L'analista si occupa della raccolta e dell'analisi dei requisiti, interagendo con il proponente per chiarimenti. Le attività che l'analista svolge sono:

- **Raccolta Requisiti:** Interagire con l'azienda proponente per comprendere e documentare i requisiti funzionali e non funzionali del sistema e la loro importanza (obbbligatori, desiderabili, facoltativi)

- **Redazione Analisi dei Requisiti:** redigere il documento di analisi dei requisiti, valutandone la fattibilità e le implicazione dei requisiti attraverso la stesura dei casi d'uso che li soddisfano assicurando chiarezza e completezza. Valutare la fattibilità tecnica e le implicazioni dei requisiti raccolti Collaborare con il team di verifica per assicurare che i requisiti siano testabili e verificati correttamente
- **Redazione Glossario:** redigere e aggiornare il glossario del progetto, definendo i termini tecnici e le abbreviazioni utilizzate nel contesto del progetto

5.1.2.4 Progettista

Il progettista, partendo dai requisiti definiti nel documento di analisi dei requisiti, definisce il design architetturale del sistema. Le attività che il progettista svolge sono:

- **Progettazione Architetturale:** definire l'architettura di alto livello del sistema, identificando i componenti principali e le loro interazioni
- **Progettazione di Dettaglio:** sviluppare specifiche dettagliate per ciascun componente, incluse le interfacce e i protocolli di comunicazione
- **Redazione Specifiche tecniche:** redigere il documento di specifiche tecniche in cui si riportano le scelte architetturali e tecnologiche (design pattern adottati, schema UML delle classi, tecnologie usate)

5.1.2.5 Sviluppatore

Lo sviluppatore si occupa di sviluppare il software/implementa le funzionalità del software secondo le specifiche progettuali. Le attività che lo sviluppatore svolge sono:

- **Scrittura del Codice:** scrivere codice che segui le specifiche ottenute durante la progettazione seguendo le linee guida (best practices) di codifica stabilite
- **Test delle Unità:** sviluppare e eseguire test unitari per garantire che ogni componente funzioni correttamente in isolamento.
- **Documentazione del Codice:** mantenere una documentazione aggiornata del codice, per facilitarne la comprensione e la manutenzione

5.1.2.6 Verificatore

Il verificatore si assicura che i documenti siano corretti, verifica il corretto funzionamento del software attraverso la scrittura di test e verifica della qualità del prodotto, assicurandosi che soddisfi i requisiti. Le attività che il verificatore svolgere sono:

- **Test del software:** eseguire e implementare test funzionali, di integrazione e di sistema per identificare difetti e garantire la conformità ai requisiti
- **Bug reporting:** registrare e tracciare i difetti riscontrati durante i test
- **Verifica dei Documenti:** verifica i verbali delle riunioni interne ed esterne, il piano di progetto e il piano di qualifica

5.1.3 Riunioni

Le riunioni si dividono in interne ed esterne. A fine delle quali l'amministratore (scelto a giro tra i membri del gruppo che non svolgono attualmente la carica di repsonsabile) redige un verbale in cui vengono documentate le decisioni prese, le azioni assegnate e i punti discussi. Tale verbale viene in seguito verificato da un verificatore e approvato dal responsabile; se si tratta di un verbale esterno viene mandato all'azienda proponente per la firma e l'approvazione.

5.1.3.1 Riunioni interne

Le riunioni interne sono incontri tra i membri del gruppo che avvengono almeno una volta a settimana, solitamente il martedì pomeriggio. Una riunione interna nella maggior parte dei casi si svolge a distanza in modalità virtuale e si svolge nel seguente modo:

- **Apertura:** viene presentando l'ordine del giorno, deciso solitamente durante la riunione precedente e aggiornato durante la settimana in base alle problematiche e i dubbi emersi
- **Aggiornamenti sullo stato del progetto:** ogni membro del gruppo aggiorna gli altri membri riguardo il lavoro svolto, le problematiche e dubbi incontrati
- **Discussione delle problematiche:** si discutono le problematiche e i dubbi emersi e si cercano delle soluzioni condivise
- **Pianificazione delle attività future:** si pianificano le attività da svolgere fino alla prossima riunione, assegnando compiti specifici ai membri del gruppo
- **Chiusura:** si chiude la riunione, riassumendo le decisioni prese e le azioni da intraprendere

5.1.3.2 Riunioni esterne

Le riunioni esterne sono incontri tra i membri del gruppo e l'azienda proponente (VarGroup S.r.l.) che avvengono su richiesta del gruppo o dell'azienda. Una riunione esterna si svolge solitamente in modalità virtuale tramite Microsoft Teams oppure in presenza presso la sede di VarGroup e serve per presentare il lavoro svolto, i dubbi riacontrati e ricevere feedback e consigli tecnici dall'azienda proponente.

5.1.4 Gestione delle comunicazioni

I canali di comunicazione usati per la comunicazione interna tra membri del gruppo e la comunicazione con l'azienda proponente sono diversi.

5.1.4.1 Comunicazioni interne

Per la comunicazione interna tra i membri del gruppo vengono utilizzati principalmente due canali:

- **Discord**: utilizzato per la comunicazione sincrona tra i membri del gruppo (riunioni interne) e la condivisione di materiale di riferimento
- **WhatsApp**: utilizzato per comunicazioni asincrona tra i membri del gruppo, soprattutto per questioni organizzative e logistiche (organizzazzione delle riunioni, aggiornamento sullo stato del verbale, messaggi per esterni).

5.1.4.2 Comunicazioni esterne

Per la comunicazione con l'azienda proponente (VarGroup S.r.l.) vengono utilizzati principalmente due canali:

- **Slack**: utilizzato per la comunicazione asincrona con l'azienda proponente, per concordare riunioni, chiarire dei dubbi, scambiare materiale
- **Microsoft Teams**: utilizzato per le riunioni virtuali (comunicazione sincrona) con l'azienda proponente
- **Email**: utilizzata per inviare documenti ufficiali all'azienda proponente, come i verbali delle riunioni esterne e la documentazione di progetto

5.2 Gestione delle infrastutture e degli strumenti

Per la gestione delle infrastrutture e degli strumenti il gruppo utilizza i seguenti strumenti:

- GitHub
- Git
- Visual Studio Code
- LaTeX Workshop
- Discord
- Slack
- Microsoft Teams
- WhatsApp
- Gmail
- Google Documents
- Google fogli
- Draw.io

5.2.1 GitHub

GitHub è una piattaforma di hosting per repository Git utilizzata per il versionamento del codice sorgente e della documentazione di progetto tramite repository dedicati. Viene inoltre impiegata per la gestione delle attività mediante dashboard Kanban e per l'automazione di processi attraverso GitHub Actions, come per l'automazione del processo di compilazione della documentazione (tali workflow prelevano i file sorgenti in formato LaTeX dalla cartella source e generano i corrispondenti file PDF, che vengono collocati nella cartella principale del repository).
L'utilizzo di GitHub consente di garantire tracciabilità, collaborazione strutturata e integrazione continua.

5.2.2 Git

Git è un sistema di controllo di versione distribuito utilizzato per gestire le modifiche al codice sorgente e alla documentazione.
Permette di mantenere uno storico delle versioni, facilitare il lavoro collaborativo e prevenire conflitti durante lo sviluppo.

5.2.3 Visual Studio Code

Visual Studio Code è l'ambiente di sviluppo integrato (IDE) principale adottato dal gruppo per lo sviluppo del codice sorgente e per la redazione della documentazione in LaTeX.
La sua flessibilità e l'ampia disponibilità di estensioni lo rendono adatto a supportare le diverse attività del progetto.

5.2.4 LaTeX Workshop

LaTeX Workshop è un'estensione di Visual Studio Code utilizzata per la compilazione, il controllo sintattico e l'anteprima dei documenti scritti in LaTeX.
L'uso di tale strumento consente di migliorare la qualità e l'affidabilità della documentazione prodotta.
Per l'uso di questa etensione è necessario prima installare:

- **distribuzione LaTex** installata localmente: TeX Live per sistemi Linux, MiKTeX per sistemi Windows, MacTeX per sistemi macOS
  Al suo interno devono essere disponibili:
  - almeno un compilatore LaTeX (pdflatex, xelatex o lualatex);
  - strumenti ausiliari per la gestione dei riferimenti e degli indici (es. bibtex o biber, makeindex).
- **Perl:** in sistemi Linux e macOS è solitamente già installato invece in Windows va installato Strawberry Perl

5.2.5 Discord

Discord è utilizzato come piattaforma di comunicazione interna tra i membri del gruppo. Supporta comunicazioni sincrone e asincrone, favorendo il coordinamento delle attività e la condivisione di materiali.

5.2.6 Slack

Slack è impiegato come strumento di comunicazione ufficiale con l'azienda proponente. Consente di mantenere uno storico ordinato delle comunicazioni e di separare le interazioni formali da quelle interne al gruppo.

5.2.7 Microsoft Teams

Microsoft Teams è utilizzato per lo svolgimento di riunioni virtuali con l'azienda proponente. La piattaforma consente la condivisione dello schermo e il confronto diretto, facilitando la comunicazione durante gli incontri formali.

5.2.8 WhatsApp

WhatsApp è utilizzato per la comunicazione rapida e informale tra i membri del gruppo, principalmente per coordinare attività operative e gestire comunicazioni urgenti.

5.2.9 Gmail

Gmail è utilizzata per l'invio di comunicazioni e documenti ufficiali all'azienda proponente, garantendo un canale formale e tracciabile per le comunicazioni esterne.

5.2.10 Google Documents

Google Documents è utilizzato per la collaborazione in tempo reale su documenti condivisi tra i membri del gruppo, in particolare nelle fasi di stesura preliminare e revisione collaborativa poichè permette a più persone di lavorare contemporaneamente sullo stesso documento, vedendo in tempo reale le modifiche e integrazioni apportate.

5.2.11 Google Fogli

Google Fogli è utilizzato per la creazione e la gestione di tabelle e grafici relativi alle ore consuntivate e preventive del progetto, nonché per il calcolo del risorse uste e quelle rimaneti. Lo strumento è adottato in quanto consente l'aggiornamento condiviso e immediato dei dati, permette di automatizzare i calcoli dell'uso delle risorse applicando regole predefinite a partire dalle ore registrate e creare automaticamente dei grafici dalle tabelle.

5.2.12 Draw.io

Draw.io è utilizzato per la creazione di diagrammi dei casi d'uso e schemi UML a supporto della documentazione di progetto, facilitando la rappresentazione grafica e la comprensione delle soluzioni progettuali adottate. Lo strumento è usato perchè in esso sono già contenuti tutti gli elementi utili per la costruzione di diagrammi dei casi d'uso e schemi UML.

5.3 Processo di miglioramento

La fase di miglioramento dei processi ha l'obiettivo di garantire l'evoluzione continua del modo di lavorare del gruppo, aumentando l'efficacia e l'efficienza delle attività svolte e riducendo il rischio di errori organizzativi e operativi. Tale processo consente di individuare criticità, valutare le soluzioni adottate e aggiornare, se necessario, le norme e le procedure di progetto.
Il gruppo segue il principio di miglioramento continuo, ovvero il processo di miglioramento è visto come un processo continuo e iterativo basato su revisioni periodiche dei processi e delle pratiche adottate. Il momento principale di revisione avviene durante il momento di retrospettiva alla fine di ogni sprint, in cui si analizzano le difficoltà incontrate, le soluzioni adottate e si identificano possibili miglioramenti.
Il processo adottato segue il ciclo di miglioramento continuo definito dal modello PDSA (Plan-Do-Study-Act) e si articole nelle seguenti 4 fasi:

- **Plan (Pianificazione):** vengono individuate le criticità, inefficienze o opportunità di miglioramento emerse durante lo svolgimento delle attività; vengono analizzate le cause e vengono identificate delle possibili azioni correttive o migliorative.

- **Do (Esecuzione):** le azioni correttive o migliorative individuate vengono implementate e messe in pratica durante lo svolgimento delle attività
- **Study (Analisi):** viene analizzato l'impatto delle azioni implementate, valutando se hanno portato ai miglioramenti attesi e se hanno risolto le criticità individuate
- **Act (Azione):** veine deciso se adottare definitivamente le azioni implementate, modificarle ulteriormente o tornare alla fase di pianificazione per individuare nuove soluzioni

5.4 Formazione

Il processo di formazione serve a garantire che tutti i membri del gruppo:

- acquisiscano le competenze necessarie al fine di svolgere le attività a loro assegnate o necessarie al fine di completare il progetto
- siano allineati sulle tecnologie, processi e strumenti usati

Il processo di formazione è un processo continuo che si attiva ogni qualvolta venga introdotto nel progetto un nuovo strumento, processo o tecnologia. Tale processo di può dovodere nelle seguenti fasi:

- **identificazione dei bisogni formativi:** identificazione degli strumenti, processi, tecnologie o competenze che ci si aspetta dai vari membri del gruppo
- **pianificazione delle attività:** si pianifica l'attività di formazione, assicurandosi che i vari membri abbiano il tempo per svolgerla.

- **svolgimento della formazione:** le attività di formazione possono essere di vario tipo:
  - studio individuale
  - interazioni con l'azienda proponente
  - discussione e scambio delle competenze acquisite tra i vari membri del gruppo
  - condivisione di materiale
  - realizzazione di casi di studio e prototipi di prova, per sperimentare direttamente l'utilizzo delle tecnologie e degli strumenti in modo da acquisire una comprensione pratica e operativa
- **Verifica dell'apprendimento:** l'efficacia della formazione viene valutata attraverso l'osservazione dell'operato dei membri del gruppo e alla qualità dei risultati prodotti; in caso di necessità, vengono pianificate attività ulteriori formative.

In particolare per lo studio delle tecnologie da usare nel progetto il gruppo ha seguito il seguente processo:

- **incontri di formazione:** partecipazione ai 4 incontri di formazione organizzati dall'azienda proponente:
  - **Backend** (9/12/1015): containers e docker, REST, NestJS
  - **AWS** (16/12/2025): panoramica sui vari servizi offerti da AWS
  - **Frontend** (17/12/2025): framework React, uso degli Hooks

  - **Introduzione su AI** (18/12/2025): AI generativa, LLM, tecniche di prompting e customization, agenti (framework langchain o langgraph), demo

- **studio individuale:** studiato individuale delle tecnologie proposte dall'azienda
- **creazione di un prototipo:** apprendimento delle tecnologie tramite la realizzazione di un prototipo più o meno complesso in cui principalmente si esplora l'utilizzo di agenti, il linguaggio TypeScript, il framework React
- **condivisione dei risultati:** discussione tra i membri del gruppo riguardo le competenze apprese, le difficoltà incontrate e gli eventuali dubbi emersi

6 Metriche e Standard di Qualità

Le metriche e i parametri per definire la qualità del prodotto e del processo si basano sugli standard ISO/IEC 9126. Questo standard definisce 6 caratteristiche generali di qualità del software, ognuna delle quali viene misurata attraverso specifiche metriche.

6.1 Funzionalità

La **funzionalità** si riferisce alla capacità del software di soddisfare i requisiti specificati e le esigenze degli utenti. In particolare descrive:

- **Adeguatezza**: il grado in cui le funzioni del software soddisfano i requisiti specificati.
- **Accuratezza**: la capacità del software di fornire risultati corretti e precisi.
- **Interoperabilità**: la capacità del software di interagire con altri sistemi o componenti.
- **Sicurezza**: la capacità del software di proteggere i dati e le risorse da accessi non autorizzati.
- **Conformità**: il grado in cui il software aderisce a standard, regolamenti e linee guida pertinenti.

6.2 Affidabilità

L'**affidabilità** si riferisce alla capacità del software di mantenere le sue prestazioni nel tempo e in condizioni specifiche. In particolare descrive:

- **Maturità**: la capacità del software di evitare guasti in condizioni operative normali.
- **Tolleranza agli errori**: la capacità del software di continuare a funzionare correttamente anche in presenza di errori o guasti.
- **Recuperabilità**: la capacità del software di riprendersi rapidamente dopo un guasto o un errore.
- **Conformità**: il grado in cui il software aderisce a standard e regolamenti pertinenti relativi all'affidabilità.

6.3 Efficienza

L'**efficienza** si riferisce alla capacità del software di utilizzare le risorse in modo ottimale per fornire le prestazioni richieste. In particolare descrive:

- **Comportamento in relazione al tempo**: la capacità del software di rispondere rapidamente alle richieste degli utenti.
- **Utilizzo delle risorse**: la capacità del software di utilizzare in modo efficiente le risorse di sistema, come la memoria e la CPU.
- **Conformità**: il grado in cui il software aderisce a standard e regolamenti pertinenti relativi all'efficienza.

6.4 Usabilità

L'**usabilità** si riferisce alla facilità con cui gli utenti possono utilizzare il software per raggiungere i loro obiettivi. In particolare descrive:

- **Comprensibilità**: la facilità con cui gli utenti possono comprendere come utilizzare il software.
- **Apprendibilità**: la facilità con cui gli utenti possono imparare a utilizzare il software.
- **Operabilità**: la facilità con cui gli utenti possono operare il software per raggiungere i loro obiettivi.
- **Attrattività**: il grado in cui il software è attraente e piacevole da usare.
- **Conformità**: il grado in cui il software aderisce a standard e linee guida pertinenti relativi all'usabilità.

6.5 Manutenibilità

La **manutenibilità** si riferisce alla facilità con cui il software può essere modificato per correggere difetti, migliorare le prestazioni o adattarsi a nuovi requisiti. In particolare descrive:

- **Analizzabilità**: la facilità con cui il software può essere analizzato per identificare difetti o aree di miglioramento.
- **Modificabilità**: la facilità con cui il software può essere modificato per correggere difetti o migliorare le prestazioni.
- **Stabilità**: la capacità del software di resistere a modifiche senza introdurre nuovi difetti.
- **Testabilità**: la facilità con cui il software può essere testato per verificare la correttezza delle modifiche apportate.

6.6 Portabilità

La **portabilità** si riferisce alla facilità con cui il software può essere trasferito da un ambiente a un altro. In particolare descrive:

- **Adattabilità**: la facilità con cui il software può essere adattato a nuovi ambienti o piattaforme.
- **Installabilità**: la facilità con cui il software può essere installato in un nuovo ambiente.

- **Sostituibilità**: la facilità con cui il software può essere sostituito da un altro software con funzionalità simili.
- **Conformità**: il grado in cui il software aderisce a standard e regolamenti pertinenti relativi alla portabilità.

6.7 Nomenclatura delle metriche

La **nomenclatura delle metriche** si riferisce alla definizione e all'uso di termini specifici per descrivere le metriche utilizzate nel processo di sviluppo del software. È fondamentale per garantire una comunicazione chiara e coerente tra i membri del team e con le parti interessate. In particolare è stata adottata la seguente nomenclatura:

M[Tipologia][Codice]

dove:

- **M**: indica che si tratta di una metrica
- **Tipologia**: indica il tipo di metrica, che può essere:
  - **PC** (Processo): per le metriche che misurano la qualità del processo di sviluppo
  - **PD** (Prodotto): per le metriche che misurano la qualità del prodotto software
- **Codice**: un numero progressivo che identifica univocamente la metrica all'interno del tipo

7 Metriche di Qualità del Processo

Le metriche di qualità del processo hanno l'obiettivo di monitorare l'andamento delle attività, l'efficienza economica e temporale, e la stabilità dei requisiti durante l'intero ciclo di vita del software. Di seguito sono elencate le metriche adottate dal gruppo, suddivise per tipologia di processo, con i relativi codici identificativi definiti nel Piano di Qualifica.

7.1 Processi Primari

7.1.1 Fornitura

7.1.1.1 Earned Value (EV)

- **Codice:** MPC-01
- **Formula:**

$$EV = BAC \times \%C$$

  Dove BAC è il Budget At Completion e $\%C$ è la percentuale di lavoro completato.
- **Descrizione:** Rappresenta il valore del lavoro effettivamente completato rispetto al budget totale previsto. È utile per valutare se il lavoro svolto è in linea con le aspettative.

7.1.1.2 Planned Value (PV)

- **Codice:** MPC-02
- **Formula:**
$$PV = BAC \times \%P$$
Dove $\%P$ è la percentuale di lavoro pianificato alla data corrente.
- **Descrizione:** Rappresenta il valore del lavoro che si era pianificato di completare entro la data corrente. Il valore pianificato deve essere inferiore al BAC.

7.1.1.3 Actual Cost (AC)

- **Codice:** MPC-03
- **Formula:**
$$AC = \text{Costo effettivo sostenuto}$$
- **Descrizione:** Rappresenta il costo reale sostenuto per il lavoro svolto fino al momento della misurazione. È fondamentale per valutare se i costi sono in linea con le aspettative.

7.1.1.4 Cost Performance Index (CPI)

- **Codice:** MPC-04
- **Formula:**
$$CPI = \frac{EV}{AC}$$
- **Descrizione:** Rapporto tra il valore del lavoro completato e il costo effettivo sostenuto. Un valore ≥ 1 indica che il progetto sta rispettando il budget.

7.1.1.5 Schedule Performance Index (SPI)

- **Codice:** MPC-05
- **Formula:**
$$SPI = \frac{EV}{PV}$$
- **Descrizione:** Rapporto tra il valore del lavoro completato e il valore del lavoro pianificato. Un valore ≥ 1 indica che il progetto sta rispettando la pianificazione temporale.

7.1.1.6 Estimate At Completion (EAC)

- **Codice:** MPC-06
- **Formula:**
$$EAC = \frac{BAC}{CPI}$$
- **Descrizione:** Stima del costo totale del progetto alla fine dei lavori, basata sulla performance attuale. Se $CPI < 1$, l'EAC sarà maggiore del budget iniziale, indicando un probabile sforamento.

7.1.1.7 Estimate To Complete (ETC)

- **Codice:** MPC-07
- **Formula:**

$$ETC = EAC - AC$$

- **Descrizione:** Stima dei costi necessari per completare le attività rimanenti del progetto.

7.1.1.8 Time Estimate At Completion (TEAC)

- **Codice:** MPC-08
- **Formula:**

$$TEAC = \frac{\text{Durata Pianificata}}{SPI}$$

- **Descrizione:** Proiezione della durata finale del progetto basata sulla performance temporale attuale. Se $SPI < 1$, il TEAC sarà maggiore della durata pianificata.

7.1.2 Sviluppo

7.1.2.1 Requirements Stability Index (RSI)

- **Codice:** MPC-09
- **Formula:**

$$RSI = \left(1 - \frac{R_{var}}{R_{tot}}\right) \times 100$$

  Dove R_{var} è il numero di requisiti aggiunti, modificati o cancellati e R_{tot} è il numero totale dei requisiti iniziali.

- **Descrizione:** Misura la stabilità dei requisiti nel tempo. Variazioni frequenti impattano negativamente sulla progettazione e sullo sviluppo.

7.2 Processi di Supporto

7.2.1 Documentazione

7.2.1.1 Indice di Gulpease

- **Codice:** MPC-10
- **Formula:**

$$G = 89 + \frac{300 \times \text{Frasi} - 10 \times \text{Lettere}}{\text{Parole}}$$

- **Descrizione:** Indice di leggibilità per la lingua italiana. Un valore superiore a 40 indica un testo comprensibile per chi ha un'istruzione di scuola superiore.

7.2.1.2 Correttezza Ortografica

- **Codice:** MPC-11
- **Formula:**

$$C_{ort} = \text{Numero di errori ortografici}$$

- **Descrizione:** Indicatore della qualità formale della documentazione. L'obiettivo è mantenere questo valore a 0 per ogni documento rilasciato.

7.2.2 Verifica

7.2.2.1 Code Coverage

- **Codice:** MPC-12
- **Formula:**

$$Coverage = \frac{\text{Linee di codice testate}}{\text{Linee di codice totali}} \times 100$$

- **Descrizione:** Percentuale di codice sorgente coperto da test automatizzati. Si raccomanda un coverage minimo del 70%.

7.2.2.2 Test Success Rate

- **Codice:** MPC-13
- **Formula:**

$$TSR = \frac{\text{Test superati}}{\text{Test totali}} \times 100$$

- **Descrizione:** Percentuale di test eseguiti con successo. Dovrebbe mantenersi al 100% per garantire che non vi siano regressioni o funzionalità non conformi.

7.2.3 Gestione della Qualità

7.2.3.1 Quality Metrics Satisfied

- **Codice:** MPC-14
- **Formula:**

$$QMS = \frac{\text{Metriche soddisfatte}}{\text{Metriche totali}} \times 100$$

- **Descrizione:** Percentuale di metriche di qualità che rispettano le soglie di accettabilità definite nel Piano di Qualifica.

7.3 Processi Organizzativi

7.3.1 Gestione dei Processi

7.3.1.1 Time Efficiency

- **Codice:** MPC-15

- **Formula:**
$$TE = \frac{\text{Ore produttive}}{\text{Ore totali}} \times 100$$
- **Descrizione:** Valutazione del rapporto tra le ore effettivamente utilizzate per attività produttive e le ore totali dedicate al progetto.

7.3.1.2 Sprint Goal Achievement

- **Codice:** MPC-16
- **Formula:**
$$SGA = \frac{\text{Obiettivi raggiunti}}{\text{Obiettivi pianificati}} \times 100$$
- **Descrizione:** Misura la percentuale di obiettivi dello Sprint (Issue completate o milestone raggiunte) che sono stati soddisfatti al termine dell'iterazione rispetto a quelli pianificati.

8 Metriche di Qualità del Prodotto

8.1 Funzionalità

8.1.1 Requisiti obbbligatori soddisfatti

- **Codice**: MPD01
- **Formula**: Requisiti obbbligatori soddisfatti $= \frac{\text{Requisiti obbligatori soddisfatti}}{\text{Totale requisiti obbligatori}} \times 100$
- **Descrizione**: Questa metrica misura la percentuale di requisiti obbligatori che sono stati soddisfatti dal prodotto software rispetto al totale dei requisiti obbligatori definiti nell'analisi dei requisiti.

8.1.2 Requisiti desiderabili soddisfatti

- **Codice**: MPD02
- **Formula**: Requisiti desiderabili soddisfatti $= \frac{\text{Requisiti desiderabili soddisfatti}}{\text{Totale requisiti desiderabili}} \times 100$
- **Descrizione**: Questa metrica misura la percentuale di requisiti desiderabili che sono stati soddisfatti dal prodotto software rispetto al totale dei requisiti desiderabili definiti nell'analisi dei requisiti.

8.1.3 Requisiti opzionali soddisfatti

- **Codice**: MPD03
- **Formula**: Requisiti opzionali soddisfatti $= \frac{\text{Requisiti opzionali soddisfatti}}{\text{Totale requisiti opzionali}} \times 100$
- **Descrizione**: Questa metrica misura la percentuale di requisiti opzionali che sono stati soddisfatti dal prodotto software rispetto al totale dei requisiti opzionali definiti nell'analisi dei requisiti.

8.2 Affidabilità

8.2.1 Statement Coverage

- **Codice**: MPD04
- **Formula**: Statement Coverage $= \frac{\text{Numero di statement eseguiti}}{\text{Totale numero di statement}} \times 100$
- **Descrizione**: Questa metrica misura la percentuale di istruzioni del codice sorgente che sono state eseguite durante i test rispetto al totale delle istruzioni presenti nel codice.

8.2.2 Branch Coverage

- **Codice**: MPD05
- **Formula**: Branch Coverage $= \frac{\text{Numero di rami eseguiti}}{\text{Totale numero di rami}} \times 100$
- **Descrizione**: Questa metrica misura la percentuale di rami di controllo (come quelli derivanti da istruzioni condizionali) che sono stati eseguiti durante i test rispetto al totale dei rami presenti nel codice.

8.2.3 Condition Coverage

- **Codice**: MPD06
- **Formula**: Condition Coverage $= \frac{\text{Numero di condizioni valutate a vero e falso}}{\text{Totale numero di condizioni}} \times 100$
- **Descrizione**: Questa metrica misura la percentuale di condizioni logiche che sono state valutate sia a vero che a falso durante i test rispetto al totale delle condizioni presenti nel codice.

8.3 Efficienza

8.3.1 Response Time

- **Codice**: MPD07
- **Formula**: Response Time $= \frac{\sum \text{Tempo di risposta}}{\text{Numero di richieste}}$
- **Descrizione**: Questa metrica misura il tempo medio di risposta del sistema alle richieste degli utenti.

8.4 Usabilità

8.4.1 Facilità di utilizzo

- **Codice**: MPD08
- **Formula**: Facilità di utilizzo $= \frac{\sum \text{Punteggio di facilità di utilizzo}}{\text{Numero di utenti}}$
- **Descrizione**: Questa metrica misura il punteggio medio di facilità di utilizzo del software, basato su feedback

8.4.2 Tempo medio di apprendimento

- **Codice**: MPD09
- **Formula**: Tempo medio di apprendimento $= \frac{\sum \text{Tempo di apprendimento}}{\text{Numero di utenti}}$
- **Descrizione**: Questa metrica misura il tempo medio necessario agli utenti per imparare a utilizzare il software in modo efficace.

8.5 Manutenibilità

8.5.1 Accoppiamento Moduli

- **Codice**: MPD10
- **Formula**: Accoppiamento Moduli $= \frac{\sum \text{Numero di dipendenze tra moduli}}{\text{Numero totale di moduli}}$
- **Descrizione**: Questa metrica misura il grado di dipendenza tra i moduli del software, indicando quanto un modulo dipende da altri moduli.

8.5.2 Linee per Metodo

- **Codice**: MPD11
- **Formula**: Linee per Metodo $= \frac{\sum \text{Numero di linee di codice per metodo}}{\text{Numero totale di metodi}}$
- **Descrizione**: Questa metrica misura il numero medio di linee di codice per metodo, indicando la complessità dei metodi nel software.

8.5.3 Parametri per Metodo

- **Codice**: MPD12
- **Formula**: Parametri per Metodo $= \frac{\sum \text{Numero di parametri per metodo}}{\text{Numero totale di metodi}}$
- **Descrizione**: Questa metrica misura il numero medio di parametri per metodo, indicando la complessità e la flessibilità dei metodi nel software.

8.5.4 Attributi per Classe

- **Codice**: MPD13
- **Formula**: Attributi per Classe $= \frac{\sum \text{Numero di attributi per classe}}{\text{Numero totale di classi}}$
- **Descrizione**: Questa metrica misura il numero medio di attributi per classe, indicando la complessità e la capacità di rappresentazione delle classi nel software.

8.5.5 Structure Fan-In

- **Codice**: MPD14
- **Formula**: Structure Fan-In $= \frac{\sum \text{Numero di moduli che utilizzano un modulo}}{\text{Numero totale di moduli}}$

- **Descrizione**: Questa metrica misura il numero medio di moduli che utilizzano un modulo specifico, indicando la centralità e l'importanza di quel modulo nel sistema.

8.5.6 Structure Fan-Out

- **Codice**: MPD15
- **Formula**: Structure Fan-Out $= \frac{\sum \text{Numero di moduli utilizzati da un modulo}}{\text{Numero totale di moduli}}$
- **Descrizione**: Questa metrica misura il numero medio di moduli utilizzati da un modulo specifico, indicando la dipendenza e l'interazione di quel modulo con altri moduli nel sistema.