Norme di Progetto
Versione 1.2.0

Stato	Approvato per RTB
Responsabile	Samuele Esposito
Verificatore	Emanuele Artusi Lorenzo Stefani Samuele Esposito Sara Ferraro
Redattori	Marco Piccoli Matteo Schievano
Distribuzione	*ALimitedGroup*

Descrizione

Questo documento contiene le *Norme di Progetto* seguite da il gruppo *ALimitedGroup* per il progetto C6 proposto dall'azienda **M31**

Registro delle Modifiche

Vers.	Data	Autore	Verificatore	Descrizione
1.2.0	2025-03-11	M. Piccoli	S. Ferraro	Modificato il documento secondo quanto riportato nella valutazione della RTB, aggiungendo per ogni processo la sezione «Strumenti a supporto».
1.1.0	2025-02-28	M. Schievano	E. Artusi	Aggiornata la Sezione 2.1.3 e la Sezione 2.2.4 con le informazioni del Verbale Interno del 24-02-2025.
1.0.0	2025-02-18	M. Schievano	E. Artusi	Correzioni minori. Approvazione da parte di S. Esposito.
0.15.0	2025-02-02	M. Schievano	E. Artusi	Aggiunte Sezione 5, Sezione 6 e Sezione 7.
0.14.0	2025-01-06	M. Schievano	E. Artusi	Aggiunto processo di Validazione (Sezione 3.4.2).
0.13.0	2025-01-04	M. Schievano	E. Artusi	Aggiunto processo di Accertamento Qualità e di Verifica (Sezione 3.3 e Sezione 3.4.1).
0.12.0	2024-12-31	M. Schievano	S. Ferraro	Sistemazione titolazione di alcuni paragrafi non correttamente numerati. Descritto il processo di Miglioramento e quello di Formazione (Sezione 4.3 e Sezione 4.4).
0.11.0	2024-12-16	M. Schievano	S. Ferraro	Proseguo redazione processo di Sviluppo (Sezione 2.2). Proseguo redazione Sezione 4.2.
0.10.0	2024-12-16	M. Piccoli	S. Ferraro	Miglioramento delle tabelle del documento.
0.9.0	2024-12-15	M. Schievano	S. Ferraro	Miglioramento Sezione "Documentazione Fornita" del processo di Fornitura (Sezione 2.1.3). Iniziata redazione processo di Sviluppo (Sezione 2.2).

Vers.	Data	Autore	Verificatore	Descrizione
0.8.0	2024-12-15	M. Schievano	S. Ferraro	Ristrutturazione documento, accorpamento di alcune sezioni.
0.7.0	2024-12-03	M. Piccoli	S. Esposito	Ristrutturata la sezione ruoli (Sezione 4.1.3). Migliorie generali nel documento.
0.6.0	2024-12-02	M. Piccoli	L. Stefani	Ristrutturata ulteriormente la suddivisione dell'indice. Migliorie generali.
0.5.0	2024-11-26	M. Piccoli	L. Stefani	Ristrutturata la suddivisione dell'indice. Migliorie generali.
0.4.0	2024-11-23	M. Schievano	L. Stefani	Aggiunto processo di infrastruttura (Sezione 4.2) e migliorie.
0.3.0	2024-11-22	M. Schievano	L. Stefani	Aggiunta sezione Valutazione della Configurazione (Sezione 3.2.6). Aggiunta introduzione ai processi organizzativi e processo "Gestione dei Processi" (Sezione 4.1).
0.2.0	2024-11-20	M. Schievano	L. Stefani	Aggiunte informazioni sulla Sezione 2.1. Redatta descrizione attività di redazione e manutenzione del processo "Documentazione" (Sezione 3.1). Aggiunte convenzioni usate sulla documentazione e informazioni sulla documentazione prodotta (Sezione 3.1.6.1).Iniziata la redazione delle informazioni sulla "Gestione delle Configurazioni" e parte delle sue attività (Sezione 3.2).
0.1.0	2024-11-15	M. Piccoli	L. Stefani	Prima redazione del documento (Sezione 1).

Indice

Lista delle tabelle

Lista delle immagini

1 - Introduzione

1.1 - Scopo del documento

Questo documento nasce per descrivere il *Way of Working*[G] adottato da parte di *ALimitedGroup* durante lo svolgimento del progetto didattico.

Per realizzare il *Way of Working*[G], i componenti hanno deciso di prendere come riferimento lo standard *ISO/IEC 12207:1995*, che identifica tre tipologie di processi:

- **Primari**: i processi senza il quale un progetto non può definirsi tale;
- **Supporto**: i processi che coadiuvano i processi primari nello svolgimento delle rispettive azioni;
- **Organizzativi**: processi di carattere più generale che aiutano la realizzazione dei progetti.

La stesura di questo documento è incrementale, cioè una stesura passo passo con modifiche, aggiunte e cancellazioni a seguito di miglioramenti del metodo di lavoro.

I membri dell'intero gruppo si impegnano a visionare costantemente questo documento e a rispettare rigorosamente le regole definite in esso, per svolgere il progetto in modo professionale, coerente ed uniforme.

1.2 - Scopo del prodotto

La gestione ottimale dell'inventario in una rete logistica distribuita è fondamentale per garantire la disponibilità continua delle risorse lungo tutta la catena operativa: un contesto in cui magazzini geograficamente distribuiti devono mantenere un flusso costante di materiali e prodotti, richiede un sistema di gestione in grado di minimizzare i tempi di risposta e di ottimizzare la distribuzione delle scorte.
Il capitolato[G] numero C6 di M31 propone di sviluppare un sistema distribuito e scalabile, basato su architettura[G] a microservizi, che favorisca l'interoperabilità tra i diversi magazzini e la centralizzazione delle informazioni in modo efficiente e sicuro, anche in scenari di elevato carico di dati e richieste simultanee.
L'obiettivo che si è posto questo gruppo è realizzare questo progetto entro il **31 Marzo 2025** con un budget a disposizione di: **Euro 12'930**.

1.3 - Glossario

La realizzazione di un sistema *software* richiede, ancora prima della scrittura del codice, un'importante operazione di confronto, analisi e progettazione: per supportare e facilitare il lavoro asincrono tutte le informazioni derivanti da questa attività saranno appositamente documentate.

Per evitare ogni tipo di ambiguità o incomprensioni riguardanti la nomenclatura adottata in tutti i documenti visionabili, viene utilizzato un Glossario in cui è trascritta, per ogni parola, la definizione.
La nomenclatura utilizzata, come descritto nel verbale interno del **26 Novembre 2024**, per segnalare che la definizione di una parola è contenuta nel glossario è la seguente:

`parola`[G]

ALimitedGroup si impegna a visionare il Glossario periodicamente, per permetter la più completa comprensione di ogni tipo di documento pubblicato dal gruppo.

1.4 - Riferimenti

1.4.1 - Riferimenti normativi

- **Capitolato d'appalto C6: Sistema di gestione di un magazzino distribuito**
  https://www.math.unipd.it/~tullio/IS-1/2024/Progetto/C6.pdf
  Ultimo Accesso 6 Febbraio 2025
- **Standard ISO/IEC 9126**
  https://it.wikipedia.org/wiki/ISO/IEC_9126
  Ultimo Accesso 6 Febbraio 2025
- **Standard ISO/IEC 12207:1995**
  https://www.math.unipd.it/~tullio/IS-1/2009/Approfondimenti/ISO_12207-1995.pdf
  Ultimo Accesso 6 Febbraio 2025

1.4.2 - Riferimenti informativi

- **I processi di ciclo di vita del *software***
  https://www.math.unipd.it/~tullio/IS-1/2024/Dispense/T02.pdf
  Ultimo Accesso 6 Febbraio 2025
- **Glossario**:
  https://alimitedgroup.github.io/Glossario.pdf
  Ultimo Accesso 6 Febbraio 2025

2 - Processi Primari

Per sviluppare un buon sistema *software* la sola scrittura di codice e lo svolgimento di alcuni Test[G] non garantisce affatto la realizzazione di un prodotto che possa essere considerato buono, ossia che continuerà ad essere utilizzato da molte persone: per realizzare un prodotto che possa essere in grado di raggiungere tale obiettivo è necessario adottare un modello che fornisca a sua volta dei processi da seguire.

Tra i processi primari definiti dallo *standard ISO/IEC 12207* si distinguono i processi di:

- **Fornitura**
- **Sviluppo**

2.1 - Fornitura

La fornitura è il processo primario adottato dal fornitore del futuro prodotto finale che si occupa di analizzare le azioni da intraprendere per la sua realizzazione.
Questo processo prevede un primo studio dei requisiti che il progetto dovrà, nelle componenti prodotte, soddisfare. Ciò produce il materiale necessario per poter effettuare una contrattazione dei requisiti con il proponente[G], e poter comunicare allo stesso una possibile pianificazione del lavoro da svolgere con probabile data di consegna prevista.

2.1.1 - Strumenti a supporto

Per svolgere le attività previste abbiamo deciso di utilizzare i seguenti strumenti:

- **Google Calendar**: per segnalare tutti gli appuntamenti previsti;
- **GitHub**[G] :per la gestione del *Backlog* e il sistema di *ticketing*, utili per tenere traccia e sotto controllo quanto fatto e quanto è ancora da fare. GitHub[G] offre inoltre una visualizzazione specifica per avere dei diagrammi di Gantt, utili per fini di pianificazione;
- ***Discord***[G] e ***Telegram***: per concordare internamente pianificazioni mediante riunioni interne.

Le comunicazioni verso l'azienda proponente[G] richiedono, invece, l'utilizzo di:

- ***Google Mail*** per le comunicazioni scritte;
- ***Microsoft Teams*** per le riunioni totalmente e parzialmente svolte in remoto con M31

2.1.2 - Attività previste

La fornitura prevede varie attività, qui di seguito descritte:

- **Inizializzazione**: questa prevede l'analisi, da parte del fornitore, delle richieste da parte del proponente[G], tenendo in considerazione eventuali vincoli organizzativi o di altra natura. È Il momento in cui il fornitore valuta la capacità di realizzare quanto proposto, determinando gli eventuali requisiti da contrattare con il proponente[G]
- **Preparazione risposte**: questa attività prevede la realizzazione di una contro-proposta per il proponente[G] che tenga conto di quanto derivato dall'attività di inizializzazione;
- **Contrattazione**: è l'attività che prevede un colloquio con il proponente[G] durante il quale verranno presentate al proponente[G] le risposte precedentemente realizzate, con l'obiettivo di giungere alla formalizzazione di un contratto;
- **Pianificazione**: il fornitore, stabiliti i requisiti finali, deve adesso stabilire un'organizzazione e un metodo di lavoro in grado di assicurare la qualità del sistema da realizzare, scegliendo, qualora non lo sia da contratto, il modello di ciclo di vita del *software* da seguire. La pianificazione include anche l'individuazione delle risorse e le

tecnologie necessarie allo sviluppo, considerando anche i potenziali rischi ad esse associate;
- **Esecuzione e controllo**: Il fornitore deve, messo a documenti la pianificazione, realizzare quanto stabilito, monitorando nel frattempo la qualità di quanto prodotto e il progresso raggiunto;
- **Revisione e valutazione**: il fornitore deve, anche durante lo sviluppo, tenersi in contatto con il proponente[G]: questo è necessario per avere feedback su quanto realizzato, con annessa quindi una valutazione sullo stato di lavoro;
- **Consegna e completamento**: il fornitore, completato il progetto, deve fornire quanto prodotto al proponente[G], garantendogli supporto.

2.1.3 - Documentazione Fornita

Vengono ora elencati tutti i documenti che *ALimitedGroup* consegnerà all'azienda M31 e ai committenti *Prof. Tullio Vardanega* e *Prof. Riccardo Cardin*.

2.1.3.1 - Analisi dei Requisiti

L'**Analisi dei Requisiti[G] v1.2.0** è il documento in cui *ALimitedGroup* descriverà tutti i requisiti obbligatori, desiderabili e opzionali previsti dal progetto. Nello specifico il documento contiene, dopo una breve introduzione, i Casi d'Uso rilevati con i relativi attori, i requisiti legati ai Casi d'Uso e al capitolato[G] e, infine, informazioni utili al loro tracciamento.

Analisi dei Requisiti	
Redattore	Analista
Destinatari	M31, *ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 1: Proprietà del documento «Analisi dei Requisiti»

2.1.3.2 - Dichiarazione degli Impegni

La **Dichiarazione degli Impegni** è il documento in cui *ALimitedGroup* ha stimato i costi del progetto, dall'impegno orario per persona e per ruolo, al costo complessivo del progetto e dei ruoli che i componenti del gruppo ricopriranno.

Dichiarazione degli Impegni	
Redattore	Responsabile
Destinatari	M31, *ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 2: Proprietà del documento «Dichiarazione degli Impegni»

2.1.3.3 - Glossario

I componenti di *ALimitedGroup* ritengono necessario avere un documento di facile consultazione per ricordare, in maniera rapida ed efficace, i termini utilizzati nei vari ambiti di realizzazione del progetto di Ingegneria del *software*.

Per questo motivo, il gruppo ha prodotto un **Glossario**, rivelatosi molto utile nel corso del tempo.

Glossario	
Redattore	Amministratore
Destinatari	*ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Interno

Tabella 3: Proprietà del documento «Glossario»

2.1.3.4 - Lettera di Candidatura

La **Lettera di Candidatura** è il documento con cui *ALimitedGroup* ha ufficialmente presentato la sua candidatura al capitolato[G] proposto dall'azienda M31. Essa contiene, in forma riassunta, i punti chiave che hanno spinto i componenti del gruppo a scegliere questo capitolato[G], nonché le informazioni generali riguardanti i *repository*[G] del gruppo contenente tutti i documenti relativi al progetto.

Lettera di Candidatura	
Redattore	Responsabile
Destinatari	M31, *ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 4: Proprietà del documento «Lettera di Candidatura»

2.1.3.5 - Lettera di Presentazione

Di due tipologie:

- La Lettera di presentazione per la ***Requirements and Technology Baseline[G] (RTB)***
- La lettera di presentazione per la ***Product Baseline[G] (PB)***

Lo scopo del documento è quello di presentare formalmente la candidatura di *ALimitedGroup* alle rispettive *Baseline*[G].

Lettera di Presentazione	
Redattore	Responsabile
Destinatari	M31, *ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 5: Proprietà del documento «Lettera di Presentazione»

2.1.3.6 - Norme di Progetto

Il presente documento: stabilisce il *Way of Working*[G] e le pratiche di sviluppo adottate dal gruppo.

Norme di Progetto	
Redattore	Amministratore
Destinatari	*ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Interno

Tabella 6: Proprietà del documento «Norme di Progetto»

2.1.3.7 - Piano di Progetto

Il **Piano di Progetto[G] v1.1.0** è un documento che elenca in maniera ordinata tutte le informazioni riguardanti la pianificazione del gruppo. Conterrà dunque le attività da svolgere di *sprint*[G] in *sprint*[G], nonché l'analisi dei rischi associati a ciascuna attività.
Dati i requisiti del progetto, questo documento avrà anche il compito di aggiornare, volta per volta, il consumo orario previsto ed effettivo e dunque, conseguentemente a questo, anche il costo derivato.

Piano di Progetto	
Redattore	Responsabile
Destinatari	M31, *ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 7: Proprietà del documento «Piano di Progetto»

2.1.3.8 - Piano di Qualifica

Descrive i metodi di qualifica (Verifica e Validazione[G]) adottate da *ALimitedGroup*, nonché i Test[G] effettuati sul prodotto e i rispettivi esiti.

Piano di Qualifica	
Redattore	Amministratore
Destinatari	M31, *ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 8: Proprietà del documento «Piano di Qualifica»

2.1.3.9 - Specifica Tecnica

Lo scopo della **Specifica Tecnica[G]** è quello di descrivere le caratteristiche progettuali delle componenti del Sistema sviluppato.

Specifica Tecnica	
Redattore	Progettisti
Destinatari	M31, *ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 9: Proprietà del documento «Manuale Utente»

2.1.3.10 - Valutazione dei Capitolati

La **Valutazione dei Capitolati** è un documento pubblicato da *ALimitedGroup* in data 2024-10-31.
Quest'ultimo fornisce, per ogni capitolato[G], una dettagliata analisi evidenziando i suoi punti di forza e le sue criticità. È suddiviso nelle seguenti sezioni:

- **Panoramica**: che indica l'azienda proponente[G], il nome del capitolato[G] e delle informazioni generali sul prodotto da realizzare;
- **Scopo**: indica che vantaggi porta la realizzazione del prodotto;
- **Punti di forza**
- **Criticità evidenziate**
- **Conclusioni**: motivazioni del gruppo sulla scelta/non scelta del capitolato[G].

Valutazione dei Capitolati	
Redattore	Responsabile
Destinatari	*ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 10: Proprietà del documento «Valutazione dei Capitolati»

2.1.3.11 - Verbale Esterno

Verbale delle riunioni svolte con persone esterne al gruppo.

Verbale Esterno	
Redattore	Amministratore
Destinatari	M31, *ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Esterno

Tabella 11: Proprietà del documento «Verbale Esterno»

2.1.3.12 - Verbale Interno

Verbali delle riunioni svolte senza la presenza di persone esterne al gruppo.

Verbale Interno	
Redattore	Amministratore
Destinatari	*ALimitedGroup*, Prof. Vardanega, Prof. Cardin
Uso	Interno

Tabella 12: Proprietà del documento «Verbale Interno»

2.2 - Sviluppo

Il **Processo di Sviluppo** stabilisce le attività che hanno come scopo quello di Analisi dei Requisiti[G], la progettazione, la codifica del *software*, l'installazione e l'accettazione di quanto prodotto.

2.2.1 - Strumenti a supporto

Per le attività previste nel processo di Sviluppo, abbiamo deciso di utilizzare i seguenti strumenti:

- **GO** e **Visual Studio Code**: per la codifica del *software* e dei microservizi;
- **Draw.io**: per realizzare i diagrammi *UML*[G] degli *use-case* individuati in fase di analisi, che verranno pubblicati nel documento di Analisi dei Requisiti[G], e per progettare l'architettura di tutto l'ecosistema riguardante il *software* e i microservizi che andremo a realizzare.

2.2.2 - Attività previste

Le attività previste dal **Processo di Sviluppo** in base allo standard *ISO/IEC 12207:1995* sono le seguenti:

- **Implementazione del processo**: ovvero la scelta del **Ciclo di Vita** del *software* più appropriato in base allo scopo, l'importanza e la complessità del progetto;

- **Analisi dei Requisiti[G]**: consiste nell'identificazione e nella definizione delle necessità dell'utente finale in relazione alle funzionalità[G] che il *software* deve offrire. Un'Analisi dei Requisiti[G] completa deve descrivere le funzionalità[G] del Sistema, i bisogni degli utilizzatori finali e vincoli imposti dal proponente[G]
- **Progettazione dell'architettura**: ovvero l'individuazione degli elementi *Hardware* e *software* del prodotto finale, affinché tutti i requisiti individuati siano soddisfatti (a questo proposito, fondamentale è il tracciamento dei requisiti stessi);
- **Analisi dei Requisiti[G] *software***: ovvero l'analisi del modo in cui il *software* soddisfa i requisiti lato utente. Deve includere anche le caratteristiche di qualità: caratteristiche funzionali (includendo anche eventuali requisiti prestazionali), le interfacce di ogni elemento *software* e requisiti di sicurezza;
- **Progettazione dell'architettura *software***: consiste nel definire le diverse componenti del Sistema e il loro funzionamento, ponendo l'attenzione sulla struttura generale, non nel dettaglio implementativo;
- **Progettazione in dettaglio del *software***: ovvero la progettazione in dettaglio delle singole componenti *software*, fino ad individuare le singole unità di ciascuna;
- **Codifica e *testing* del *software***: ovvero la produzione di tutte le unità di tutte le componenti individuate. Ciascuna di queste parti dovrà essere adeguatamente testata per assicurare il suo corretto funzionamento;
- **Integrazione del *software***: ovvero l'integrazione delle varie parti di una componente nella sua componente completa. Essenziali sono i Test[G] per assicurare il corretto funzionamento;
- **Test di qualifica del *software***: ovvero la realizzazione di appositi Test[G] per assicurare la conformità del *software* agli obiettivi di qualità attesi;
- **Integrazione del Sistema**: ovvero l'integrazione di tutte le componenti realizzati nel Sistema finale;
- **Test di qualifica del Sistema**: ovvero il Test[G] dell'intero Sistema per assicurarne il corretto funzionamento;
- **Installazione del *software***: ovvero la fornitura di quanto realizzato al cliente finale nell'ambiente concordato;
- **Supporto per approvazione[G] del *software***: ovvero l'attività per cui il fornitore dovrà supportare l'utilizzatore finale al fine di comprendere se nell'effettivo tutti i requisiti richiesti siano effettivamente soddisfatti.

Apportando lo scopo del processo alle *Baseline*[G] previste dal progetto (*Requirements and Technology Baseline*[G] *(RTB)* e *Product Baseline*[G] *(PB)*), riteniamo particolarmente di interesse le attività di **Analisi dei Requisiti[G]** e **Codifica** per la *RTB*[G] mentre **Progettazione dell'architettura**, **Progettazione dell'architettura *software*** e **Codifica** per la *PB*[G] : *ALimitedGroup* ha quindi deciso di discutere maggiormente nel dettaglio queste attività, aggiungendo e aggiornando le loro sezioni nel momento opportuno (nel dettaglio, le attività di **Progettazione dell'architettura** e **Progettazione dell'architettura *software*** saranno inserite a *RTB*[G] raggiunta).

2.2.3 - Analisi dei Requisiti

L'**Analisi dei Requisiti[G]** è tra le attività cardine della *Requirements and Technology Baseline*[G] *(RTB)* e ha come fine l'individuazione di tutti i requisiti che il Sistema da noi sviluppato dovrà soddisfare. Tale analisi, reperibile nell'apposito documento visualizzabile su **Analisi dei Requisiti[G] v1.2.0**, espone nel dettaglio tutte le informazioni necessarie, che saranno poi

fondamentali per supportare il lavoro dei progettisti e dei programmatori nelle rispettive attività di progettazione dell'architettura e codifica della stessa (fondamentale riferimento sarà il **Piano di Qualifica[G] v1.1.0** che, contenendo i Test[G] e il loro stato alla **Sezione 4**, permetterà di determinare quali requisiti sono soddisfatti e quanti ancora non lo sono).

In particolar modo, il documento raggruppa tutti i Casi d'Uso rilevati e i requisiti ad essi associati. Per una più rapida consultazione sarà ora discussa la nomenclatura nel dettaglio.

2.2.3.1 - Casi d'Uso

Per i casi d'Uso viene utilizzata la seguente nomenclatura:

`UCPrincipale.Secondario`

dove:

- **UC** è un acronimo stante per **U**se **C**ase, ovvero la traduzione inglese di Caso d'Uso[G]
- **Principale** è un numero crescente stante ad indicare che il Caso d'Uso[G] non è correlato ad altri Casi d'Uso oppure è utilizzato da più Casi d'Uso mediante inclusione.
- **Secondario** è un numero crescente stante ad indicare che il Caso d'Uso[G] è correlato esclusivamente al Caso d'Uso[G] identificato dal valore **Principale**

Si noti che **Principale** è **univoco** tra tutti i Casi d'Uso, dunque non può sussistere l'esistenza di due Casi d'Uso aventi stesso valore **Principale** mentre è possibile che uno stesso valore **Secondario** sia ripetuto, ma **mai** per la stesso valore **Principale**.

Si noti, inoltre, che un Caso d'Uso[G] **Secondario** può avere a sua volta delle inclusioni: in tal caso la nomenclatura **Principale**.**Secondario** sarà la parte **Principale** di tale inclusione e seguirà le regole sopra esposte.

La nomenclatura utilizzata è volta ad assicurare l'unicità di ogni Caso d'Uso[G].

Ogni Caso d'Uso[G] è inoltre accompagnato da un nome che ne riassume lo scopo e una descrizione: per maggiori informazioni si consiglia la lettura della parte introduttiva del documento di **Analisi dei Requisiti[G] v1.2.0** (**Sezione 2.1**).

2.2.3.2 - Requisiti

Identificati i Casi d'Uso, il documento di **Analisi dei Requisiti[G] v1.2.0** si concentra sull'individuazione dei requisiti deducibili dagli stessi e dal capitolato[G] **(Sezione 3)**. I requisiti sono anch'essi identificati da una nomenclatura:

`R-Numero-Tipologia-Priorità`

dove:

- **R** è per abbreviare la parola **R**equisito;
- **Numero** è un valore univoco che identifica il requisito;
- **Tipologia** indica il tipo di requisito. I possibili valori sono:
  - **F** per **F**unzionale;
  - **Q** per **Q**ualità;
  - **V** per **V**incolo;
- **Priorità** indica la priorità di sviluppo che quel requisito assume. I valori possibili sono:
  - **Ob** per **Ob**bligatorio;
  - **De** per **De**siderabile;
  - **Op** per **Op**zionale;

Per maggiori informazioni circa la tipologia e la priorità si consiglia la lettura della parte introduttiva del documento di **Analisi dei Requisiti[G] v1.2.0 (Sezione 1.1)**.

2.2.4 - Codifica

2.2.4.1 - Scopo

La codifica svolta dai programmatori di *ALimitedGroup* ha come scopo la realizzazione di quanto progettato dagli analisti e dai progettisti.

Questa sezione descrive le regole e le convenzioni che i programmatori devono seguire per garantire la qualità del codice prodotto.

2.2.4.2 - Stile di codifica: Golang

2.2.4.2.1 - Formattazione del codice

La struttura di un *file* sorgente Golang deve seguire lo standard prodotto dall'eseguibile **gofmt**, un tool sviluppato dal team del linguaggio Golang.

Tale eseguibile dev'essere eseguito in automatico sia nell'ambiente locale di uno sviluppatore (possibilmente dopo l'azione di salvataggio del *file*) sia in ambiente di CI/CD tramite GitHub[G] Action[G] ,le quali bloccheranno l'errata introduzione di codice non formattato secondo lo standard all'interno del *branch*[G] principale.

2.2.4.2.2 - Buone pratiche di programmazione

- **Variabili** globali: evitare l'uso di variabili globali dove possibile;
- **Funzioni**: evitare funzioni troppo lunghe, preferire funzioni brevi e ben definite;
- **Lingua**: usare la lingua inglese per tutti i costrutti e commenti;
- **Nomi**: usare nomi significativi per variabili, funzioni, metodi e strutture;
- **Commenti**: commentare il codice in modo chiaro e conciso, ogni funzione deve avere un commento che ne descriva lo scopo e i suoi argomenti;
- **Indentazione**: usare quattro spazi per l'indentazione;
- **Istruzioni per linea di codice**: evitare più istruzioni su una linea in quanto rendono difficile la lettura del codice.

Sarà inoltre necessario sfruttare il *framework* **fx** per applicare il *pattern* della *Dependency Injection*[G].

2.2.4.2.3 - Convenzioni sulle nomenclature e sulla posizione dei file

Considerando la necessità di sviluppare microservizi ad architettura[G] esagonale e dato l'alto numero di componenti, si adottano le seguenti convenzioni di nomenclatura:

- **Package**: ogni cartella deve contenere oggetti esistenti all'interno dello stesso *package*, che deve essere diverso per ogni cartella;
- **Interfacce**: il nome delle interfacce devono sempre essere precedute dalla lettera **I** (ad esempio, `IUseCase`);
- **Componenti dell'architettura esagonale**: le tre componenti principali dell'architettura esagonale devono avere, nella parte finale del nome, i seguenti nomi:
  - **Controller**, per indicare la componente responsabile[G] dell'*Application Logic*;
  - **Service**, per indicare la componente responsabile[G] della *Business Logic*;
  - **Repository**[G] ,per indicare la componente responsabile[G] della *Persistance Logic*;
  - **UseCase**, per indicare le interfacce con cui il *Controller* comunica con il *Service*;
  - **Port**, per indicare le interfacce con cui il *Service* comunica con il *Repository*[G] .

- **Oggetti dell'architettura esagonale**: è preferibile, nella misura possibile, utilizzare, nella parte finale del nome degli oggetti, le sigle **DTO** e **CMD** per indicare gli oggetti dell'*Application Logic* e della *Business Logic* rispettivamente

In merito al posizionamento dei file, è necessario mantenere le seguenti convenzioni:

- **Interfacce**: devono essere inserite in file separati rispetto alla restante parte del codice sorgente;
- **Strutture che implementano interfacce**: devono essere poste su file separato rispetto al restante codice sorgente;
- **Strutture che rappresentano attributi di altre strutture**: devono essere poste su un file separato rispetto al restante codice sorgente;
- **Strutture utilizzate dai *Controller***: devono essere poste all'interno di una cartella comune.

3 - Processi di Supporto

Tra i processi di supporto utilizzati nel progetto distinguiamo:

- **Documentazione**
- **Gestione delle configurazioni**
- **Accertamento Qualità**
- **Qualifica**, formata a sua volta da:
  - **Verifica**[G]
  - **Validazione**[G]

Saranno ora descritte nel dettaglio le attività previste.

3.1 - Documentazione

Il processo di documentazione è parte fondamentale di tutti i processi primari: il prodotto di tale processo è infatti essenziale per tenere traccia delle decisioni intraprese e per favorire il lavoro asincrono, molto più produttivo di quello sincrono nel nostro ambito.

Nel dettaglio, il processo di documentazione si occupa della registrazione delle informazioni derivanti da un processo o da un'attività nel ciclo di vita.
Riguarda dunque l'insieme delle attività che pianificano, progettano, sviluppano, producono, modificano, distribuiscono e mantengono i documenti necessari a tutti gli interessati.

3.1.1 - Strumenti a supporto

Per redigere la documentazione il gruppo utilizza due strumenti in particolare:

- ***Typst***: si tratta di un linguaggio *mark-up* molto recente ma che si sta rivelando una valida alternativa a **LaTeX**[G] per la redazione di documenti a carattere scientifico e non solo. *Typst*[G] permette la realizzazione di una anteprima istantanea, senza necessità di compilare ogni volta il documento, inoltre la sintassi è molto vicina ai linguaggi di programmazione. *ALimitedGroup* lo utilizza in tutti i documenti. Nello specifico, la redazione dei documenti sfrutta funzioni di *Typst*[G] appositamente implementate dai componenti di *ALimitedGroup* presenti all'interno della cartella *lib* del *repository*[G].
  La descrizione dei vari template nel dettaglio è consultabile nella Sezione 4.2.3.10, in questa sezione saranno invece descritte le strutture dei vari documenti.
- **GitHub**[G] :il gruppo ha deciso di utilizzare il sistema di *ticketing* e le *pull request*[G] di *GitHub*[G] per permettere la redazione, Verifica[G] e approvazione[G] di tutti i documenti. Per ulteriori dettagli consultare la Sezione 4.2.

3.1.2 - Attività previste

Il processo di documentazione è un processo assai delicato, e molto importante. Le attività cardine di questo processo sono due:

- **Produzione**: l'attività che stabilisce con quale metodo il documento deve essere redatto, per ulteriori informazioni vedere la Sezione 3.1.6;
- **Manutenzione**[G]: l'attività che definisce come un documento viene, eventualmente, modificato; per ulteriori informazioni vedere la Sezione 3.1.7.

3.1.3 - Verbali

La redazione di un verbale sfrutta il *template* nel *file* *verbale.typ*

Generalmente un verbale possiede questa struttura iniziale:

- **Pagina di copertina**, con al suo interno:

    - Logo del gruppo;
    - Tipo di verbale (interno o esterno) con annessa data della riunione verbalizzata;
    - Stato del documento;
    - Persone presenti alla riunione;
    - Distribuzione (ossia, i destinatari);
    - Ordine[G] del giorno (in formato riassuntivo);
- Allegato con **tabella delle versioni**;
- **Indice del documento**, generato automaticamente da *Typst*[G].

Un verbale successivamente prevede sempre una prima sezione con alcune informazioni generali tra cui:

- **Modalità**, distinta tra «in presenza» o «virtuale» (se virtuale si intende avvenuta sulla piattaforma *Discord*[G] salvo diversamente specificato);
- **Data della riunione**;
- **Orario di inizio**;
- **Orario di fine**.

cui seguirà, in elenco, l'ordine del giorno nei vari punti.

La seconda sezione, che segue quanto appena scritto, esplicita quanto discusso per ogni punto dell'ordine del giorno.

Segue quindi la penultima sezione, denominata «Esiti della riunione», che riassume quanto concordato.
L'ultima sezione è dedicata alla *tabella delle decisioni e delle azioni*, che riassume in modo strutturato tutte le decisioni prese e le azioni concordate.
Queste informazioni vengono integrate e tracciate nel *sistema di ticketing*.

La tabella utilizza, per tracciare le decisioni, la seguente nomenclatura:

```
ID#
```

- **ID** è un codice univoco che rappresenta la decisione o l'azione. Questo valore può assumere significati diversi:
    - **DI**[G] ovvero **Decisione Interna**[G]: viene utilizzato per indicare una decisione intrapresa con effetto immediato: potrebbe, per questo motivo, non avere un'*issue* associata;
    - **AP**[G] ovvero **Attività Passata**[G]: viene utilizzato per segnalare un'attività (dunque una decisione che *dovrebbe avere* associata una *issue*[G]) ma intrapresa prima che il gruppo decidesse di utilizzare il sistema di ticketing (o che, per errore, non è stata associata ad una *issue*[G] prima di procedere ad eventuali modifiche);
    - **DOCS**, indica una decisione che ha un'*issue* associata nel *repository*[G] dei documenti;
    - **POC**, indica una decisione che ha un'*issue* associata nel *repository*[G] del *Proof Of Concept*;
    - **MVP**[G] ,indica una decisione che ha un'*issue* associata nel *repository*[G] del *Minimum Viable Product*[G].
- **#** è un numero crescente da 1, univoco a seconda dell'ID.

La tabella è creata tramite la funzione «tabella-decisioni».

In ultima istanza, i **verbali esterni** devono includere una sezione dedicata all'approvazione esterna.
Ogni pagina, ad eccezione della copertina, deve riportare il numero e un *header* con:

- **Nome del gruppo**;
- **Tipo verbale**;
- **Data della riunione**.

3.1.4 - Diari di bordo

Nell'ambito del progetto di Ingegneria del *software*, il *Prof. Tullio Vardanega* ha previsto delle attività volte ad accertare, in maniera condivisa, lo stato di progresso per ogni singolo gruppo iscritto al I lotto.

In genere, la struttura di un diario di bordo è composta dalle seguenti parti:

- **Risultati raggiunti e confronto con le previsioni**, che elenca quanto abbiamo fatto nel periodo corrente;
- **Obiettivi e attività per il periodo successivo**, che elenca le attività future da completare;
- **Difficoltà riscontrate e questioni aperte**, che ci permette di esporre le problematiche che abbiamo affrontato e quelle su cui ancora nutriamo dei dubbi.

3.1.5 - Altri documenti

Per tutti gli altri documenti la struttura è composta da:

- **Pagina di copertina**, con:
  - Nome del documento;
  - Versione;
  - Stato del documento;
  - Data di ultima modifica;
  - Distribuzione (destinatari);
- **Registro delle modifiche**, che viene realizzata automaticamente invocando la funzione per la copertina;
- **Indice dei contenuti**, viene realizzato automaticamente con la stessa funziona menzionata sopra;
- **Elenchi di tabelle e figure**, se presenti nel documento;
- **Corpo del documento**, con sezioni e sotto-sezioni che vengono scritte manualmente.

Ogni pagina, ad eccezione della copertina, dovrà inoltre contenere:

- ***Header*** con:
  - **Nome del gruppo**;
  - **Titolo del documento**;
  - **Versione del documento**;
- **Numero di pagina**.

3.1.6 - Produzione

La produzione di un documento segue questi passaggi:

- **Creazione della issue[G] e del branch[G] secondario**: innanzitutto viene aperta una issue[G] su GitHub[G], essenziale per tenere traccia dei compiti da svolgere, in seguito viene creato un branch[G] secondario rispetto al principale nel sistema di versioning, per poter eseguire modifiche senza intaccare il ramo stabile. La denominazione del *branch*[G] e delle *issue*[G] segue una nomenclatura specifica decisa nel **verbale interno** del **18 Novembre 2024** e riportata nella Sezione 4.2.3.3.

- **Assegnazione della issue[G] e inizio stesura**: in base al ruolo (vedi la Sezione 2.1.3 per maggiori informazioni) viene assegnata l'issue di redazione e si inizia la stesura del documento nel branch[G] dedicato.
- **Verifica**[G] : terminata la redazione, il redattore[G] apre una *pull request*[G] su GitHub[G] per chiedere, ad un verificatore[G], la Verifica[G] del documento.
  L'esito dell'attività potrà essere positivo oppure negativo.
  Se negativo, il verificatore[G] allega un elenco delle modifiche da apportare ai redattori.

A questo punto il processo diviene differente per i verbali e i rimanenti documenti:

- **Approvazione e pubblicazione (verbali)**: A modifiche ultimate o solo in caso di Verifica[G] positiva, il verificatore[G] richiede al Responsabile[G] di approvare il documento.

Il Responsabile[G], una volta approvato il documento, risolve la pull request[G] e procede quindi al merge del branch[G] secondario in quello principale. L'azione scatenerà l'avvio di una ***GitHub Action*** che aggiornerà il sito web del progetto con i nuovi documenti approvati.
Dal momento che un verbale descrive un evento trascorso, è poco probabile debba essere cambiato in futuro, perciò il *ALimitedGroup* ha deciso che i verbali saranno pubblicati solamente quando approvati.

- **Pubblicazione (altri documenti)**: una volta verificato, il documento può essere pubblicato nella sua versione attuale completando il *merge* e risolvendo la *Pull Request*[G].
  Questo processo differisce da quello dei verbali, in quanto questi ultimi possono essere integrati nel branch[G] principale (main) solo dopo l'approvazione formale.
  Al contrario, gli altri documenti, essendo di consultazione più frequente, possono essere pubblicati nel branch[G] principale semplicemente dopo la Verifica[G], per garantire l'accesso alla versione più aggiornata.
- **Approvazione (altri documenti)**: l'approvazione di un documento implica il raggiungimento di determinati standard di qualità. Similmente a quanto avviene per i verbali, sarà il Responsabile[G] ad approvare il documento a seguito di opportuna *Pull Request*[G].

Nell'ambito della redazione dei documenti vengono utilizzate alcune convenzioni, definite nella prossima sezione.

3.1.6.1 - Denominazione e datazione dei documenti

Come osservato nel **verbale interno** del **4 Novembre 2024**, **Sezione 2.3.1** e in seguito estesa a tutti i documenti, la denominazione dei file seguirà la seguente sintassi:

`TIPO AAAA-MM-GG # VERSIONE`

dove:

- **TIPO** è una sigla che identifica il tipo di documento. Le possibili scelte sono:
  - **VI** per **V**erbale **I**nterno;
  - **VE** per **V**erbale **E**sterno;
  - **DB** per **D**iario di **B**ordo;
  - **NP** per **N**orme di **P**rogetto;
  - **PP** per **P**iano di **P**rogetto;
  - **AR** per **A**nalisi dei **R**equisiti;
  - **PQ** per **P**iano di **Q**ualifica;

    - **MU** per **M**anuale **U**tente;
    - **SP** per **S**pecifica **T**ecnica.
- **AAAA-MM-GG** indica la data in formato anno-mese-giorno (con due cifre per giorno e mese e 4 cifre per l'anno);
- **#** è un modificatore, ossia un numero crescente a partire da 2 per indicare eventuali documenti dello stesso tipo redatti lo stesso giorno. Viene omesso sempre per il primo documento in ordine[G] cronologico;
- **VERSIONE** indica la versione corrente del documento.

Le date nei documenti dovranno essere scritte sempre nel formato **GG-MM-AAAA** (giorno-mese-anno, con due cifre per giorno e mese e quattro cifre per l'anno).
Eventuali identificatori dovranno sempre seguire la regola del ***Kebab Case***, ossia parole separate dal carattere **-**.

3.1.7 - Manutenzione

L'attività di manutenzione[G] è necessaria nel momento in cui un documento ha bisogno di essere aggiornato con nuove informazioni. Il processo di aggiornamento non differisce di molto rispetto al processo di prima redazione, prevedendo infatti la realizzazione di una issue[G] dedicata e di un processo di Verifica[G] mediante *Pull Request*[G].

Per le regole sulla nomenclatura da utilizzare per *branch*[G] e *Issue*[G] si veda Sezione 4.2.3.3 nella parte precedente di questo documento.

3.2 - Gestione delle Configurazioni

La gestione delle configurazioni è un processo fondamentale in un progetto: essa permette di identificare le singole componenti del sistema sviluppato e controllarne le modifiche e i rilasci, nonché registrare lo stato di sviluppo di ciascuna. Complessivamente, il processo permette di avere un buon sviluppo.

3.2.1 - Strumenti a supporto

Per ogni attività prevista nel processo di Manutenzione[G] ,*ALimitedGroup* utilizza:

- **GitHub**[G]: per gestire i cambiamenti da effettuare ai documenti e al codice, nonché per garantire a tutti i membri di *ALimitedGroup* di accedere ai compiti da svolgere;

Per ulteriori informazioni si veda la Sezione 4.2.2.

3.2.2 - Attività previste

L'attività di gestione delle configurazioni svolta dal gruppo si avvale delle seguenti attività:

- **Identificazione della configurazione** ovvero l'identificazione delle componenti che formeranno il prodotto da sviluppare;
- **Controllo della configurazione** ovvero il controllo dei cambiamenti con opportuni metodi di approvazione[G] e rifiuto degli stessi;
- **Registrazione dello stato di configurazione** ovvero il come rappresentare la storia dei cambiamenti subiti da ciascun elemento sviluppato;
- **Valutazione della configurazione** ovvero come determinare l'efficacia del prodotto sviluppato, ossia la sua conformità ai requisiti;

3.2.3 - Identificazione della configurazione

Il capitolato[G] sviluppato dal nostro gruppo è formato da molte parti, motivo per cui l'identificazione della configurazione si renderà essenziale: per questo motivo, durante la fase di progettazione, sarà necessario schematizzare le varie parti dell'architettura del sistema da sviluppare e allegare il risultato di quanto compreso in un documento apposito da determinare durante la *Product Baseline*[G] *(PB)*.

Per quanto riguarda invece la parte di documentazione, le parti componenti tale sezione del progetto sono state descritte nella Sezione 3.1.

3.2.4 - Controllo della configurazione

Il **Controllo della configurazione** è l'attività che disciplina le richieste di modifica che devono essere notificate per poi essere o meno approvate. Un buon controllo della configurazione dovrebbe prevedere un sistema che permetta di tracciare le modifiche da approvare, consigliando eventuali suggerimenti per ottenere l'approvazione.

ALimitedGroup ha deciso di attuare questa attività mediante l'utilizzo delle ***issue***, delle ***board***[G] e delle ***Pull Request***[G] di ***GitHub***, così come descritto qui di seguito:

- ***Issue***: ogni modifica da apportare deve essere documentata mediante una *issue*[G] da assegnare al componente che prenderà in carico la modifica o la redazione di un documento o di una parte di codice.

Generalmente una issue[G] possiede:
- un numero univoco nella *repository*[G] che funge da identificativo della *issue*[G]
- un nome che esplicita in cosa consiste la *issue*[G]
- una descrizione facoltativa;
- una *milestone*[G] o una *baseline*[G] associata;
- un assegnatario che prenderà in carico il lavoro da effettuare;
- un legame con la *board*[G] del progetto;

Una *issue*[G] può essere chiusa solo nel momento in cui l'attività che prevedeva è stata verificata e portata dunque nel ramo principale dei *repository*[G] :tale operazione può essere effettuata in automatico inserendo, nel commento del *commit* che esegue lo spostamento delle modifiche al ramo principale, il numero della *issue*[G] preceduto da `fix #` (esempio `fix #1` chiude in automatico la issue[G] numero 1).

- ***Board***: serve per stabilire se una *issue*[G] presa in carico è ancora da iniziare, in sviluppo o terminata. La ***Board***[G] identifica in quale periodo è necessario svolgere le attività e, mediante dei diagrammi di Gantt generati da *GitHub*[G], aiuta a comprendere la loro durata preventivata ed effettiva, individuando eventuali ritardi;
- ***Pull Request***[G]: un sistema che permette di chiedere la Verifica[G] e (eventualmente) l'approvazione prima di fondere la modifica nel ramo principale del *repository*[G]. Le *Pull Request*[G] sono il cuore del controllo della configurazione: permettono infatti di generare un elenco dettagliato e una discussione sulle modifiche da effettuare, permettendo di suggerire cambiamenti prima di considerare le modifiche apportate come definitive.
- ***Teams di GitHub***: la struttura del *repository*[G] permettere solo ai verificatori (e al Responsabile[G] per eventuali approvazioni) la possibilità di poter trasportare delle modifiche nel ramo principale (previa verifica[G]).Per queste esigenze, i *repository*[G] prevedono tre categorie:
  - Responsabile[G]

- Verificatori;
- Redattori.

3.2.5 - Registrazione dello stato di configurazione

ALimitedGroup è consapevole della necessità di tracciare i cambiamenti effettuati a documentazione e codice.

L'attività di **Registrazione dello Stato di Configurazione** si occupa proprio di questo e i componenti di *ALimitedGroup* hanno deciso di implementarla come segue.

Come verbalizzato nel **verbale interno** del **4 Novembre 2024** ed esteso poi a tutta la documentazione, i documenti utilizzeranno il seguente sistema di versionamento:

`MAJOR.MINOR.PATCH`

dove:

- **MAJOR**: subisce un incremento solo ad approvazione[G] del file;
- **MINOR**: subisce un incremento solo quando è completato sia il processo di modifica che quello di Verifica[G]
- **PATCH**: subisce un incremento per modifiche di entità minore, quali correzioni ortografiche e/o di sintassi.

Lo spunto d'uso di tale sistema proviene dal sito semverdoc.org.

In aggiunta, ogni documento avrà un allegato contenente l'elenco cronologico di tutte le modifiche apportate per ogni versione, come anche descritto nella Sezione 3.1.3 e Sezione 3.1.5.

3.2.6 - Valutazione della configurazione

La **Valutazione della configurazione** è l'attività che prevede di controllare la completezza del *software* prodotto rispetto ai suoi requisiti e al design scelto.

ALimitedGroup intende realizzare un **Tracciamento dei Requisiti** per controllare l'aderenza di quanto sviluppato con le aspettative. Più nello specifico, l'obiettivo è realizzare un Prodotto che sia efficiente, cioè soddisfa tutti i requisiti e sufficiente nel senso che soddisfa solo i requisiti necessari. Il gruppo, durante la parte di sviluppo effettiva, si impegnerà a tracciare, anche nel codice tramite commenti, cosa intende realizzare.

3.3 - Accertamento Qualità

L'**Accertamento della Qualità** (detto anche ***Quality Assurance***) è un processo che ha come scopo quello di attestare che la realizzazione dei processi e dei prodotti finali sia stata fatta a regola d'arte: i processi e prodotti devono soddisfare determinati requisiti minimi affinché siano efficienti ed efficaci.

Per assicurare qualità *ALimitedGroup* eseguirà, come suggerito dallo standard *ISO/IEC 12207:1995*, le attività di Verifica[G] (Sezione 3.4.1) e Validazione[G] (Sezione 3.4.2).

3.3.1 - Strumenti a supporto

Per verificare ed attestare che i prodotti finali siano stati realizzati a regola d'arte, *ALimitedGroup* utilizza delle metriche che si differiscono in base al processo coinvolto. In questo modo siamo consapevoli della effettiva qualità del lavoro svolto, essendo tali metriche puramente oggettive.
Tutte le metriche adottate dal gruppo sono presentate e descritte interamente nel **Piano di**

Qualifica[G] v1.1.0 (Sezione 5).
Inoltre, durante il processo di Accertamento della Qualità verrà utilizzata una funzionalità[G] di **Grafana**[G] :le *dashboard*[G] .
Quest'ultima, permette di visualizzare, analizzare e monitorare in tempo reale come vengono memorizzati i dati e, soprattutto, come risponde il *software* realizzato con livelli di carico alti.

3.3.2 - Attività previste

Le attività previste dal processo di **Accertamento della qualità** sono le seguenti:

- **Implementazione del processo**: che prevede l'individuazione degli standard di qualità da rispettare, nonché le procedure da eseguire per raccogliere le valutazioni;
- **Accertamento qualità prodotto**: che prevede il controllo di tutto il materiale prodotto per verificare che questo rispetti gli standard decisi;
- **Accertamento qualità processo**: che prevede il controllo di tutti i processi eseguiti dal gruppo per verificare che questi rispettino gli standard decisi.

ALimitedGroup per assicurare qualità ha individuato specifiche misure discusse alla Sezione 5. Si è deciso, per una consultazione più rapida, di definire in una sezione apposita tali misurazioni per renderle più velocemente consultabili.

Le misurazioni dovranno essere realizzate alla fine di ogni *sprint*[G] e adeguatamente riportate nel **Piano di Qualifica[G] v1.1.0 (Sezione 5)**.

Le metriche per la qualità di prodotto sono disponibili alla Sezione 7 mentre quelle di processo alla Sezione 6.

3.4 - Qualifica

3.4.1 - Verifica

Il processo di **Verifica**[G] ha come scopo quello di verificare che quanto prodotto sia stato svolto a regola d'arte, in conformità con i requisiti imposti.

Possiamo racchiudere l'obiettivo di questo processo nella risposta alla domanda:

«Did I build the System right?»

ovvero, *«Ho realizzato il Sistema correttamente?*.

Tutti gli obiettivi e gli esiti di questo processo dovranno essere adeguatamente riportati nel **Piano di Qualifica[G] v1.1.0 (Sezione 4)**.

3.4.1.1 - Strumenti a supporto

Per verificare che quanto prodotto dal gruppo, sia stato realizzato ed implementato secondo i requisiti richiesti ed imposti da M31, ovvero che risponda correttamente alla domanda citata precedentemente.
Per questo, all'interno del documento del **Piano di Qualifica[G] v1.1.0** nella sezione 4, vengono esposti e descritti dettagliatamente tutti i *test*[G] che il gruppo effettuerà, per verificare che il prodotto realizzato sia conforme ai requisiti imposti da M31.

Per il calcolo dell'indice di Gulpease è stato utilizzato il seguente servizio:

farfalla-project.org/readability_static/

3.4.1.2 - Attività previste

In base allo standard *ISO/IEC 12207:1995*, le attività previste da questo processo sono:

- **Implementazione del processo**, che prevede la realizzazione di quanto necessario al perseguimento dell'attività di Verifica[G], dalla rilevazione delle criticità del Progetto da analizzare sino alla rilevazione delle problematiche e ai metodi di mitigazione;
- **Attività di Verifica**, ovvero l'attività di Verifica[G] vera e propria in cui è necessario controllare l'efficacia di **Processi**, **Requisiti** (in termini di esaustività e consistenza), **Progettazione** (in termini di efficacia[G] rispetto ai requisiti ed esigenze di M31), **Codice** (in termini di grado di soddisfacimento dei requisiti che una parte di codice dovrebbe soddisfare), **Integrazione** (delle varie parti del Sistema tra loro) e **Documentazione**.

3.4.1.3 - Implementazione del processo

ALimitedGroup ha concluso che la principale necessità è assicurare che il Progetto soddisfi le esigenze di M31 e per questo ha intenzione di realizzare approfonditi Test[G], dalla Documentazione al Codice prodotto.

In merito alla verifica[G] delle componenti del progetto, *ALimitedGroup* ha stabilito che nessun documento o parte di codice possa essere portato nel ramo principale del *repository*[G] del gruppo senza che questo sia prima stato adeguatamente verificato e/o testato ove possibile, così come descritto nella Sezione 3.4.1.4. Si comprende dunque che il ramo principale del *repository*[G] non possa contenere elementi che sono, in quanto potenzialmente non corretti, non di qualità.

Eventuali problematiche di qualsiasi natura non singolarmente risolvibili saranno oggetto di discussione in sede di riunione interna.

Ai fini di descrivere al meglio la parte di *Way of Working*[G] che *ALimitedGroup* intende adottare per il processo di Verifica[G], saranno ora esposti i vari aspetti dell'attività.

3.4.1.4 - Attività di Verifica

Iniziando dalla **Documentazione**, attività in cui il gruppo si è particolarmente concentrato nella prima fase del Progetto, *ALimitedGroup* ha stabilito che ogni documento sia sottoposto alla fase di verifica[G] dopo essere redatto, prima che questo possa essere portato nel ramo principale del *repository*[G]. Questa attività consiste nel controllare la correttezza grammaticale e sintattica del documento, ma anche la correttezza del contenuto.

Per maggiori informazioni sul processo di redazione di un documento si consiglia la lettura della Sezione 3.1.6.

In merito invece alle verifiche relative al **Codice**, questo sarà un argomento che *ALimitedGroup* affronterà maggiormente nel dettaglio non appena raggiunta la *Requirements and Technology Baseline*[G].

In generale tutte le informazioni relative alla Verifica[G], dalle misurazioni agli esiti dei Test[G] effettuati, dovranno essere riportati nel **Piano di Qualifica[G] v1.1.0 (Sezione 4)**.

La Verifica[G], in generale può essere realizzata in due modi: mediante **Analisi Statica** e mediante **Analisi Dinamica**, qui di seguito descritte.

3.4.1.4.1 - Analisi Statica

L'**Analisi Statica** è così chiamata in quanto sono attività di Verifica[G] che **non** prevedono l'esecuzione dell'oggetto da verificare. Si tratta dunque di una Verifica[G] che si concentra

maggiormente sulla sintassi e sulla correttezza effettiva di quanto scritto, al fine di rilevare potenziali problemi ancor prima che questi si possano verificare mediante l'esecuzione.

L'**Analisi Statica** può essere realizzata mediante **metodi formali**, ovvero mediante prove matematiche, oppure per **metodi di lettura**, meno formali ma comunque efficaci.

I **metodi di lettura** sono due:

- **Walkthrough**, che suppone di essere consapevoli dell'esistenza di un problema, ma di non sapere di che cosa si tratta e dove si trova. Richiede dunque un'analisi approfondita di tutto l'oggetto di verifica[G] :è una metodologia costosa e poco applicabile perché difficilmente automatizzabile;
- **Ispezione**, si basa sulla conoscenza dei possibili problemi senza però verificarne la presenza, risulta meno efficace rispetto al primo sistema, in quanto non è completamente esaustiva, soprattutto per attività appena avviate. Tuttavia, consente di creare una lista di controllo da applicare all'oggetto di verifica[G], permettendo così di individuare potenziali errori o problemi.

Tra i due, data l'alta mole di oggetti da verificare, l'**ispezione** è preferibile in quanto, potendola realizzare mediante una lista di controllo, è automatizzabile.

3.4.1.4.2 - Analisi Dinamica

L'**Analisi Dinamica** è così chiamata perché prevede, per poter essere effettuata, l'esecuzione dell'oggetto da verificare. L'intento è quello di rimuovere i *fault*, ovvero elementi che hanno determinato un comportamento inatteso, mediante l'esecuzione del codice e la rilevazione delle *failure*, ovvero esecuzioni il cui esito non era quello atteso.

L'**Analisi Dinamica** assolve il proprio compito mediante i Test[G], che devono essere **ripetibili** (poiché se si presenta un *failure* e i *fault* che lo ha causato viene risolto è necessario eseguire nuovamente il Test[G] per verificarne l'effettiva correzione) e automatizzabili (questo ottenibile mediante *driver* che chiamano le parti di codice non direttamente eseguibili, *stub* che sostituiscono parti di codice richiamabili per testare la componente che le chiama e *logger* che registrano l'esito dei Test[G] in un formato comprensibile da un automa).

Per essere corretto, un Test[G] deve essere indipendente dall'ambiente di esecuzione e restituire, dato uno stesso input, lo stesso output, che deve essere corretto rispetto all'input fornito.

Le principali tipologie di Test[G] sono:

- **Test di Unità**;
- **Test di Integrazione**;
- **Test di Sistema**;
- **Test di Regressione**.

La nomenclatura utilizzata per descrivere tali Test[G] (reperibili nel **Piano di Qualifica[G] v1.1.0** alla **Sezione 4**) è la seguente:

`T-#-Tipo`

dove:

- **T** indica la parola **T**est
- **#** è un numero crescente che identifica, all'interno del tipo, un determinato Test[G]
- **Tipo** classifica il Testin una delle seguenti tipologie:

- **U** per Test[G] di **U**nità
- **I** per Test[G] di **I**ntegrazione
- **S** per Test[G] di **S**istema
- **A** per Test[G] di **A**ccettazione

Ogni Test[G] ha poi uno stato tra i seguenti:

- **S** ovvero **S**uperato
- **I** ovvero **I**mplementato
- **NI** ovvero **N**on **I**mplementato

3.4.1.4.2.1 - Test di Unità

I **Test di Unità** verificano la correttezza delle unità del codice, ovvero parti del Sistema che, eventualmente avvalendosi di moduli, costituiscono una componente piccola ma con una responsabilità unica, non condivisa e sufficientemente grande per poter essere testata.

Questi Test[G] sono di due tipologie:

- **Test Funzionali**: sono Test[G] che verificano gli input e gli output dati al sistema, ma senza verificare la logica che porta alle conclusioni di tali risultati. Non guardando la logica dell'unità, sono per questo detti anche **black-box**. Per realizzare dei buoni Test[G] funzionali è necessario realizzare, data una unità, Test[G] su cinque classi di valori in input: uno non ammissibile «inferiore» all'insieme dei valori ammissibile, uno tra i valori «estremi inferiori» ammissibili, uno tra i valori ammissibili, uno tra i valori «estremi superiori» ammissibili e uno tra i valori non ammissibili «superiori».
- **Test Strutturali**: sono Test[G] che invece concentrano l'attenzione sulla logica del programma, su quanti e quali *statement* del codice vengono effettivamente eseguiti.

3.4.1.4.2.2 - Test di Integrazione

Una volta attestato che le varie unità funzionano correttamente, è necessario assicurarsi che le varie unità «parlino» correttamente tra di loro, verificandone quindi l'integrazione. Per effettuare tale integrazione è possibile seguire due metodologie:

- **Top - Down**: prevede di verificare prima le componenti con maggior numero di dipendenze, ovvero quelle che hanno maggiore responsabilità verso l'esterno del Sistema;
- **Bottom - Up**: ovvero a partire dalle componenti con minore dipendenze d'uso, solitamente dunque quelle con maggiore importanza all'interno del Sistema.

3.4.1.4.2.3 - Test di Sistema

Sono Test[G] che mirano a verificare la correttezza del Sistema nel suo complesso, per controllare l'effettiva efficacia[G] del Sistema rispetto ai suoi requisiti.

3.4.1.4.2.3.1 - Test di Regressione

Durante l'esecuzione dei Test[G] descritti in precedenza, è possibile che uno di questi segnali un problema: in tal caso è necessario analizzare il problema, sviluppare una soluzione, codificarla e infine controllare che il problema sia stato risolto ripetendo il Test[G].

È tuttavia necessario assicurarsi che una correzione apportata non introduca un problema precedentemente risolto: a questo scopo non ci si limita all'esecuzione del solo Test[G] fallito, bensì di tutti i Test[G], per evitare, di fatto, una regressione.

3.4.2 - Validazione

Lo scopo del processo di **Validazione**[G] è quello di tracciare l'efficacia di quanto sviluppato, ovvero comprendere se quanto realizzato soddisfa le esigenze di M31. Potremmo riassumere lo scopo della **Validazione**[G] nella risposta alla domanda:

«Did I build the right System?»

ovvero, *«Ho realizzato il giusto Sistema?*.

3.4.2.1 - Attività previste

In base allo standard *ISO/IEC 12207:1995*, le attività previste da questo processo sono:

- **Implementazione del processo**
- **Attività di Validazione**

3.4.2.1.1 - Implementazione del processo

ALimitedGroup ha studiato le esigenze di M31 e ha racchiuso tutti i requisiti da soddisfare all'interno dell'**Analisi dei Requisiti[G] ver. 1.2.0 (Sezione 3)**.

Come già previsto da quanto descritto alla Sezione 3.2.6, il **Tracciamento dei Requisiti** sarà un elemento valido anche al processo di Validazione[G]: permette infatti di controllare se il prodotto funziona correttamente (un requisito è soddisfatto solo se il codice che lo implementa funziona correttamente) ed è conforme ai requisiti (perché il loro soddisfacimento è, per l'appunto, tracciato).

ALimitedGroup, per realizzare l'attività di Validazione[G] vera e propria, si impegna a **tracciare i Requisiti** ed ad effettuare **Test di Accettazione** per controllare l'efficacia di quanto sviluppato e verificato, simulando scenari di utilizzo da parte di M31.

3.4.2.1.2 - Attività di Validazione

ALimitedGroup, per realizzare l'attività di Validazione[G] vera e propria, di realizzare dunque il **tracciamento dei Requisiti** ed eventuali **Test di Accettazione** per controllare l'efficacia di quanto sviluppato e già verificato, simulando scenari di utilizzo da parte di M31.

4 - Processi Organizzativi

I **Processi Organizzativi** sono dei processi a supporto del progetto che assicurano il buon andamento dell'intero progetto. Le attività previste assicurano la buona esecuzione di tutti i processi adottati e l'adozione di eventuali miglioramenti, la gestione delle infrastrutture utilizzate e la formazione del team nei compiti da seguire.

Tra i processi organizzativi si individuano:

- **Gestione dei Processi**
- **Infrastruttura**
- **Miglioramento**
- **Formazione**

4.1 - Gestione dei Processi

La **Gestione dei Processi** ha l'obiettivo di individuare i compiti da svolgere e i ruoli ai quali questi saranno assegnati, nonché permettere una comunicazione interna ed esterna efficace e altresì garantire lo svolgimento delle varie attività in maniera efficace mediante un'opportuna pianificazione.

4.1.1 - Strumenti a supporto

Per le attività previste nella Gestione dei Processi, abbiamo deciso di utilizzare i seguenti strumenti:

- **Git** e **GitHub**[G] :per la gestione delle attività programmate e assegnate ad ogni singolo componente del gruppo, tramite *issue*[G] ,nonchè la pianificazione degli *sprint*[G] ,delle attività e *milestone*[G] future;
- **Google Keep**: per tenere traccia di tutte le attività completate, tra quelle programmate, e gestire nel miglior modo possibile le *task* all'interno di ogni *sprint*[G] .

4.1.2 - Attività previste

Le attività principali, osservate da *ALimitedGroup*, previste da questo processo sono:

- **Inizializzazione**: è necessario stabilire i requisiti delle attività da svolgere, cercando di comprendere quali risorse esse richiedono. È compito del **Responsabile**[G] determinare queste caratteristiche che vengono comunque discusse internamente durante le riunioni interne;
- **Pianificazione**: è necessario, stabiliti i requisiti, comprendere il tempo necessario per completare le attività, così come è necessario anche stimare i costi economici e temporali necessari allo svolgimento. Per facilitare questo compito, è stato scelto di dividere le responsabilità in vari ruoli, vedi Sezione 4.1.3.
- **Esecuzione e controllo**: l'esecuzione delle attività è affidata quindi ai vari ruoli e il **Responsabile**[G] dovrà costantemente monitorare lo stato di progresso e avanzamento complessivo;
- **Revisione e valutazione**: una volta effettuata l'attività è necessario controllare la conformità di quanto prodotto: questo avviene per opera del **Verificatore**[G]. Per un dettaglio più specifico delle attività di revisione si veda la Sezione 3.1.6 per quanto riguarda la documentazione mentre Sezione 3.2.6, Sezione 3.4.1 e Sezione 3.4.2 per il codice;
- **Finalizzazione**: un'attività è da reputarsi conclusa solo nel momento in cui viene definitivamente approvata. Come anticipato nelle Sezioni precedenti, questa operazione viene eseguita al pari di una normale attività di verifica[G], con la sola eccezione che la

risoluzione della *Pull Request*[G] associata determina la chiusura della *Issue*[G] e del *branch*[G] utilizzato per la redazione/modifica della componente.

Nell'interesse dello scopo del presente documento saranno ora descritti i ruoli prima citati e le loro responsabilità, previsti in particolar modo dall'attività di **pianificazione**. Sarà quindi dato spazio ad una parte fondamentale del gruppo, ovvero i metodi di **coordinamento**.

4.1.3 - Ruoli

Ruolo	Compiti e responsabilità
Responsabile	Al Responsabile spetta il compito di comprendere, tenendo in considerazione quanto già portato a termine, quanto da realizzare nei periodi di sviluppo successivi, individuando le attività, i costi e rischi associati e infine assegnandone lo svolgimento ai componenti che vestono in quel periodo il ruolo più appropriato. Il Responsabile è anche colui che ha il compito, a nome di tutto il gruppo, di dialogare con le parti esterne ad *ALimitedGroup*, quali, a mero titolo esemplificativo, l'azienda proponente del capitolato M31. In ultima istanza, è sempre compito di questo ruolo **approvare** i vari documenti prodotti da *ALimitedGroup*.
Amministratore	Noto anche come *System Administrator* o *SysAdmin*, ha l'importante compito di gestire e migliorare l'infrastruttura che i membri di *ALimitedGroup* utilizza per portare a compimento le varie attività del progetto didattico, a partire dal sistema di ticketing sino agli strumenti utilizzati per la verifica e Validazione di quanto realizzato. È questo ruolo che copre la responsabilità di risolvere quanto prima eventuali problematiche legate alle infrastrutture. Per via della sua approfondita conoscenza del *Way of Working* adottato dal gruppo, necessario per poter al meglio gestire le infrastrutture, è di questo ruolo il compito di redigere il presente documento e il **Piano di Qualifica**, ma può risultare un ruolo adatto anche a redigere sia i verbali interni sia quelli esterni. Redige anche il **Manuale Utente** con l'aiuto dei Progettisti.
Verificatore	Ha il compito di garantire che tutto ciò che è prodotto, dalla documentazione alla più piccola attività, sia svolta a regola d'arte. Si occupa di eseguire Test approfonditi e revisioni del *software*, identificando eventuali aree di miglioramento specie in ambito qualitativo. È sempre di questo ruolo la responsabilità di verificare la correttezza dei vari documenti ad ogni modifica effettuata.
Analista	Questo ruolo è Responsabile nell'identificazione dei requisiti obbligatori, desiderabili e facoltativi del progetto, considerando quanto discusso nelle riunioni esterne con l'azienda proponente M31. È sempre affidato a questo ruolo il compito di redigere l'**Analisi dei Requisiti**.

Ruolo	Compiti e responsabilità
Programmatore	Il Programmatore è Responsabile dello sviluppo del *software*, traducendo il design architetturale in codice funzionante. Collabora strettamente con il progettista per assicurarsi che tutte le funzionalità siano implementate secondo le specifiche. Il suo lavoro è fondamentale per il progresso del progetto e richiede una buona conoscenza delle tecnologie adottate. È affidata a questo ruolo anche la responsabilità di realizzare Test automatici per verificare il corretto funzionamento del codice sviluppato.
Progettista	È un ruolo cruciale per lo svolgimento del capitolato: esso infatti ha il compito di trasformare requisiti in un design architetturale, producendo documenti e schemi esplicativi e definendo le scelte tecnologiche. Ha il compito, inoltre, di scrivere la **Specifica Tecnica**.

Tabella 13: Compiti e responsabilità di ogni singolo ruolo

4.1.4 - Coordinamento

Una parte fondamentale del progetto risiede nelle capacità di coordinamento del gruppo, sia internamente che esternamente.

Per svolgere adeguatamente questa attività è necessario prevedere e svolgere riunioni apposite e avere canali di comunicazione funzionanti e utili allo scopo.

4.1.4.1 - Riunioni

Le riunioni sono di due tipi: **interne** ed **esterne**.

ALimitedGroup realizza periodicamente, generalmente all'inizio di ogni *sprint*[G], riunioni interne per il coordinamento interno: durante le stesse viene effettuato un punto della situazione sul progetto, un momento in cui i componenti si aggiornano sulle attività svolte e su quelle non terminate in tempo. L'attività di dialogo permetterà al **Responsabile**[G] di avere una panoramica attuale su quanto svolto, permettendogli così di gestire al meglio il periodo successivo. In queste occasioni è anche possibile effettuare la rotazione dei ruoli in caso di necessità (quali, a mero titolo di esempio, il termine delle ore di un determinato ruolo per una determinata persona oppure l'esaurimento di compiti appartenenti a quel ruolo).

Durante lo svolgimento del progetto è inoltre utile avere degli incontri con M31: si tratta dunque di riunioni **esterne** dove i componenti di *ALimitedGroup* hanno l'occasione di presentare il lavoro sino a quel momento svolto e chiarire eventuali dubbi riscontrati. Contrariamente alle riunioni interne, non è prevista una periodicità: gli incontri sono fissati di volta in volta per mezzo di richiesta ad M31 mediante posta elettronica.

L'esito degli incontri dovrà sempre essere documentato mediante la redazione di appositi **verbali**, rispettivamente **interni** ed **esterni**: per maggiori informazioni si veda la Sezione 2.1.3.

4.1.4.2 - Comunicazioni

In merito ai metodi comunicativi, *ALimitedGroup* utilizza **Telegram**[G] e **Discord**[G] per, rispettivamente, comunicazioni asincrone e sincrone (riunioni). In genere una comunicazione urgente ma che non richiede approfondita discussione o per comunicazioni di servizio

minori (come, a titolo di esempio, dubbi minori e non urgenti su particolari argomenti) i componenti utilizzeranno l'apposito gruppo realizzato su **Telegram**[G] o messaggi diretti ad un componente specifico sempre utilizzando la medesima piattaforma, mentre la realizzazione di riunioni interne e la discussione di criticità complesse richiederanno lo svolgimento di una riunione presso il *Server* **Discord**[G] appositamente realizzato.

Per quanto riguarda invece le comunicazioni esterne, queste verranno sempre realizzate dal **Responsabile**[G] mediante l'utilizzo dell'indirizzo di posta elettronica di *ALimitedGroup* alimitedgroup@gmail.com.

Per maggiori informazioni si consiglia la lettura del processo di **Infrastruttura** alla Sezione 4.2.

4.2 - Infrastruttura

Il processo di **Infrastruttura** è responsabile[G] della creazione e del mantenimento dei componenti (di qualsiasi natura, sia *Hardware* che *software*) necessari per permettere tutti gli altri processi.

4.2.1 - Attività previste

Si compone delle seguenti attività:

- **Implementazione**
- **Creazione**
- **Manutenzione**[G]

Tali attività saranno descritte nel dettaglio nelle prossime Sezioni.

4.2.2 - Implementazione

ALimitedGroup ha compreso, durante lo svolgimento del progetto didattico, la necessità di adottare appositi strumenti per permettere il lavoro asincrono dei suoi componenti. Di seguito vengono elencati l'insieme di tutti gli strumenti utilizzati.

- *Discord*[G]
- *Git*
- *GitHub*[G]
- *Google Calendar*
- *Google Fogli*
- *Google Mail*
- *Microsoft Teams*
- Script in *Python*
- *Telegram*[G]
- *Typst*[G]

Strumento	Descrizione
Discord	*Discord* è un programma di messaggistica istantanea e videoconferenza utilizzato da *ALimitedGroup* per realizzare le proprie riunioni interne in modalità **virtuale**.
Git	Programma originariamente sviluppato da Linus Torvalds per il versionamento del codice: *ALimitedGroup* ha deciso di impiegarlo come strumento di versionamento per il sorgente della propria documentazione e il codice prodotto. Git è dunque un potente *Version*

Strumento	Descrizione
	Control System (VCS) che permette di versionare efficacemente tutto ciò che viene prodotto grazie anche alla possibilità di organizzare gli sviluppi in *branch* separati.
GitHub	**GitHub** è un prodotto che permette principalmente la memorizzazione su dispositivo remoto di *repository* Git, ma non solo: esso infatti si è evoluto nel corso del tempo per permettere la collaborazione asincrona tra *developer*. *ALimitedGroup* utilizza *GitHub* per sincronizzare gli sviluppi tra i vari membri, tenere traccia del *backlog* mediante il sistema di *issue* e *project board*, nonché permettere la verifica di quanto redatto prima che tali documenti o codice raggiunga il *branch* principale. Per i dettagli d'uso si rimanda alla Sezione 3.2.4.
Google Calendar	Per condividere con il gruppo tutti gli appuntamenti *ALimitedGroup* utilizza **Google Calendar**.
Google Fogli	*ALimitedGroup* ha deciso di utilizzare un documento di **Google Fogli** per tenere traccia delle attività svolte settimanalmente durante gli *sprint*
Google Mail	Il servizio di posta elettronica utilizzato da *ALimitedGroup* per tutte le comunicazione verso l'esterno
Microsoft Teams	**Microsoft Teams** viene utilizzato da *ALimitedGroup* per realizzare le riunioni esterne con l'azienda proponente M31.
Script in Python	Per automatizzare l'aggiornamento del sito web con i nuovi documenti e la compilazione dei file Typst, *ALimitedGroup* ha deciso di realizzare degli appositi script in Python. Un ulteriore script inoltre permette di realizzare una sostituzione parziale delle parole nei documenti per collegarle ai termini contenuti nel glossario
Telegram	Telegram viene utilizzato dal gruppo per comunicare in maniera diretta tra i membri in caso di dubbi minori per cui non è necessaria una riunione interna.
Typst	Typst viene utilizzato da *ALimitedGroup* per la redazione di tutti i documenti. Vengono compilati automaticamente utilizzando gli script prodotti direttamente dal gruppo.

Tabella 14: Strumenti componenti l'Infrastruttura

4.2.3 - Creazione

L'attività di **creazione** guida la realizzazione dell'infrastruttura. Di seguito i dettagli per ogni prodotto utilizzato.

4.2.3.1 - Discord

Per utilizzare **Discord**[G] è stato realizzato un server con un canale testuale e un canale vocale apposito per le riunioni.

4.2.3.2 - Git

Git non richiede particolari modifiche: deve essere configurato inserendo username e email con cui il componente interagisce normalmente con il *repository*[G] GitHub[G] del progetto.

4.2.3.3 - GitHub

Il *repository* **GitHub**[G] dedicato alla documentazione è strutturato in maniera da favorire la produzione dei documenti. All'interno è possibile trovare varie *directories*, qui di seguito verranno descritte quelle di maggior importanza:

- **.github/workflows**: contiene lo script in *Python* che si occupa di compilare i files *Typst*[G] dei documenti ed aggiornare il sito di conseguenza. Contiene inoltre lo script python per applicare i riferimenti delle parole presenti nel Glossario;
- **.vscode**: contiene informazioni utili per l'*IDE* di sviluppo *Microsoft Visual Studio Code*, se questo viene utilizzato per la stesura dei documenti;
- **01-candidatura**: contiene, con le eventuali *sub-directories* i file della candidatura, ossia la prima fase del progetto;
- **02-RTB**: contiene, con le eventuali *sub-directories* i file della *Requirements and Technology Baseline*[G] (RTB), ossia la seconda fase del progetto;
- **assets**: contiene *files* utili per il sito web e i documenti, come loghi e *fonts*;
- **lib**: contiene i file *template* per la redazione dei documenti;
- **website**: contiene i file relativi al sito web del gruppo.

È inoltre possibile trovare il file ***.gitignore*** (utile per evitare il tracciamento di alcuni file specifici), il file ***README.md*** (che permette di realizzare la descrizione nella pagina principale del *repository*[G]) e ***docs.typ*** (altro file di utilità per i documenti).

Come descritto nella Sezione 3.1.6, la redazione o modifica di un documento richiede la creazione di un branch[G] secondario. Tale branch[G] avrà un nome che segue il seguente schema:

`#-azione-documento-data`

Dove:

- Al posto di **#** va inserito il numero della *Issue*[G] le cui modifiche operate nel branch[G] determineranno la chiusura
- Al posto di **azione** va inserita l'azione fatta nel *branch*[G] ossia:
  - **redazione** per indicare la redazione di un nuovo documento
  - **aggiornamento** per indicare la modifica di un documento esistente
- Al posto di **documento** va inserito la tipologia di documento interessato dalla modifica, come, a mero scopo esemplificativo, **verbale** oppure **norme-progetto**
- Al posto di **data** la data del documento, se tale documento la prevede (ad esempio i **verbali**)

Il gruppo ha poi deciso di utilizzare le *Issue*[G] di GitHub[G] per tracciare le attività da fare. Generalmente, il nome di una *Issue*[G] segue il seguente schema:

`azione documento data`

Dove **azione**, **documento** e **data** hanno lo stesso significato della nomenclatura utilizzata per i *branch*[G]. Ogni *Issue*[G] è inoltre legata ad una *Project Board*, uno strumento di GitHub[G] che permette di vedere velocemente che attività ci sono ancora da svolgere e quali invece sono in corso.

La chiusura di una *Issue*[G] avviene sempre ed esclusivamente mediante l'apertura di una *Pull Request*[G] e al seguito di una verifica[G] con esito positivo: in tal caso il **Verificatore**[G] che ha effettuato la verifica[G] o il **Responsabile**[G] potrà procedere alla chiusura della stessa con

conseguente entrata delle modifiche nel *branch*[G] principale e questa azione determinerà la chiusura automatica della *Issue*[G]. Qualora *GitHub*[G] non segnalasse tale automazione, sarà necessario procedere ad aggiungere al commento di chiusura della *Pull Request*[G] la dicitura

`fix #issue`

dove al posto di ***issue***[G] va inserito il numero della issue[G] associata alla *Pull Request*[G].

È bene sottolineare che seppur la chiusura di una *Issue*[G] può essere effettuata manualmente in caso di necessità, questo **compromette fortemente** la **tracciabilità** di quanto effettuato.

In ultima istanza, il gruppo ha configurato anche la funzionalità[G] fornita da GitHub[G] denominata **GitHub Actions**, che permette di realizzare azioni automatiche quando un commit viene realizzato nel branch[G] di sviluppo principale (**main**): nello specifico, l'action esegue, grazie anche all'ausilio di uno script Python, la compilazione dei documenti e la pubblicazione nel sito web del gruppo.

4.2.3.4 - Google Calendar

In merito al calendario condiviso, è affidato al **Responsabile**[G] il compito di aggiungere gli eventi del gruppo, dai diari di bordo alle riunioni interne ed esterne programmate.

Il calendario è condiviso tra i vari membri, che riceveranno un promemoria almeno un giorno prima rispetto l'evento in questione.

Rimane responsabilità di ogni membro controllarlo periodicamente.

4.2.3.5 - Google Fogli

Il file condiviso di **Google Fogli** prevede tre schede:

- **Dashboard**[G]: fornisce un prospetto aggiornato delle ore preventivate, quelle utilizzate e quelle rimanenti per ogni ruolo, nonché un dettaglio per ogni *sprint*[G]. È responsabilità di ogni componente aggiornare l'impegno orario effettivo durante lo *sprint*[G] ad ogni ora produttiva svolta, annotando l'elenco delle attività svolte;
- **Grafico**: fornisce una rappresentazione grafica delle informazioni contenute nella **Dashboard**[G].

4.2.3.6 - Google Mail

La casella di Google Mail non ha richiesto particolari configurazioni.

4.2.3.7 - Microsoft Teams

La piattaforma **Microsoft Teams** viene controllata direttamente da M31, in quanto tale non necessita di alcuna operazione.

4.2.3.8 - Script in Python

In merito agli Script di *Python*, questi non necessitano di particolare modifiche manuali: possono essere eseguiti direttamente senza necessità di alcuna operazione aggiuntiva.

In merito allo Script sulla compilazione dei file *Typst*[G] e aggiornamento del sito, questo viene eseguito automaticamente come descritto nella sezione Sezione 4.2.3.3.

4.2.3.9 - Telegram

ALimitedGroup ha realizzato un gruppo **Telegram**[G] per le comunicazioni di minore importanza. Tale gruppo è configurato in maniera tale da escludere il possibile ingresso di

persone esterne: ad eccezione di questo, non richiede operazioni di interesse per questo documento.

4.2.3.10 - Typst

L'ambiente per la realizzazione dei documenti, **Typst**[G], è stato personalizzato a partire dalla realizzazione di *template*, contenenti funzioni utili alla stesura dei documenti e conservati nella cartella *lib* del *repository*[G].

In merito ai **verbali**, le funzioni sono contenute nel file ***verbale.typ*** e sono, in elenco:

- **verbale**: inserisce la pagina di copertina, la tabella delle modifiche e l'indice;
- **inizio-verbale-interno**: permette di inserire un testo con le informazioni iniziali della riunione (come, a titolo di esempio, la data, il luogo e la durata);
- **inizio-verbale-esterno**: permette di inserire un testo con le informazioni iniziali della riunione (come, a titolo di esempio, la data, l'azienda con cui è stata fatta la riunione, il luogo e la durata);
- **approvazione-esterna**: permette di inserire il testo per certificare, nella sezione esterna, l'approvazione da parte dell'azienda con cui si è svolto l'incontro (solo in caso di verbale esterno);
- **tabella-decisioni**: permette di inserire la **tabella delle decisioni e delle azioni**.

In merito ai **diari di bordo**, il file ***diario.typ*** possiede funzioni primarie e secondarie per la realizzazione delle presentazioni. In esso vi è un'unica funzione di interesse primario, **presentazione**, che permette di generare automaticamente l'intero documento se ad essa vengono forniti i punti per ciascuna delle sezioni.

Per i rimanenti **documenti** funzioni di interesse sono contenute nel template **common.typ**, tra cui:

- **prima-pagina**: Inserisce la pagina di copertina e il registro delle modifiche;
- **indice**: genera l'indice del documento;
- **indice-tabelle**: genera un elenco di tutte le tabelle inserite nel documento;
- **indice-immagini**: genera un elenco di tutte le immagini presenti nel documento

Per i seguenti documenti:

- **Norme di Progetto**[G];
- **Piano di Progetto**[G];
- **Piano di Qualifica**[G];
- **Analisi dei Requisiti**[G];

che assumono, all'interno del progetto, un'importanza significativa rispetto ad altri (verbali, diari di bordo etc...) abbiamo realizzato un *template* apposito, che permette di automatizzare la realizzazione della struttura adatta a loro.
Tuttavia, l'utilizzo di tale *template*, denominato ***importantdocs.typ*** non è obbligatorio e può essere integrato con l'utilizzo di ulteriori *template*.

I vari documenti sono spesso accomunati da esigenze particolari servibili mediante l'utilizzo di funzioni non caratteristiche di alcun documento.
Tali funzioni sono sempre nel file ***common.typ***:

- **p** : permette l'inserimento delle informazioni riguardanti ogni persona coinvolta in questo progetto, quali i componenti del gruppo o i docenti di Ingegneria del *software*, utile per evitare erroneamente di inserire informazioni non veritiere;

- **M31** : permette di visualizzare il nome dell'azienda proponente[G]. Il nome è personalizzato con il font «*Futura*» e messo sempre in grassetto;
- **abbrev** : preso un *output* fornito dalla funzione *p*, inserisce nome e cognome della persona selezionata;
- **prof** : permette la visualizzazione, da un *output* della funzione *p*, del nome e cognome del docente selezionato preceduti dalla dicitura «Prof.»;
- **issue**[G] : fornisce la possibilità di inserire il *link* che si riferisce ad una determinata *issue*[G], tramite il numero associato a quest'ultima (e opzionalmente il *repository*[G], altrimenti viene di *default* inserito quella della documentazione);
- **pr**[G] : come sopra, ma per indicare le *pull request*[G];
- **doc** : che, preso in *input* il nome del documento e il testo da inserire, fornisce il *link* per quel documento con al suo posto il testo dato.

4.2.4 - Manutenzione

A causa dei continui sviluppi nel progetto *ALimitedGroup* è consapevole che l'infrastruttura subirà nel tempo cambiamenti e potrà causare possibili problemi: è per questo affidato all'**Amministratore** il compito di presiedere al controllo del regolare funzionamento della stessa, aggiornandone le funzionalità[G] qualora errori o cambiamenti lo rendano necessario.

4.3 - Processo di miglioramento

Il Processo di **Miglioramento** consiste, in base allo standard *ISO/IEC 12207:1995*, nello stabilire, consolidare, misurare, controllare e migliorare i processi utilizzati durante il ciclo di sviluppo di un *software*.

4.3.1 - Strumenti a supporto

Per le attività previste nel processo di Miglioramento, abbiamo deciso di utilizzare i seguenti strumenti:

- **GitHub**[G] :per tenere traccia, tramite un sistema di *ticketing*, con *issue*[G] e *milestone*[G] le attività svolte e il loro tempo per completarle.In aggiunta a questo, successivamente il periodo di «Retrospettiva», vengono individuate delle *issue*[G] per delle attività volte a migliorare il contenuto di ogni singolo documento prodotto dal grouppo.
  Inoltre, vengono automatizzate alcune azioni ripetitive nel processo di sviluppo come, per esempio, la *build* degli artefatti migliorando sia l'efficienza produttiva sia l'efficienza complessiva.

4.3.2 - Attività previste

Il processo prevede tre attività:

- **Inizializzazione del processo**, ovvero stabilire i processi organizzativi e realizzare l'opportuna documentazione;
- **Valutazione del processo**, ovvero stabilire un modo per valutare e documentare le revisioni dei vari processi con lo scopo di comprenderne l'efficienza e l'efficacia;
- **Miglioramento del processo**, ovvero stabilire come un processo inefficace o inefficiente possa essere migliorato, osservando le revisioni effettuati nei periodi in cui questo era in atto;

4.3.3 - Inizializzazione del processo
Come anticipato, è necessario stabilire dei processi organizzativi che guidino la realizzazione del progetto durante lo svolgimento di tutte le sue attività. Tali processi devono essere documentati ed è lo scopo principale del presente documento.

4.3.4 - Valutazione del processo
Stabiliti i processi, si rivela essenziale controllare l'andamento degli stessi: è necessario dunque individuare delle misurazioni appropriate allo scopo di valutare l'efficienza e l'efficacia dei processi adottati e realizzare periodici controlli sui dati provenienti da queste metriche. Per maggiori informazioni si veda la Sezione 6.

4.3.5 - Miglioramento del processo
Effettuate le misurazioni e controllate le stesse è necessario dedurre i processi che presentano problematiche e individuare opportune soluzioni che consentano di migliorare gli stessi. La documentazione dovrà dunque essere aggiornata per riflettere questi cambiamenti.

4.4 - Processo di Formazione
Il processo di formazione ha lo scopo di mantenere i membri di *ALimitedGroup* aggiornati sui cambiamenti effettuati, nonché definire come gli stessi debbano apprendere le nozioni necessarie allo svolgimento del Progetto.

4.4.1 - Strumenti a supporto
Per facilitare il processo di Formazione, è previsto che ogni singolo membro del gruppo acquisisca le competenze e le conoscenze fondamentali per il corretto svolgimento del progetto. Questo comprende sia le tecnologie necessarie per il progetto, sia ogni aspetto coinvolto nell'ingegneria del *software*.
In più, all'interno del gruppo, oltre alla formazione individuale, viene utilizzato il ***pair programming***[G] per consolidare l'utilizzo delle tecnologie e condividere le conoscenze apprese tra i membri del gruppo.
Tutti i membri del gruppo hanno accesso ai *repository*[G] relativi alla documentazione, *Poc*[G] e *MVP*[G] che consentono di poter lavorare ed accedere alle risorse in modo facile e ordinato. Inoltre, vengono utilizzate anche *repository*[G] private per testare aspetti nuovi delle tecnologie e condividere con il gruppo quanto imparato.

4.4.2 - Attività previste
Il processo di formazione include le seguenti attività:

- **Implementazione del processo**: ovvero la realizzazione dei meccanismi necessari alla formazione del personale;
- **Sviluppo di materiale per la formazione**: ovvero lo sviluppo di materiale atto alla formazione del gruppo;
- **Implementazione del piano per la formazione**: ovvero la realizzazione di un piano per formare e mantenere formato il gruppo.

4.4.3 - Implementazione del processo
In base a quanto previsto dallo standard *ISO/IEC 12207:1995* è necessario venga realizzata una revisione dei requisiti del progetto per comprendere le competenze che i membri di *ALimitedGroup* dovranno sviluppare per portare a compimento il progetto didattico.

ALimitedGroup ha perciò sin da subito analizzato i requisiti e compreso quali tecnologie sarà necessario approfondire dal punto di vista del progetto vero e proprio:

- Il linguaggio di programmazione **Go**[G]
- Il sistema di messaggistica per sistemi distribuiti **NATS**[G]
- Il sistema di containerizzazione **Docker**[G].

e dal punto di vista della produzione della documentazione:

- Il linguaggio di marcatura **Typst**[G]
- Il linguaggio di programmazione **Python**.

Saranno inoltre utili i seguenti servizi:

- Il *software* per il versionamento **Git**;
- Il servizio di *hosting* per progetti **GitHub**[G].

4.4.4 - Sviluppo di materiale per la formazione

Data la natura del progetto, ovvero a fini didattici, *ALimitedGroup* ha optato per la ricerca di materiale utile allo studio rispetto alla produzione propria. In particolar modo, il gruppo ha concluso che lo studio sarà affrontato sfruttando le seguenti risorse:

Strumento	Risorsa
Go	Risorse presenti sul sito del linguaggio; in particolare, l'autore di questa sezione ha trovato molto utile il *Tour of Go*
NATS	Videocorso realizzato dagli stessi sviluppatori del prodotto, reperibile su YouTube
Docker	Si consiglia di installare Docker Desktop, aprire un terminale, avviare l'immagine `docker/getting-started` (istruzioni presenti al link), ed infine aprire http://localhost in un browser. Si troverà un tutorial che spiega le basi di Docker.
Typst	La documentazione ufficiale ha un tutorial base al seguente link https://typst.app/docs/tutorial. Purtroppo, c'è una relativa carenza di risorse più avanzate, che tuttavia non dovrebbero essere necessarie per scrivere base e/o usando template già fatti.
Python	Python, data la sua caratteristica di essere un linguaggio molto facile ed estremamente diffuso, ha moltissime risorse disponibili su Internet. Appunto la sua facilità porta l'autore a pensare che una *cheatsheet* come https://learnxinyminutes.com/python/ sia ampiamente sufficiente per usufruire del linguaggio.
Git e **GitHub**	Viene consigliata la lettura del materiale messo a disposizione per il corso di **Metodi e Tecnologie per lo Sviluppo *software*** reperibili nel Moodle STEM dell'Università di Padova

Tabella 15: Materiale per la formazione dei membri di *ALimitedGroup*

4.4.5 - Implementazione del piano per la formazione

ALimitedGroup si promette di permettere durante la durata del progetto appositi spazi per permettere la formazione dei membri: durante i vari *sprint*[G] sarà possibile sfruttare i momenti in cui è stata assegnata un'attività di peso minore per poter dare maggior peso alla formazione personale.

ALimitedGroup inoltre si riserva, qualora lo ritenesse necessario, di pianificare degli spazi temporali durante gli *sprint*[G] dedicati esclusivamente allo studio di queste tecnologie, pur restando consapevole che un forte aiuto sarà dato dalla realizzazione effettiva del *Proof of Concept*[G] e del progetto finale.

5 - Metriche e standard per la Qualità

ALimitedGroup ha deciso di seguire lo *standard ISO/IEC 9126* per definire le metriche e i parametri che determinano la qualità di quanto realizzato. Lo standard identifica sei caratteristiche generali:

- **Funzionalità**[G]
- **Affidabilità**;
- **Efficienza**[G]
- **Usabilità**;
- **Manutenibilità**;
- **Portabilità**.

5.1 - Funzionalità

In merito alla **Funzionalità**[G], lo scopo è misurare quanto un prodotto soddisfa le esigenze del proponente[G]. Nello specifico misura:

- **Adeguatezza**, ovvero se il prodotto offre le funzioni necessarie a svolgere i compiti prefissati dal proponente[G]
- **Accuratezza**, ovvero se il prodotto fornisce i precisi effetti richiesti;
- **Interoperabilità**, ovvero se il prodotto riesce ad interagire con altri Sistemi definiti;
- **Conformità**, ovvero se il prodotto aderisce a determinati standard richiesti dal dominio d'uso del prodotto;
- **Sicurezza**, ovvero se il prodotto mette in sicurezza i dati degli Utenti, impedendone il reperimento ad individui non autorizzati.

5.2 - Affidabilità

In merito all'**Affidabilità**, è la capacità del prodotto di mantenere un determinato livello di prestazioni in qualsiasi momento, anche in periodi d'uso particolarmente intensi. Nello specifico misura:

- **Maturità**, ovvero l'arginare la possibilità che si verifichino errori o malfunzionamenti;
- **Tolleranza agli errori**, ovvero l'obiettivo di mantenere la prestazionalità del prodotto anche in caso di malfunzionamenti;
- **Recuperabilità**, ovvero la capacità del prodotto di ritornare al normale comportamento prestazionale a seguito di un malfunzionamento;
- **Aderenza**, ovvero la capacità di aderire a standard di affidabilità.

5.3 - Efficienza

L'**Efficienza** misura la capacità di fornire delle prestazioni che siano direttamente proporzionate alle risorse utilizzate. Nello specifico misura:

- **Comportamento temporale**, ovvero la capacità di fornire una risposta in tempi adeguati;
- **Utilizzo delle risorse**, ovvero un uso proporzionale delle risorse in rapporto all'utilizzo;
- **Conformità**, ovvero l'aderenza del prodotto a standard riguardanti l'efficienza.

5.4 - Usabilità

L'**Usabilità** consiste nel misurare quanto l'Utente finale è in grado di apprendere la modalità di utilizzo del prodotto. Si misura:

- **Comprensibilità**, ovvero quanto è facile comprendere i concetti del prodotto;

- **Apprendibilità**, ovvero quanto è semplice apprendere l'uso delle funzionalità[G] del prodotto;
- **Operabilità**, ovvero se è semplice per gli Utenti utilizzare il prodotto;
- **Attrattività**, ovvero se l'Utente trova positivo l'uso del prodotto;
- **Conformità**, ovvero se il prodotto aderisce a standard relativi all'usabilità.

5.5 - Manutenibilità

La **Manutenibilità** misura quanto il prodotto è semplice da modificare per permetterne l'evoluzione o la correzione. Nello specifico misura:

- **Analizzabilità**, ovvero la facilità d'ispezione del codice con lo scopo di cercare un errore;
- **Modificabilità**, ovvero quanto è difficile apportare una modifica;
- **Stabilità**, ovvero se il prodotto è in grado di arginare effetti indesiderati determinati da modifiche non corrette;
- **Testabilità**, ovvero la semplicità con cui il prodotto può essere testato.

5.6 - Portabilità

La **Portabilità** è la facilità con cui il prodotto può essere spostato da un ambiente di esecuzione ad un altro. Specificatamente si misura:

- **Adattabilità**, ovvero con che facilità il prodotto può essere adattato ad un diverso ambiente d'esecuzione;
- **Installabilità**, ovvero con che facilità il prodotto può essere installato su un altro ambiente d'esecuzione;
- **Conformità**, ovvero se il prodotto aderisce a convenzioni sulla portabilità;
- **Sostituibilità**, ovvero quanto è semplice sostituire un prodotto esistente con il prodotto da noi sviluppato.

5.7 - Nomenclatura delle Metriche

La nomenclatura utilizzata per le metriche è la seguente:

`MTipo##`

dove:

- **M** sta per **M**etrica
- **Tipo** può assumere uno tra questi valori:
  - **PC** ovvero **P**ro**c**esso
  - **PD** ovvero **P**ro**d**otto
- **##** è un numero crescente da 0. Il conteggio per il tipo **PC** e **PD** è separato.

Per tutte le definizioni, acronimi e abbreviazioni utilizzati in questo documento, si faccia riferimento al **Glossario**, fornito come documento separato, che contiene tutte le spiegazioni necessarie per garantire una comprensione uniforme dei termini tecnici e dei concetti rilevanti per il progetto.

6 - Metriche di Qualità del Processo

6.1 - Processi primari

6.1.1 - Fornitura

6.1.1.1 - *Earned Value* (EV)

- **Codice**: MPC01
- **Formula**: $\text{Earned Value} = \text{Budget at Completion} * \text{Percentuale di lavoro completato nello sprint}$
- **Descrizione**: L'indicatore Earned Value rappresenta il valore del lavoro *completato* rispetto al budget totale previsto.
  L'indicatore è utile per monitorare l'andamento del progetto e valutare se il lavoro svolto è in linea con le aspettative.

6.1.1.2 - *Planned Value* (PV)

- **Codice**: MPC02
- **Formula**: $\text{Planned Value} = \text{Budget at Completion} * \text{Percentuale di lavoro pianificato nello sprint}$
- **Descrizione**: L'indicatore Planned Value rappresenta il valore del lavoro *pianificato* rispetto al budget totale previsto.
  L'indicatore è utile per monitorare l'andamento del progetto e valutare se la pianificazione è rispettata. Il valore pianificato non può essere negativo e deve essere inferiore al BAC

6.1.1.3 - *Actual Cost* (AC)

- **Codice**: MPC03
- **Formula**: $\text{Actual Cost} = \text{Costo effettivo sostenuto nello sprint}$
- **Descrizione**: L'indicatore Actual Cost rappresenta il costo effettivo sostenuto per completare il lavoro nello sprint[G].
  L'indicatore è utile per monitorare l'andamento del progetto e valutare se i costi sono in linea con le aspettative.

6.1.1.4 - *Cost Performance Index* (CPI)

- **Codice**: MPC04
- **Formula**: $\text{Cost Performance Index} = \frac{\text{Earned Value}}{\text{Actual Cost}}$
- **Descrizione**: Il Cost Performance Index rappresenta il rapporto tra il valore del lavoro completato e il costo effettivo sostenuto.
  Un valore maggiore di 1 indica che il progetto sta rispettando il budget, un valore minore di 1 indica che il progetto sta superando il budget.

6.1.1.5 - *Schedule Performance Index* (SPI)

- **Codice**: MPC05
- **Formula**: $\text{Schedule Performance Index} = \frac{\text{Earned Value}}{\text{Planned Value}}$

- **Descrizione**: Lo Schedule Performance Index rappresenta il rapporto tra il valore del lavoro completato e il valore del lavoro pianificato.
  Un valore maggiore di 1 indica che il progetto sta rispettando la pianificazione, un valore minore di 1 indica che il progetto sta accumulando ritardi.

6.1.1.6 - *Estimate At Completion* (EAC)

- **Codice**: MPC06
- **Formula**: $\text{Estimate At Completion} = \frac{\text{Budget at Completion}}{\text{Cost Performance Index}}$
- **Descrizione**: La metrica Estimate At Completion rappresenta una proiezione del costo finale totale del progetto basata sulla performance attuale. Utilizza il CPI come indicatore di efficienza per correggere la stima iniziale (BAC). Se CPI < 1, EAC sarà maggiore del BAC, indicando un probabile sforamento del budget.

6.1.1.7 - *Estimate To Complete* (ETC)

- **Codice**: MPC07
- **Formula**: $\text{Estimate To Complete} = \text{Estimate At Completion} - \text{Actual Cost}$
- **Descrizione**: La metrica Estimate To Complete stima quanto costerà completare il lavoro rimanente del progetto. Si calcola sottraendo i costi già sostenuti (AC) dalla stima del costo finale totale (EAC). Utile per la pianificazione del budget residuo necessario.

6.1.1.8 - *Time Estimate At Completion* (TEAC)

- **Codice**: MPC08
- **Formula**: $\text{Time Estimate At Completion} = \frac{\text{Durata pianificata}}{\text{Schedule Performance Index}}$
- **Descrizione**: La metrica Time Estimate At Completion proietta la durata finale del progetto basandosi sulla performance temporale attuale. Utilizza SPI come indicatore di efficienza temporale per correggere la stima iniziale. Se SPI < 1, TEAC sarà maggiore della durata pianificata, indicando un probabile ritardo.

6.1.2 - Sviluppo

6.1.2.1 - *Requirements Stability Index*

- **Codice**: MPC09
- **Formula**: $\text{Requirements Stability Index} = \left(\frac{\text{NINIZ}-(\text{NAGG}+\text{NCAM}+\text{NCAN})}{\text{NINIZ}}\right) * 100$
- **Descrizione**:
  - **NINIZ**: **N**umero **Iniz**iali di Requisiti
  - **NCAM**: **N**umero **Cam**biamenti di Requisiti
  - **NCAN**: **N**umero **Can**cellati di Requisiti
  - **NAGG**: **N**umero **Agg**iunti di Requisiti

  Permette di misurare il numero di cambiamenti apportati ai requisiti nel corso del tempo.

6.2 - Processi di supporto

6.2.1 - Documentazione

6.2.1.1 - Indice di Gulpease

- **Codice**: MPC10
- **Formula**: $\text{Indice Gulpease} = 89 - \frac{\text{numero di lettere}}{\text{numero di parole}} * 100 + \frac{\text{numero di frasi}}{\text{numero di parole}} * 300$
- **Descrizione**: L'Indice Gulpease è un indice di leggibilità del testo. È utile per capire quanto un testo sia facile o difficile da leggere per un lettore medio. La formula tiene conto del numero di lettere, parole e frasi nel testo.

  Intervalli e interpretazioni

  - Indice ≥ 80: Il testo è molto facile da leggere, comprensibile per chi ha completato solo la scuola elementare.
  - Indice compreso 60 e 80: Il testo è di media difficoltà, adatto a chi ha completato la scuola dell'obbligo.
  - Indice compreso 40 e 60: Il testo è abbastanza difficile, comprensibile per chi ha almeno un'istruzione di livello superiore.
  - Indice < 40: Il testo è molto difficile da leggere, comprensibile per lettori con un'istruzione universitaria.

6.2.1.2 - Correttezza ortografica

- **Codice**: MPC11
- **Formula**: $\text{Correttezza ortografica} = \text{numero di errori ortografici}$
- **Descrizione**: La correttezza ortografica è un indicatore della qualità della documentazione. I documenti devono essere privi di errori ortografici.

6.2.2 - Verifica

6.2.2.1 - *Code Coverage*

- **Codice**: MPC12
- **Formula**: $\text{Code Coverage} = \left(\frac{\text{Linee di codice testate}}{\text{Linee di codice totali}}\right) * 100$
- **Descrizione**: Percentuale di codice coperto da Test automatizzati. Si raccomanda un coverage minimo del 80%.

6.2.2.2 - *Test Success Rate*

- **Codice**: MPC13
- **Formula**: $\text{Test Success Rate} = \left(\frac{\text{Test passati}}{\text{Test totali}}\right) * 100$
- **Descrizione**: Percentuale di Test automatizzati che passano con successo. Dovrebbe mantenersi al 100% data la natura del progetto

6.2.3 - Gestione della Qualità

6.2.3.1 - *Quality metrics satisfied*

- **Codice**: MPC14

- **Formula**: $\text{Quality metrics satisfied} = \left(\frac{\text{Numero metriche soddisfatte}}{\text{Numero metriche totali}}\right) * 100$
- **Descrizione**: Percentuale di soddisfacimento delle metriche

6.3 - Processi organizzativi

6.3.1 - Gestione dei Processi

6.3.1.1 - *Time Efficiency*

- **Codice**: MPC15
- **Formula**: $\text{Time Efficiency} = \left(\frac{\text{Ore produttive}}{\text{Ore totali}}\right) * 100$
- **Descrizione**: Valutazione del rapporto tra le ore utilizzate e quelle effettivamente produttive.

7 - Metriche di Qualità del Prodotto

7.1 - Funzionalità

7.1.1 - Requisiti obbligatori soddisfatti

- **Codice**: MPD01
- **Formula**: $\text{Requisiti obbligatori soddisfatti} = \frac{\text{Numero di requisiti obbligatori soddisfatti}}{\text{Numero di requisiti obbligatori totali}} * 100$
- **Descrizione**: L'indicatore Requisiti obbligatori soddisfatti rappresenta la percentuale di requisiti obbligatori soddisfatti rispetto al totale dei requisiti obbligatori.
  L'indicatore è utile per monitorare il grado di soddisfacimento dei requisiti essenziali del progetto.

7.1.2 - Requisiti desiderabili soddisfatti

- **Codice**: MPD02
- **Formula**: $\text{Requisiti desiderabili soddisfatti} = \frac{\text{Numero di requisiti desiderabili soddisfatti}}{\text{Numero di requisiti desiderabili totali}} * 100$
- **Descrizione**: L'indicatore Requisiti desiderabili soddisfatti rappresenta la percentuale di requisiti desiderabili soddisfatti rispetto al totale dei requisiti desiderabili.
  L'indicatore è utile per monitorare il grado di soddisfacimento dei requisiti opzionali del progetto.

7.1.3 - Requisiti opzionali soddisfatti

- **Codice**: MPD03
- **Formula**: $\text{Requisiti opzionali soddisfatti} = \frac{\text{Numero di requisiti opzionali soddisfatti}}{\text{Numero di requisiti opzionali totali}} * 100$
- **Descrizione**: L'indicatore Requisiti opzionali soddisfatti rappresenta la percentuale di requisiti opzionali soddisfatti rispetto al totale dei requisiti opzionali.
  L'indicatore è utile per monitorare il grado di soddisfacimento dei requisiti aggiuntivi del progetto.

7.2 - Affidabilità

7.2.1 - *Branch Coverage*

- **Codice**: MPD04
- **Formula**: $\text{Branch Coverage} = \left(\frac{\text{Rami testati}}{\text{Rami totali}}\right) * 100$
- **Descrizione**: Percentuale di rami del codice coperti da Test automatizzati. Si raccomanda un coverage minimo del 60%.

7.2.2 - *Statement Coverage*

- **Codice**: MPD05
- **Formula**: $\text{Statement Coverage} = \left(\frac{\text{Istruzioni testate}}{\text{Istruzioni totali}}\right) * 100$
- **Descrizione**: Percentuale di istruzioni del codice coperte da Test automatizzati. Si raccomanda un coverage minimo del 70%.

7.2.3 - *Failure Density*

- **Codice**: MPD06
- **Formula**: $\text{Failure Density} = \left(\frac{\text{Numero di difetti rilevati}}{\text{KLOC}}\right)$

- **Descrizione**: Numero di failure per 1000 linee di codice (**KLOC**, ***K**ilo **L**ines **O**f **C**ode*). Un valore superiore a 0.5 indica possibili problemi di affidabilità.

7.3 - Usabilità

7.3.1 - *Time on Task*

- **Codice**: MPD07
- **Formula**: $\text{Time on Task} = \text{Tempo medio per completare un'attività}$
- **Descrizione**: Tempo medio impiegato per completare un'attività. Indica l'usabilità del prodotto.

7.3.2 - *Error Rate*

- **Codice**: MPD08
- **Formula**: $\text{Error Rate} = \left(\frac{\text{Errori totali}}{\text{Azioni totali}}\right) * 100$
- **Descrizione**: Percentuale di errori commessi durante l'utilizzo del prodotto. Dovrebbe essere inferiore al 5%.

7.4 - Efficienza

7.4.1 - *Response Time*

- **Codice**: MPD09
- **Formula**: $\text{Response Time} = \text{Tempo medio di risposta}$
- **Descrizione**: Tempo medio impiegato per rispondere a una richiesta. Indica l'efficienza del prodotto. Un tempo di risposta inferiore a 2 secondi è considerato accettabile, mentre un tempo inferiore a 1 secondo è considerato ottimo.

7.5 - Manutenibilità

7.5.1 - *Code Smells*

- **Codice**: MPD10
- **Formula**: $\text{Code Smells} = \left(\frac{\text{Numero di code smells}}{\text{KLOC}}\right)$
- **Descrizione**: Numero di code smells per 1000 linee di codice. Un valore superiore a 10 indica possibili problemi di manutenibilità.

7.5.2 - *Coefficient of Coupling* (CoC)

- **Codice**: MPD11
- **Formula**: $\text{Coefficient of Coupling} = \left(\frac{\text{Numero di dipendenze}}{\text{Numero di componenti}}\right)$
- **Descrizione**: Numero medio di dipendenze tra le componenti del sistema. Un valore superiore a 0.4 indica un accoppiamento eccessivo tra le componenti.

7.5.3 - *Cyclomatic Complexity*

- **Codice**: MPD12
- **Formula**: $\text{Cyclomatic Complexity} = E - N + P$
- **Descrizione**:
  - E = numero di archi nel grafo di controllo

- N = numero di nodi nel grafo di controllo
- P = numero di componenti connesse da ogni arco

Misura la complessità del codice contando i percorsi linearmente indipendenti. Un valore superiore a 10 indica codice complesso che potrebbe richiedere refactoring.