Norme di Progetto

Jackpot Coding

March 10, 2024

Versione	Data	Autore	Verificatore	Modifica
V1.0.0	10/03/2024	-	R. Simionato	Approvazione versione 1.0.0
V0.1.1	3/03/2024	R. Simionato	M. Gobbo	Aggiunto corsivo per i termini in inglese e apice G per Glossario
V0.1.0	2/03/2024	-	R. Simionato	Verifica documento
V0.0.10	10/02/2024	M. Gobbo	R. Simionato	Stesura della sezione 4.3 Formazione
V0.0.9	12/01/2024	M. Camillo	R. Simionato	Stesura della sezione 4.1 Gestione di processo
V0.0.8	06/01/2024	M. Gobbo	R. Simionato	Stesura parte 2.1, stesura iniziale parte 2.2
V0.0.7	04/01/2024	M. Gobbo	R. Simionato	Stesura parte 1.4, stesura iniziale parte 3.2, 3.3, 3.4
V0.0.6	28/11/2023	E. Gallo	R. Simionato	Stesura delle sezioni 1.1, 1.2, 1.3 dell'Introduzione
V0.0.5	27/11/2023	G.Moretto	R. Simionato	Aggiornamento sezioni da 3.1.8 a 3.1.12
V0.0.4	19/11/2023	G.Moretto	R. Simionato	Aggiunti *link* all'indice Aggiornamento sezioni 3.1.4, 3.1.5, 3.1.6 , 3.1.7
V0.0.3	17/11/2023	R.Simionato	G.Moretto	Aggiunta sezione 4.2 Infrastruttura
V0.0.2	16/11/2023	G. Moretto	R. Simionato	Aggiunta sezione 3.1 documentazione e stesura preliminare
V0.0.1	15/11/2023	G. Moretto	R. Simionato	Creata struttura del documento

Indice

1 Introduzione

1.1 Scopo del documento

Il presente documento ha lo scopo di individuare le *best practices* di progetto e definire correttamente il *Way of Working*[G] dell'attività produttiva, in modo tale da permettere un lavoro omogeneo, coeso e attuabile in modo asincrono. Per garantire un approccio incrementale, vengono indicate nel registro delle modifiche, tutte le versioni e lo storico del documento, seguendo le regole riportate nel documento stesso. Nei prossimi punti vengono precisamente descritte le convenzioni sull'uso degli strumenti utilizzati, sia di comunicazione sia di realizzazione dei processi di lavoro interno ed esterno.

1.2 Scopo del prodotto

L'obiettivo del progetto è quello di realizzare un'applicazione che esplori a fondo la capacità dei *Large Language Models* (*LLM*[G]) di generare codice più o meno corretto. In questo particolare caso, il capitolato[G] cerca di approfondire l'uso degli *LLM* e la tecnica di *prompting*[G] per generare codice *SQL*[G]. L'applicativo deve:

1. Archiviare la descrizione della struttura di un *database*[G].
2. Fornire un modo per richiedere, tramite linguaggio naturale[G], una visione di alcuni dati del *database* a scelta.
3. Combinare la richiesta con le varie informazioni necessarie della struttura del *database*, creando un *prompt*[G] che verrà poi sottoposto ad un sistema di *AI*[G].

Sia il linguaggio di programmazione sia l'*LLM* utilizzato dalla piattaforma, sono a discrezione del gruppo. Come linguaggio di programmazione è stato scelto *Python*[G], per semplicità e la presenza di numerose librerie. Oltre ai requisiti[G] obbligatori, l'azienda Zucchetti S.p.A., da la possibilità di sviluppare l'applicazione in maniere più approfondita. I requisiti opzionali sono:

1. Visualizzare la frase *SQL* prodotta dal sistema *AI*.
2. Sviluppare l'*input*[G] vocale della frase in linguaggio naturale.
3. Verificare la correttezza della frase *SQL* prodotta dal sistema di *AI*.
4. Implementare la gestione di più basi di dati.
5. Utilizzare modelli[G] *LLM* alternativi a *ChatGPT*[G].
6. Addestrare un modello in modo specifico per la traduzione di frasi di interrogazione in italiano a frasi *SQL*.

1.3 Glossario

Per evitare incomprensioni relative ai vari termini utilizzati nel documento, viene fornito un Glossario, nell'omonimo *file*, in modo da dare una definizione precisa di ogni termine che possa risultare ambiguo. Ogni termine presente nel Glossario verrà indicato nel documento con una lettera G come apice.

1.4 Riferimenti

1.4.1 Riferimenti normativi

Capitolato C9 *ChatSQL*: creare frasi *SQL* da linguaggio naturale

- `https://www.math.unipd.it/~tullio/IS-1/2023/Progetto/C9.pdf`

1.4.2 Riferimenti informativi

Slide dell'insegnamento del corso di Ingegneria del Software:

- `https://www.math.unipd.it/~tullio/IS-1/2023/Progetto/T3.pdf`
- `https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/T4.pdf`
- `https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/T5.pdf`

2 Processi primari

2.1 Fornitura

2.1.1 Descrizione, Scopo e Aspettative

Il processo di fornitura è il fondamento che orienta ogni fase del progetto, mirando a una comprensione completa delle richieste del proponente^{G} per soddisfarle in modo impeccabile. Questo approccio si sviluppa attraverso diverse fasi chiave:

- Avvio: inizia con un'approfondita comprensione delle richieste del proponente e una valutazione della loro fattibilità, culminando nell'instaurazione di un accordo contrattuale.
- Contrattazione: definisce chiaramente i termini e le condizioni, garantendo una base solida per la collaborazione.
- Pianificazione: determina compiti, tempistiche e risorse necessarie per la realizzazione del progetto.
- Esecuzione: attua il piano e monitora costantemente il progresso per assicurare il rispetto dei tempi e degli obiettivi.

- Revisione e valutazione: valuta le fasi precedenti per apportare eventuali correzioni o miglioramenti.
- Consegna e completamento: conclude il processo garantendo la piena soddisfazione del proponente.

Il processo di fornitura mira a eliminare ogni dubbio legato alle richieste del proponente, perseguendo la massima comprensione delle sue esigenze al fine di soddisfarle appieno. Questo approccio si concentra sull'instaurazione e il mantenimento di un dialogo attivo per garantire una collaborazione efficace e senza incomprensioni.
Durante l'intero svolgimento del progetto ci impegniamo a mantenere un costante e proficuo scambio con il proponente, l'azienda Zucchetti S.p.A. Questo coinvolgerà:

- Determinare le necessità del prodotto finale per assicurare una chiara comprensione delle aspettative del proponente rispetto al prodotto finale.
- Stabilire i tempi di consegna e definire con precisione le scadenze per garantire un processo fluido.
- Ricevere *feedback* sul lavoro svolto e accogliere i commenti per migliorare continuamente il nostro impegno.
- Chiarire dubbi e incomprensioni per risolvere ogni possibile incertezza attraverso un dialogo costante e chiaro.
- Definire vincoli e requisiti per stabilire parametri chiari per il prodotto finale e per ogni fase intermedia del processo.

2.1.2 Piano di Qualifica

Il documento rappresenta i compiti e le attività fondamentali legate al progetto che devono essere condotte dal Verificatore. Queste attività sono cruciali per assicurare la qualità del prodotto finale mediante l'utilizzo di approcci di verifica e validazione.
In dettaglio, il Piano di Qualifica^{G} è strutturato come segue:

- Qualità di Prodotto: stabilisce le metriche per valutare e garantire la qualità del prodotto finale.
- Qualità di Processo: definisce le metriche per il controllo della qualità dei processi adottati.
- Specifica dei *Test*: descrive in maniera specifica i *test* eseguiti sul prodotto per assicurare il soddisfacimento dei requisiti stabiliti.
- Resoconto attività di verifica: fornisce un riassunto delle attività di verifica svolte in un determinato periodo di tempo o fase del progetto
- Valutazioni attività di verifica: fornisce un'analisi critica delle attività di verifica svolte durante il processo di sviluppo del *software*.

2.1.3 Piano di Progetto

Il Piano di Progetto[G], situato nella fase iniziale di pianificazione, funge da strumento essenziale per la definizione delle attività, delle risorse necessarie e delle tempistiche del progetto stesso. Trattandosi di un documento ufficiale, è soggetto a versionamento[G] e approvazione, e rappresenta una guida chiara e concisa degli obiettivi e degli elementi chiave necessari al raggiungimento di tali obiettivi. Il Piano di Progetto abbraccia diverse componenti fondamentali:

- Analisi dei Rischi: valuta e anticipa le possibili difficoltà che potrebbero insorgere durante lo sviluppo del progetto, al fine di prevenire ostacoli e ritardi. Questi rischi sono classificati in due categorie principali:
  - Rischi Tecnologici.
  - Rischi Interni
    - Rischi Organizzativi.
    - Rischi Comunicativi.
- Modello di Sviluppo: determina il modello da adottare per la gestione dello sviluppo.
- Pianificazione: definisce una sequenza temporale delle attività di progetto, stabilendo le relative scadenze.
- Preventivo: sintetizza in modo schematico l'aspetto economico e temporale complessivo del progetto, fornendo anche una suddivisione per fasi.
- Consuntivo: monitora l'andamento del progetto rispetto al preventivo iniziale.

2.2 Sviluppo

2.2.1 Descrizione, Scopo e Aspettative

Il processo di sviluppo comprende tutte le attività coinvolte nello sviluppo del prodotto *software*, tra cui l'analisi dei requisiti, la progettazione, la codifica, l'integrazione, i *test*, l'installazione e l'accettazione, necessari per il completamento del *software*.

L'obiettivo principale del processo di sviluppo è definire i compiti e le attività necessarie per la codifica del prodotto *software* richiesto, delineando in modo chiaro e definito i ruoli da adottare.
Il Gruppo *"Jackpot Coding"* si impegna a determinare gli obiettivi di sviluppo e *design* necessari per l'implementazione corretta del prodotto finale, garantendo che rispetti le richieste del proponente provenienti dall'Analisi dei Requisiti[G] e dal *design ER*, oltre a superare i *test* di verifica e validazione.

Nel perseguire il processo di sviluppo, ci aspettiamo:

- Definizione dei Vincoli Tecnologici: chiara definizione dei requisiti tecnologici.
- Determinazione degli Obiettivi di Sviluppo: identificazione chiara degli obiettivi relativi allo sviluppo.
- Definizione dei Vincoli di *Design*: specificazione dei vincoli di *design*.
- Conclusione di un Prodotto Finale Che Soddisfi le Richieste del Proponente: garanzia che il prodotto finale superi i *test* e soddisfi appieno i requisiti richiesti dal proponente.

2.2.2 Analisi Requisiti

L'analisi dei requisiti è una fase preliminare fondamentale nel definire chiaramente il prodotto finale. Gli Analisti raccolgono informazioni dettagliate sul contesto d'uso e sugli obiettivi del prodotto per:

- Identificare il fine del prodotto, in linea con le richieste del proponente.
- Definire le funzionalità, identificando chi interagisce con il sistema e come (attori e casi d'uso).
- Stabilire i criteri per la qualifica e il controllo dei *test*.
- Effettuare confronti interni ed esterni, delineando una visione generale del prodotto e quantificando gli impegni basandosi sui ruoli.

L'obiettivo è capire appieno la specifica del capitolato, arricchendola tramite dialogo col proponente per garantire una corretta realizzazione del prodotto.
Questa attività punta a individuare tutti i requisiti diretti e indiretti richiesti per il prodotto, suddividendo il problema in requisiti elementari per semplificarne lo sviluppo.
La documentazione finale raccoglie tutti i requisiti richiesti dal proponente, fungendo da guida per gli sviluppatori e come riferimento per i *test*, ottenendo così una stima della mole di lavoro necessaria.
Per identificare questi requisiti, è essenziale leggere attentamente il capitolato e mantenere un dialogo costante col proponente.

Casi d'uso
I casi d'uso[G] rappresentano il modo in cui il prodotto viene utilizzato e il suo comportamento. Sono delineati attraverso diagrammi *UML*[G] e ognuno comprende:

- Un codice identificativo univoco.
- L'attore principale coinvolto.
- Condizioni iniziali necessarie.

- Condizioni finali una volta completato il caso.
- Lo scenario principale del caso d'uso.
- Possibili scenari alternativi o casi simili.
- Estensioni che possono verificarsi durante l'esecuzione del caso d'uso.

Il nostro gruppo ha scelto di denominarli secondo uno schema preciso:

UC[Numero Caso D'Uso].[Numero Sottocaso]-[Nome Caso D'Uso]

Requisiti
I requisiti derivano da varie fonti, tra cui:

- L'esame approfondito della documentazione iniziale del progetto.
- Discussioni e confronti tra i membri del *team*.
- Dialogo diretto e confronto con il cliente o il proponente del progetto.
- L'analisi dei modi in cui il prodotto viene utilizzato, come espresso nei casi d'uso.

Anche per i requisiti il gruppo ha definito uno *standard* da adottare che è il seguente:

R[Tipo][Numero Requisito]

Dove il tipo può essere:

- **F** - Requisito Funzionale
- **Q** - Requisito Qualità
- **V** - Requisito Vincolo
- **P** - Requisito Prestazionale

2.2.3 Progettazione

La fase di progettazione costituisce un passaggio fondamentale per delineare la struttura essenziale del progetto, basandosi sui requisiti individuati nell'analisi e poi definiti nell'Analisi dei Requisiti. Questo compito è affidato ai progettisti che si dedicano a pianificare l'implementazione di tutti i requisiti specificati. Il nostro *team*, in questa fase del ciclo di vita del *software*, ha le seguenti aspettative:

- Tradurre tutti i requisiti in specifiche dettagliate che coprano ogni aspetto del sistema.
- Assicurare una comprensione agevole per facilitare la manutenzione.

- Ottenere l'approvazione per procedere alla fase di sviluppo.

Questo processo si articola in tre livelli:

1. *Design* dell'interfaccia: qui si opera a un livello elevato di astrazione rispetto al funzionamento interno del sistema. L'attenzione è focalizzata sulle tecnologie che saranno utilizzate nella fase di sviluppo, portando alla creazione di un *Proof of Concept*[G].

2. Progettazione architetturale: si seleziona la struttura generale del sistema, definendo un quadro a alto livello che trascura i dettagli interni dei principali componenti del prodotto. Inoltre, si definiscono i *test* di integrazione.

3. Progettazione dettagliata: si specificano gli elementi interni di tutti i principali componenti e le specifiche architetturali del prodotto. Si definiscono anche i diagrammi delle classi e i *test* di unità per ciascun componente. Questa fase costituisce il fondamento del prodotto (*Product Baseline*[G]).

Il fine dell'attività di progettazione è concretizzare l'architettura del sistema. Inizialmente come già detto quindi ci si avvale di un *Proof of Concept*, che funge da demo prototipale del sistema, presentato alla *Requirements & Technology*[G] per approvazione.

2.2.4 Codifica

Una volta completata la fase di progettazione del prodotto, i membri del gruppo con il ruolo di Programmatori passeranno alla fase di codifica, traducendo le specifiche dei requisiti e i documenti di progettazione in codice effettivo. L'obiettivo primario di questa fase è rendere tangibile il prodotto *software* desiderato attraverso il processo di programmazione.
Durante la fase di codifica, le aspettative includono:

- Completare lo sviluppo del prodotto finale rispettando le richieste del proponente e garantendone la qualità.

- Assicurare che il codice prodotto sia facilmente comprensibile per chi lo legge.

La codifica mira a concretizzare la progettazione trasformando il concetto in *software* funzionante. L'obiettivo è generare un prodotto *software* che soddisfi le caratteristiche e i requisiti concordati con il proponente. Affinché il codice sia facilmente gestibile nelle fasi successive di manutenzione, modifica, verifica e validazione, è importante rispettare alcune convenzioni che ne agevolino la comprensione.

3 Processi di supporto

3.1 Documentazione

3.1.1 Descrizione, Scopo e Aspettative

Si descrive come documentazione *software* come illustrazioni e/o testo che accompagnano il progetto *software*, sia come documenti separati che come commenti nel codice sorgente.
Il suo scopo è quello di rendere comprensibile il progetto nella sua interezza per chi lo sviluppa, chi lo mantiene ed il suo utilizzatore finale. Questo viene fatto descrivendo le attività effettuate nel ciclo di vita del *software* nel modo più chiaro possibile.
Le aspettative per questo processo sono:

- Documentare tramite verbale ogni incontro interno ed esterno
- Avere una struttura fissa per ogni tipo di documento
- Utilizzare una procedura per la verifica dei documenti
- Facilitare il lavoro autonomo per la gestione dei documenti

3.1.2 Ciclo di vita di un documento

Il ciclo di vita di un documento è diviso nelle seguenti fasi:

- Pianificazione: la struttura ed il contenuto vengono decisi dai componenti del gruppo, varie sezioni sono divise e assegnate tramite la piattaforma *Jira*.
- Impostazione: viene implementata la struttura del documento.
- Scrittura: le persone incaricate scrivono il contenuto del documento seguendo la struttura e le indicazioni per il tipo di documento. Il documento viene scritto nel linguaggio *LaTeX*[G] e caricato nella *repository*[G] in un *branch*[G] dedicato.
- Verifica: le persone incaricate verificano il contenuto e, se necessario, apportano modifiche allo stesso.
- Approvazione: il responsabile di progetto approva il documento dopo la verifica. Una volta approvato il documento viene aggiunto al ramo[G] *main* della *repository*.

3.1.3 Modelli dei documenti

Viene creato un modello (o *template*) in formato *LaTeX* per ogni tipo di documento, in questo vengono definite l'intestazione e le sezioni del documento.

3.1.4 Struttura di un documento

Intestazione
L'intestazione contiene il logo dell'università di Padova, indicando il corso di laurea, il corso relativo al progetto e l'anno accademico. Inoltre è presente il logo, il nome e l'indirizzo email del gruppo (*Jackpot Coding*). Sono prensenti i dati che identificano il documento: data del documento, oggetto, redattori, verificatori, il responsabile, lo scriba ed in fine l'uso (interno o esterno) e, se esterno, i destinatari del documento.

Cambiamenti
La sezione cambiamenti contiene una tabella con le seguenti colonne:

- Versione: la versione raggiunta
- Data: quando sono state effettuate le modifiche
- Autore: chi ha effettuato le modifiche
- Verificatore: chi ha verificato le modifiche
- Modifica: quali modifiche sono state apportate

Indice
L'indice riporta l'elenco delle sezioni e sotto-sezioni del documento riportando a quale pagina si trovano e contenendo un *link* per la navigazione rapida.

3.1.5 Struttura di un verbale

Oltre alle sezioni sopra citate vengono inserite le seguenti sezioni.

Orario
Contiene l'ora di inizio e fine dell'incontro.

Partecipanti
Contiene una tabella che presenta il nome e cognome dei partecipanti e la durata della loro presenza all'incontro.

Sintesi dell'incontro
Contiene una versione riassuntiva degli argomenti trattati durante l'incontro.

Decisioni
Contiene le decisioni prese durante l'incontro, queste si traducono i compiti assegnati ai membri del gruppo.

3.1.6 Nome dei *file*

Il nome del *file* deve essere esplicativo, indicandone il contenuto, e la versione.

3.1.7 Stile del testo

Grassetto: applicato ai titoli e a parole di particolare importanza.
`Monospace`: applicato a frammenti di codice e percorsi di cartelle e *file*.
Sottolineato: applicato a *link* presenti nel documento.
Corsivo: applicato ai termini inglesi.

3.1.8 Norme tipografiche

- Gli elementi di un elenco finiscono con il carattere punto.
- La prima lettera degli elementi di un elenco è sempre maiuscola.
- I nomi delle sezioni e sottosezioni iniziano con una lettera maiuscola.

3.1.9 Glossario

Il Glossario è un documento che contiene termini e/o sigle il quale significato è importante per il progetto. Il documento viene caricato sulla *repository* per facilitare la consultazione e la stesura da parte del gruppo. I termini presenti nel glossario presentano una G in apice.

3.1.10 Sigle

3.1.11 Immagini

Le immagini presenti nei documenti sono caricate nella cartella `/docs/src/assets` della *repository*.

3.1.12 Strumenti utilizzati

Overleaf
Viene utilizzato per la scrittura collaborativa di documenti in formato *LaTeX*.

GitHub
Nella *repository* ufficiale vengono caricate le versioni dei documenti utilizzando la procedura descritta dal ciclo di vita come da sezione 3.1.2.

Editor *LaTeX*
Ogni componente del gruppo ha la libertà di utilizzare l'*editor LaTeX* che preferisce (viene consigliato TeXstudio).

3.2 Gestione della configurazione

3.2.1 Descrizione e scopo

La gestione della configurazione è un processo cruciale per mantenere la coerenza del *software* nel tempo, garantendo il corretto funzionamento del sistema nonostante le eventuali modifiche apportate. Problemi nella configurazione potreb-

bero causare incoerenze o problemi di aderenza alle normative, compromettendo le operazioni aziendali, è quindi importante perseguire questo punto.

3.2.2 Sistema di versionamento

Il versionamento offre la possibilità di registrare tutte le modifiche apportate a un documento. Questo sistema consente di ripristinare una versione precedente del documento e visualizzare chiaramente tutte le modifiche fatte nel corso del tempo, incluso l'autore di ciascuna modifica.
Nel nostro gruppo, abbiamo adottato il seguente codice per identificare le diverse versioni di un documento:

[x].[y].[z]

La versione viene indicata con tre numeri separati da un punto, come visto sopra, indicati in questa sezione come x.y.z.

x) Il numero viene aumentato solo quando il documento è pronto per essere caricato nel *main branch*[G].

y) Il numero viene aumentato solo quando le modifiche sono verificate.

z) Il numero viene aumentato quando il documento subisce una modifica.

3.2.3 *Repository*

Tecnologie utilizzate
Il nostro *team* ha scelto di utilizzare *GitHub*[G] come strumento per gestire la configurazione, facendo affidamento sul sistema di controllo di versione distribuito di *Git*[G].
La *repository* è accessibile pubblicamente tramite il seguente *link*:

`https://github.com/Jackpot-Coding`

Per agevolare la consultazione e la navigazione dei documenti, abbiamo creato un sito dedicato tramite *GitHub*. È possibile accedervi utilizzando il seguente riferimento:

`https://jackpot-coding.github.io/chatSQL/`

Struttura *Repository*
La struttura della *repository* si basa sulla strategia **GitHub Flow** che distingue due tipi di rami:

- Ramo principale ***Main*** in cui è presente la documentazione ed il codice verificato e funzionante;
- Rami secondari utilizzati per la redazione dei documenti e l'implementazione di nuovo codice che verranno uniti al ramo principale tramite il *merging*[G] una volta terminate e verificate le modifiche.

Nel ramo *Main* all'interno della cartella "docs" si trovano i documenti divisi nelle sottocartelle:

- "interni", che contiene i documenti utili al gruppo;
- "esterni", che contiene i documenti che verranno condivisi con i committenti e il proponente;
- "src", che contiene i *file* sorgente e le immagini o *assets* necessari per la compilazione dei documenti.

Sempre nel ramo *Main* all'interno della cartella "src" si trovano i *file* sorgenti del prodotto.

3.3 Gestione della qualità

3.3.1 Descrizione, scopo

La gestione della qualità in un progetto è fondamentale per assicurare che i processi e i prodotti soddisfino le richieste del cliente con la massima qualità possibile. È altrettanto importante perseguire un miglioramento continuo attraverso monitoraggi e valutazioni retrospettive. Questo processo coinvolge una serie di attività volte a garantire che i risultati e le *performance* del progetto siano allineati agli obiettivi e ai requisiti stabiliti. Questa sezione si concentra sull'impegno del gruppo nella gestione della qualità del progetto, cercando di assicurare un'implementazione accurata e coerente di tali processi.

3.3.2 Piano di qualifica

Per assicurare un mantenimento della qualità il gruppo si avvale del documento Piano Di Qualifica nel quale vengono descritti:

- Obiettivi della qualità di Prodotto;
- Obiettivi della qualità di Processo;
- Metodi per la misurazione di questi tramite metriche;
- Definizione dei *test* da effettuare;
- Cruscotto^{G} per la visione dello stato del raggiungimento degli obiettivi;

3.4 Verifica

3.4.1 Descrizione, scopo

La verifica del *software* è il processo valutativo che assicura l'accuratezza della fase di sviluppo per costruire il prodotto desiderato. Si esegue durante lo sviluppo per rilevare difetti precocemente e garantire che il *software* soddisfi i requisiti del cliente. L'obiettivo della verifica è controllare la correttezza e

completezza del prodotto, garantendo che sia conforme alle aspettative. Si basa su analisi e *test* per assicurare l'assenza di errori nel *software* e nella documentazione. Questa sezione mira a definire l'approccio scelto dal gruppo per attuare il processo di verifica.

3.4.2 Analisi statica

L'analisi statica consiste nell'esaminare il codice o la documentazione prima dell'esecuzione del *software*, garantendo il soddisfacimento dei requisiti specificati. Questo metodo si applica non solo al codice ma anche ai documenti, focalizzandosi sugli aspetti statici del sistema *software*, come le convenzioni del codice o le metriche del *software*. Include sia *test* manuali che automatizzati, come l'analisi di coerenza. Questo tipo di analisi si divide in due metodi principali:

- *Walkthrough*: prevede una lettura ampia per individuare possibili errori senza conoscere la loro posizione esatta.
- *Inspection*: punta a individuare errori specifici con un approccio mirato.

In particolare, nell'analisi dei documenti, si utilizzano tecniche simili di *Walkthrough* e *Inspection*.
Inizialmente, l'attività di *Walkthrough* è prevalente rispetto all'*Inspection*, ma con il progredire del progetto e la ripetizione delle verifiche sulla documentazione, si sviluppa una lista degli errori comuni (Lista di Controllo), consentendo l'adozione più ampia dell'*Inspection*, che si rivela più efficiente.

3.4.3 Analisi dinamica

L'analisi dinamica si svolge simultaneamente all'esecuzione del *software* ed è principalmente costituita dalla fase di *test*. A differenza dell'analisi statica, coinvolge l'esecuzione effettiva del sistema e dei suoi componenti.
Per garantire che il prodotto funzioni correttamente durante l'esecuzione, il gruppo utilizza una serie di *test* eseguibili in tempo reale. È essenziale che tali *test* siano ripetibili, pertanto si identificano strumenti per automatizzarne l'esecuzione.
Questo approccio dinamico è cruciale per rilevare problemi e difetti durante l'esecuzione effettiva del *software*, fornendo una maggiore sicurezza sul corretto funzionamento del prodotto.

4 Processi Organizzativi

4.1 Gestione Di Processo

4.1.1 Scopo

Il processo mira a sviluppare il documento noto come Piano di Progetto. Questo strumento è essenziale per organizzazione e la gestione dei ruoli di ciascun mem-

bro del gruppo coinvolto. L'obiettivo principale è fornire una guida chiara e strutturata che consenta ai membri del *team* di comprendere appieno le loro responsabilità e contribuire in modo efficace al successo del progetto.

4.1.2 Aspettative

Le aspettative chiave legate al processo di gestione dei processi sono:

- **Creazione di un documento dettagliato che delinei le fasi, le attività e le risorse necessarie per il progetto:** questo documento serve come guida chiara per tutti i membri del *team*.
- **Definizione dei ruoli dei membri del gruppo:** chiara definizione e assegnazione dei ruoli e delle responsabilità di ciascun membro del *team*. Questo contribuisce a garantire una distribuzione equa del lavoro e una comprensione completa delle responsabilità individuali.
- **Definizione di un piano per l'esecuzione dei compiti programmati:** sviluppo di un piano operativo che stabilisca le tempistiche, le scadenze e le sequenze di attività per garantire l'esecuzione efficiente e tempestiva dei compiti programmati. Questo passo è fondamentale per mantenere il progetto in linea con le tempistiche previste e per affrontare eventuali deviazioni in modo proattivo.

4.1.3 Coordinamento

Comunicazioni Nel contesto delle comunicazioni interne tra gli studenti del gruppo, si fa uso di diversi canali di comunicazione. La selezione di tali canali è stata attentamente valutata per evitare sovrapposizioni e suddividere i compiti specifici di ciascun canale, al fine di evitare confusioni nella ricerca delle informazioni scambiate. I principali canali utilizzati includono:

- ***Discord*:**
  - **Testuale:** utilizzato per lo scambio di informazioni testuali, come *file*, *link* e bozze di appunti.
  - **Canale "Chat-Generale":** adibito allo scambio informale di dettagli riguardanti note riunioni e la condivisione di documenti.
  - **Canale "Info":** riservato allo scambio formale di informazioni relative al nucleo centrale del progetto.
  - **Vocale:** utilizzato per condurre incontri interni attraverso canali vocali.
- ***Telegram*:** utilizzato per aggiornamenti rapidi tra i membri del gruppo, richieste di chiarimenti, informazioni, organizzazione di incontri e ottenere risposte tempestive.

4.1.4 Gestione degli incontri

Incontri Interni
Gli incontri interni coinvolgono esclusivamente i membri del gruppo e si tengono regolarmente, generalmente almeno una volta a settimana, adattandosi alle esigenze e allo sviluppo del lavoro autonomo di ciascun componente. In situazioni particolari, sono organizzati incontri *extra*. Un impegno costante è posto nel coordinare gli impegni di tutti, cercando di garantire la partecipazione di ogni membro a tali incontri.
Nel caso in cui la partecipazione risulti impossibile per qualche membro, viene assicurato l'**accesso al verbale** dell'incontro. Inoltre, il responsabile si impegna a fornire tutti gli aggiornamenti essenziali a chi non ha potuto partecipare, garantendo così una piena condivisione delle informazioni all'interno del gruppo.
Per ottimizzare l'efficienza del tempo durante le riunioni interne, vengono adottati i seguenti accorgimenti:

- **Scaletta *Standard*:** ogni riunione segue una scaletta predefinita, garantendo un flusso organizzato e la discussione completa di tutti gli argomenti pianificati.
- **Preparazione:** ciascun membro del gruppo si impegna attivamente e produttivamente durante le riunioni. Questo richiede una preparazione preliminare da parte di ogni partecipante. L'uso del tempo sincrono è considerato prezioso, e pertanto si incoraggia la partecipazione consapevole su argomenti di competenza individuale. Ciò implica avere un'idea chiara di "cosa so" e "cosa non so" per favorire discussioni informate.

Incontri Esterni
Gli incontri esterni vengono organizzati con il committente[G] e il proponente e vengono richiesti dal gruppo in situazioni in cui si necessita di opinioni più esperte. Data la preziosità del tempo degli esperti, si fa uno sforzo per minimizzare la frequenza di tali incontri, assicurandosi che siano ben preparati e focalizzati.
L'utilizzo del **tempo degli esperti è ottimizzato** riducendo al minimo il numero di incontri e assicurandosi che le domande rivolte siano dettagliate e concise. L'obiettivo è ottenere tutte le informazioni necessarie nel minor tempo possibile.
Gli incontri esterni avvengono virtualmente attraverso la piattaforma di comunicazione ***Zoom*** e coinvolgono tutti i membri del gruppo presenti al momento dell'incontro, insieme al referente dell'azienda. Questo approccio permette una partecipazione flessibile e una comunicazione efficace senza la necessità di spostamenti fisici.

4.1.5 Reperibilità Membri

È richiesto che ogni membro del nostro gruppo sia disponibile per riunioni sincrone dal lunedì al venerdì, nel pomeriggio. In caso di impossibilità di parte-

cipazione da parte di uno o più membri alla data programmata, è previsto che si avverta tempestivamente il Responsabile di Gruppo. Inoltre, la disponibilità può essere estesa al fine settimana in caso di necessità particolari.
Nel corso del progetto, ciascun membro gode della libertà di **gestire in modo autonomo** i compiti assegnati **asincroni** nel rispetto delle scadenze stabilite. È incoraggiato a organizzare il proprio lavoro tenendo conto degli impegni accademici e personali, garantendo al contempo il rispetto delle tempistiche concordate.

4.1.6 Ruoli Di Progetto

All'interno del progetto, ogni membro ha assegnati specifici ruoli da ricoprire per un periodo congruo rispetto a quanto preventivato. Di seguito sono elencati i ruoli previsti:

- **Responsabile di Progetto:** figura professionale, punto di riferimento sia per il committente sia per il fornitore, con lo scopo di mediare tra le due parti.
  - Guida il progetto a livello macroscopico e gestisce i processi.
  - Rimane costantemente aggiornato sullo stato di avanzamento del progetto.
  - Gestisce la pianificazione delle attività per ciascuna *milestone*^{G}.
  - Approva le attività completate e verificate, compresi processi primari e di supporto.
  - Gestisce il calendario condiviso e si occupa delle comunicazioni esterne.
- **Amministratore:** figura professionale con l'incarico delle procedure di controllo e amministrazione dell'ambiente di lavoro, con piena responsabilità sulla capacità operativa e sull'efficienza.
  - Garantisce l'efficacia ed efficienza dei processi.
  - Redige documenti come Norme di Progetto^{G}, Piano di Progetto, Piano di Qualifica.
  - Gestisce l'infrastruttura e gli strumenti utilizzati.
  - Automatizza i processi.
  - Individua punti di miglioramento nei processi.
- **Analista:** figura professionale con maggiori competenze riguardo il dominio applicativo del problema.
  - Gioca un ruolo chiave nelle fasi iniziali del progetto.
  - Comprende a fondo le necessità del proponente e identifica i requisiti fondamentali.

  - Redige il documento di Analisi dei Requisiti.
  - Studia il dominio applicativo relativo alle richieste del proponente.
  - Scompone le esigenze del proponente in elementi atomici.

- **Progettista:** figura professionale che gestisce gli aspetti tecnologici e tecnici del progetto.
  - Modella i requisiti identificati nella fase di analisi e li ricompone in un'architettura soddisfacente.
  - Produce un'architettura che soddisfa i requisiti richiesti.
  - Approfondisce conoscenze tecniche e ricerca strumenti tecnologici utili.
  - Realizza una soluzione con un alto livello di manutenibilità, seguendo le *best practices* note.
  - Progetta l'architettura di un sistema riducendo al minimo le dipendenze tra componenti.

- **Programmatore:** figura professionale incaricata alla codifica del progetto e delle componenti di supporto che verranno utilizzate per eseguire prove di verifica e validazione sul prodotto.
  - Implementa l'architettura prodotta nella fase di progettazione.
  - Scrive codice conforme alle specifiche di progettazione.
  - Applica le *best practices* nella scrittura di codice, promuovendo l'alta manutenibilità con versionamento e documentazione.
  - Scrive i *test* relativi al codice prodotto.
  - Redige il Manuale Utente.

- **Verificatore:** figura professionale con l'incarico di sorveglianza sul lavoro svolto dagli altri componenti del gruppo, sulla base delle proprie competenze tecniche, esperienza e conoscenza delle norme.
  - Si occupa di controllare che le attività svolte rispettino il livello di qualità atteso.
  - Non effettua il controllo delle attività svolte personalmente per ovvie ragioni.

4.1.7 Gestione Delle Task

Nel contesto della gestione di progetti, ogni obiettivo fondamentale è suddiviso in ***milestone***, o tappe chiave, alle quali sono associate specifiche attività. Questo approccio consente un monitoraggio dettagliato dello stato di avanzamento interno di ciascuna *milestone*. Per facilitare la gestione delle attività, il nostro *team* fa uso del *software* ***Jira***, fornendo un'interfaccia intuitiva e flessibile.
Processo di Gestione delle Attività con *Jira*:

- **Creazione:** una nuova attività viene identificata e definita con una descrizione dettagliata. La *task*[G] è inserita nella lista "*To Do*".
- **Assegnazione:** un membro del *team*, tramite il proprio *account Jira*, si prende in carico la *task*. L'assegnazione può avvenire autonomamente o essere decisa durante gli incontri interni del *team*.
- **Sviluppo:** una volta che il membro assegnato inizia a lavorare sulla *task*, questa viene spostata nella lista "*In Progress*". Tale transizione riflette l'avvio effettivo dell'attività.
- **Completamento:** una volta che la *task* è completata viene associato il *branch GitHub* correlato al lavoro svolto.
- **Verifica:** un revisore incaricato esegue la verifica della qualità dell' implementazione.
- **Accettazione:** se la verifica ha esito positivo, la *task* viene spostata nella lista "*Done*".

Categorizzazione delle Attività: le attività vengono categorizzate in base alla loro dimensione e alla loro importanza nel contesto del progetto:

- **Processo Primario:** codifica di una classe.
- **Processo di Supporto:** stesura di una sezione di un documento.
- **Processo Organizzativo:** creazione e configurazione delle attività di una *milestone*.

Calendarizzazione e Pianificazione:

- Le attività vengono pianificate in base a una calendarizzazione predefinita.
- La dimensione delle attività è commisurata alla loro complessità e al carico di responsabilità associato.

Questo processo strutturato e l'uso di *Jira* consentono al *team* di mantenere un controllo efficace sullo sviluppo del progetto, garantendo una gestione efficiente delle attività e una chiara visibilità sullo stato di avanzamento complessivo.

4.1.8 Metodo Di Lavoro

Il gruppo, al fine di minimizzare ritardi e massimizzare lo svolgimento delle proprie attività, ha deciso di implementare in modo ***Agile***[G] le tecniche di miglioramento continuo. Dopo una prima revisione, constatando il ritardo accumulato, il gruppo ha optato per l'introduzione della pratica degli *Sprint*[G] nel proprio metodo di lavoro per rendere più efficace il lavoro svolto.
Il nuovo metodo prevede la suddivisione del tempo di lavoro in **brevi intervalli** di una settimana ciascuno, noti come ***Sprint***. Questi *Sprint* sono caratterizzati da fasi chiave che guidano il processo di sviluppo e valutazione.
***Sprint Planning*:**

- ***Brainstorming* delle idee:**
  - Ogni membro del gruppo esprime le proprie idee su ciò che è più rilevante per l'immediato futuro.
- **Obiettivi e *issue*:**
  - Il Responsabile crea lo *Sprint Backlog*[G] definendo gli obiettivi specifici per lo *Sprint*.
  - Le corrispondenti *issue* vengono inserite su *GitHub*.
- **Preventivi:**
  - Ciascun membro del gruppo indica il proprio preventivo orario basato sulle attività da svolgere durante lo *Sprint*.

***Sprint Review*:**

- **Consuntivo e produttività individuale:**
  - Ogni membro esprime il lavoro svolto durante lo *Sprint*, evidenziando successi e insuccessi.
  - Possibilità di mostrare i risultati e i dubbi sull'attività svolta.
- **Obiettivi raggiunti:**
  - Viene stilata una lista degli obiettivi raggiunti in seguito alle attività svolte dai membri.
- **Obiettivi non raggiunti:**
  - Si elenca una lista degli obiettivi non completati durante lo *Sprint*, che saranno affrontati nello *Sprint* successivo.

***Sprint Retrospective*:**

- **Valutazione generale:**
  - Si conclude definitivamente lo *Sprint* appena svolto, valutandone l'andamento generale.
- ***Feedback* e miglioramenti:**
  - Si stila una lista "*Good*" per ciò che è andato bene e una "*To Improve*" per gli aspetti migliorabili.
  - Si definiscono azioni necessarie per iniziare, smettere o continuare attività specifiche.

Questo approccio *Agile*, integrando elementi di miglioramento continuo e gli *Sprint*, consente al gruppo di **adattarsi in modo flessibile**, valutare le prestazioni e implementare **miglioramenti costanti**, contribuendo così a un ciclo di sviluppo efficiente e iterativo.

4.2 Infrastruttura

L'infrastruttura organizzativa comprende tutti gli strumenti utilizzati per la comunicazione, la divisione dei compiti e il coordinamento tra i componenti per svolgere in modo efficace e strutturato i processi di organizzazione.

4.2.1 Strumenti

GitHub
Servizio di versionamento e *hosting* della *repository* scelto per mantenere la documentazione e il codice del progetto.

Discord
Strumento utilizzato per la comunicazione sincrona e asincrona interna al gruppo. All'interno del *server* sono presenti due tipologie di canali:

- **Canali Testuali**: utilizzati principalmente per lo scambio di risorse e le comunicazioni testuali sincrone e asincrone;
- **Canali Vocali**: utilizzati per la comunicazione vocale sincrona durante le riunioni interne, consentono inoltre di condividere il proprio schermo con gli altri membri presenti.

In base al periodo e alle necessità il numero di ciascun tipo di canale può variare, dando la possibilità di creare canali adibiti ad argomenti specifici.

Telegram
Principale strumento di comunicazione interna testuale asincrona che avviene tramite una *chat* condivisa. Le comunicazioni sono per lo più informali e volte all'organizzazione interna. La *chat* permette di mettere in evidenza i messaggi importanti, menzionare altri membri del gruppo e condividere sondaggi e/o *file*.

Jira
Strumento utilizzato per la gestione delle *issues* e la suddivisione dei compiti. Sono presenti tre liste che rappresentano lo stato di avanzamento di una *issue*:

- ***To Do***: stato iniziale, qui la *issue* attende fino a quando non viene presa in carico da un membro del gruppo. Verrà successivamente spostata alla lista "*In Progress*" appena inizierà la sua lavorazione;
- ***In Progress***: la *issue* è in lavorazione. Potrà essere spostata alla lista "*Done*", solamente quando verrà terminata e verificata;
- ***Done***: stato finale, la *issue* è stata completata e verificata.

È compito del *Responsabile di Progetto* verificare che le *issue* vengano divise correttamente ai diversi membri del gruppo.

Overleaf
Editor online per i *file LaTeX*, principalmente usato per la stesura dei *file* in maniera condivisa.

Google Drive
Strumento utilizzato dal gruppo per la condivisione di *file* non ufficiali contenenti bozze e appunti con *Google Docs*, tabelle per la divisione interna dei ruoli e tracciamento del tempo con *Google Sheets* e presentazioni per i Diari di Bordo con *Google Slides*.

Zoom
Strumento utilizzato principalmente per videochiamate esterne con committente e proponente.

Google Mail
Utilizzato per le comunicazioni esterne asincrone verso il proponente e il committente con l'indirizzo mail del gruppo jackpotcoding@gmail.com.

4.3 Formazione

Per garantire un costante miglioramento delle nostre attività e il mantenimento efficace delle stesse in un ambiente asincrono, è essenziale che tutti i membri del gruppo si impegnino nell'autoapprendimento delle tecnologie e delle procedure operative pertinenti. Questo approccio accelererà il processo di apprendimento e di familiarizzazione con gli strumenti impiegati. A tale scopo, si elencano di seguito alcuni documenti utilizzati durante lo sviluppo, sia a livello di documentazione che di organizzazione, per assicurare una comprensione completa del contesto operativo:

- *GitHub*
- *Git*
- *LaTeX*
- *Python*