Norme di progetto

v0.7.0

Farmacode

farmacode.swe.unipd@gmail.com

Registro delle modifiche

Versione	Data	Scrittori	Revisori	Descrizione
0.7.0	2-12-2023	Rosson Lorenzo	Favaron Riccardo	Realizzata prima stesura sezione 4
0.6.0	25-11-2023	Rosson Lorenzo	Favaron Riccardo	Realizzata prima stesura sezione 3.2, apportate modifiche alla sezione 3.1.5
0.5.0	21-11-2023	Baggio Matteo	Carraro Alessandro	Trasferimento da LaTeX a Typst del documento
0.4.0	20-11-2023	Passarella Alessandro	Carraro Alessandro	Completamento stesura sezione 3.1
0.3.0	18-11-2023	Baggio Matteo	Carraro Alessandro	Completamento stesura sezione 1
0.2.0	15-11-2023	Baggio Matteo Passarella Alessandro	Carraro Alessandro	Stesura indice
0.1.0	12-11-2023	Bomben Filippo Rosson Lorenzo	Baggio Matteo	Stesura iniziale del documento

Indice

1) Introduzione al documento

Questo documento è stato creato per identificare le best practices di progetto e per stabilire una metodologia di lavoro chiara nel corso dell'attività produttiva. L'obiettivo è garantire una gestione omogenea e coesa del lavoro. Per facilitare il monitoraggio del progresso e consentire un approccio incrementale, vengono registrate le diverse versioni del documento.

1.1) Scopo del prodotto

Lo scopo del prodotto è creare un'applicazione dove sia possibile verificare i possibili interessi di un cliente nei confronti di un prodotto. Al giorno d'oggi l'ambito degli e-commerce si sta sempre più espandendo ed evolvendo. La presenza di negozi virtuali permette di accedere a molti dati legati agli acquisti, alle preferenze ed al comportamento degli utenti. Questi dati se analizzati propriamente permettono di prevedere preferenze e comportamenti futuri degli utenti, dando spazio ad operazioni di marketing mirate.

Il prodotto sarà dunque un'applicazione attraverso la quale l'amministrazione di un e-commerce sarà in grado di accedere ai risultati dell'analisi dei dati relativi all'utilizzo della suddetta attività. Il lavoro principale di questa applicazione non sarà dunque svolto dal lato dell'utente, il quale avrà solo accesso ad un'analisi dei dati e potrà garantire feedback sulla loro correttezza, ma sarà svolto da un algoritmo non visibile né accessibile all'utente. Questo algoritmo utilizzerà la tecnologia dell'intelligenza artificiale per analizzare i dati forniti dall'azienda con lo scopo di trovare e definire le correlazioni tra i vari prodotti, tra i vari utenti e tra utenti e prodotti. Queste correlazioni trovate su più livelli di profondità permetteranno di creare un altro set di dati, dal quale l'utente dell'applicazione potrà accedere ai dati che necessita, principalmente questi dati saranno gli N prodotti che potrebbero interessare ad un X utente e gli N utenti che potrebbero essere interessati ad un X prodotto.

Questa applicazione inoltre per comodità d'uso sarà sviluppata sotto la forma di una webapp che potrà essere accessibile utilizzando diversi dispositivi, sistemi e browser.

1.2) Glossario

Si fa inoltre notare la presenza di un Glossario nel quale sono riportati i termini utilizzati nei documenti. Con questo principio le best practices per la creazione del prodotto riusciranno facilmente ad essere rispettate garantendo il più possibile l'omogeneità del prodotto.

1.3) Miglioramenti e maturità del documento

Questo documento è stato creato seguendo un approccio incrementale, il che implica la sua natura adattabile e suscettibile di modifiche nel tempo. Queste modifiche saranno apportate in risposta alle esigenze concordate tra i membri del gruppo e il proponente. Pertanto, questa versione del documento non deve essere considerata come una versione definitiva o completa, ma piuttosto come un punto di partenza che sarà ulteriormente sviluppato e aggiornato per meglio rispondere alle mutevoli esigenze del progetto.

1.4) Riferimenti

2) Processi primari

2.1) Acquisizione

3) Processi di supporto

3.1) Documentazione

3.1.1) Descrizione

La documentazione software è l'insieme di informazioni, raccolte testualmente, volte allo scopo di spiegare a quali funzionalità assolve un software, come è strutturato e implementato e come lo si utilizza. Nel contesto del team di sviluppo è necessaria per facilitare il lavoro dei componenti, tenendo traccia e documentando tutti i processi e attività presenti andando a facilitare anche la manutenzione migliorando la qualità del risultato finale.

È bene quindi che vengano definite delle regole chiare e concise utili per la stesura di un documento, da seguire durante tutto il ciclo di vita del progetto allo scopo di garantire maggiore comprensione.

3.1.2) Strumenti

- Typst: scelto per la definitiva formattazione dei documenti per via della comodità con cui effettuare il versionamento dei documenti stessi;
- Overleaf (LaTeX): utilizzato nelle prime fasi del progetto per la realizzazione dei documenti necessari, successivamente cambiato con typst;
- UML: per la creazione di diagrammi UML il team ha deciso di utilizzare StarUML.

3.1.3) Grafiche

- Template: le nostre grafiche per i documenti sono state realizzate con photoshop;
- Tabelle: le tabelle presentano una classica intestazione dei contenuto, i nomi delle colonne in grassetto e nessun altra particolarità, si è scelto di utilizzare una filosofia minimale per non appesantire i documenti.

3.1.4) Norme tipografiche

- Nome file: I nomi dei file hanno tutti una notazione omogenea tra di loro, ovvero, nomi descrittivi del contenuto, lettera iniziale è sempre maiuscola e il resto tutto minuscolo, le parole sono separate da degli underscore. La data viene scritta in formato AAAA-MM-GG;
- Stile del testo: divisione in sezioni X.X.X e in caso di ulteriori suddivisioni si utilizza un elenco puntato, la sezione X.X.1 è sempre la descrizione del contenuto di quella sezione. Si cerca sempre di rendere il tutto più semplice possibile per facilitarne la lettura e mantenere ordinato il documento;
- Glossario:

3.1.5) Struttura

I documenti ufficiali hanno una struttura precisa e comune che deve essere rigorosamente rispettata per i motivi citati nella descrizione.

- Prima pagina: sempre composta dal template esclusivo del team, il logo dell'università, l'anno accademico in cui viene svolto il progetto, il nome del documento, il nome del team con la mail e i componenti;

- Registri modifiche (changelog): composti da versionamento, data della modifica effettuata, descrizione della modifica, ruolo dei componenti che hanno effettuato la modifica e i loro nomi.
- Indice: ogni documento presenta un indice nella pagina seguente al registro delle modifiche, la strutture è divisa in sezioni X.X.X con il numero della pagina in cui inizia la sezione. La divisione X.X.X presenta i macro-argomenti suddivisi nei loro vari paragrafi a loro volta suddivisi in sezioni più specifiche;
- Contenuto: esposto con la maggiore chiarezza e semplicità, rigorosamente diviso in sezioni secondo i principi di indicizzazione;
- Pié pagina: solamente il numero della pagina in questione.

3.1.6) Caratterizzazione

- Formali: Sono i documenti che andranno a formare la documentazione software del prodotto. In quanto tali sono sottoposti a versionamento e a processi di verifica e approvazione. Essi comprendono documenti interni, utili quindi ai membri del team di sviluppo, ed esterni, destinati a proponente e committente.

  Complessivamente ne fanno parte:

  - Interni:
    - Norme di progetto, rappresentano il "way of working";
    - Verbali interni e esterni, a uso consultativo;
  - Esterni:
    - Analisi dei Requisiti;
    - Piano di Progetto;
    - Piano di Qualifica;
    - Glossario;
    - Verbali interni e esterni, attestanti di quanto discusso.
- Informali: Sono i documenti interni non destinati alla divulgazione con esterni e fini a loro stessi. Perciò non necessitano di versionamento. Spesso sono bozze in preparazione a documenti formali, o note e appunti generiche.

3.2) Gestione della configurazione

3.2.1) Descrizione

Il concetto di "gestione della configurazione" abbraccia tutte le pratiche essenziali per gestire lo stato di un prodotto software e di tutti i suoi componenti, compresi sorgenti e documentazione. Questo insieme di norme e procedure non solo fornisce informazioni sullo stato di avanzamento del progetto, ma offre anche un resoconto dettagliato dell'evoluzione nel tempo del prodotto, garantendo nel contempo che il sistema operi secondo le attese. Un'efficace gestione della configurazione è cruciale per preservare l'integrità e le prestazioni del prodotto software durante il suo avanzamento. Inoltre, dovrebbe facilitare la risoluzione di problematiche e conflitti, assicurando una gestione fluida e efficiente del ciclo di vita del software.

3.2.2) Versionamento

Il versionamento è una procedura fondamentale per la gestione di un progetto. Oltre a tracciare i cambiamenti di ogni *artefatto*, documento o sorgente che sia, permette il rispristino di quest'ultimo ad una sua fase precedente rendendo molto più semplice la gestione di errori. Il changelog o "registro delle modifiche", strettamente collegato al concetto di versionamento, espone al lettore, il ciclo di vita dell'artefatto, le modifiche effettuate, le problematiche sorte, e infine anche la distribuzione dei lavori tra i componenti del team di sviluppo.
Ogni documento oltre a essere dotato di un changelog è identificato da un numero di versione così composto:

vX.Y.Z

dove :

- X rappesenta fasi del documento che suddividono e raccolgono i cambiamenti più significativi apportati all'artefatto anche detti "major".
- Y rappresenta modifiche minori come ad esempio la realizzazione di una sezione o feature le quali si pensa non siano sufficienti a stabilire una nuova "fase" del documento. Sono anche identificati con l'appellativo "minor".
- Z rappresenta piccoli aggiustamenti (fixes) o migliorie generali.

Si noti che ogni versione rappresenta non solo un'aggiunta di tipo prettamente produttivo, ma anche al sua revisione.

3.2.2.1) Lato documentazione

Nella documentazione è possibile aggiornare la versione andando semplicemente ad aggiungere un nuovo record nella sezione di changelog del rispettivo file sorgente. Qui sotto un esempio:

```
changelog: (

    "0.5.0", "21-11-2023", p.baggio, p.carraro, "Stesura sezione 3.2",
    "0.4.0", "20-11-2023", p.passarella, p.carraro, "Stesura sezione 3.1",

)
```

Una volta fatto, la compilazione automatica, attuata grazie ad una github action realizzata ad hoc, insieme alle funzionalità di scripting che fornisce typst, andrà a creare effettivamente la tabella del registro delle modifiche con all'interno tutte le informazioni specificate e richieste. Si noti che *p* è una variabile d'ambiente contenente tutti i nominativi dei componenti del gruppo di lavoro e di ulteriori nominativi utili e ripetuti molteplici volte nel corso del progetto.

3.2.2.2) Lato software

3.2.3) Repository

Per la gestione della configurazione e versionamento il progetto si poggia sul uso di un repository Github. Qui sotto un link alla documentazione ufficiale:

Github Docs.

3.2.3.1) Struttura

L'attuale struttura del repository è suddivisa in 3 branch:

- main;
- approval;
- sources.

main:
E' definibile come il branch di presentazione, nel quale sono presenti solo artefatti revisionati e approvati dal responsabile di progetto corrente. Su esso è applicata una "branch protection rule" che non ne permette i push diretti e protegge il ramo.

approval:
Come intuibile dal nome, è il branch che rappresenta il main durante lo sviluppo e che garantisce che ciò che entra nel ramo principale sia completamente revisionato e approvato. I suoi contenuti verrano uniti a quelli del main tramite un processo di merge una volta che il responsabile di progetto lo ritenga possibile. Le pull request da qualsiasi ramo verso quello di presentazione verranno infatti reinderizzate a quest'ultimo, che ne andrà a valutare la qualità, accettando il lavoro svolto o rimandandolo al mittente con direttive sul come migliorarlo.

sources:
E' il branch relativo alla produzione della documentazione, perciò contiene solo file di tipo .typ e non è pensato per una sua supervisione esterna. I file sorgenti verranno compilati e resi disponibili automaticamente nel ramo approval.

Nel branch main è disponibile un README.md che ne descrive la struttura di cartelle. Qui sotto un link al repository:

Repository di progetto.

3.3) Accertamento della qualità

3.3.1) Descrizione

3.4) Qualifica

3.4.1) Verifica

3.4.2) Validazione

3.5) Revisioni congiunte con il cliente

3.5.1) Descrizione

3.6) Verifiche interne

3.6.1) Descrizione

3.7) Risoluzione dei problemi

3.7.1) Descrizione

3.7.2) Gestione dei problemi

3.7.3) Gestione dei cambiamenti

3.7.4) Utilizzabilità

3.7.5) Valutazione del prodotto

4) Processi organizzativi

4.1) Gestione dei processi

4.1.1) Obiettivo

L'obiettivo dei processi organizzativi è quello di arrivare alla creazione del documento denominato "Piano di progetto" nella sua forma il più completa possibile, andando a definire ruoli, attività e loro collocazione nel tempo. Il documento, oltre ad essere utile al team per gestire l'organizzazione e la gestione dei ruoli di ogni componente, fa da bacheca al commitente di quanto appena citato. Il piano di progetto punta a comprendere tutte le pratiche e metodi riguardati il processo organizzativo e di pianificazione, descrivendone l'applicazione.

4.1.2) Descrizione

In questa sezione vengono trattate tutte le normative utili alla stesura e redazione del documneto "Piano di progetto", gli argomenti che ne fanno parte sono:

- Ruoli e relativa organizzazione: con conseguente descrizione ed esposizione della strategia utilizzata per la rotazione degli stessi.
- Gestione degli incontri e delle comunicazioni: comprendente loro suddivisone in tipologia.
- Gestione dell'organizzazione: informazioni relative alle pratiche per la suddivisone in attività, e loro collocazione temporale, del corso del progetto.

4.1.3) Ruoli e relativa organizzazione

La suddivisione in ruoli segue le norme definite nel "Regolamento progetto didattico" sottopostoci dal nostro committente. Qui sotto un breve riepilogo:

Ruolo	**Costo orario**	**Responsabilità**
Responsabile	30	• Coordina l'elaborazione di piani e scadenze • Approva il rilascio di prodotti parziali o finali (SW, documenti) • Coordina le attività del gruppo
Amministratore	20	• Assicura l'efficienza di procedure, strumenti e tecnologie a supporto del way of working
Analista	25	• Svolge le attività di analisi dei requisiti
Progettista	25	• Svolge le attività di progettazione (design)
Programmatore	15	• Svolge le attività di codifica
Verificatore	15	• Svolge le attività di verifica

Si noti come i ruoli possano svolgere anche mansioni al di fuori della loro responsabilità in caso di necessità, ovviamente senza venire meno alle pratiche di tracciabilità adottate normalmente dal team.

La loro assegnazione viene gestita dal Responsabile di progetto corrente, il quale confrontandosi con gli altri componenti del gruppo, va a stabilire una rotazione conforme al regolamento. Ogni membro del team dovrà infatti ricoprire ogni carica almeno una volta.

Segue una descrizione più dettagliata di ogni ruolo e rispettive mansioni:

- Responsabile di progetto
- Amministratore
- Analista
- Progettista
- Programmatore
- Verificatore

4.1.3.1) Gestione dei "cold start"

Al fine di evitare rallentamenti durante il corso del progetto, dovuti a delle situazioni di "cold start", il team si impegna ad adottare le seguenti pratiche:

- Documentazione dettaglita:

  Ogni membro è tenuto a documentare ogni azione ritenuta non banale e avente valenza e dipendenze future. Prima in documenti informali, questo per non rallentare troppo i tempi, successivamente da integrare nella documentazione ufficiale.

- Formazione e Condivisione delle Conoscenze:

  Ogni membro è tenuto, qualora si ritenga necessario, a condividere, oltre che in forma scritta attraverso la documentazione, verbalmente le conoscenze e le competenze apprese durante gli sviluppi, o pregresse.

- Rotazione Graduale:

  Le rotazioni dei ruoli, quando ritenute necessarie, avverranno in modo graduale. Ciò consentirà a coloro che hanno già sviluppato una certa dimestichezza in un determinato ambito di supportare chi si avvicina a quel ruolo per la prima volta. In pratica, questa modalità di rotazione riflette l'approccio XP, emulando la pratica del "pair programming".

4.1.4) Gestione degli incontri e delle comunicazioni

4.1.4.1) Reperibilità dei membri

Ogni membro del gruppo si impegna ad essere reperibile per riunioni sincrone durante la settimana, dal lunedì al giovedì, nel pomeriggio, il venerdì durante la mattinata. In caso di impossibilità di partecipare alle riunioni nelle date stabilite, è obbligatorio informare tempestivamente il Responsabile di progetto. Inoltre, la disponibilità può essere estesa anche durante il weekend in casi di necessità, solitamente preferendo la domenica al sabato.

Durante il corso di uno sprint ogni mebro è libero di gestire le proprie attività di progetto in modo asincrono, a meno che esse non richiedano la collaborazione di più componenti. Ogni membro si assume responsabilmente la gestione di impegni accademici e personali, rispettando le scadenze imposte dal relativo sprint.

Per facilitare l'assegnazione delle attività in relazione agli impegni di ogni componente, il team ha a disposizione un file "Google Fogli" dove sono visualizzabili le proprie disponibilità giornaliere (inserite al inizio del progetto), e dove nel eventualità è possibile segnare altri impegni inderogabili e sorti in un secondo momento.

4.1.4.2) Comunicazioni

- **Interne**

  Le comunicazioni interne sono quelle coinvolgenti solo il team, o alcuni dei suoi membri, avvengono tramite mezzi quali:

– Telegram: Il gruppo è usato per comunicazioni di tipo più breve e repentino. Solitamente viene utilizzato per stilare un breve ordine del giorno, e per organizzare incontri interni. Mentre in privato avvengono comunicazioni che interessano solo le parti conivolte in modo da non intasare il gruppo.

– Discord: Il server è suddiviso in vari canali testuali:

- General: utilizzato per comunicazioni generali;
- Link-utility: in esso sono riversati tutti i link utili al gruppo perchè spesso visitati o consultati (ad esempio: link alla tabella condivisa per la gestione costi/ore);
- Link-brainstorming: racchiude tutti i link a fonti di tipo informativo (ad esempio: link alla documentazione della libreria di python surprise);
- Domande: è utilizzato per contenere le domande dei vari componenti del team, sia rivolte verso il team stesso, si verso l'esterno (proponente/committenti);
- Todo-reminder: contiene delle annotazioni su cose da fare nel breve periodo.

Sono inoltre presenti molteplici canali vocali per permettere di lavorare in piccoli sottogruppi.

- **Esterne**

Le comunicazioni esterne avvengono tramite i seguenti mezzi:

– E-mail: Usate per le comunicazioni con proponente e commitenti. Principalmente hanno la funzione di concordare meeting, o di esporre quesiti e dubbi.

4.1.4.3) Incontri o Meetings:

- **Interni**

Per una migliore gestione degli imprevisti e in generale della pianificazione e organizzazione delle attività, il gruppo ha deciso di adottare 2 tipologie differenti di incontri interni: "Scheduled Meeting", e "Daily Call". Per questioni di efficenza e praticità si è concordato di adoperare Discord come mezzo tramite, la modalità di questi incontri è quindi "da remoto".

- **"Scheduled meeting"**

Sono i meeting interni che solitamente prevedono la messa a verbale. Vengono fissati con cadenza settimanale con data variabile a seconda delle disponibilità dei membri del team, quest'ultima viene regolarmente concordata alla fine del incontro precedente. La loro durata è variabile, e tutte le componenti sono tenute a presenziarvi. Per una migliore gestione del tempo a disposizione è stato deciso di strutturare i meeting come segue:

– Struttura:

1) Ordine del giorno: Al inizio di ogni meeting chi ha partecipato alle lezioni del giorno, condivide informazioni utili con il team, aggiornadolo su quanto stato discusso con i docenti.

2) Retrospettiva: Prima di discutere le varie domande e dubbi, il responsabile di progetto corrente stila un breve riassunto di quanto stato fatto nel ultimo sprint, evidenziandone aspetti positivi e negativi. Per facilitare queste operazioni, il responsabile di progetto è tenuto ad arrivare "preparato" al incontro, in modo che sia tutto più scorrevole.

3) Domande e dubbi: Successivamente vengono discusse domande e dubbi per le quali si ritenga necessaria una discussione collettiva.

4) Pianificazione prossima: Infine viene effettuata, avendo un miglior cruscotto sul avanzamento del progetto, una pianificazione più mirata per il prossimo sprint, andando a definire effettivamente le attività, ruotando i ruoli e spartendo queste ultime.

5) Post meeting: Alla fine di ogni meeting il responsabile di progetto fa un breve resoconto di quanto discusso e lo condivide sul gruppo telegram in modo da aggiornare chi eventualmente non fosse riuscito a presenziare al incontro. Successivamente si adopera anche ad una prima stesura del relativo verbale. Per facilitare queste operazioni ogni meeting interno viene registrato.

- **"Daily Call"**

  Sono incontri di durata mediamente minore, che avvengono giornalmente quando e se ne sorge la necessità. Possono essere richiesti da qualsiasi membro del gruppo, e la partecipazione è richiesta solamente ai sottoinsiemi coinvolti. Solitamente non prevedono la stesura di relativo verbale, ma ciò dipende dagli argomenti discussi e dalla presenza o meno di decisioni importanti. Vengono anche utilizzati per sessioni di lavoro "in pair".

- **Esterni**

  – Proponente:

  Questi incontri prevedono sempre la stesura di relativo verbale necessitante validazione ed approvazione dal partecipante esterno attraverso la sua firma. Solitamente vengono richiesti dal team. Data e orario, sono concordati a priori durante il meeting precedente, o tramite E-mail, tenendo conto delle disponibilità del proponente e del gruppo. Gli argomenti trattati sono di vario tipo e seguono gli sviluppi del progetto, per facilitarne la discussione vengono esplicitati al proponente tramite mail alcuni giorni prima del incontro.

  – Committenti:

  nd.

4.1.5) Gestione dell'organizzazione

4.1.5.1) Metodologia e pratiche

In modo da migliorare la collaborazione e una migliore gestione del ritmo di avanzamento dei lavori, il team ha preso la decisione di adottare un approccio agile nello sviluppo del progetto, ispirandosi a framework e metodologie ben consolidati come Scrum e XP, ampiamente utilizzati in contesti lavorativi reali.

La filosofia che sottende le strategie di tipo agile è incentrata sull'adozione di pratiche di Continuous Integration/Continuous Deployment (CI/CD).

Questa scelta mira a fornire diversi vantaggi e valori aggiunti:

- Favorire il Lavoro di Gruppo:

  Promuove la collaborazione e la comunicazione all'interno del team, incoraggiando la condivisione delle idee e delle competenze.

- Sviluppo Individuale:

  Favorirsce la crescita individuale a livello di conoscenze e competenze, consentendo a ciascun membro di contribuire al massimo delle proprie capacità indipendentemente dal ruolo corrente.

- Miglioramento Continuo:

  Promuove il miglioramento continuo attraverso pratiche di retrospezione, identificando e risolvendo le problematiche in modo tempestivo e continuo.

- Organizzazione Efficace:

  Migliora e facilita l'organizzazione tra i membri del team, assicurando una distribuzione efficace delle responsabilità e delle attività.

- Trasparenza per Proponenti e Committenti:

  Garantisce trasparenza al proponente, consentendo un costante flusso di feedback e una maggiore comprensione del processo di sviluppo. Facilita anche l'analisi da parte del committente per una migliore valutazione del progresso.

L'adozione di pratiche CI/CD si inserisce in questo contesto, contribuendo a garantire una integrazione continua del codice e una distribuzione continua delle nuove funzionalità, riducendo al minimo rischi e migliorando la qualità del software. Questo approccio agile mira a fornire al team una struttura dinamica e flessibile per affrontare le sfide e rispondere alle esigenze del progetto in modo efficiente e tempestivo.

4.1.5.2) Milestone e Sprint

Le tempistiche del periodo di progetto sono scandite da milestone e sprint. Il team, sempre rifacendosi ad un approccio di tipo agile, ha definito conseguentemente queste ultime.

- Milestone:

  Rappresentano le revisioni di progetto, e gli sprint necessari al loro compimento. Ragionevolmente, ogni sprint verrà a posteriori suddiviso in macro attività pianificandone le fasi, la cui granularità e specificità verranno esplorate durante un periodo più a ridosso dello stesso, avendo un istantanea migliore dello stato del progetto.

- Sprint:

  Definiscono finestre di tempo durante le quali il team lavora per portare a termine relative attività, rispettandone le scadenze. La durata degli sprint deve essere tassativamente fissata, in modo che ogni componente percepisca l'incobenza delle scadenze e quindi si adegui di conseguenza, evitando così "pigrizie". Questo senza andare a discapito della salute mentale di ogni componente, la quale andrebbe ad incidere sulla qualità di quanto prodotto. Solo in casi eccezionali, come durante gli albori del periodo relativo al RTB, considerato, di "assestamento", sono ammesse delle variazioni, perlopiù per facilitare l'adeguamento di ogni membro al respettivo ruolo corrente. In modo da mantenere un workflow scorrevole, si è concordata la durata di 1 settimana per sprint. Così facendo ogni membro ha la possibilità di esplorare più ruoli, e di gestire il proprio tempo in modo più produttivo.

La loro definizione e collocazione temporale è definita nel documento "Piano di progetto", redatto dal responsabile di progetto.

4.1.5.3) Gestione di attività e Issue

Per la gestione delle attività relative ai processi primari e di supporto, si utilizza il sitema integrato di Issues Tracking System (ITS), di Github. Il ciclo di vita delle azioni segue i seguenti passaggi:

Creazione: L'attività viene definita come un compito da svolgere e viene registrata come una issue su GitHub. Le issue devono essere collegate alla/e board di progetto, ed al rispettivo sprint utilizzando la milestone allegata. Inoltre devono essere contrassegnate da label identificative consone. Ne segue la lista che ne norma l'uso:

- approval: da utilizzare solo per issue che identificano attività di approvazione, da svolgere dal Responsabile corrente. Solitamente una finale per sprint.
- bug-fix: denota una issue la cui rispettiva attività mira alla correzione di un bug, o altre problematiche.
- code: rappresentano una nuova feature o integrazione nel codice.
- documentation: rappresentano issue legate alla stesura di documentazione.
- revision: da utilizzare solo per issue legate alle attività di revisone.
- RTB: deve essere usata solo per issue rappresentanti sprint legati al periodo di RTB.
- PB: deve essere usata solo per issue rappresentanti sprint legati al periodo di PB.

Infine le issue devono avere un nome significativo e possedere una descrizione definita come segue:

(h3 in markdown) desc: (plain text) testo della descrizione.

Assegnazione: Le issue vengono assegnate in modo da rispettare la configurazione ruolistica corrente.

Completamento: L'attività viene completata dalla persona incaricata, per poi essere spostata nello stato "ReadyToReview" nella rispettiva board di progetto, in modo da notificarne la revisione. Successivamente verrà chiusa attraverso l'apposita funzionalità di chiusura dell'issue in GitHub da chi ne svolge la revisone.

Verifica: Chi ne è incaricato procede a verificare quanto svolto utilizzando strumenti automatici o meno, a seconda di quanto prodotto (Documentazione, codice, …). Nel caso in cui il verificatore non sia soddisfatto del lavoro svolto, informerà l'autore e il Responsabile. L'autore dovrà quindi ritornare su quanto fatto e apportare le modifiche suggerite dal verificatore. Una volta finito con esito positivo il processo di verifica il verificatore procederà chiduendo la issue relativa sia all'attività corrisponendete, sia la issue relativa alla verifica di tale attività.

Si noti che è stato scelto di avere una issue specifica per la verifica per rendere ai verificatori più facile organizzare e pianificare il proprio lavoro, al inizio di ogni sprint.

Accettazione: Dopo la conferma positiva da parte del Verificatore, a ridosso della fine di ogni sprint, il Responsabile effettua un controllo aggiuntivo, procedendo quindi in caso a chiudere l'issue di approvazione corrispondente, e a eseguire il merge nel branch principale di presentazione.

Le singole attività vengono valutate in base alla loro dimensione e alla pianificazione definita, considerando sia il carico di lavoro che le responsabilità associate.

Una più dettagliata descrizione della gestione delle board di progetto è visionabile nel prossimo sotto-paragrafo.

4.2) Infrastrutture

4.2.1) Strumenti di supporto ai processi

4.3) Miglioramento

4.3.1.1) Manutenzione migliorativa dei processi

4.4) Formazione

4.4.1) Complementi all'auto-formazione