.7

NORME DI PROGETTO

Versione 3.0.0

Arena Ivan Antonino
Baesso Nicola
Bousapnamene Ruth Genevieve
Calabrese Luca
Garon Martina
Liva Noemi
Marchiante Marco

Progetto Ingegneria del Software

Dipartimento di Matematica
Università degli Studi di Padova

12 giugno 2023

Contatti: dotseventeam@gmail.com

Registro delle versioni

Versione	Data	Autore	Ruolo	Motivazione
3.0.0	12.06.2023	Ivan Antonino Arena	Responsabile	Validazione
2.0.6	17.05.2023	Nicola Baesso	Amministratore	Modifica $2.2.6.3 e $3.4.4.3 (CAP-184 e CAP-189). Verificato da Martina Garon
2.0.5	10.05.2023	Noemi Liva	Amministratore	Modifica $4.1.4.1 e stesura 2.1.5, 2.1.6 2.2.7, 2.2.8 e 3.3.2, 3.3.3 (CAP-186). Verificato da Luca Calabrese
2.0.4	08.05.2023	Martina Garon	Amministratore	Modifica $3.4.4 (CAP-147). Verificato da Ivan Antonino Arena
2.0.3	20.04.2023	Marco Marchiante	Amministratore	Modifica $3.2.4.1 (CAP-136). Verificato da Nicola Baesso
2.0.2	07.04.2023	Ivan Antonino Arena	Amministratore	Inserimento procedure, attività e strumenti per ciascun processo (CAP-135). Verificato da Luca Calabrese
2.0.1	04.04.2023	Martina Garon	Amministratore	Modifica a $3.1.3.4 (CAP-134) e $3.2.3 (CAP-138). Verificato da Nicola Baesso
2.0.0	14.02.2023	Martina Garon	Responsabile	Validazione
1.2.0	24.01.2023	Luca Calabrese	Verificatore	Verifica $2
1.1.3	19.01.2023	Noemi Liva	Amministratore	Stesura $2
1.1.2	10.01.2023	Martina Garon	Verificatore	Verifica $3.2.4.2
1.1.1	05.01.2023	Nicola Baesso	Amministratore	Modifiche $3.2.4.2

Tabella 1: registro di versionamento del documento

Versione	Data	Autore	Ruolo	Motivazione
1.1.0	04.01.2023	Nicola Baesso	Verificatore	Verifica generale
1.0.1	04.01.2023	Noemi Liva	Amministratore	Stesura $3.2.4.2 e modifiche $3.1.5.2
1.0.0	19.12.2022	Noemi Liva	Responsabile	Validazione
0.3.0	17.12.2022	Ivan Antonino Arena	Verificatore	Verifica generale
0.2.1	15.12.2022	Ruth Genevieve	Amministratore	Stesura $3.4.3
0.2.0	15.12.2022	Ivan Antonino Arena	Verificatore	Verifica generale
0.1.2	10.12.2022	Marco Marchiante	Amministratore	Modifica $3
0.1.1	10.12.2022	Luca Calabrese	Amministratore	Completamento $3
0.1.0	30.11.2022	Ruth Genevieve	Verificatore	Verifica generale
0.0.2	21.11.2022	Noemi Liva	Amministratore	Prima stesura $1, $3 e $4
0.0.1	20.11.2022	Noemi Liva	Amministratore	Struttura del documento

Tabella 2: registro di versionamento del documento

Indice

1 Introduzione

1.1 Scopo del documento

Lo scopo di questo documento è di definire le norme, le convenzioni e le procedure adottate da tutti i membri di **DotSeven Team**, in modo da poter ottenere un metodo di lavoro comune. Per raggiungere questo scopo, ogni membro è tenuto a visionare periodicamente il documento e a rispettare tutte le norme in esso presenti. La visione va fatta periodicamente essendo che questo è un documento incompleto e le norme verranno definite passo per passo partendo dalle più urgenti.
Il presente documento è riferimento normativo per il *team*.

1.2 Scopo del prodotto

Data la sempre maggiore influenza di intelligenze artificiali sempre più complesse e sistemi informatici robotizzati, è importante sviluppare dei metodi che permettano di distinguere se la persona che sta interagendo con un sistema sia effettivamente una persona fisica o dimostri i comportamenti di uno strumento automatico.
L'obbiettivo del Team **Dot Seven** e dell'azienda **Zucchetti s.p.a** è quindi quello di creare un sistema CAPTCHA (*Completely Automated Public Test to tell Computers and Humans Apart*) in grado di distinguere le macchine dall'umano.

1.3 Glossario

Per evitare ambiguità relative alle terminologie utilizzate è stato creato un documento denominato "Glossario". Questo documento contiene tutti i termini tecnici scelti dal gruppo e utilizzati nei vari documenti con le relative definizioni.

1.4 Riferimenti

1.4.1 Riferimenti normativi

- Presentazione del capitolato C1: https://www.math.unipd.it/ tullio/IS-1/2022/Progetto/C1.pdf (data ultimo accesso al link: 12 giugno 2023);
- Regolamento del progetto didattico: https://www.math.unipd.it/ tullio/IS-1/2022/Dispense/PD02.pdf (data ultimo accesso al link: 12 giugno 2023);
- Verbali interni.

1.4.2 Riferimenti informativi

- Lezione sui processi del ciclo di vita del SW: lezione T02 (data ultimo accesso al link: 12 giugno 2023);
- Discord: https://discord.com/ (data ultimo accesso al link: 12 giugno 2023);
- Telegram: https://telegram.org (data ultimo accesso al link: 12 giugno 2023);
- Jira: https://www.atlassian.com/it/software/jira (data ultimo accesso al link: 12 giugno 2023).

2 Processi Primari

2.1 Fornitura

2.1.1 Scopo

Il processo di Fornitura ha lo scopo di definire e trattare le norme e i termini che i membri del gruppo sono tenuti a rispettare per rivestire adeguatamente il ruolo di fornitore nei confronti dell'azienda proponente **Zucchetti** S.p.A. e dei committenti.

2.1.2 Aspettative

Nel corso dell'intero progetto, il gruppo intende instaurare con **Zucchetti S.p.A.**, in particolare con il relativo referente Dr. Gregorio Piccoli, un rapporto di costante collaborazione al fine di:

- Determinare aspetti chiave per soddisfare i bisogni del proponente;
- Determinare eventuali vincoli sui requisiti e sui processi;
- Effettuare una stima, in tempo e in denaro, dei costi;
- Garantire che il prodotto soddisfi le richieste, concordandone la qualifica.

2.1.3 Materiale fornito

Il materiale che il gruppo fornirà al proponente ed ai committenti è:

- **Analisi dei Requisiti**: contiene l'analisi dei casi d'uso e dei requisiti con lo scopo di:
  - Determinare tutte e sole le funzionalità che saranno offerte dal prodotto finale;
  - Chiarire ogni ambiguità che potrebbe sorgere nella comprensione del capitolato;
- **Piano di Progetto**: contiene la pianificazione preventiva dei tempi, l'analisi dei rischi, il consuntivo di periodo, la data di consegna e i costi previsti;
- **Piano di Qualifica**: contiene le modalità adottate in verifica e validazione, assicurando che la qualità dei processi e dei prodotti rispetti le aspettative;
- **Proof of Concept**: piccolo software di esempio che servirà al gruppo per determinare la fattibilità pratica e dimostrare la fondatezza e applicabilità di concetti fondamentali e costituenti in relazione al prodotto finale.

2.1.4 Procedure

Nel processo di fornitura, si attuano le seguenti procedure:

- Definizione degli argomenti da trattare nei documenti;
- Redazione dei documenti;
- Caricamento dei documenti in area pubblica;
- Colloquio con Proponente:
- Colloquio con docenti.

2.1.4.1 Definizione degli argomenti da trattare nei documenti

Gli argomenti da trattare nei documenti sono discussi collegialmente in riunioni interne. Se necessario, si possono indire più riunioni, per approfondire specifici dettagli.
Le attività necessarie sono quindi:

- Riunione interna, con specifico "ordine del giorno";
- Redazione del verbale della riunione, contenente i punti salienti;
- Approfondimenti autonomi e/o collegiali, in merito ad argomenti specifici.

Gli strumenti utilizzati sono:

- Discord, per le riunioni interne;
- Overleaf, per la redazione dei documenti in Latex, con relative credenziali d'accesso;
- Dispositivo con connessione Internet.

2.1.4.2 Redazione dei documenti

I documenti vengono redatti in Overleaf. Dopo la loro scrittura, devono essere verificati e successivamente validati.
Le attività necessarie sono quindi:

- Scrittura del documento in Latex, tramite il servizio Overleaf;
- Verifica del documento;
- Validazione del documento.

Gli strumenti utilizzati sono:

- Overleaf, per la redazione degli appunti, con relative credenziali d'accesso;
- Dispositivo con connessione Internet.

2.1.4.3 Caricamento dei documenti in area pubblica

I documenti validati e che completano una determinata *milestone* sono caricati nella *repo* di Github.
Le attività necessarie sono quindi:

- Individuazione dei documenti da caricare;
- Fare l'upload nella *repo*.

Gli strumenti utilizzati sono:

- Account Github, con relative credenziali;
- Overleaf, per la generazione dei documenti;
- Dispositivo con connessione Internet.

2.1.4.4 Colloquio con Proponente

Tutti gli argomenti relativi al progetto sono discussi con il Proponente. Spesso, gli argomenti vengono anticipati via email al Proponente, in modo da avere un argomento comune.
Le attività necessarie sono quindi:

- Pianificare una riunione con il Proponente, con specifico "ordine del giorno";
- Redazione del verbale della riunione, contenente i punti salienti;
- Approfondimenti autonomi e/o collegiali, in merito ad argomenti specifici;
- Eventuali modifiche ai documenti, con nuove indicazioni ricevute durante il colloquio.

Gli strumenti utilizzati sono:

- Zoom, per le riunioni con il Proponente;
- Overleaf, per la redazione dei verbali;
- Gmail, per l'invio delle email per concordare l'appuntamento;
- Dispositivo con connessione Internet.

2.1.4.5 Colloquio con docenti

In caso di dubbi o revisioni, è possibile chiedere un colloquio con i docenti.
Le attività necessarie sono quindi:

- Pianificare una riunione con il docente, con specifico "ordine del giorno";
- Redazione del verbale della riunione, contenente i punti salienti;
- Approfondimenti autonomi e/o collegiali, in merito ad argomenti specifici;
- Eventuali modifiche ai documenti, con nuove indicazioni ricevute durante il colloquio.

Gli strumenti utilizzati sono:

- Zoom, per le riunioni con il docente;
- Overleaf, per la redazione dei verbali;
- Gmail, per l'invio delle email per concordare l'appuntamento;
- Dispositivo con connessione Internet.

2.1.5 Strumenti

Per il processo di fornitura sono stati utilizzati i seguenti strumenti:

- Jira Software, per gestire le task da svolgere e i membri del gruppo a cui erano assegnate;
- Zoom, per le riunioni con i docenti ed il Proponente;
- Overleaf, per la redazione di documenti;
- Gmail, per l'invio delle email per concordare gli appuntamenti.

2.1.6 Metriche

Per il processo di Fornitura sono state individuate le seguenti metriche. Analizzate nel dettaglio nel Piano di Qualifica (versione 1.0.0):

- MPC01 - Budget cost of work scheduled (BCS);
- MPC02 - Actual cost of work performed (ACS);
- MPC03 - Scheduled variance;
- MPC04 - Budget variance;
- MPC05 - Bugs for Line of Code.

2.2 Sviluppo

2.2.1 Scopo

Il processo di sviluppo definisce le attività e i compiti necessari per ottenere un prodotto finale che soddisfi i requisiti del proponente e le regole dell'Ingegneria del Software.

2.2.2 Aspettative

Una corretta implementazione di tale processo si basa sulle seguenti aspettative:

- Realizzazione di un prodotto finale:
  - Che sia conforme alle richieste del proponente;
  - Che superi i test di verifica descritti nel Piano di Qualifica (versione 1.0.0);
  - Che superi i test di validazione descritti nel Piano di Qualifica (versione 1.0.0);
- Fissaggio dei vincoli tecnologici e di design;
- Fissaggio degli obiettivi di sviluppo.

2.2.3 Procedure di sviluppo ed attività

Il processo di Sviluppo, in accordo con lo standard *ISO/IEC 12207*, si compone delle seguenti procedure, composte da singole attività, qui descritte:

- Analisi dei requisiti;
- Progettazione;
- Codifica.

2.2.4 Analisi dei requisiti

2.2.4.1 Scopo

Gli Analisti hanno il compito di individuare ed elencare in modo formale i requisiti del capitolato, i quali possono essere estrapolati da più fonti. Il documento contenente tali informazioni è l'Analisi dei Requisiti, il quale espone:

- **Descrizione generale del prodotto:** dove vengono definiti gli obiettivi del prodotto e i requisiti minimi e opzionali estrapolati dal capitolato d'appalto;

- **Casi d'uso:** dove vengono identificati i casi d'uso individuati sulla base delle potenziali funzionalità dell'applicativo;
- **Requisiti:** dove vengono elencati i requisiti individuati.

2.2.4.2 Attività

Le attività necessarie sono:

- Comprendere ed estrarre i requisiti dal Capitolato, fornito dal Proponente;
- Colloqui con il Proponente, per definire ulteriori requisiti non espliciti;
- Approfondimenti autonomi e/o collegiali, in merito ad argomenti specifici;
- Definizione degli *Use Case* e dei relativi diagrammi;
- Organizzazione dei requisiti in tipologie;
- Redazione del documento Analisi dei Requisiti;
- Presentazione dell'Analisi dei Requisiti al Proponente, per accettazione finale;
- Presentazione dell'Analisi dei Requisiti ai docenti, per la valutazione.

2.2.4.3 Denominazione dei Requisiti

$$R[Tipologia][Codice]$$

Composto da:

- R: acronimo di requisito;
- Tipologia:
  - F - Funzionale;
  - Q - Qualitativo;
  - P - Prestazionale;
  - V - Vincolo;
- Codice: ID del requisito.

2.2.4.4 Denominazione dei casi d'uso

$$UC[NominativoCaso][CodiceCasoBase].[CodiceSottoCaso]$$

Composta da:

- UC: acronimo di *"Use Case"*;
- NominativoCaso:
  - C: *use cases* relativi al CAPTCHA;
  - CE: estensioni di *use cases* relativi al CAPTCHA;
  - W: *use cases* relativi all'applicazione web;
  - WE: estensioni di *use cases* relativi all'applicazione web;
- CodiceCasoBase: ID del caso d'uso;
- CodiceSottoCaso: ID opzionale per i sottocasi di un caso d'uso.

2.2.4.5 Struttura dei casi d'uso

Ogni caso d'uso è descritto da:

- **Id:** codice identificativo del caso d'uso, stabilito come enunciato sopra;
- **Nome:** stringa titolo del caso d'uso posta dopo l'id;
- **Diagramma UML:** diagramma per rappresentare graficamente il caso d'uso;
- **Attori:** entità esterne al sistema che interagiscono con esso. Ne esistono due varianti:
  - Primario: interagisce con il sistema per raggiungere un obiettivo;
  - Secondario: aiuta il primario a raggiungere l'obiettivo. Non utilizzato.
- **Precondizione:** descrive lo stato del sistema prima del verificarsi del caso d'uso;
- **Postcondizione:** descrive lo stato del sistema dopo che si è verificato il caso d'uso;
- **Scenario principale:** elenco numerato che descrive il fusso degli eventi del caso d'uso;
- **Scenario secondario/alternativo:** elenco numerato che descrive il flusso degli eventi del caso d'uso dopo un evento imprevisto che lo ha deviato dal caso principale. Può non esserci o possono esserci più di uno;
- **Estensioni:** utilizzate negli scenari alternativi. Se si verifica una determinata situazione, il caso d'uso collegato all'estensione viene interrotto.

2.2.5 Progettazione

2.2.5.1 Scopo

L'attività di Progettazione avviene ad opera dei Progettisti, i quali hanno il compito di definire le caratteristiche essenziali del prodotto software richiesto, in funzione di quanto esposto nell'Analisi dei Requisiti.

2.2.5.2 Descrizione e procedure

La progettazione si articola nelle seguenti procedure:

- Progettazione della *Technology Baseline*: nella quale viene eseguita una prima analisi ad alto livello delle tecnologie che verranno coinvolte nello sviluppo del prodotto, la quale porta alla produzione di un PoC e di una conseguente *Technology Baseline*;
- Progettazione Architetturale: nella quale verrà eseguita una definizione ad alto livello dell'architettura del prodotto e delle sue componenti, insieme alla definizione dei test di integrazione;
- Progettazione di Dettaglio: nella quale verrà eseguita una definizione delle specifiche di dettaglio dell'architettura del prodotto e delle sue componenti, scomposte in unità, insieme ai diagrammi atti a descriverle e ai test di verifica. Tali informazioni costituiranno la *Product Baseline*.

2.2.5.3 Progettazione della *Technology Baseline*

La *Technology Baseline* è una analisi ad alto livello delle tecnologie necessarie, per individuare quali tecnologie utilizzare, verificare se sono adatte a soddisfare gli scopi del progetto e l'integrazione tra tutte le tecnologie.
Le attività necessarie sono quindi:

- Analisi delle necessità tecnologiche;

- Approfondimento sulle singole tecnologie;
- Valutazione della curva di apprendimento;
- Test di integrazione tra le tecnologie scelte;
- Composizione di un PoC;
- Redazione del documento "Scelta delle tecnologie".

Gli strumenti utilizzati sono:

- Overleaf, per la redazione del documento "Scelta delle tecnologie";
- Ambienti di sviluppo, per fare i vari test;
- Account Github con relative credenziali;
- Dispositivo con connessione Internet.

2.2.5.4 Progettazione Architetturale

La progettazione architetturale è una progettazione ad alto livello, che definisce le parti del prodotto e la sua architettura.
Le attività necessarie sono quindi:

- Approfondimenti autonomi e/o collegiali, in merito ad argomenti specifici;
- Suddivisione del progetto un parti più piccole;
- Progettazione architetturale delle parti più piccole;
- Definizione dei test di integrazione;
- Colloquio con docenti, per chiarire lacune;
- Unione delle progettazioni minori in un'unica progettazione.

Gli strumenti utilizzati sono:

- Zoom, per le riunioni con il docente;
- Overleaf, per la redazione delle progettazioni minori e la scrittura dei testi di integrazione;
- Draw.io, per creare qualche diagramma generico;
- Dispositivo con connessione Internet.

2.2.5.5 Progettazione di Dettaglio

La progettazione in dettaglio riguarda i dettagli del prodotto e le sue componenti.
Le attività necessarie sono quindi:

- Definizione in parti più piccole il progetto e delle sue componenti;
- Creazione dei diagrammi UML;
- Scelta dei test di verifica;
- Colloquio con docenti, per chiarire lacune;
- Redazione del documento "Specifica Tecnica".

Gli strumenti utilizzati sono:

- Zoom, per le riunioni con il docente;
- Overleaf, per la redazione dei verbali;
- Gmail, per l'invio delle email per concordare eventuali appuntamenti con i docenti;
- Dispositivo con connessione Internet.

2.2.6 Codifica

2.2.6.1 Scopo

Lo scopo del processo di codifica è l'effettiva realizzazione del prodotto software, svolto dal Programmatore.
Le procedure che si attuano sono le seguenti:

- Preparazione degli strumenti necessari alla codifica;
- Codifica e *testing*.

2.2.6.2 Preparazione degli strumenti necessari alla codifica

Ogni membro del team deve avere accesso allo stesso ambiente di sviluppo ed alle risorse necessarie.
Le attività necessarie sono quindi:

- Definizione dell'ambiente di sviluppo;
- Assicurarsi che tutti i membri abbiano le stesse risorse disponibili.

Gli strumenti utilizzati sono:

- Ambiente di sviluppo;
- Dispositivo con connessione Internet.

2.2.6.3 Codifica e *testing*

In caso di dubbi o revisioni, è possibile chiedere un colloquio con i docenti.
Le attività necessarie sono quindi:

- Pianificare una riunione con il docente, con specifico "ordine del giorno";
- Redazione del verbale della riunione, contenente i punti salienti;
- Approfondimenti autonomi e/o collegiali, in merito ad argomenti specifici;
- Eventuali modifiche ai documenti, con nuove indicazioni ricevute durante il colloquio.

Gli strumenti utilizzati sono:

- Ambiente di sviluppo;
- Dispositivo con connessione Internet;
- Definizione dei test di unità;
- *GitHub flow*, ovvero *workflow* per la codifica;
- Codifica secondo la progettazione.

2.2.7 Metriche

Per il processo di sviluppo sono state individuate le seguenti metriche, spiegate nel dettaglio nel Piano di Qualifica (versione 1.0.0):

- MPC06 - Indice Gulpease;
- MPC07 - Quality Metrics Satisfied;
- MPC08 - Code Coverage;
- MPC09 - Passed test cases percentage.

2.2.8 Strumenti

Gli strumenti utilizzati sono:

- Zoom, per le riunioni con i docenti ed il Proponente;
- Overleaf, per la redazione di documenti;
- Gmail, per l'invio delle email per concordare gli appuntamenti;
- Draw.io, per la creazione dei diagrammi;
- Google Drive, per condividere i diagrammi;
- Node.js: *runtime* JavaScript largamente utilizzata per la creazione di applicazione di rete;
- Docker: software per eseguire il codice in ambienti isolabili, minimali e facilmente distribuibili chiamati container Linux; *runtime* JavaScript largamente utilizzata per la creazione di applicazione di rete;
- React.js: Libreria JavaScript per la creazione di UI, miglioramento delle performance di renderizzazione delle pagine, facilitazione di manutenibilita e testing; ‘
- Visual Studio Code: IDE performante ed estendibile scelto dal gruppo per lo sviluppo del codice.
- Git: sistema di versionamento del codice.

3 Processi di Supporto

3.1 Documentazione

3.1.1 Scopo

Lo scopo del presente processo è la definizione degli standard e degli strumenti necessari alla stesura di tutti i documenti del progetto.

3.1.2 Ciclo di vita dei documenti ed attività per la loro realizzazione

Ogni documento sviluppato dal gruppo **Dot Seven** segue le seguenti fasi di vita (attività):

- Strutturazione: viene stesa una scaletta preliminare elencante i macro argomenti da trattare e sviluppare;
- Redazione: il documento viene scritto utilizzando un approccio incrementale. Viene considerato redatto una volta scritto nella sua interezza;
- Verifica: ogni sezione redatta viene verificata da almeno una persona, diversa dal redattore della stessa;
- Approvazione: il Responsabile di Progetto stabilisce la validità, il documento è completo in ogni sua parte e pronto per essere rilasciato, rendendolo disponibile per il consulto esterno.

3.1.3 Strumenti necessari

Gli strumenti necessari sono:

- Overleaf, per la redazione dei verbali;
- Discord, per eventuali discussioni o riunioni su uno specifico argomento;
- Google Documenti, per la creazione di qualche bozza;
- Dispositivo con connessione Internet.

3.1.4 Struttura dei documenti

Ogni documento sviluppato segue la stessa struttura.

3.1.4.1 Prima Pagina

La prima pagina è uguale per tutti i documenti e presenta i seguenti elementi:

- Logo del gruppo, posizionato in alto e centrale;
- Titolo del documento e versione attuale;
- Nome dei componenti del gruppo;
- Breve intestazione: "Progetto Ingegneria del Software, Dipartimento di Matematica, Università degli Studi di Padova" e logo dell'Università;
- Data dell'ultima validazione;
- Email di riferimento del gruppo.

3.1.4.2 Indice

L'indice permette di avere una visione completa del documento e di individuare le varie parti, ogni voce è un collegamento ipertestuale alla parte del documento in cui viene trattata.

3.1.4.3 Tabella di versionamento

Tutti i documenti soggetti a continue integrazioni e modifiche devono contenere un registro delle modifiche dopo la prima pagina. Questa tabella contiene le seguenti colonne:

- Versione: versione che il documento raggiunge dopo la modifica;
- Data: data della modifica;
- Autore: membro del gruppo che ha apportato la modifica;
- Ruolo: ruolo ricoperto dall'autore che modifica;
- Motivazione: descrizione breve e concisa della modifica apportata.

Fanno eccezione i verbali interni ed esterni, i quali vengono redatti contemporaneamente agli incontri e non sono soggetti a modifiche successive.
Si è scelto di escludere anche il Glossario dal versionamento, essendo un documento che contiene definizioni, non necessita di modifiche al testo già steso, ma sono di aggiungere nuove parole.
Ogni nuovo termine inserito verrà subito verificato in modo che il documento sia costantemente in una forma stabile e pronto alla validazione.
In qualsiasi documento, quando si cita un altro documento, si deve inserire il numero di versione del documento.

3.1.4.4 Struttura dei verbali

I verbali dopo la prima pagina e l'indice seguono una struttura diversa dai documenti generici, sistematica e metodica:

- Informazioni di corredo, che comprendono:
  - Presenti: interni ed esterni al team;
  - Assenti;
  - Luogo della riunione;
  - Data;
  - Orario di inizio e di fine riunione;
  - Conduttore della riunione.
- Ordine del giorno: dichiarazione numerata e scelta a priori degli argomenti in discussione;
- Svolgimento: riassunto schematico per punti degli argomenti trattati con riferimento all'ODG e contenente i punti salienti della conversazione;
- Conclusioni: elemento discorsivo che racconta in breve la retrospettiva dello svolgimento. Serve ad avere un *overview* sulla riunione;
- Attività dello Sprint successivo: deve essere presente un elenco delle attività assegnate, complessivo di codice identificativo all'interno dell'*Issue Tracking System* adottato.

3.1.5 Classificazione dei documenti

I documenti redatti sono divisi in due categorie: documenti ad uso interno ed ad uso esterno.
Documenti ad uso interno
Sono documenti ad uso interno quelli pensati unicamente per l'uso da parte di membri del gruppo:

- Verbali interni;
- Norme di progetto;
- Bozze varie di appunti.

Documenti ad uso esterno
Sono documenti ad uso esterno quelli destinati ad essere divulgati al di fuori del gruppo:

- Verbali esterni;
- L'Analisi dei Requisiti;
- Il Piano di Progetto;
- Il Piano di Qualifica;
- Il Glossario.

3.1.6 Norme tipografiche

3.1.6.1 Nomenclatura dei documenti

Ogni documento prodotto viene denominato utilizzando la stessa convenzione:

- Le iniziali delle parole avranno la lettera maiuscola;
- Gli articoli saranno scritti totalmente in minuscolo;
- Le parole saranno separate da underscore;
- Non sarà indicata la versione, le date di modifica o di creazione.

Esempio: [Nome_del_Documento]

Per i verbali la nomenclatura funzionerà nel seguente modo:

- "int"/"est" per distinguere il verbale tra interno o esterno;
- Data della riunione in questo formato [YYYY-MM-DD];
- In caso di verbali esterni viene aggiunto il nome dell'entità esterna con la quale si è tenuta la riunione;
- Ognuna di queste componenti viene separata da un underscore.

esempio: [int_2022-12-16] oppure [est_2022-12-16_Zucchetti]

3.1.7 Citazione

Ogni volta che un documento cita un'altro documento (presente in *repo*), è necessario indicare anche la versione a cui ci si riferisce.
Ogni volta che si inserisce un *link* (collegamento ipertestuale) è necessario indicare anche la data di ultimo accesso al sito stesso.

3.1.8 Strumenti

Per la stesura dei documenti ufficiali o con un minimo di importanza vengono utilizzate le seguenti tecnologie:

- LaTeX: https://www.latex-project.org/ (data ultimo accesso al link: 12 giugno 2023);
- Overleaf: https://it.overleaf.com (data ultimo accesso al link: 12 giugno 2023).

Per la stesura di brevi bozze, documenti ufficiosi e note si è deciso di utilizzare la suite Google Workspace.

3.2 Gestione della configurazione

3.2.1 Scopo

La gestione della configurazione è un processo che pone come obiettivo la gestione e il controllo della produzione del codice e dei documenti in maniera ordinata e metodica.
Per ogni oggetto sottoposto a configurazione viene garantito il versionamento e una posizione specifica nella repository per permettere il mantenimento dell'integrità del prodotto.

3.2.2 Stadi della configurazione

Il gruppo ha scelto di usare una configurazione a 5 stadi:

- *TODO*: nel *backlog*;
- *IN PROGRESS*: assegnata ad uno *sprint*;
- *DEV*: l'assegnatario sta lavorando correntemente;
- *VERIFY*: l'assegnatario ha concluso ed il verificatore può iniziare la verifica;
- *DONE*: è verificata e pronta ad essere integrata.

Questi stadi sono definiti nell'*Issue Tracking System* adottato (Jira). Ogni attività viene inserita nello stadio "TODO" e successivamente transiterà in altri stadi.

3.2.3 Versionamento

Ogni documento attraversa varie versioni di produzioni che sono classificate in [X].[Y].[Z], dove :

- X corrisponde a una *milestone* o*major release*, approvata e pronta al rilascio. La numerazione parte da 0;
- Y indica una *minor release*, modifiche importanti. La numerazione parte da 0 e si azzera ad ogni incremento di X;
- Z viene incrementato ad ogni modifica incrementale minore. La numerazione parte da 0 e si azzera ad ogni incremento di X o Y.

Ogni documento racchiude solo contenuti verificati: pertanto, il numero di versione sarà incrementato solo a seguito di ogni verifica di attività andata a buon fine.

3.2.4 Tecnologie e strumenti

Per il progetto viene utilizzato il sistema di controllo di versione distribuito *Git*, nello specifico con il servizio *GitHub*.

3.2.4.1 Struttura del repository documentale

```
├── **Nome della revisione**
│   ├── Documenti interni
│   │   ├── Verbali interni
│   │   │   └── int_**YYYY-MM-DD**_[partecipante esterno].pdf
│   │   └── **Nome_del_Documento_Interno**.pdf
│   ├── Documenti esterni
│   │   ├── Verbali esterni
│   │   │   └── est_**YYYY-MM-DD**_[partecipante esterno].pdf
│   │   └── **Nome_del_Documento_Esterno**.pdf
│   └── Lettera_di_Presentazione_[Sigla della revisione].pdf
└── README.md
```

Qui sotto elencate le *repositories* con il loro contenuto:

1. dotseven-captcha
2. demo-captcha
   (a) res
3. project-docs
   (a) Candidatura
      i. Documenti esterni
      ii. Documenti interni
   (b) *Requirements and Technology Baseline*
      i. Documenti esterni
      ii. Documenti interni
   (c) *Product Baseline*
      i. Documenti esterni
      ii. Documenti interni

3.2.4.2 Caricamenti dei documenti nel repository *"project-docs"*

Il repository contiene i documenti necessari, che vengono caricati dopo una modifica rilevante o dopo un insieme di modifiche minori, ritenute sufficienti per giustificare un aggiornamento dei documenti prodotti.
Gli aggiornamenti dei vari documenti possono essere sollecitati sia dal responsabile che dal team stesso, tale azione comporta la notifica di entrambe le parti tramite i canali di comunicazione previsti.
Ogni caricamento è necessariamente preceduto dalla validazione del documento, o dei documenti, che s'intende caricare.
La validazione, come spiegato successivamente, richiede che il documento riceva un'analisi approfondita ed accurata. Il caricamento avviene solamente se tale procedura ha esito positivo.

3.3 Gestione della qualità

3.3.1 Scopo

Lo scopo del processo di gestione della qualità è di assicurare che i requisiti di qualità individuati dagli *stakeholder* e le esigenze espresse dal proponente vengano rispettate dai prodotti e processi da sviluppare. La gestione della qualità verrà approfondita nel documento "Piano di Qualifica" (versione 1.0.0).

3.3.2 Aspettative

Obiettivo del presente processo è:

- Avere un livello quantificabile della qualità dei processi attuati;
- Valutare le attivita del gruppo;
- Avere un controllo continuo della qualità del prodotto, in modo da poter verificare che esso rispetti le aspettative del proponente.

3.3.3 Denominazione metriche

Tutte le metriche che sono definite nel Piano di Qualifica (versione 1.0.0) sono identificate univocamente dalla seguente convenzione:

$$M[Tipologiametrica][CodiceMetrica]$$

Composta da:

- M: acronimo di Metrica;
- Tipologiametrica:
  - PC: la metrica fa parte della qualità di processo;
  - PD: qualità di prodotto.
- CodiceMetrica: ID della metrica.

3.4 Verifica

3.4.1 Scopo

Lo scopo è definire come attuare il processo di verifica, per accertarsi che non ci siano errori durante lo sviluppo del prodotto. La verifica viene attuata ad ogni processo in esecuzione.

3.4.2 Norme interne

- Se la verifica non va a buon fine si avvia un dialogo tra implementatore e verificatore che deve portare ad una soluzione pacifica; il responsabile ha l'ultima parola in caso di disaccordi;
- Ogni implementatore si impegna ad iterare il suo compito qualora la verifica fallisse e a comprenderne le cause entro i limiti del possibile.

3.4.3 Verifica della documentazione

Ogni membro del gruppo che modifica un documento deve seguire le indicazioni riportate nell'*issue* relativa. Una volta terminata, deve aggiornare lo stato su *Jira* in *DA VERIFICARE*, aggiungendo il link al documento come commento. È buona norma notificare che il documento è pronto alla verifica, sul canale *discord* appropriato.
Il verificatore dovrà fare una lista delle modifiche che vanno apportate e inserirle attraverso la funzione di *Review* di overleaf; dopodiché, dovrà commentare l'*issue* su *Jira* indicando che ha verificato e cambiare lo stato del documento in *VERIFICATO* notificando al redattore che ci sono delle notifiche da apportare.
Il documento deve essere verificato nella sua interezza, in modo da segnalare tutti gli errori ortografici.
Le frasi devono essere ben formate e non troppo lunghe, in modo da non generare confusione.
Il verificatore deve anche controllare che le date siano corrette, che la tabella di versionamento sia stata aggiornata e che le modifiche aggiunte siano state effettivamente riportate.

3.4.3.1 Strumenti necessari

Gli strumenti necessari sono:

- Overleaf, per la lettura dei documenti ed eventuali modifiche;
- Jira, per individuare la *issue* e la comunicazione riguardante la stessa.

3.4.4 Verifica del codice

La verifica del codice prodotto è molto importante, poiché controlla che non siano stati inseriti degli errori nel prodotto finale.
La verifica del codice viene resa automatizzata e ripetibile, tramite strumenti appositi.
La verifica avviene tramite due tipi di analisi:

- Analisi statica, che non richiede l'esecuzione del codice;
- Analisi dinamica, che richiede l'esecuzione del codice.

I test vengono eseguiti in locale, in modo da avere un riscontro immediato di qualche errore. Successivamente, al *push* del codice, si attivano le *GitHub Actions*, che eseguono nuovamente tutti i test sul codice nella *repository*.
I test effettuati sono descritti nel Piano di Qualifica (versione 3.3.1).

3.4.4.1 Tipologie di test

I test eseguiti sul codice sono differenti, a seconda dello scopo del test stesso:

- Test di unità: sono eseguite sulle singole unità di codice. I test sono redatti dallo sviluppatore stesso, durante l'attività di sviluppo;
- Test di integrazione: il loro scopo è di accertarsi che le unità che compongono il prodotto riescano ad interfacciarsi ed interagire;
- Test di sistema: viene eseguito quando tutte le sue unità sono state integrate e dopo che tutti i test di integrazione hanno avuto esito positivo. Si accerta che tutti i requisiti concordati siano stati soddisfatti;
- Test di regressione: verifica che una modifica al codice non interrompa il funzionamento di altre componenti, facendo "regredire" il prodotto in termini di funzionalità;
- Test di accettazione (detto anche "collaudo"): viene fatto in presenza del committente, che si accerta che tutti i requisiti utente siano stati soddisfatti.

3.4.4.2 Denominazione dei test

La denominazione dei test segue la seguente nomenclatura:

$$T[Tipologia][Codice]$$

Composto da:

- T: iniziale del termine "test";
- Tipologia:
  - U - Unità;
  - I - Integrazione;

  - S - Sistema;
  - R - Regressione;
  - A - Accettazione;
- Codice: ID del requisito.

3.4.4.3 Strumenti necessari

Gli strumenti necessari si dividono in

- **Strumenti per l'analisi statica**, tra questi rientrano
  - **Complilatore Java**, alla versione 8.0.660.18, che si occupa di tradurre il codice sorgente in un codice di più basso livello;
  - **JaCoCo**, alla versione 0.8.10, che permette la *Continuos Integration* in Java.
- **Strumenti per l'analisi dinamica**, tra cui
  - **GitHub Actions**, che permette l'automazione della *pipeline* di compilazione, test e distribuzione;
  - **React Testing Library**, alla versione 14.0.0, per i test dei componenti sviluppati in ReactJS;
  - **JaCoCo**, alla versione 0.8.10, che permette la *Continuos Integration* in Java.

3.5 Validazione

3.5.1 Scopo

Lo scopo della validazione è stabilire se il prodotto è in grado di soddisfare l'obiettivo per il quale è stato creato.

3.5.2 Descrizione

Questo processo avviene dopo il processo di verifica. L'obbiettivo è accertare la correttezza delle attività di verifica svolte durante tutto il ciclo di vita del progetto e assicurare che il prodotto finale soddisfi tutti i requisiti individuati.
Le attività necessarie sono quindi:

- Individuazione dei requisiti da soddisfare;
- Lettura critica del documento;
- Se la validazione ha esito positivo, incrementare il numero di versione.

Gli strumenti utilizzati sono:

- Overleaf, per la lettura e modifica dei documenti;
- Jira, per avere lo stato di validazione dei documenti;
- Dispositivo con connessione Internet.

4 Processi Organizzativi

4.1 Gestione dei processi

4.1.1 Scopo

La gestione dei processi contiene le attività e i compiti generici necessari ai membri del gruppo per l'organizzazione e la gestione dei ruoli di ogni componente. Questo processo porta alla creazione del documento Piano di Progetto (versione 2.0.0).

4.1.2 Ruoli di progetto

Il Responsabile di Progetto ha il compito di suddividere i ruoli e l'assegnazione oraria per i membri del gruppo, garantendo che ognuno di essi assuma, nel corso del progetto, almeno una volta ogni ruolo.
Per semplificare la redazione di altri documenti si useranno le seguenti sigle: responsabile (RE), amministratore (AM), analista (AN), progettista (PT), programmatore (PR), verificatore (VE).
I ruoli richiesti dal progetto sono qui di seguito descritti, con le attività che deve svolgere.

4.1.2.1 Responsabile

È la figura professionale di riferimento sia per il committente sia per il fornitore e assume il ruolo di intermediario tra i due. Le sue mansioni sono:

- Elaborare piani e scadenze;
- Approvare il rilascio di prodotti parziali o finali (SW, documenti);
- Coordinare le attività del gruppo.

4.1.2.2 Amministratore

È la figura professionale incaricata del controllo e dell'amministrazione di tutto l'ambiente di lavoro, con piena responsabilità sulla capacità operativa e sull'efficienza. Le sue mansioni sono:

- Ricercare, studiare e mettere in opera risorse per migliorare l'ambiente di lavoro e automatizzarlo ove possibile;
- Assicurare l'efficienza di procedure, strumenti e tecnologie a supporto del *way of working*.

4.1.2.3 Analista

È la figura professionale che possiede maggiori conoscenze riguardo il dominio applicativo del problema. Le sue mansioni sono:

- Studiare il problema e il relativo contesto applicativo;
- Redigere l'analisi dei requisiti.

4.1.2.4 Progettista

È la figura professionale che gestisce gli aspetti tecnologici e tecnici del progetto sulla base di competenze costantemente aggiornate. Le sue mansioni sono:

- Definire il design della soluzione al problema a partire dai requisiti definiti dall'Analista;
- Effettuare scelte riguardanti gli aspetti tecnici e tecnologici del progetto, favorendone l'efficacia e l'efficienza.

4.1.2.5 Programmatore

È la figura professionale responsabile della codifica del progetto e delle componenti di supporto che serviranno per effettuare le prove di verifica e validazione sul prodotto. Le sue mansioni sono:

- Scrivere un codice pulito e facile da mantenere che rispetti le Norme di Progetto;
- Partendo dal design e dalle indicazioni del Progettista, implementare le varie componenti del software.

4.1.2.6 Verificatore

È la figura professionale incaricata del controllo del lavoro svolto dagli altri componenti del gruppo sulla base delle proprie competenze tecniche, esperienza e conoscenza delle norme. Le sue mansioni sono:

- Esaminare i prodotti in fase di revisione, con l'ausilio delle tecniche e degli strumenti definiti nelle Norme di Progetto;
- Verificare la conformità dei prodotti ai requisiti funzionali e di qualità.

4.1.3 Coordinamento

L'attività di coordinamento è responsabile della gestione delle comunicazioni interne, esterne e delle riunioni.

4.1.3.1 Norme da seguire e strumenti da utilizzare

1. **Controllare frequentemente i canali di comunicazione**
   Con il termine "frequentemente" si intende almeno una volta al giorno perché potrebbe esserci un'emergenza o qualcuno che richiede supporto. I canali di comunicazione sono Telegram, Discord, Jira e Gmail;
2. **Rispondere con un ACK ai messaggi che richiedono conferma di presa visione**
   Come "ACK" si intende: almeno una reazione (thumb up) al messaggio, oltre ad un'eventuale richiesta di chiarimento o ad un commento. Richiede conferma qualunque messaggio che la espliciti o che riguardi l'organizzazione interna del gruppo;
3. **Essere concisi e specifici nelle comunicazioni**;
4. **Utilizzare appropriatamente i canali di comunicazione:**
   le interazioni non riguardanti il progetto devono avvenire su *Telegram* o sull'apposito canale *Discord*.

4.1.4 Comunicazione

4.1.4.1 Comunicazioni interne

Le comunicazioni interne riguardano esclusivamente i membri del gruppo e avvengono attraverso i seguenti strumenti:

- *Telegram*: per le comunicazioni più frivole, non riguardanti il progetto in sé, ma magari le lezioni, i ritardi ecc;
- *Discord*: viene usato per le informazioni riguardanti il progetto, è stato creato un canale per ogni macro-argomento, in modo che tutti i messaggi siano ordinati e classificati.

Lista dei canali:

- Avvisi;
- Analisi-requisiti;
- Analisi-problemi;
- *Useful-links*;
- Generale;
- Gregorio;
- Diario-di-bordo;
- Organizzazione-interna;
- *Proof-of-concept*;
- RTB;
- PB;
- Manuale-utente;
- Manuale-sviluppatore;
- Specifica-tecnica;
- Progettazione;
- Norme-di-progetto.

- Commenti di *Jira*: viene usato esclusivamente per la comunicazione tra chi prende in carico e completa una *issue* e chi deve verificare il lavoro svolto. Queste comunicazioni avvengono attraverso i commenti della *issue* interessata.

4.1.4.2 Comunicazioni esterne

Le comunicazioni esterne si svolgono unicamente attraverso l'email del gruppo: dotseventeam@gmail.com.

4.1.5 Riunioni

4.1.5.1 Riunioni esterne

Alle riunioni esterne oltre ai membri del gruppo parteciperanno anche soggetti esterni (proponente e/o committente). La richiesta di una riunione potrebbe provenire da entrambe le parti attraverso l'email ufficiale del gruppo. Il Responsabile di Progetto dovrà nominare uno scriba, incaricato di prendere appunti per poi occuparsi della stesura del verbale. Le riunioni esterne si svolgono tramite Zoom.

4.1.5.2 Riunioni interne

Le riunioni interne al team vengono fatte da remoto su piattaforma *Discord* e verbalizzate.
Le riunioni devono prevedere un ordine del giorno, che deve essere seguito fedelmente.
Dati i vari impegni lavorativi dei membri del gruppo è stato fissato un incontro settimanale per fine ed inizio *scrum* ogni lunedì ore 18.00-19.00. Se una persona non potrà partecipare deve avvertire il team con quanto più anticipo possibile.
Le riunioni avvengono tramite Discord. Quanto discusso nella riunione deve affrontare anche i temi di retrospettiva e pianificazione.
É dunque necessario rendicontare le ore di queste riunioni, in quanto tempo effettivo di progetto. Il ruolo con cui vengono rendicontate si basa sul ruolo assunto nel periodo precedente, in quanto é facilmente intuibile che eventuali dubbi o difficoltà possano sorgere prevalentemente dal lavoro affrontato nel periodo precedente. La verbalizzazione delle riunioni avviene tramite creazione di verbale in Overleaf.

4.1.5.3 Struttura delle riunioni interne

Le riunioni interne si svolgeranno in questo modo:

1. Prima di iniziare la riunione verranno scelti lo scriba (che si occuperà poi anche di verbalizzare), il *timekeeper* e lo *scrum master* (per comodità lo farà il responsabile del periodo);
2. Viene stipulato l'O.D.G. con rispettive tempistiche. Il responsabile propone la sua idea di odg e gli altri possono aggiungere punti o modificarli;
3. Si attua la chiusura dello sprint, facendo il resoconto e la verifica delle ore e dei progressi svolti. Ogni componente avrà già segnato sul foglio apposito le ore fatte, il responsabile controlla e conferma;
4. Si passa alla retrospettiva, momento in cui se sono stati riscontrati dei problemi vanno resi pubblici al gruppo;
5. Inizio dello *sprint* successivo con rispettivi compiti e tempistiche.

4.2 Formazione

4.2.1 Scopo

I membri del gruppo hanno il compito di informarsi e formarsi in modo autonomo sulle tecnologie e sugli strumenti necessari al completamento del progetto.
Si svolgeranno dei seminari interni al team sui seguenti argomenti:

- *Git* e *GitHub*;
- Stesura dei documenti in *LaTeX*.

Inoltre se uno dei componenti si trova in difficoltà con l'utilizzo di una tecnologia deve chiedere aiuto al gruppo che darà il pieno sostegno, in modo da rimanere sempre tutti sullo stesso piano.