



4OURSQUARED

4oursquared.unipd@gmail.com

Lumos Minima  
Imola Informatica

## Norme di Progetto

### *Informazioni*

<i>Redattori</i>	Alberti Nicolas
	Brotto Romina
	Ceccato Francesco
	Soldà Matteo
<i>Versione</i>	0.0.4
<i>Uso</i>	interno

### Descrizione

Questo documento contiene le procedure, gli strumenti e i criteri di qualità che verranno usati nel progetto.

Versione	Data	Redattore	Verificatore	Descrizione
0.0.4	22/04/2023	Brotto Romina		Stesura iniziale sezioni "3.3, 3.4, 4, 4.2"
0.0.3	22/04/2023	Alberti Nicolas		Stesura iniziale sezioni "1.1, 2.1, 2.2"
0.0.2	20/04/2023	Ceccato Francesco	Soldà Matteo	Terminata Sezione Convenzioni di Codifica
0.0.1	19/04/2023	Soldà Matteo		Terminata Sezione Documentazione
0.0.0	24/03/2023	Soldà Matteo	Brotto Romina	Prima stesura.

# Contents

<b>1</b>	<b>Introduzione</b>	<b>1</b>
1.1	Scopo del Documento	1
1.1.1	Scopo del prodotto	1
1.1.2	Glossario	1
<b>2</b>	<b>Processi Primari</b>	<b>1</b>
2.1	Fornitura	1
2.1.1	Scopo	1
2.1.2	Aspettative	1
2.1.3	Documenti	1
2.1.4	Strumenti	2
2.1.5	Repository Pubblica	2
2.1.6	Contatti	2
2.2	Sviluppo	2
2.2.1	Scopo	2
2.2.2	Aspettative	3
2.2.3	<i>Analisi dei Requisiti</i>	3
2.2.4	Organizzazione dei File	4
2.2.5	Convenzioni di Codifica	4
<b>3</b>	<b>Processi Secondari</b>	<b>6</b>
3.1	Documentazione	6
3.1.1	Ciclo di Vita	6
3.1.2	Strumenti Utilizzati	6
3.1.3	Struttura	6
3.1.4	Convenzioni Documentali	8
3.2	Versionamento	9
3.2.1	Repository Github	9
3.2.2	Convenzioni di Versionamento	9
3.2.3	Comandi Utili	9
3.3	Verifica	9
3.3.1	Regole Generali	9
3.3.2	Metodi di Verifica	10
3.4	Change Management	10
<b>4</b>	<b>Processi Organizzativi</b>	<b>10</b>
4.1	Gestione dei Processi	10
4.2	Miglioramento dei processi	10

# 1 Introduzione

## 1.1 Scopo del Documento

Il documento si prefigge lo scopo di definire i metodi e le attività chiave legate al *Way of Working*. Tutti i membri del gruppo si impegnano a seguire quanto riportato, al fine di ottenere consistenza e coerenza in tutti i documenti e in tutte le attività del gruppo, per raggiungere in modo efficace ed efficiente la realizzazione del prodotto finale.

### 1.1.1 Scopo del prodotto

TODO: inserire quello che abbiamo previsto nella candidatura e quello di cui abbiamo parlato nella presentazione.

### 1.1.2 Glossario

TODO: redigere un Glossario in un documento a parte oppure integrarlo qui; Indicare termine nel glossario con testo *corsivo* e con una 'G' in pedice.

# 2 Processi Primari

## 2.1 Fornitura

### 2.1.1 Scopo

Lo scopo del processo di fornitura è quello di descrivere e determinare ogni compito ed attività svolta dal fornitore, al fine di comprendere e soddisfare le richieste del proponente. La suddivisione e le tempistiche delle attività da svolgere saranno definite dal documento *Piano di Progetto*.

### 2.1.2 Aspettative

Le aspettative determinate dal processo di fornitura sono le seguenti:

- Individuare e definire i bisogni che il prodotto deve soddisfare;
- Definire i requisiti e i vincoli dei processi;
- Stimare i costi per la realizzazione del prodotto;
- Ottenere dei feedback da proponente e committenti riguardanti il lavoro svolto;
- Chiarire eventuali dubbi sorti durante il progetto;

### 2.1.3 Documenti

#### *Piano di Progetto*

Il *Piano di Progetto* è il documento che viene redatto e mantenuto durante tutta la durata del progetto. Al suo interno comprende:

- **Analisi dei rischi:** vengono analizzati i rischi che possono essere individuati durante il corso del progetto e vengono descritte le strategie messe in atto al fine di contenerne la gravità;
- **Pianificazione del lavoro;**
- **Preventivo e consuntivo delle ore e dei costi.**

## *Piano di Qualifica*

Al fine di produrre materiale di qualità, viene redatto il documento *Piano di Qualifica*, che descrive le norme riguardanti la qualità dei prodotti e dei processi del gruppo. Al suo interno comprende:

- **Qualità di processo;**
- **Qualità di prodotto;**
- **Specifiche dei test;**
- **Resoconto delle attività di verifica.**

### 2.1.4 Strumenti

Nel processo di fornitura, il gruppo utilizza i seguenti strumenti per produrre i documenti precedentemente elencati:

- **Google Calendar:** sistema di calendari, usato per inserire le scadenze e gli eventi significativi riguardanti il gruppo;
- **Microsoft PowerPoint (TODO: Google Presentazioni?):** è un software facente parte del pacchetto suite Microsoft Office, permette la realizzazione delle presentazioni del gruppo.
- **TODO: programma per Gantt**

### 2.1.5 Repository Pubblica

Durante lo sviluppo del progetto, il codice e la documentazione saranno depositati in una repository pubblica. La parte di interesse del proponente e del committente riguarda la cartella *public*.

La cartella sopra citata contiene la documentazione necessaria alle revisioni di periodo con il committente. Tale cartella rispecchia la struttura della cartella *docs*, situata nella parte privata.

I file che prevedono versionamento, riporteranno al loro interno la numerazione relativa alla versione attuale e un riassunto delle versioni precedenti.

Ogni modifica della parte pubblica deve essere approvata, tramite *pull request*, dal responsabile.

### 2.1.6 Contatti

Tutti i contatti ufficiali, sia con il proponente che con il committente, avverranno tramite la mail del gruppo [4oursquared.unipd@gmail.com](mailto:4oursquared.unipd@gmail.com).

Le mail per la presentazione ad una revisione di periodo necessita di una conferma unanime, valutata la preparazione. Questa mail deve contenere la lettera di presentazione con i link ai vari documenti aggiunti/aggiornati nella parte pubblica della repository.

## 2.2 Sviluppo

### 2.2.1 Scopo

Il processo di sviluppo ha come scopo quello di descrivere le attività e i task di analisi, progettazione, codifica, test, installazione e accettazione riguardanti il prodotto software in sviluppo.

### 2.2.2 Aspettative

Le aspettative prevedono:

- Determinazione dei requisiti del prodotto;
- Determinazione dei vincoli tecnologici e di design;
- Determinazione degli obiettivi di sviluppo;
- Realizzazione del prodotto finale, superando tutti i test individuati e soddisfacendo i requisiti e le aspettative del proponente.

### 2.2.3 Analisi dei Requisiti

#### Scopo

Lo scopo dell'*Analisi dei Requisiti* è quello di:

- Identificare requisiti obbligatori ed auspicabili richiesti dal proponente, tramite lo studio del capitolato;
- Supportare l'attività di pianificazione, fornendo informazioni utili al computo della mole di lavoro;
- Supportare l'attività di verifica, facilitandone il tracciamento dei requisiti.

#### Aspettative

L'attività si pone come risultato la creazione di un documento contenente tutti i requisiti richiesti dal proponente, includendo anche il punto di vista dell'utente che utilizzerà i prodotti. Sarà presente una sezione legata al tracciamento dai requisiti ai casi d'uso e viceversa.

#### TBD: Casi d'Uso

Un Caso d'Uso è l'insieme degli scenari che prevedono uno stesso obiettivo per un utente. Vengono descritte le interazioni con il sistema da parte di uno o più attori, senza fornire alcun dettaglio implementativo. È formato da:

- **Diagramma *UML*** (opzionale): indica la relazione con altri Casi d'Uso;
- ***TBD: Intestazione***, nel formato:

UC [Codice] - [Numero Caso d'Uso].[Sottocaso] [Titolo]

dove:

- UC indica "Use Case", ovvero "Caso d'Uso";
- [Codice] è il codice identificativo del Caso d'Uso,
- [Titolo] è il titolo del Caso d'Uso.

- **Attore/i** primario/i;
- **Attore/i** secondario/i (opzionale);
- **Descrizione**;
- **Scenario principale**;
- **Estensioni**;
- **Inclusioni**;
- **Precondizioni**;
- **Postcondizioni**;
- **Generalizzazioni**.

### TBD: Requisiti

I requisiti ricavati dall'analisi approfondita del capitolato, dalle discussioni con i componenti del gruppo e dal confronto con il proponente, vengono suddivisi in tre tipologie:

- **Funzionali:** descrivono le funzionalità e il comportamento che il prodotto deve presentare;
- **Qualità:** descrivono i vincoli sulla qualità del prodotto e dei suoi componenti;
- **Vincolo:** descrivono i vincoli legati all'implementazione del prodotto, come ad esempio sulla tecnologia da utilizzare.

Ogni requisito ha una determinata importanza, di seguito descritta:

- **Obbligatorio**
- **Desiderabile**
- **Opzionale**

Ogni requisito viene presentato nel formato univoco descritto di seguito: **TBD**

R[Tipologia][Caso d'Uso Relativo].[Sottocaso] - [Importanza]

Dove:

- **R:** Acronimo di "Requisito";
- **[Tipologia]:** indica la tipologia del requisito tra le seguenti:
  - **F:** Funzionale;
  - **Q:** Qualità;
  - **V:** Vincolo;
- **[Importanza]:** indica l'importanza del requisito tra le seguenti:
  - **O:** Obbligatorio;
  - **D:** Desiderabile;
  - **P:** Opzionale.

### 2.2.4 Organizzazione dei File

Tutta la documentazione sarà contenuta all'interno della cartella *docs/*. Questa sarà inoltre suddivisa in *interna* ed *esterna*, le quali conterranno rispettivamente la documentazione interna al gruppo e quella da condividere con il proponente e con il committente. I verbali interni saranno conservati nella sottocartella *docs/interni/verbali* mentre quelli esterni saranno conservati in *docs/esterni/verbali*.

Per ogni documento esisterà una nuova sottocartella omonima.

### 2.2.5 Convenzioni di Codifica

#### Convenzioni Linguistiche

- Ogni elemento del codice deve essere scritto in lingua inglese, ad eccezione dei commenti, che dovrebbero essere scritti in italiano per evitare incomprensioni.

## Convenzioni di Nomenclatura File

- Un file che contiene l'interfaccia o l'implementazione di una sola ed esclusiva classe (ad eccezione di classi interne e sottoclassi) dovrebbe essere individuato nel seguente modo: `NomeClasse.estensione`.

## Convenzioni Stilistiche

	Java	JavaScript	Python	TypeScript
Funzioni e Metodi	camelCase()	camelCase()	snake_case()	camelCase()
Classi	PascalCase			
Interfacce	PascalCase	PascalCase	NA	PascalCase
Namespace e Package	lowercase	NA	NA	camelCase
Costanti	SCREAMING_SNAKE_CASE			
Variabili Locali e Attributi	camelCase	camelCase	snake_case	snake_case
Variabili Globali	NA	camelCase	snake_case	snake_case

A seguire, in JavaScript, le funzioni e i metodi privati anteporranno il simbolo di underscore prima del nome.

Per quanto riguarda invece i linguaggi di markup, i tag e gli attributi andranno scritti in lowercase se primitivi o in kebab-case se custom o complessi.

Inoltre:

- I membri di una classe dovrebbero essere dichiarati e/o definiti nel seguente ordine:
  - Variabili di classe (statiche)
  - Variabili di istanza (attributi)
  - Nel seguente ordine (ove possibile):
    1. public
    2. protected
    3. private
  - Costruttore
  - Distruttore (se richiesto)
  - Metodi
- Un blocco di codice delimitato da parentesi graffe dovrebbe essere strutturato apponendo le parentesi graffe ognuna in una nuova riga;
- Si predilige l'uso della tabulazione per indentare la porzione di codice inclusa in un blocco.

## CSS



- Mantenere un file chiamato `nomepagina.css` per la versione desktop di ciascuna pagina;
- Mantenere un file chiamato `global.css` per le regole che si applicano ad ogni aspetto del sito, in particolare ai moduli universali quali, a titolo esemplificativo e non esaustivo, header e footer;
- Mantenere un file chiamato `mini.css` con accorgimenti per la versione mobile
- Specificare le unità di misura in *em*, *rem*, *%*, *vw*, *vh* (unità relative), a parte per i valori delle seguenti proprietà, i quali possono essere espressi in *px*:
  - *text-shadow*
  - *box-shadow*

## 3 Processi Secondari

### 3.1 Documentazione

#### 3.1.1 Ciclo di Vita

Ogni documento seguirà il seguente flusso:

- **Pianificazione**  
Il documento viene progettato ad alto livello secondo quanto richiesto.
- **Redazione**  
Un membro del gruppo, nel ruolo di redattore, stila il documento.
- **Revisione**  
Un membro del gruppo, nel ruolo di revisore, controlla l'assenza di errori grammaticali e il contenuto sia conforme alle norme di progetto.
- **Approvazione**  
Un membro del gruppo, nel ruolo di responsabile, verifica che il contenuto del documento sia corretto.

#### 3.1.2 Strumenti Utilizzati

Per la stesura dei documenti verrà utilizzato il linguaggio L<sup>A</sup>T<sub>E</sub>X.

Affinché i documenti risultino stilisticamente coerenti, dovranno includere al loro interno il file `docs/template/style.tex` per lo stile generale e il file `docs/template/front_page.tex` per utilizzare il template della prima pagina.

#### 3.1.3 Struttura

Ogni documento sarà così strutturato:

1. Pagina di intestazione
2. Pagina con lista delle versioni, ove necessario
3. Pagina indice, ove necessario
4. Una o più pagine di contenuto

Le varie sezioni saranno così definite:

#### **Pagina di Intestazione**

- Logo e nome del gruppo
- Nome del progetto e azienda cliente
- Mail di contatto
- Titolo del documento
- Info generali
  - Redattori
  - Revisori
  - Responsabile
  - Versione, ove necessario
  - Destinazione d'uso (interno o esterno)
- Sinossi

### Elenco delle Versioni

L'elenco delle versioni è una tabella così definita:

- Versione
- Data
- Redattore
- Revisore
- Descrizione

Le righe della tabella sono organizzate in ordine cronologico inverso.

### Indice

L'indice è stato creato utilizzando il comando `\tableofcontents`

### Norme di Progetto

- **Scopo:** Lo scopo del documento è quello di definire le procedure, gli strumenti e criteri di qualità al fine di stabilire un *Way of Working*. Ogni membro del gruppo sarà tenuto a rispettare le indicazioni del documento presentate
- **Titolo:** Norme di Progetto
- **Nome del File:** *norme\_di\_progetto.tex*

### Verbali

- **Scopo:** Lo scopo dei verbali è quello di rendicontare ciò che viene discusso durante una riunione, sia interna che esterna, delineando gli impegni che ne derivano
- **Titolo:** Ogni verbale sarà titolato con "*Verbale del* < data > [*con* < esterni >]"
- **Nome del File:** Tutti i file saranno chiamati < yyyy > \_ < mm > \_ < dd > \_ < tipo > .tex dove < yyyy > \_ < mm > \_ < dd > indica la data nella quale si è effettuata la riunione nel formato anno-mese-giorno; {tipo} indica se la riunione è avvenuta tra i soli membri del gruppo "I" o con persone esterne "E".
- **Indice:** Non presente in quanto non utile
- **Struttura:**
  - Data, ora e durata;
  - Luogo;

- Partecipanti (interni ed esterni, raggruppati per appartenenza e ordinati alfabeticamente secondo il cognome);
  - Ordine del giorno;
  - Conclusioni derivanti dalla riunione;
  - Impegni assunti.
- **Stesura:** All'inizio di ogni riunione verrà nominato un membro del gruppo che redigerà il verbale e un membro che lo validerà. Chi redige il verbale avrà 24 ore per completare il lavoro, lo stesso tempo sarà concesso al validatore per approvarlo e segnalare le modifiche da applicare.

### Piano di Progetto

- **Scopo:** Lo scopo del documento è quello di definire gli obiettivi da raggiungere, stabilendo le tempistiche e il responsabile, tenendo traccia dei progressi e valutando i costi sostenuti rispetto a quelli preventivati.
- **Titolo:** Piano di Progetto
- **Nome del File:**

### Piano di Qualifica

- **Scopo:** Lo scopo del documento è quello di definire le metriche e i requisiti minimi di qualità affinché un prodotto del progetto possa essere approvato.
- **Titolo:** Piano di Qualifica
- **Nome del File:**

#### 3.1.4 Convenzioni Documentali

- **Riferimento a file o cartelle:**
  - Per fare riferimento al nome di un file o di una cartella si utilizza il **testo monospaziato**;
  - Il nome di una cartella deve sempre terminare con `/`.
- **Stringhe e nomi:**
  - Le stringhe vanno scritte tra doppi apici;
  - Per indicare un parametro rappresentate una stringa si usa il testo tra parentesi angolari. Tale parametro non può contenere spazi;
  - Per indicare una parte di nome o stringa opzionale si usa il testo racchiuso tra parentesi quadre.
- **Riferimento tra documenti:**
  - Riferimenti interni al documento: comando `\ref`;
  - Riferimenti esterni al documento: nome della sezione del documento a cui si fa riferimento in corsivo.

## 3.2 Versionamento

### 3.2.1 Repository Github

Per il versionamento del progetto si è deciso di utilizzare *Git* sulla piattaforma *GitHub*. La repository è raggiungibile tramite l'indirizzo <https://github.com/4ourSquared/LumosMinima>.

### 3.2.2 Convenzioni di Versionamento

Per il versionamento si farà uso del Versionamento Semantico, secondo le linee guida. Questo tipo di versionamento si fonda sulle seguenti caratteristiche principali:

1. Un numero di versione deve essere nella forma  $X.Y.Z$  dove  $X, Y, Z$  sono numeri interi non negativi e non devono contenere zeri iniziali;
2. La versione *major zero* (es:  $0.X.Y$ ) serve solo per lo sviluppo iniziale e indica una versione non stabile del prodotto;
3. La versione *patch*  $Z$  (es:  $x.y.Z$ ) deve essere incrementata solo se sono state introdotte correzioni retrocompatibili del testo (modifica da parte del redattore);
4. La versione *minor*  $Y$  (es:  $x.Y.z$ ) deve essere incrementata se nella documentazione è stata introdotta una nuova modifica retrocompatibile (approvazione da parte del verificatore);
5. la versione *major*  $X$  (es:  $X.y.z$ ) deve essere incrementata solo se nella documentazione è stata introdotta una modifica non retrocompatibile con le versioni precedenti (pubblicazione da parte del responsabile).

### 3.2.3 Comandi Utili

- **Sincronizzazione con la repository remota:** `git pull`;
- **Creazione di un nuovo branch locale:** `git branch <nome_branch>`;
- **Passaggio ad un branch locale esistente:** `git checkout <nome_branch>`;
- **Aggiunta delle modifiche alla stage area:** `git add .`;
- **Creazione del commit con le modifiche:** `git commit -m <descrizione>`;
- **Push sul remote in un branch remoto non esistente:** `git push --set-upstream origin <nome_branch>`;
- **Push sul branch remoto esistente omonimo del branch locale:** `git push`.

Escludendo per il momento l'utilizzo di *git flow*, questi risultano i comandi da utilizzare.

## 3.3 Verifica

### 3.3.1 Regole Generali

- Il Verificatore si occupa di controllare le Pull Request aperte dagli altri membri del gruppo;
- Il Verificatore NON contribuisce ai prodotti che verifica;
- Il Verificatore NON modifica i prodotti che verifica;
- Il Verificatore, nel caso in cui trovi errori, è tenuto a commentare le PR usando la funzione apposita di GitHub.

### 3.3.2 Metodi di Verifica

Il verificatore può far uso dei test automatici presenti nel Piano di qualifica.

Per le sezioni di codice è consigliato un walkthrough del prodotto, similmente si dovrebbe chiedere a chi abbia scritto il codice eventuali delucidazioni su elementi poco chiari.

## 3.4 Change Management

Nel caso si dovessero cambiare sezione del codice bisogna controllare che eventuali modifiche non vadano ad impattare altri elementi.

Per evitare ciò si farà uso del Diagramma delle classi presente nelle Specifiche architeturali.

## 4 Processi Organizzativi

### 4.1 Gestione dei Processi

Si è scelto il modello Scrum con sprint di durata settimanale. Si è deciso di effettuare una riunione interna ogni mercoledì dove discutere dei seguenti punti:

- Resoconto dello Sprint precedente;
- Eventuali problemi riscontrati e soluzioni adottate;
- Resoconto di ore e costi per ogni componente;
- Pianificazione degli sprint successivi.

I resoconti dovranno essere preparati dai singoli membri prima di ogni esposizione.

Ogni incontro dovrà produrre un aggiornamento delle milestones (e diagramma di Gantt). La pianificazione dei prossimi sprint consisterà nel dividere il lavoro in item e decidere il numero di membri adatto per item.

Gli item pianificati non potranno subire modifiche o cancellazioni durante il ciclo di sprint e dovranno essere portati a termine entro il ciclo successivo. In caso emergessero importanti contestazioni o criticità il responsabile deciderà come agire.

Ogni ciclo di Scrum prevederà un nuovo responsabile, scelto per cognome in ordine alfabetico. Il responsabile dovrà assegnare le persone agli item considerando la rotazione dei ruoli e dovrà poi creare le issues su GitHub relative agli item e assegnarvi i membri.

Le issues dovranno essere collegate alle milestones di appartenenza dal responsabile, che si occuperà di aggiornare preventivo e consultivo totale, compilando nel Piano di Progetto la sezione del ciclo di Scrum corrispondente.

Si è inoltre deciso che in caso di problemi singolarmente insormontabili andrà notificato il responsabile su Telegram che si occuperà di assegnare un membro per la risoluzione.

Il responsabile dovrà creare la issue su GitHub relativa al problema assegnandovi la persona scelta. Le soluzioni adottate per la risoluzione del problema andranno documentate dal responsabile stesso.

### 4.2 Miglioramento dei processi

Durante l'attività di consuntivo di periodo dell'incontro settimanale, viene discusso il lavoro fatto fino ad ora, dando particolare peso ai problemi riscontrati.

In base a ciò si valutano le eventuali migliorie e correzioni applicabili.