**PROJECT PLAN**

***1. INTRODUZIONE***

Lo scopo del progetto è creare un sistema software per la gestione di un ospedale, in particolare per tenere traccia dei pazienti operati e degli interventi a cui devono sottoporsi. L'obiettivo principale è raggiungere una forma stabile e funzionante del programma, anche se non ancora commercializzabile, soprattutto dal punto di vista grafico.

*Spiegazione Problema*

Il paziente va in visita dal medico e si decide che deve essere operato; il medico inserisce il paziente nel sistema, il quale contiene una pagina anagrafica per ogni paziente (all’interno sono contenute le informazioni anagrafiche-mediche del paziente ed è importante che in ogni momento si possa accedere a queste informazioni).

Il paziente entra in una lista di graduatoria dell’intervento specifico a cui si deve sottoporre. La cabina di regia contatta il paziente per fornirgli le date degli esami che dovrà eseguire prima dell’intervento. Il paziente svolge gli esami e, una volta completati, viene chiamato per fare l’intervento; a questo punto la cabina di regia lo inserisce nella lista delle persone pronte per fare l’intervento, in modo tale che il medico, una settimana prima, può preparare la lista operatoria, che schedula giorno per giorno chi verrà operato (le liste operatorie possono essere modificate in qualsiasi momento dal medico).

Nel giorno dell’intervento si tiene traccia di tutti i movimenti, i rischi, l'anestesia adoperata, la diagnosi e la procedura relative all'operazione eseguita sul paziente in un verbale; nel verbale sono contenuti gli orari di inizio e fine dell'intervento, le azioni effettuate durante l'operazione e i membri dell'equipe medica cha ha effettuato l'operazione (i ruoli e le competenze rispetto al verbale possono variare per ogni componente).

Il software deve tenere traccia di tutte le liste a cui il paziente appartiene e monitorare come vengono gestiti i vari pazienti e medici per ogni intervento.

***2. MODELLO DEL PROCESSO***

Il Modello adottato per la costruzione del progetto è un framework SCRUM.

Lo **sprint** (lasso di tempo prestabilito per completare il proprio lavoro) dura una settimana, ma ogni settimana viene effettuato un **weekly scrum**, per discutere sull’andamento dell’intero progetto. Il ruolo di **Scrum Master** viene ricoperto da tutti i membri del gruppo, a seconda della porzione di progetto da completare (Project Plan, Diagrammi, Codice…).

Per svolgere al meglio il lavoro e per non creare confusione nella stesura, è necessario definire alcuni requisiti come prioritari (**product backlog**), i quali andranno soddisfatti per prima rispetto al resto.

I tre pilastri fondamentali nell’implementazione del modello SCRUM sono: *Trasparenza, Ispezione e Adattamento*.

Prima della stesura del progetto stesso, è necessaria un'attenta analisi per capire le varie problematiche legate ad esso.

***3. ORGANIZZAZIONE DEL PROGETTO***

Il gruppo di lavoro è composto da tre persone:

* Nicola Cattaneo
* Matteo Mangili
* Marco Saini

Ogni collaboratore è *scrum master* di una delle parti del progetto, secondo quanto segue:

- Stesura del Project Plan, Presentazione Progetto: Mangili

- Creazione dei Diagrammi, Interfaccia Grafica: Saini

- Stesura Codice Sorgente, Creazione Database: Cattaneo

I tre membri del team collaborano per lo sviluppo delle varie parti, tenendosi aggiornati quotidianamente, e in settimana si incontrano per discutere sull'andamento del progetto.

GitHub è lo strumento prescelto sia per la gestione della documentazione relativa al progetto sia per la condivisione di codice, diagrammi e richieste di modifica.

***4. STANDARD, LINEE GUIDA, PROCEDURE***

Il software implementato è un’architettura *Client-Server*:

*-**Client*, riferimento ai pazienti e ai medici che devono trovare le informazioni nel sistema.

*- Server*, riferimento al sistema stesso, che deve essere di facile utilizzo, comprensibile e sicuro.

Il Client e il Server devono essere sempre cooperanti tra di loro, al fine di non creare collisioni nell’utilizzo del software.

***5. ATTIVITÀ DI GESTIONE***

Per gestire tutte le parti prodotte nello sviluppo del progetto, è necessario fissare delle attività di gestione:

* Ogni membro del gruppo lavora sui suoi compiti e in caso di difficoltà, si rivolge ai colleghi, al fine di risolvere eventuali problematiche.
* Ogni settimana i membri del gruppo si incontrano per definire eventuali dubbi, sistemare gli errori e per fare un resoconto sull’andamento del progetto nella settimana passata.

Tali attività permettono un controllo continuo sullo sviluppo del progetto, rispondendo ad obiettivi e priorità fissati per il progetto.

Prima di apportare una modifica, ogni membro deve confrontarsi con gli altri collaboratori, al fine di capire se tale modifica è necessaria e, in caso affermativo, ottenere la loro approvazione a procedere.

***6. RISCHI***

I rischi maggiori sono:

* mancare la consegna finale del progetto causa impegni universitari e non di ciascun membro del team.
* difficoltà nel trovare giorni della settimana in cui tutti i membri del gruppo potevano incontrarsi (per effettuare un confronto sull’andamento del progetto), causa impegni extra-scolastici di ognuno.
* difficoltà a sviluppare un software che rispondesse a tutte le esigenze/requisiti del problema proposto.

***7. PERSONALE***

I tre membri del gruppo si dividono le varie competenze del progetto, secondo quanto segue:

* Nicola Cattaneo (project manager, progettista database, product owner, backend)
* Matteo Mangili (frontend, statical models developer, tester)
* Marco Saini (frontend, backend, reviewer, statical models developer)

Il ruolo di Scrum Master viene condiviso da tutti i membri del gruppo, per far apprendere la mansione e migliorare l’eperienza.

***8. METODI E TECNICHE***

Per modellare il programma con un'ampia visione dell'insieme di funzioni del progetto, si sfruttano vari tipi di Diagrammi in UML.

Per prima cosa, vanno identificati i casi d'uso e gli attori del problema proposto, al fine di costruire lo *USE CASE DIAGRAM*.

Vengono, in seguito, definite le varie classi, le relazioni tra di esse e tutti gli attributi/metodi di ciascuna classe, che verranno implementate nel *CLASS DIAGRAM*.

I legami che uniscono le singole classi possono essere visti come messaggi, ciò permette di definire lo *SEQUENCE DIAGRAM*, con cui è possibile identificare l'ordine corretto di istruzioni da eseguire per raggiungere lo scopo del problema.

Se si vuole vedere nel dettaglio lo stato di un oggetto del problema, si sfrutta lo *STATE CHART DIAGRAM.*

Per visualizzare/comprendere le azioni che specificano il comportamento definito dall'utente rispetto al sistema generale, si sfrutta l'*ACTIVITY DIAGRAM*.

Nel corso dell’intera progettazione, è sempre necessario usare un test di *Verifica & Validazione* (V&V):

* *Verifica:* il processo deve rispondere ai requisiti dettati dall’utente a inizio compilazione
* *Validazione:* il processo deve soddisfare i propri requisiti specificati

***9. GARANZIA DI QUALITÀ***

Per la progettazione del software applicativo, si fa riferimento alla norma ISO 9001: specifica i requisiti di un sistema di qualità per l'organizzazione; deve dimostrare la propria capacità di fornire prodotti che soddisfino i requisiti dei clienti e alle sue varie voci (in particolar modo alla sezione Pianificazione), per garantire il successo della creazione e utilizzo del progetto.

I punti fondamentali da tenere conto per la gestione della qualità del software sono: *Funzionamento del Prodotto, Revisione del Prodotto e Transizione del Prodotto* (a loro volta suddivisi in vari fattori che rispondono a determinate domande, al fine di migliorare l’intero progetto).

***10. PACCHETTI DI LAVORO***

Il Progetto è stato suddiviso in diverse parti distinte:

1. Intervista ad un'infermiera dell'Ospedale Papa Giovanni XXIII di Bergamo (BG), per raccogliere le informazioni generali alla stesura dell'intero progetto.

2. *Project Plan,* per definire i vari passaggi della stesura del progetto.

3. *Diagrammi,* per avere una rappresentazione schematica del progetto.

4. *Data Base,* per organizzare meglio le varie informazioni per ogni classe/attore del sistema.

5. *Codice*, per elaborare il programma attraverso un linguaggio di alto livello, anche con l’ausilio di un’ottima interfaccia grafica.

Ogni membro del gruppo collabora in maniera attiva allo sviluppo dei vari componenti.

***11. RISORSE***

Per la realizzazione finale del progetto il team ha usufruito di varie risorse:

* Ogni membro ha sfruttato il proprio computer portatile (PC) o il proprio tablet per lavorare sulle singole componenti dell'intero progetto.
* Utilizzo di varie applicazioni, quali *Google Drive, Microsoft Teams, Microsoft OneDrive* per raccogliere le informazioni necessarie al fine di realizzare un buon progetto (spesso si fa riferimento ai lucidi sul corso di Ingegneria del Software).
* Utilizzo di applicazioni di messagistica, per la comunicazione tra i membri del gruppo.
* *Eclipse IDE*, per la stesura del codice.
* *StarUML,* per la creazione dei vari diagrammi.
* *GitHub,* per condividere il progetto e notificare/gestire eventuali modifiche.

***12. BUDGET***

Il tempo è molto importante per la stesura dell'intero progetto: stimando che ogni membro del gruppo lavorerà 60 h per conto proprio (per un totale complessivo di 180 h), la maggior parte di esse verrà impegnata nella progettazione e sviluppo dell’intero progetto, mentre le restanti saranno di studio (al fine di raccogliere nozioni utili) e revisione (per controllare/risolvere eventuali errori).

***13. CAMBIAMENTI***

Ogni cambiamento apportato al progetto viene discusso tra i membri del team: qualora un membro volesse modificare una parte del progetto, è tenuto ad avvisare i restanti membri della modifica effettuata, attraverso l'uso di app di messaggistica. Le riunioni settimanali permettono di comprendere meglio le modifiche apportate al progetto e di operare sul sistema in presenza di tutti i membri del team.

L'uso di GitHub permette ai membri del gruppo di poter essere sempre aggiornati sullo sviluppo del progetto, qualora la comunicazione diretta non fosse possibile. Inoltre, su GitHub vengono notificate, attraverso delle issue, modifiche importanti da apportare (vengono assegnante ad uno/più membri che devono portarle a compimento).

***14. CONSEGNA***

La consegna finale del Progetto avverrà entro il 10/02/2024, tenendo aperta la possibilità di eventuali manutenzioni e modifiche entro il 15/02/2024.

Verranno sfruttate delle issue per effettuare la consegna del Project Plan e del progetto intero.