Project Plan

Indice

1. Introduzione

1.1 Contesto e motivazioni

Al giorno d'oggi, con la diffusione sempre più capillare della tecnologia e, in particolare, dei social network, le distrazioni sono sempre dietro l'angolo.

Per questo riteniamo sia necessario un supporto allo studio per tutti coloro che hanno difficoltà a mantenere la concentrazione: un'applicazione desktop finalizzata all'aumento della produttività. Presupponendo che l'eliminazione totale delle distrazioni non è realistica, l'idea è quella di unire un approccio ludico a un approccio produttivo in termini di studio, rendendo quest'ultimo più piacevole.

1.2 Obiettivi del progetto

Nel concreto, la componente ludica consiste nella costruzione di una città virtuale, il cui sviluppo è direttamente proporzionale al tempo di studio, misurato fintanto che l'utente resta all'interno dell'applicazione. L'utente sarà poi in grado di visualizzare i propri progressi ammirando la città costruita con il proprio impegno.

L'obbiettivo di questo progetto, in termini di qualità, è sviluppare un prodotto intuitivo e facilmente testabile (per la definizione di questi aggettivi facciamo riferimento ai fattori di qualità secondo McCall).

1.3 Responsabili del progetto

Per questo progetto i responsabili sono: Christian Miele, Filippo Monzani, Lorenzo Corbellini e Sergio Pedercini.

2. Modello di processo

2.1 Scelta del modello ibrido

Adottiamo un modello di processo ibrido, frutto dell'unione tra RAD e SCRUM.

2.2 Dettagli su RAD e SCRUM

In particolare, utilizziamo il concetto di time box per identificare la durata intera del progetto, la quale a sua volta è suddivisa in diversi sprint di circa due settimane ciascuno. Inoltre, sfruttiamo lo schema di priorità MoSCoW di RAD e il modello di pianificazione basato su KANBAN BOARD di SCRUM.

3. Organizzazione del progetto

3.1 Struttura del team

Il team di sviluppo segue le linee guida del team SWAT e le responsabilità sono divise equamente tra i vari membri.

3.2 Filosofia e motto del team

Il nome del nostro team è "Bug Busters" e la nostra visione è esemplificata dal motto "dove c'è un bug c'è una soluzione"; crediamo che tutti gli errori possano essere risolti con sforzo e duro lavoro.

4. Standard, linee guida e procedure

4.1 Convenzioni di codifica

Seguiamo le convenzioni Oracle circa la programmazione Java, reperibili al link:
https://www.oracle.com/technetwork/java/codeconventions-150003.pdf

5. Attività di gestione

5.1 Monitoraggio dello stato di avanzamento

Per monitorare lo stato di avanzamento del progetto la comunicazione è costituita dal Daily SCRUM, che nel nostro caso verrà effettuato con cadenza settimanale.

5.2 Uso del sistema MoSCoW e GitHub

Lo sviluppo del progetto seguirà un andamento prioritario dettato dallo schema MoSCoW, verificabile tramite le operazioni di commit rintracciabili su GitHub.

6. Rischi

6.1 Aspetti critici della GUI

Uno degli aspetti più rischiosi è la GUI, in quanto il team non possiede né competenze pregresse né strumenti necessari allo sviluppo di grafica di qualità. Pertanto, tale aspetto potrebbe risultare poco accattivante all'utente.

6.2 Limitazioni di tempo e complessità

Un ulteriore rischio è la brevà della time box rispetto alla complessità del progetto; per questo motivo qualche funzionalità potrebbe essere sacrificata.

6.3 Funzionalità non realizzabili

Infine, la funzionalità che prevede il conteggio delle ore di studio solamente se l'utente si trova all'interno dell'applicazione potrebbe non essere realizzabile con gli strumenti a disposizione del team e, dunque, essere sostituita da una funzione più semplice.

7. Personale

7.1 Ruoli e responsabilità

Durante tutta la durata del progetto saranno impiegati quattro sviluppatori nonché i responsabili del progetto, ognuno con competenze diverse come richiesto dal paradigma del team SWAT.

8. Metodi e tecniche

8.1 Ingegneria dei requisiti

Adottiamo tre diversi tecniche di elicitazione:

1. Analisi basata sullo scenario, tramite la quale ricaviamo i requisiti costruendo casi d'uso probabili
2. Derivazione da un sistema esistente; in questo caso l'ispirazione proviene dall'applicazione mobile "Forest App", dalla quale abbiamo tratto l'idea di base e i requisiti generali

3. Prototipazione, al fine di utilizzare prototipi per estrarre i requisiti in modo incrementale

8.2 Progettazione

Adottiamo lo stile architettonico MVC che prevede la separazione tra interfaccia utente e logica di programma. Tale decisione è stata presa per gestire al meglio la parte grafica dell'applicazione, rendendo quest'ultima indipendente dal resto.

8.3 Testing

Alla fine di ogni sprint verranno effettuate verifica e validazione del prodotto. In particolare, la verifica avverrà tramite test automatici mentre la validazione tramite test manuali.

9. Garanzia di qualità

9.1 Metodi per assicurare qualità e facilità d'uso

Per rispettare i requisiti dichiarati al punto 1 (ovvero sviluppare un prodotto intuitivo e facilmente testabile), facciamo affidamento a test manuali e automatici: con i primi cerchiamo di garantire la facilità d'uso del software; i secondi serviranno a controllare la correttezza del codice.

10. Pacchetti di lavoro

10.1 Blocchi principali del progetto

In accordo con il punto 7 e secondo la filosofia del team SWAT, ciascun membro ha gli stessi compiti e responsabilità circa la stesura del codice. Pertanto, non vi è un'esplicita suddivisione dei compiti. Di conseguenza, i membri del team collaborano in contemporanea sugli stessi pacchetti di lavoro.

Abbiamo individuato principalmente tre blocchi:

1. Logica di login e interfaccia
2. Logica dell'applicazione e interfaccia
3. Database
4. Networking

11. Risorse

11.1 Hardware e software utilizzati

Per il nostro progetto utilizzeremo quattro computer, un server GitHub remoto per la condivisione del codice e vari tool di supporto per lo sviluppo, ad esempio la gestione delle dipendenze con Maven, la fase di test con JUnit e l'analisi statica con SonarLint.

12. Budget e programma

12.1 Pianificazione temporale

Il modello di processo è agile, pertanto la timeline non è definita a priori; tuttavia, in linea generale, seguirà il seguente programma: ingegneria dei requisiti (dal 28 novembre al 15 dicembre), progettazione (dal 16 dicembre al 31 dicembre), implementazione e testing (dal 1° gennaio fino a scadenza).

13. Cambiamenti

13.1 Gestione delle modifiche

Utilizzando un modello di processo agile, le modifiche non prevedono una documentazione formale bensì l'apertura di apposite issues sul repository del progetto. Se uno sviluppatore realizza funzionalità in un branch secondario, deve presentare al team una Change Request che, se approvata, comporta l'integrazione del branch secondario con il branch Main.

14. Consegna

14.1 Data e modalità di consegna

La consegna del progetto è prevista entro e non oltre il 19 gennaio 2025. La consegna avverrà tramite condivisione del repository su GitHub.