Estimate evaluation

team 8

Stabilità dei requisiti

I requisiti individuati a inizio a inizio progetto non hanno risentito di modifiche importanti. Essendo stata applicata una metodologia agile, è stato comunque possibile ridiscutere e rivedere le implementazioni delle user stories con l'obiettivo di migliorare il prodotto a seguito dell'esperienza acquisita e al continuo confronto con il cliente.
A metà del progetto sono stato discusse e create nuove user stories dovute a nuovi requisiti (o meglio, requisiti non ben identificati in fase di progettazione) da parte del cliente. Essendo stati espressi durante uno sprint, si è aspettata la conclusione dello sprint prima di discuterne all'interno del team.
Dopo essersi confrontati con il cliente, rispetto alla fase di progettazione è stato modificato il calcolo della complessità in quanto non soddisfacente, scegliendo un algoritmo più collaudato e stabile.

Metriche applicate

Le LOC inserite a seguito rappresentano solamente le linee di codice prodotte manualmente dai membri del team, senza contare le linee derivanti da altri file generati automaticamente da processi di Django, documentazione e altro.
Questo permette di avere una migliore idea sul lavoro effettuato realmente dal team.

Sprint 1

LOC

Al termine dello sprint 1 le linee di codice sono state: 1.012.

Story Points

Al termine dello sprint 1 i punti per storia realizzati sono stati: 19.

Sprint 2

LOC

Al termine dello sprint 2 le linee di codice sono state: **4.541** (3529 nuove) .

Story Points

Al termine dello sprint 2 i punti per storia realizzati sono stati: 20.

Sprint 3

LOC

Al termine dello sprint 3 le linee di codice sono state: 6.716 (2.175 nuove).

Story Points

Al termine dello sprint 3 i punti per storia realizzati sono stati: 32.

Sprint 4

LOC

Al termine dello sprint 4 le linee di codice sono state: **7.823** (1.107 nuove).

Story Points

Al termine dello sprint 4 i punti per storia realizzati sono stati: 19.5.

Riflessioni

Questi numeri fanno capire che la maggior parte della struttura del prodotto è stata realizzata al termine del primo sprint ufficiale (sprint 1 e sprint 2) e questo rispecchia anche le linee guida della metodologia agile, che richiede il prima possibile il rilascio di un MVP funzionante e testabile dal cliente. Nel secondo sprint ufficiale (sprint 3 e sprint 4) invece, c'è stata una maggior attenzione al refactoring ed alla realizzazione delle nuove richieste da parte del cliente, che in rapporto hanno prodotto un minor quantitativo di linee di codice.

Per quanto riguarda i punti per storia, si evince che il team ha da subito compreso la sua effettiva velocità e questo ha permesso di distribuire il carico di lavoro in maniera più o meno corretta e non sovrastimata e anche questo dato fa capire che il team ha compreso e messo in pratica al meglio ciò che la metodologia agile insegna.

NOTA: C'è una grande differenza nei punti dello sprint 3 ma questo probabilmente è dovuto ad una sovrastima del punteggio di alcune user stories che hanno portato ad

una conclusione anticipata dei task dello sprint (inizialmente il punteggio totale era 22) e ad una inclusione di nuove user stories che hanno fatto *impennare* il punteggio a 32.

Riteniamo queste metriche abbastanza attendibili e utili (soprattutto le story point), in quanto permettono di dare un'idea al team della propria velocità e dello stato di avanzamento del progetto; le stime iniziali risultano praticamente rispettate.

Per avere un'idea generale sulla grandezza del progetto, contando anche i file non creati esclusivamente dal team, le ***LOC*** dell'intero progetto sono **11804**.

Analisi dei rischi

Il team ha prodotto un'analisi dei rischi iniziale che si è rivelata abbastanza corretta e molto efficace nel corso del progetto. Sono però comparse nuove criticità a cui non si era pensato inizialmente, mentre alcuni impatti/difficoltà sono variati rispetto alla stima iniziale.
Il team ha dunque evidenziato in rosso i cambiamenti rispetto all'analisi dei rischi prodotta in fase di progettazione.

Nome criticità	Impatto (1 - 10)	Difficoltà (1 - 10)
Analisi frequenza parole e complessità testo	**10** - Core dell'applicativo, senza esso l'intero applicativo fallisce.	**7** - Lavorato poco sull'analisi di testi, poche conoscenze (conoscenza degli indici di testo) (4) - Questo tipo di operazioni sono ben integrate in Python ed è bastato documentarsi su di esse. Il calcolo della complessità è stato affidato ad un algoritmo noto (ARI).
Salvare nel db la frequenza delle parole	**9** - Molto importante, permette di salvare in modo permanente i dati	**5** - Conoscenze pregresse sull'uso di database, difficoltà nel capire come ideare bene le tabelle e le relazioni tra loro
Utilizzare e integrare API traduttore per tradurre una parola	**6** - Importante (richiesto dal cliente) ma non essenziale ai fini dello scopo dell'applicazione	**6** - Nessuna conoscenza su esso, ma molta documentazione online (2) - Facile integrazione dell'API nel progetto, con funzioni semplici per la

		traduzione e rilevamento lingua.
Creazione interfaccia per cercare testo da confrontare	**7** - Relativamente importante	**5** - Conoscenze pregresse nello sviluppo web, ma complicazioni date da interrogazioni dinamiche al database (7) - Si è rivelato difficile integrare le varie interfacce per il confronto di testi, con il bisogno di un buono studio preliminare sulle tecniche da poter usare.
Ricerca parole chiave	**8** - Abbastanza importante. Ad essa sono collegate buona parte delle funzionalità dell'applicazione.	**3** - Conoscenze pregresse sull'interrogazione di database
Gestione raccolte di testi	**1** - Marginale. Non richiesta esplicitamente.	**6** - Chiesto studio e lavoro front-end approfondito
Calcolare complessità generica di un utente	**2** - Poco importante. Non richiesta esplicitamente.	**4** - Necessità di capire come calcolare la complessità generica delle analisi di un utente dal database. (2) - Più semplice del previsto, complessità gestita in maniera più facile ed immediata di quanto si fosse pensato inizialmente.
Salvataggio temporaneo testi inseriti tramite cookie per rimediare a chiusura erronea del browser prima del salvataggio	**2** - Poco importante. Non richiesta esplicitamente.	**9** - Mai lavorato con cookies, probabile documentazione su Internet ma argomento mai trattato.
Uso di funzioni Ajax per salvataggio testi, traduzione	**7** - Decisamente importante. Fondamentale per la traduzione dinamica (senza caricamento della pagina) e il salvataggio non obbligatorio dopo l'analisi.	**8** - E' stato necessario capire a fondo i meccanismi di richiesta Ajax per poterli sfruttare a nostro vantaggio e in diversi casi d'uso (dal ritornare una singola stringa ad un'intera lista di

		testi già formattata adeguatamente)
Inserimento di grafici nelle pagine dell'app	**7** - Importante in quanto richiesta espressamente dal cliente.	**5** - Buona documentazione online, facilità di implementazione ma difficoltà di integrazione di più grafici in una sola pagina.
Hosting online dell'applicazione	**10** - Cruciale. Senza di esso l'applicazione non sarebbe stata fruibile dagli utenti.	**6** - E' stato necessario un notevole sforzo per comprendere appieno i criteri di funzionamento del server su cui *installare* l'applicativo.
Uso di CSS e Bootstrap per rendere il sito visualizzabile da smartphone	**2** - Poco importante al fine del progetto che non richiede esplicitamente un sito ottimizzato per smartphone.	**5** - Relativamente difficile. Non è stato del tutto compreso come rendere ogni componente totalmente responsive e ottimizzata per i dispositivi di piccole dimensioni.
Django per sviluppo app e testing	**10** - Framework usato dal team per lo sviluppo dell'applicativo	**4** - Buona conoscenza del framework da parte del team ma poca esperienza sia con esso che con la parte di testing, necessario studio preliminare su alcune sue funzionalità e testing.
*Utilizzo di un software di versione di controllo (**git**)*	**10** - Permette lo sviluppo dello stesso software da più persone contemporaneamente	**3** - Molto complicato nei primi tempi, ma una volta apprese le basi del funzionamento e della gestione dei conflitti non sono state evidenziate ulteriori complicazioni.

Debito tecnico

Il team ha cercato di concludere ogni sprint risolvendo il maggior numero di problemi possibili (con anche ampio uso di refactoring), per evitare accumulo di debito tecnico.
Nonostante ciò, sono state comunque individuate alcune problematiche:

- servizio di **cloud hosting** → soluzione gratuita ed immediata, ma sicuramente con evidenti limiti tecnici come la memoria disponibile, il tempo di CPU a disposizione e la mancata indicizzazione nel motore di ricerca di google (si veda il documento "**Post-Retrospective**", sezione "**Sprechi identificati**"). Questo debito sarebbe risolvibile utilizzando ad esempio un server dell'azienda o pagando un canone mensile ad un servizio di cloud hosting a pagamento.
- Uso di **SQLite** come database → Soluzione di facile implementazione e uso che mette comunque a disposizione una buona quantità di memoria però non permette una grande concorrenza tra utenti, cosa che potrebbe creare problemi nel momento in cui l'applicativo dovesse essere usato da molti utenti contemporaneamente.
  La soluzione sarebbe l'uso di un database più potente (ad esempio PostgreSQL).
- **Dark mode** → In fase di caricamento della pagina si vede un brevissimo flash bianco, dovuto all'applicazione dinamica del tema scuro.
- Per l'aggiunta di una **nuova categoria** da admin bisogna riavviare l'applicativo per renderla visibile nella barra di ricerca.
- **Smartphone** → Il sito è responsive ed è possibile visitarlo da smartphone. Tuttavia, la visualizzazione da smartphone non è ottimizzata e presenta alcuni difetti di layout o difficoltà di utilizzo. Sarebbe possibile risolverle documentandosi maggiormente sulle modalità di gestione del layout responsive e implementando alcune modifiche agli stili dell'applicazione.
- **Registrazione** → Quando un utente crea un account non viene mandata una email di verifica. Si potrebbe risolvere gestendo un invio automatico di una email in fase di registrazione.

Al momento, ogni debito tecnico è relativamente poco impattante sul progetto (l'applicativo funziona comunque egregiamente); in una visione più a lungo termine, sicuramente i più impattanti sono l'uso di SQLite come database e l'hosting cloud (attualmente su pythonanywhere.com).

Alla conclusione del primo sprint, il team aveva accumulato un debito tecnico non indifferente dovuto ad alcuni problemi tra cui la creazione statica delle categorie nella barra di ricerca; ciò portava a molti bug all'interno dell'applicativo.
Il debito tecnico accumulato durante il primo sprint è stato però risolto durante il secondo sprint.

Preventivo, consuntivo e costo

Progettazione

Il team ha stimato all'inizio della progettazione di lavorare 2 ore circa al giorno per ogni membro, con un prezzo di 45 € lordi all'ora per ciascun componente.

Preventivo

Descrizione	Valore
Confronti con il cliente	Quotidiani
Periodo di lavoro richiesto	10 giorni lavorativi
Costo preventivato della progettazione	3.510,00 €
IVA (22%)	990,00 €
Totale	4.500,00 €

Il consuntivo non è necessario in quanto il team si è attenuto al preventivo.

Codifica

Il team ha stimato all'inizio della codifica di lavorare 3 ore circa al giorno per ogni membro, con un prezzo di 40 € lordi all'ora per ciascun componente.

Preventivo 1 - a prezzo fisso del progetto (non necessario consuntivo in quanto pagato interamente)

Descrizione	Valore
Rilascio di versioni testabili dal cliente	Settimanale
Periodo di lavoro richiesto	31 giorni lavorativi
Costo preventivato del progetto	14.508,00 €
IVA (22%)	4.092,00 €
Totale	18.600,00 €

Preventivo 2 - pagamento a rilascio di MVP

Descrizione	Valore
Rilascio di versioni testabili dal cliente	Settimanale
MVP stimati	5
Costo per MVP	2.340,00 €
IVA (22%)	660,00 €
Totale	3.000,00 €

Costi aggiuntivi possibili:

Voce costo	Valore
Server + Dominio (fonte: Aruba)	25.99 € + IVA / anno
Dominio web (fonte: Aruba)	9.99 € + IVA / anno
Spese vive (elettricità, benzina...)	200 € + IVA / mese

Costi di futura assistenza:

Voce costo	Valore
Programmatore	35 € / ora lordi

Consuntivo (Preventivo 2) a MVP

Descrizione	Valore
MVP prodotti	4
Costo per MVP	2.340,00 €
IVA (22%)	660,00 €
Totale per MVP	3.000,00 €
Totale	12.000 €

Suddivisione interna del ricavato

Totale ore (340)	Programmatore	Preventivo 1 (18.600 €)	Preventivo 2 (12.000 €)
85 - 25%	Davide Carletti	4.650,00 €	3.000,00 €
83 - 24,42%	Gabriele Mattioli	4.542,12 €	2.930,40 €
70 - 20,59%	Marco Incerti	3.829,74 €	2.470,80 €
55 - 16,18%	Filippo Rinaldi	3.009,48 €	1.941,60 €
47 - 13,81%	Filippo Fontana	2.568,66 €	1.657,20 €