UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Esame di Ingegneria, Gestione ed Evoluzione del Software

PROGETTO CODESMILE

Master Test Plan

TEAM MEMBERS

Dario Mazza - 0522501553

Nicolò Delogu - 0522501556

REPOSITORY

`https://github.com/xDaryamo/smell_ai`

VERSIONE

1.0

Anno Accademico 2024-2025

1 Introduzione

Questo documento definisce la strategia generale, l'approccio e le attività pianificate per il processo di testing del progetto CodeSmile. Il piano copre i diversi livelli di testing, inclusi test di unità, integrazione e di sistema.

1.1 Premessa relativa alla Change Request 1

La decisione di concentrare inizialmente gli sforzi sul testing solo a livello di sistema prima della transizione a codice orientato agli oggetti è stata guidata dalla necessità di valutare il comportamento globale del sistema senza introdurre complessità aggiuntive. Poiché il progetto attuale è privo di una test suite e basato su un'architettura procedurale, il testing di sistema consente di identificare eventuali criticità funzionali e di stabilire una baseline per il comportamento atteso. Dopo la ristrutturazione, l'applicazione del testing di unità, di integrazione e di sistema, con confronto rispetto al precedente, garantirà una verifica più completa, sia a livello di singole componenti che di funzionalità complessive, favorendo così un approccio graduale e ben strutturato al miglioramento del tool.

2 Strategie di Testing

2.1 Testing di Unità

L'approccio applicato in questa attività sarà quello del white-box, facendo uso della branch coverage come indicatore. Lo scopo è quello di coprire il maggior numero di branch, selezionando input opportuni con cui possano essere raggiunti.

2.2 Testing di Integrazione

La seguente attività fa riferimento a un insieme di test che hanno l'obbiettivo di combinare le singole componenti, al fine di verificare la loro corretta interazione.

2.3 Testing di Sistema

Per questa attività verrà adottata la tecnica di black box, con un focus particolare sul category partitioning. I dati saranno organizzati in classi di test e, per ottimizzare il processo, si utilizzerà la tecnica WECT, che permetterà di ridurre il numero di combinazioni da considerare, escludendo quelle non valide.

2.4 Testing E2E

Per il modulo WebApp, è previsto un testing end-to-end (E2E) finalizzato a verificare il corretto funzionamento dei principali flussi utente. Questo tipo di test garantirà che tutte le componenti interagiscano correttamente, simulando scenari reali di utilizzo. Per l'implementazione del testing E2E, verranno utilizzati strumenti come Cypress, con l'obiettivo di automatizzare e ottimizzare il processo di verifica.

3 Organizzazione delle funzionalità testate

3.1 Componenti incluse dal Testing

Tutte le componenti della nuova architettura e della WebApp saranno oggetto di testing di unità e integrazione. Inoltre l'applicazione web sarà oggetto di testing E2E.

3.2 Componenti escluse dal Testing

Verranno escluse dal testing tutte le funzionalità che riguardano il modello di intelligenza artificiale e la creazione del dataset.

3.3 Coverage

Tenendo conto che la copertura attuale del codice ammonta al 0%, riteniamo opportuno raggiungere una percetuale di branch coverage maggiore o uguale al 70%, escludendo dal conteggio i moduli relativi al modello e alla creazione del dataset (come gia specificato nelle sezioni precedenti).