



Università degli Studi di Salerno

Corso di Ingegneria del Software: Tecniche Avanzate

Code Smile Pre-Modification Test Incident Report Versione 1.0

Team:

Antonio Caiazza
Emanuele Iovane
Salvatore Di Martino



Progetto: Code Smile	Versione: 1.0
Documento: Pre-Modification Test Incident Report	Data: 25/11/2025

Partecipanti:

Nome	Matricola
Antonio Caiazzo	NF22500205
Salvatore Di Martino	NF22500114
Emanuele Iovane	NF22500162

Scritto da:	Antonio Caiazzo, Salvatore Di Martino, Emanuele Iovane
--------------------	--

Progetto: Code Smile	Versione: 1.0
Documento: Pre-Modification Test Incident Report	Data: 25/11/2025

Indice

1.	Introduzione.....	4
2.	Incident Report	4
2.1	Test Incident 1 – TI1	4
2.1.1	Informazioni generali.....	4
2.1.2	Descrizione del Test Incident.....	4
2.1.3	Risoluzione.....	5

Progetto: Code Smile	Versione: 1.0
Documento: Pre-Modification Test Incident Report	Data: 25/11/2025

1. Introduzione

Nel presente report vengono analizzate le criticità emerse durante l'esecuzione dei casi di test precedenti alle modifiche previste per CodeSmile, includendo non solo i fault del sistema, ma anche eventuali errori nei test stessi, indipendentemente dal loro livello: unità, integrazione o sistema.

Il documento ha l'obiettivo di fornire una descrizione accurata delle anomalie rilevate, intese come discrepanze tra il comportamento atteso (oracolo) e quello effettivamente osservato, o come difformità riconducibili a test non corretti. Per ciascun problema viene proposta un'analisi dettagliata con relativi possibili interventi correttivi, con l'intento di migliorare il sistema e rendere più efficace il processo di sviluppo, contribuendo così a garantire una qualità più elevata del codice analizzato.

2. Incident Report

2.1 Test Incident 1 – TII

2.1.1 Informazioni generali

- **Nome:** TII
- **Ambiente:** Ambiente di Test
- **Tipologia di test:** Test di unità
- **File:** *test/unit_testing/components/test_project_analyzer.py*
- **Metodo:** *test_analyze_project_empty_directory*

2.1.2 Descrizione del Test Incident

- **Problema:** Il test *test_analyze_project_empty_directory* falliva perché il mock applicato a *FileUtils.get_python_files()* non aveva effetto sul modulo *components.project_analyzer*, che manteneva il riferimento originale alla classe importata. Di conseguenza, il metodo *analyze_project()* rilevava l'assenza di file Python e sollevava una *ValueError* inattesa dal test.
- **Riproduzione del Test:** Eseguire i test di unità del modulo *components* lanciando:

```
pytest test/unit_testing/components/test_project_analyzer.py::test_analyze_project_empty_directory -v
```

Il test tenta di simulare una directory vuota, ma il mocking non va a buon fine e viene sollevata un'eccezione prima dell'assert finale.
- **Comportamento attuale:** Durante l'esecuzione, *analyze_project()* chiama *FileUtils.get_python_files()* reale (non mockato) e, trovando zero file, solleva una *ValueError* con il messaggio "*contains no Python files*", interrompendo il test e provocandone il fallimento.

Progetto: Code Smile	Versione: 1.0
Documento: Pre-Modification Test Incident Report	Data: 25/11/2025

2.1.3 Risoluzione

- **Stato:** Risolto. Il test ora passa correttamente e verifica l'eccezione attesa.
- **Cause del problema:** Il mock veniva applicato al modulo *utils.file_utils*, ma *ProjectAnalyzer* utilizza una copia della classe *FileUtils* importata in *components.project_analyzer*. Il riferimento interno non veniva quindi sostituito. Inoltre, il test aspettava un valore di ritorno pari a 0, mentre il comportamento corretto del metodo è sollevare una *ValueError*.
- **Componenti coinvolte:** *ProjectAnalyzer* (components), *FileUtils* (utils), test *test_analyze_project_empty_directory*.
- **Pianificazione:** Identificata la causa, è stato aggiornato il test per effettuare il mocking nel modulo corretto e per verificare l'eccezione prevista. Successivamente sono stati rieseguiti i test del modulo components e l'intera suite di unit test per confermare l'assenza di regressioni.
- **Soluzione:** Creata una classe mock *MockFileUtils* che restituisce una lista vuota e sostituita direttamente *FileUtils* all'interno del modulo *components.project_analyzer*. Il test è stato aggiornato per utilizzare *pytest.raises(ValueError)* e per ripristinare il riferimento originale a *FileUtils* tramite blocco *finally*.