

UC1: EseguiAnalisiCodice

- a) operazione: `avviaAnalisiCodice(projectPath: String)`
- b) riferimenti: `EseguiAnalisiCodice`
- c) pre-condizioni:
 - Il percorso `projectPath` identifica una directory esistente
 - il progetto contiene almeno un file sorgente
 - esiste almeno un linguaggio supportato dal sistema
 - il set di regole di analisi è disponibile
- d) post-condizioni:
 - viene creata un'istanza `Analisi` associata al progetto
 - viene creata o recuperata una o più istanze `FileSorgente` associate ai file presenti nel progetto
 - viene creata una o più istanze `FileAnalizzato` associate ai `FileSorgente`
 - viene costruita per ogni file analizzabile un'istanza `AST`
 - per ogni violazione rilevata viene creata un'istanza `Issue` associata a `FileSorgente` e all'istanza `Rule` (regola violata)
 - l'istanza `Analisi` mantiene l'associazione `Risultati` con la collezione di tutte le `Issue` prodotte

UC2: GeneraReportQualità

- a) operazione: `generaReportQualità(idAnalisi: String, formato: ReportFormat)`
- b) riferimenti: `GeneraReportQualità`
- c) pre-condizioni:
 - esiste un'istanza `Analisi` identificata da `idAnalisi`
 - l'analisi associata è completata
- d) post-condizioni:
 - viene creata un'istanza `Report`
 - il `Report` contiene una classificazione delle `Issue` per categoria e severità
 - viene calcolato e assegnato un valore all'attributo `ScoreQualità` del `Report`
 - il `Report` viene esportato nel formato specificato (HTML, JSON o PDF)

UC3: VisualizzaRevisione

- a) operazione: `getIssues(idAnalisi: String, filePath: String): List<Issue>`
- b) riferimenti: `VisualizzaRevisione`
- c) pre-condizioni:
 - esiste un'istanza `Analisi` identificata da `idAnalisi`
 - il file identificato da `filePath` appartiene al progetto analizzato
- d) post-condizioni:
 - viene restituito un insieme di istanze di tipo `Issue`
 - ogni `Issue` è associata a una `Analisi` e al `FileSorgente` identificato da `filePath`
 - i risultati dell'Analisi non vengono modificati