Kvalitetssikring og Test: PDF downloader

Opgavebeskrivelse

Som softwareudvikler i et konsulentfirma er du blevet bedt om at kvalitetssikre en eksisterende løsning til download af PDF-rapporter. Løsningen er blevet udviklet af en tidligere konsulent, men kunden har oplevet ustabilitet og performanceproblemer og vil derfor gerne have sikret dens funktionalitet gennem en grundig teststrategi.

Kunden har specifikt anmodet om:

- Enhedstest af kritiske funktioner i PDF-downloaderen.
- Integrationstest for at sikre samspillet mellem Excel-dataindlæsning, URL-håndtering og filhåndtering.
- Robust fejlhåndteringstest, hvor systemet valideres mod forskellige fejlscenarier (f.eks. ugyldige links, timeout, netværksfejl).
- En testrapport med identificerede fejl og forbedringsforslag baseret på din analyse af koden.

Afleveringsformat

Opgavens løsning skal præsenteres foran dine medkursister på fredag. Her skal du demonstrere hvordan du har løst opgavens dele og implementeret testing og kvalitetssikring. Det er som udgangspunkt valgfrit hvad I lægger vægt på i præsentationen, I har sammenlagt 15 min til at præsentere og sparre med de andre deltagere om jeres løsning. Det forventes dog at I kommer omkring projektets kode, testdækning, testrapport og en README-fil med en beskrivelse af hvordan man kører test suiten.

OBS: For ikke at overbelaste Specialisternes netværk - Hvis I beslutter jer for at teste PDF-downloaderens concurrency, så sørg for at teste med maks 10 PDF'er ad gangen når I er på

kontoret.

Afleveringskrav:

- Enhedstest (Unit Tests) af kritiske funktioner i PDF-downloaderen fra sidste uge
- Integrationstest mellem PDF-downloaderen og Excel/CSV-filhåndtering
- Testrapport, der viser testdækningen, forklarer hvordan fejl og mangler blev håndteret og som reflekterer over kodens kvalitet samt forbedringsforslag.
- README-fil med beskrivelse af hvordan man kører test suiten
    - Til dem af jer der kommer langt kan i påbegynde udviklingen af en CI/CD-pipeline til automatisering af test

---

Foreslået fremgangsmåde:

🟢 Del 1: Implementering af enhedstest (Unit Tests)

- Skriv unit tests for hver af PDF-downloaderens kernefunktioner.
- Anvend mocking til at isolere testene fra andre moduler og eksterne afhængigheder som netværksforbindelser og filsystemet.
- Test fejlscenarier, såsom ugyldige URL'er og netværksfejl.
- Brug testframeworks såsom **JUnit (Java), Mockito (Java), xUnit (C#), Moq (.Net) PyUnit(Python), Mock (Python),** eller lignende afhængigt af sproget.

🟢 Del 2: Integrationstest

- Test interaktionen mellem PDF-downloaderen og Excel/CSV-data.
- Sikre, at programmet korrekt læser og behandler URL'er fra inputfilen.
- Verificér, at downloadede filer bliver korrekt registreret med status i outputfilen.
- Undgå brug af mocks – testene skal interagere med faktiske filer

🥏 Del 3: Testrapport og dokumentation

- Dokumentér testresultater, herunder fejl, der er identificeret og rettet i en test rapport.
- Evaluér kodekvalitet, herunder:
- Vurder om koden følger ensartede navngivningskonventioner, og at klasser og metoder er logisk organiseret.
- Vurder om koden er let at forstå og vedligeholde. Tjek, om komplekse dele er forsynet med relevante kommentarer.
- Vurder om fejl håndteres korrekt med passende fejlmeddelelser og logging, så problemer kan identificeres og rettes effektivt.
- Evaluér testdækning og potentielle forbedringer.

I test rapporten, kommentér på kodeområder hvor testene har afsløret fejl eller ineffektiviteter. Overvej også at indskrive problemet i koden som en # TODO i koden.

🥏 Del 4 (Valgfri): CI/CD-pipeline

- Hvis du når at færdiggøre testene, kan du implementere en simpel CI/CD-pipeline.
- Brug GitHub Actions, Jenkins eller et lignende værktøj til at køre testene automatisk ved hver commit.
- Overvej at følge et Test-Driven-Development (TDD) arbejdsflow ved retning af bugs- eller forbedring af kodekvaliteten.

Pensum og ressourcer
