Kravspecifikation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Navn | Marcus | | | |
| Opgave | **PDF downloader** | | | |
| Krav/prioriteter: *Her kan du oplyse de vigtigste elementer i dit software* | * **Download af PDFer** * **Fallback på download af PDFer** * **Effektivere** * **Registrere download status** | | | |
| Kodesprog | **Python** | | | |
| Evt. frameworks |  | | | |
| Estimeret tid | **4 dage** | | | |
|  | | | | |
| Featurelist *Her kan du tilføje hvilket features som skal kodes til dit software* | **Duplicate Check Before Download**   * Programmet sikrer, at en PDF ikke downloades igen, hvis den allerede eksisterer lokalt.   **Excel Integration**   * Automatisk indlæsning og parsing af Excel-ark.   **API-baseret PDF Download**   * Understøtter download af PDF-filer via API-kald, inkl. håndtering af dynamisk URL’er og parametre.   **Fallback Mechanism for Unsuccessful Downloads**   * Implementerer en alternativ downloadmetode, hvis det primære kald fejler backup-URL.   **Robust Fejlhåndtering**   * Error -og exceptionhandling. Samt logging af success eller ej, når der bliver downloaded.   **Multithreading Support**   * Asynkron/Threading behandling af flere downloadopgaver for at forbedre hastighed og ressourceudnyttelse.   **SOLID Principles Architecture**   * Koden er struktureret efter SOLID-principperne for høj vedligeholdelsesvenlighed og genbrug.   **Indbygget Dokumentation og Readability**   * Kodekommentarer og struktureret dokumentation (docstrings, README).   **PDF-validering**   * Tjekke om der er indhold i PDF’en eller om den er korrupt. | | | |
| Work breakdown *Her kan du bryde dit software ned i arbejdsopgaver og estimere tiden det vil tage at bygge hvert et element* | Arbejdsopgave | Start dato | Forventet slutdato | Tidsforbrug |
| **Duplicate Check Before Download**  * Opret destinationsmappe og struktur. * Tjek om filnavnet allerede findes * Filter hvis de allerede existere   **Excel Integration**   * Indlæs Excel-ark med pandas.read\_excel() * Indlæs relevante kolonner, såsom BRnum, Pdf\_URL(AL), Report Html Address (AM) * Valider data såsom tomme felter.   **API-baseret PDF Download**   * Brug requests til synkrone API-kald * Dynamisk konstruerede URLs baseret på data fra Excel * Brug response til at generare PDF’en   **Fallback Mechanism for Unsuccessful Downloads**   * Definér backup-URL (fra Excel) * try/except omkring primær download – kald fallback ved fejl * Brug statuskoder (response.status\_code) til at afgøre succes * Log begge forsøg   **Robust Fejlhåndtering**   * Brug try/except omkring netværkskald, filskriveoperationer osv. * Catch og håndter diverse exceptions.   **Multithreading Support**   * Anvende concurrent.futures.ThreadPoolExecutor * Anvendes til hurtigere download. * Thread-safe logning og fejlopsamling   **Generere Excel rapport**   * Indhold af BRnummer, om den kan downloades eller ej (download status)   **Indbygget Dokumentation og Readability**   * Tilføj docstrings. * Evt. brug type hints (def fetch(url: str) -> bytes) * Dokumentér projektets struktur i en README.md * Overvej at bruge pydoc.   NOTE:  Timer går ikke op, da jeg har sat buffer tid. Da jeg er i tvivl om tiden giver mening og om jeg overser noget.  Den første halvdel af opgaverne er genbrug med en smule justering. | 06-10-2025  06-10-2025  06-10-2025  06-10-2025  06-10-2025  08-10-2025  09-10-2025  06-10-2025 | 06-10-2025  06-10-2025  06-10-2025  06-10-2025  10-10-2025  08-10-2025  09-10-2025  10-10-2025 | 0.5 timer  1 time  1 time  2 timer  Løbende      3 timer    1 time  Løbende. 2-3 timer |