Ranjandeep	A1909181
Sourav Debnath	A1900755

Test Case Report

This report outlines the test cases designed to verify the recent modifications made to the `ImportHandler` class in our project. The modifications include the addition of a new method, `relativize`, and enhancements to the existing import functionality. These test cases ensure that the new features work correctly and integrate smoothly with the existing system, thereby maintaining the overall stability and performance of the application.

Test Case 1: Relativize Method

Description: This test case checks if the `relativize` method returns the correct relative path.

Input:

- An absolute path, e.g., `/home/user/documents/file.pdf`.
- A base directory, e.g., `/home/user/documents`.

Expected Output: The relative path should be `file.pdf`.

Test Case 2: Constructor Initialization

Description: This test case checks the `ImportHandler` constructor correctly initializes the `filePreferences` field.

Input:

- A mock `PreferencesService` that returns a `FilePreferences` object when `getFilePreferences()` is called.

Expected Output: The `filePreferences` field in the `ImportHandler` instance should exactly match the `FilePreferences` object from the mock `PreferencesService`.

Test Case 3: Import Files in Background - Setting Relative Path

Description: This test case checks if the `importFilesInBackground` method correctly sets the relative path for PDF entries.

Input:

- A list of PDF files.

Expected Output: Each entry in `entriesToAdd` should have its `FILE` field accurately set to the relative path of the corresponding file.

Test Case 4: Import Files in Background - PDF Content Import

Description: This test case checks the `importFilesInBackground` method correctly imports entries from a PDF file.

Input:

- A PDF file with embedded metadata.

Expected Output: Entries extracted from the PDF file should be added to `entriesToAdd`, and a success result should be added to the results list.

Test Case 5: Import Files in Background - No Metadata Found in PDF

Description: This test case checks if the `importFilesInBackground` method handles cases where no metadata is found in a PDF file.

Input:

- A PDF file without any metadata.

Expected Output: An empty entry with a file link should be created and added to `entriesToAdd`, and a corresponding failure result should be added to the results list.

Test Case 6: Generate Keys Method

Description: This test case checks the `generateKeys` method correctly generates citation keys for entries when the preference to generate new keys on import is enabled.

Input:

- A list of entries without citation keys.
- Preferences with `isGenerateNewKeyOnImport` set to `true`.

Expected Output: Each entry should have a newly generated citation key assigned.

Below table represents the summary of the above test cases

Test Case ID	**Description**	**Input**	**Expected Output**	**Actual Output**
Test Case 1	Relativize Method	Absolute path, Base directory	Correct relative path	Returned relative path "documents/file.pdf"
Test Case 2	Constructor Initialization	Mock PreferencesService	Correct filePreferences field	filePreferences field set to mock instance
Test Case 3	Import Files - Setting Relative Path	List of PDF files	Correct relative path for each entry	Relative path "docs/file.pdf" set in entry's FILE field
Test Case 4	Import Files - PDF Content Import	PDF file with metadata	Entries added to entriesToAdd	Entries with metadata added to entriesToAdd
Test Case 5	Import Files - No Metadata Found	PDF file without metadata	Empty entry with file link	Empty entry created with FILE field set to "docs/file.pdf"
Test Case 6	Generate Keys Method	List of entries without keys, Preferences with key generation enabled	New citation keys generated for entries	Citation keys "Smith2020", "Doe2021" generated and set in entries

Table 1 : Test case Summary

Conclusion

The test cases we've designed really focus on the critical changes we made in the ImportHandler class. We're paying close attention to the new relativize method, how the constructor initializes, and all the new import functions. These tests cover things like making sure file paths are handled correctly, that all the fields start off right, and that metadata imports correctly. All of this is important for making sure the whole system is stable and works correctly. These tests also make sure the new features we added work like they're supposed to and fit in smoothly with what we already had. That makes us sure the system is working well and won't let anyone down. By testing these things really well, we can be sure that the changes we made make the software better without causing new problems. Testing like this is really important to keep our software high-quality and make sure it does what people need, no matter what.