Feature - Unit testing of added content

- Programmer: Ronnie Hoover
- Date: 4/18/18

Problem Definition

New content added the wiki required unit tests to ensure continued functionality.

Requirement

Test features added by the `ArchivePage` and `RestorePage` objects, as well as from the added function of `SearchHistory`.

Ensure that any testing does not leave pages within the database.

User Stories

User story 1 - As a developer I would like to ensure that my teams code is properly being tested so that we have a better metric as to whether changes in the code cause any issues.

Design

User Story 1

Interface

- Test Classes: test_core.WikiTestCase, test_web.TestArchivePage, and test_web.TestRestorePage
- Action: Assert cases for specific functions.

Detailed Design

- Tests created are ran alongside prior standing tests.
- In `test_store` the local file is read and that is compared to the page_file attribute within the database.
- In `test_restore` the database receives two stores for a entry of the same name, causing it to create a new entry with a new commitID.

  - The second file is different than the original, so this emulates versions of a page.
  - The test then reads and stores the content of the first "original" file.
  - The file is restored to the first commit (optimally the same as the original) then asserts if the original and restored versions are the same.
- In `test_restoreDeleted` a file is stored to the database, then attempted to restore to a file that does not exist.
  - The test asserts that the new file exists.
- In `test_searchHistory` a dictionary of history pages is created from `searchHistory` and the length is asserted to be the same as the number of distinct pages in the database.

Implementation

- `test_store` stores a example markdown file to the database, then asserts that the content is accurate.
- `teat_restore` stores an example markdown file, stores an alternate version of the same file to emulate a new commit of the same page. Then attempts to restore the first commit and asserts the original content matches
- `test_restoreDeleted` stores an example markdown file, then restores it to a new file. The asserts that the file is present.
- `test_searchHistory` retrieves the current dictionary of changes, then asserts that the number of pages is equal to the number of distinct pages.