|  |  |
| --- | --- |
| **Project:** Image DNA | Caitlan Nichols |
| **Description:** Testing image DNA | Peer tester: Caitlan Nichols |
| **Date:** 06/09/2025 | Testing **Date:** 06/09/2025 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test1 | **Image's save in own project folder and database** | | | |
| **Test Objective:** upload to folder and database successfully from HTML | **Environment:** computer | **Developer: Caitlan N**  **Date tested:** 06/09/25 | **Peer tester: Caitlan Nichols**  **Date tested:** 06/09/25 | |
| **Steps** | **Action** | **Expected results:** | **Developer pass/fail** | **Tester pass/fail**  **+ Screenshot** |
| 1 | Run the testingpypart.bat | Flask will run on your computer | yes | Yes |
| 2 | Copying the HTTPs that is running on the batch script, put it in your browser of choice | Will open to a plain HTML page with upload image functions | yes | Yes |
| 3 | Click the “Choose file”, upload an image (a simple non-identifying one will do). Click “upload and process after”. | File will be uploaded. Upload and process button is functional | yes | Yes |
| 4 | Look through folder and the SQL file database to see if it has been stored | Both will be uploaded in folder and database. | yes | Yes |
| **Comments** | Fully functional. - No errors have been found. Noted, however, Flask will update individual when image is uploaded. See image here, | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test 2 | **Image's save in own project folder and database, Edited, few changes** | | | |
| **Test Objective:** upload an EDITED image to folder and database successfully from HTML | **Environment:** computer | **Developer: Caitlan N**  **Date tested:** 06/09/25 | **Peer tester: Caitlan Nichols**  **Date tested:** 06/09/25 | |
| **Step** | **Action** | **Expected results:** | **Developer pass/fail** | **Tester pass/fail**  **+ Screenshot** |
| 1 | Run the testingpypart.bat  (or keep the prior session open by clicking enter on your web address bar) | Flask will run on your computer | yes | Yes |
| 2 | Copying the HTTPs that is running on the batch script, put it in your browser of choice | Will open to a plain HTML page with upload image functions | yes | Yes |
| 3 | Click the “Choose file”, upload an image you used prior that you edited (draw a few lines in the upper corner in MS paint or software of your choice). | File will be uploaded. Upload and process button is functional | yes | Yes |
| 4 | Look through folder and the SQL file database to see if it has been stored | Both will be uploaded in folder and database. | yes | Yes |
| 5 | Compare hashes, the image is broken into 16 parts. Take a guess as to where you drew your line(s). If done too many you may have changed the “finger print” significantly... | Slightly different hash values were made on edited parts while non edited remain the same | yes | Yes |
| **Comments** | Fully functional. - No errors have been found. | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test 3 | **Comparing hashes for saturation, hue, and resizing.** | | | |
| **Test Objective:** upload an EDITED image to ensure hashing works | **Environment:** computer | **Developer: Caitlan N**  **Date tested:** 06/09/25 | **Peer tester: Caitlan Nichols**  **Date tested:** 06/09/25 | |
| **Step** | **Action** | **Expected results:** | **Developer pass/fail** | **Tester pass/fail**  **+ Screenshot** |
| 1 | Using steps 1,2,3,4 from the prior tests, we will be editing images differently to see what changes will be needed. | Prior steps work | yes | Yes |
| 2 | Use two edited hue images that affects the entire image | Greyscale mechanism should be able to change it to make hashes match or closely match. | yes | Passed, comments below. |
| 3 | Use two edited saturation images that affects the entire image | Greyscale mechanism should be able to change it to make hashes match or closely match. | yes | Passed, comments below |
| 4 | Use two edited size/resolution images that affects the entire image | Resizing mechanism should be able to change it to make hashes match or closely match. | yes | Yes |
| **Comments** | Resizing mechanism works perfectly! The hashes match the database file to the original file.   "Passed” reasoning - The greyscale mechanism does seem to be working. However, little snippets of the hash changes which could be due to the changes of files since it can create more “gradient” changes.   More testing will be needed to ensure full functionality. Possibly, another function will be needed to ensure matching hashes meet for saturation changes and hue changes due to the harder “gradients” in images.  NOTE: FOR NEXT TESTING ROUNDdings in flask. Will be testing later to ensure the function here works as well. | | | |