Objective

For the final iteration, we would like to close the gaping security hole of having an unprotected create user screen for the slightly smaller gaping security hole of using an unprotected web service. We would also like to provide some documentation so that others can verify that our solution works as intended after deployment.

Step 1 - Write the test plan

This step needs to be done early! The person you are paired with is depending on your test plan to be able to test your solution.

Generate a test plan document. This document should provide a sequence of steps, expected inputs and outputs, to verify that your program is working as intended. Each test should be numbered so that your peer can generate a brief test report indicating which tests passed and which tests failed. You may include installation instructions in this document, however these are not required since all of our solutions should install the same way using the preflight.sh script.

Minimally your test plan should cover:

- Creating the initial users
- Login to the system
- Add two facilities
- Add two assets
- Complete the asset transit process
- Dispose of an asset
- Run each of the reports

You should add a directory named 'testdoc' to your repository. A copy of test plan in PDF format should be committed to the testdoc directory.

-- Test plan - 5 pts

Step 2 - Exchange test plans

Let your peer know that your test plan is ready. They will get their copy of the test plan by checking out your repository. While you are waiting on your peer to deliver their test plan, you can make progress on the other assignment steps.

When your peer informs you that their test plan is ready, checkout your peer's repository and attempt to execute the test plan. As you execute the testplan, keep a

text document open and note for each test wether it passed or failed. For failed tests, include a short description of what/how it failed. Send your completed test report to your peer so that it can be added to their repository.

-- Tests - 2 pts

Step 3 - Start the feature branch

This is done like we have in the previous weeks. The feature branch name should be 'assignment10'.

-- Feature branch - 0pts (just what we do)

Step 4 - Create a web service to add/revoke users

A web service to add and revoke users is needed to replace the current unprotected user creation screen. If an existing username is given when making the create user call, the service should make the existing user active and replace their password rather than making a new user record. A revoked user should not be able to login to LOST or use any access controlled pages. If the username associated with a revoke call does not exist the call should not create a user.

Should these services be done via one service call or two separate service calls? What should the routes be? What arguments are needed? Are database changes needed to support user revocation? All of these design decisions are left to you. Please add a comment in your app.py to indicate the routes that are supporting your web service and add comments to the create_tables.sql file if changes are made there.

-- Web service - 0 pts since we can't test until the clients are done (you chose the protocol so we can't write our own clients).

Step 5 - Create a CLI client to add/revoke users

Two clients are needed to interact with your web service. Add a 'clients' directory to your repository to hold your web service client implementations. One client will be implemented to create users, 'activate_user.py'. A second client will be implemented to revoke users, 'revoke_user.py'.

activate_user.py

Three arguments will be passed to activate_user.py. The first argument will be the host part of the URL for your LOST instance and will include a trailing '/' (e.g. http://127.0.0.1:8080/). The second argument will be the username to create or reactivate. The third argument will be the password to set for the user.

```
python3 activate_user.py http://127.0.0.1:8080/ smith14 password
```

revoke_user.py

Two arguments will be passed to revoke_user.py. The first argument will be the host part of the URL for your LOST instance and will include a trailing '/' (e.g. http://127.0.0.1:8080/). The second argument will be the username to revoke.

```
python3 revoke_user.py http://127.0.0.1:8080/ smith14
```

-- Web service clients - 10 pts (5pts activate_user.py working, 5pts revoke_user.py working)

Step 6 - Disable the user create screen

Remove the path from app.py that allows users to be created using the web browser. Also remove any links that occur on other pages for the create user screen.

-- Remove path - 2 pts

Step 7 - Add test report to repo

Commit the test report document received from your peer to the testdoc directory of your repositor with the filename 'test_report_wk10.txt'.

-- Include report - 1 pts

Step 8 - Commit and push everything

This is also done like in prior weeks. Merge the feature branch into the master branch, commit and push.