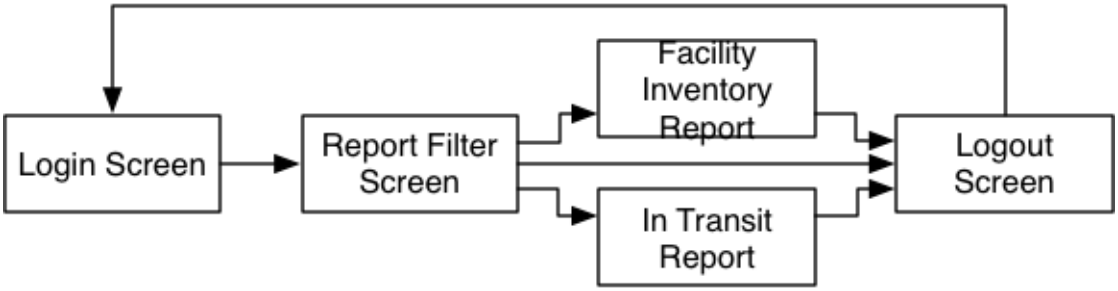
Objective

Assignment 3 is a first web application using Flask. Your web application will implement a simple page flow and retrieve some data from the database. This assignment checks that you can build a simple web application using flask and that you can connect your flask application to a database.

Step 1: Demonstration Application Description

We have some data in the database and the customer has already talked to the sales and project managers about having problems getting reports on their data. Since we imported some of their data last week, demonstrating that we can easily generate some reports that are currently very hard for them will be very impressive to them and shouldn't be too hard for us.

Screen Flow



The demonstration application will have a few screens and some state will need to be tracked between screens in a user session. An example of state is the username associated with the current session. If errors occur on any page, the user should be redirected to the logout screen

Login Screen

We're not actually going to authenticate a login yet. This screen will just accept whatever information the user gives.

Report Filter Screen

The user should be able to select the parameters to use for a report to run. Key things the customer would like to filter by include facilities and date. Compartment data would also be of interest but is not required for this demo.

Facility Inventory Report

This report should list all of the assets located at a particular facility on the given date. The report should also include the asset arrival and was expunged dates.

In Transit Report

This report should list all of the assets in transit on the given date. This report should include where and when the asset departed and where and when the asset arrived.

Logout Screen

This screen provides a goodbye message to the user and provides a link to the login screen.

Step 2: Write the Application

From the screen flow and report descriptions implement the demonstration code as a flask application or wsgi script. The final project will need to delivered using wsgi but, for the demo, running flask directly will satisfy the customer.

Source code for the application should go the 'src' directory of your repository.

Step 3: Documentation

You must include a READE.txt file in the source directory that enumerates all of the files in the src directory with a short comment regarding what capability the file provides (e.g. app.py - python code to run directly with python3). Your README.txt file must clearly state wether your demonstration code should be run using wsgi via Apache or directly using python from the commandline.

Step 4: Verify commit and push

Git has a two step process for publishing changes. The first step is to "commit", which adds a change set to the local copy of the repository. The second step is to "push", which copies all of the committed changes to the remote repository.

To verify your software is working correctly:

1. Create a new database and import the legacy data using your data import script (output of assignment 2).

2. Start your flask application (if using wsgi, copy your code to the correct path and start apache, if using flask directly invoke your flask program using python3).
3. Point your browser to http://127.0.0.1:8080 to interact with your application.
4. Login, run each of the reports once, and logout