Technical paper - BeerTracker

Frontend

HTML

- I use the templating language Jinja to dynamically generate HTML code.
- Every page is generated by using “layout.html” as a template and inserting the individual page’s HTML, which is defined in another file.
- The file “layout.html” itself contains some HTML meta tags, links to the CSS and JavaScript files and the navigation buttons, which need to be included on every page.
- To make the transaction history scrollable I created two tables, one for the head row and one one for the body rows. The body container has a limited height and becomes scrollable if it’s content doesn’t fit the height anymore.
- In order to communicate to the server whether the user gave beer or got beer, I added an invisible field in the form called “action-info” which gets sent along with the other information.

CSS

- At the beginning of the CSS file I reset some of the browser default CSS styles.
- I tried to apply the principles of BEM (Block Element Modifier) although the execution might not be perfect as this was my first time using it.

JavaScript

- I use JavaScript to show and hide the form when the user click on the “I got beer” / “I gave beer” buttons.
- I also use it to set the “action-info” field’s content depending on which button the user clicked

Backend

Database

- I use an SQLite database.
- It contains two tables, one to keep track of users and another one for the transactions (the records of all the beers given and received).
- Most of the SQL is very basic but the “SELECT” command in the “compare” function in “application.py” is a bit more complex. I use it to get the scores of the user and their friend. As I store the amount of beers in a transaction as either a positive or a negative integer depending on the direction of the transaction (user -> friend: positive, friend -> user: negative), I need to split this column into two so that I can sum them up to get the user’s score and the friend’s score separately.

Framework

- I use the framework Flask with Python 3.
- For communication with the database I use the SQL functionality that the CS50 library offers.
- The file “helpers.py” contains a few additional functions that get called in “application.py”.