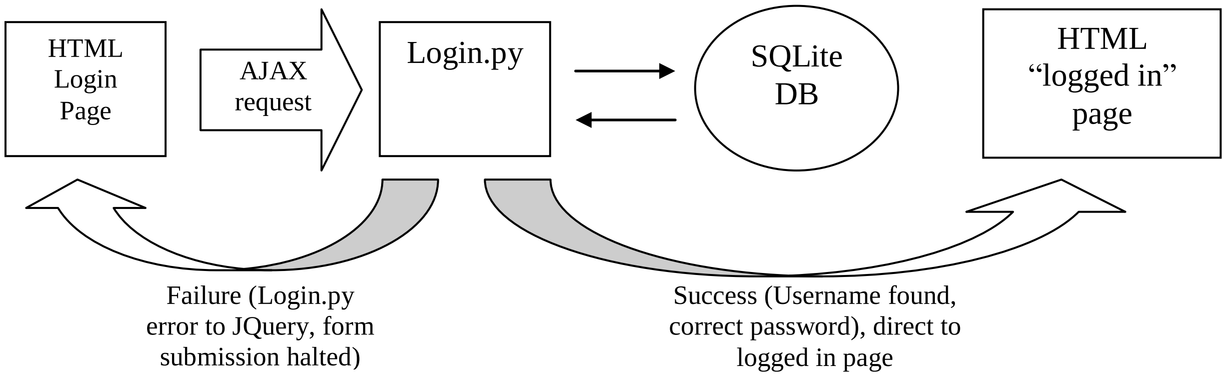
ventUr: Milestone 2 Progress Report

Ari Geller, Adam Kravitz, Thomas Jeffries, and Sam Wlody

Our project can be found at https://github.com/swlody/CSC210-ventUR.

Our login system is a combination of HTML, JavaScript, Python, and SQLite:


HTML Login Page
AJAX request
Login.py
SQLite DB
HTML "logged in" page
Failure (Login.py error to JQuery, form submission halted)
Success (Username found, correct password), direct to logged in page

When the user clicks the "submit" button, the form tries to direct the user to the "logged-in" page. However, before that happens, the JQuery LogIn.js script first makes an AJAX request to the Python script "login.py." login.py attempts to find the user in the SQLite database, and compares the given password with the password (encrypted and hashed with salt) stored in the database. If it succeeds, then JQuery will let the form submit and go on to the "logged-in" page. If the user entered a wrong username, or the passwords do not match, then the Python script will return an error to JQuery, which will then not let the form submit, and alert the user that the username/password combination was incorrect.

We think the coolest part of our project so far is how it tells the user if a username is taken when they try to sign up for a new account. When they user types in their desired username in the sign-up page, an AJAX request goes to the database to check if the username is already taken. If it is free, the user will see in green letters “Your chosen username is available!” Otherwise, the user will see in red letters “That username is already taken.” and will not be able to submit the form until they have chosen an available username. Many websites just alert you to the fact that your username is taken when you click the submit button, but this adds a little more user-friendliness to the experience, so that they know instantly whether they need to pick a new name or not.

Many parts of this milestone were challenging, since none of us have a background in web programming. In addition, no one person knew all of the languages/technologies we used, so everyone had to learn something new. It was especially frustrating to debug some of the python scripts, since they get called inside AJAX requests, and so cgitb doesn’t apply to them. We also had some difficulty in getting the headers on Python scripts to be correct; as often there would be errors in them with no easily apparent reason, just like in class.