Milestone 6

Tue Apr 5 (125 points)

1. Make sure you have the `SaltedHashedPassword` class defined as in cs691pub. Run its `main` method so you can generate a hash for whatever password you provide:

```
Create new password: secret123
43685c6394daf9a10910bacfbad5c7f2$b11e640bbdaab0d29d9e0c5cd58dc0e6dc0cba9c3c4beddde4b50
```

2. Copy the long hex string you generated in the previous program, and paste it into one of your SQL migration scripts as the password for a sample user. The way my user table is defined, it's something like:

```
insert into user (email, password) values
( 'league@acm.org'
, '43685c6394daf9a10910bacfbad5c7f2$b11e640bbdaab0d29d9e0c5cd58dc0e6dc0cba9c3c4beddde4
);
```

3. Using the `Paths` helper and `get`/`post` declarations in `SparkDemo.main`, define a login screen.

   In the `get` handler, it should display an HTML form with email address and password fields, and a submit button labeled "Log in".

   In the `post` handler, it will look up the email address in the user table of the database, construct a `SaltedHashedPassword` object from the password field, and then use the object's `check` method with the password from the login form.

   If the check succeeds, it should set a session variable to the user ID and redirect to the polls list. If not, it should display an error message (invalid user or password) and allow the user to try again.

4. Add a blurb to the page containing the polls list to say either "Welcome, {{email}}" with a logout link; or say "You are not logged in" with a link to the login page.

   We still have an example of this sort of functionality from early in the course, but it accepted whatever name the user provided without checking a password or that the account exists.