Sprint 1 - Generating QR code for given URL

<Equivalence Classes>

Acceptable Input: Any random string consist of ASCII characters except /.
Unacceptable Input: A set of ASCII characters that contains /.
Acceptable Output: A QR code image that is saved to a designated location containing correct URL information.

<Test 1>

1. Randomly generate the 10 acceptable input for testing.
2. Create the QR code for each of them and save it.
3. Check every QR code exists on the designated folder.
4. Check each QR code contains correct URL information
5. Expected Output: 10 QR images with correct URL data.

Sprint 2 - Basic signup/login/logout functionality

<Equivalent Classes - for signup>

Acceptable Input: Any non-conflicting non-null ASCII string for username, any non-null ASCII string for the password
Unacceptable Input: Null string for username or password, a username that already exists in the database.

<Test 1>

1. Create 10 username-password pairs so that usernames are not conflicting.
2. Create 10 accounts using the signup page.
3. Go to admin page and view users.
4. Check if every user account is successfully made.
5. Expected Output: 10 User account made in the admin page.

<Test 2>

1. Create a temporary account.
2. With using the same username with 1, try to create another user account.
3. Expected Output: Error message that username already exists should occur.

<Equivalent Classes - for login>

Acceptable Input: username-password pair that each consists of non-null ASCII string, and already exists in the database(created through signup).
Unacceptable Input: username-password pair that do not consist of ASCII characters, or doesn't contain the correct account information(wrong username/password)

<Test 1 >

1. Create 10 username-password pairs so that usernames are not conflicting.
2. Create 10 accounts using the signup page.
3. Go to the login page.

4. Try to log in with all of the 10 accounts, reset the system manually every time we success to log in since we don't have logout functionality yet.
5. Expected Output: Login Success for every 10 accounts.

<Test 2>
1. Create an acceptable username-password pair and create a user.
2. Try to log in with a different username, and different password
3. Expected Output: Login fails with "username or password incorrect" message.

<Equivalent Classes - for logout>

Acceptable Action: In any pages in our project, clicking the logout button.
Acceptable Output: If a user is logged in, the user is logged out from the system successfully. If the user is not logged in, nothing happens.

<Test 1>
1. Add a message to the home page that can be only viewed by logged in users.
2. Create a temporary account, log in with that account.
3. Try to log out on the home page.
4. Check if we see the message.
5. Expected Output: Redirecting to the home page with logged out.

<Test 2>
1. Create an account, but remained not logged in.
2. Try to log out on the home page.
3. Expected Output: Redirecting to the home page with logged out. (nothing changes)

Sprint 2 - permission views on course and lecture page.

<Equivalent Classes>

Acceptable Output: Contents in our course and lecture page only viewed for users that logged in.
Unacceptable Output: Viewing contents without logged in.

<Test 1>
1. Create an account, log in to that account.
2. Create temporary courses and lectures.
3. For course-detail, course-index, and lecture-index pages(this is only pages we had at this point) check the contents of the pages.
4. Log out from the page.
5. Revisit each web page to check the contents of the pages.
6. Expected Output: Every content in web pages are only viewed when the user is logged in, and if the user is not logged in, a warning message appears instead of contents.

Sprint 3 - Designing web pages.

In the design part, I didn’t devise a formalized test method, but here are the things I checked whenever I changed my design for a certain page.

1. For Navbar: For home, course, log in, logout, admin button, check if all of them works properly. And any of those redirections contains a bug or lead the user to the wrong URL.
2. For functionality in a Web page: For basic functionality existing in lecture, check if they perform the same by following the simple routine for that page. (ex. For course detail page, check if the lecture is successfully generated and viewed) This step is required because some of the design fixes might lead to buggy behavior of functionality.
3. For alignment and contents: For every design changes, check if they are fully adopted. Such as colors are correctly changed, blocks are correctly aligned, and contents are viewed without any bugs.