Assignment 1: Fix SQL injection

Correct SQL injection vulnerability in the log-in example.

Logging-in with, for instance, **student' OR '** and an arbitrary password, should not be possible.

(File `model/UserDB.php`)

Assignment 2: Fix XSS vulnerability

The form and the super-global `$_SERVER['PHP_SELF']` are not sanitized in the PHP_SELF example.

Sanitize the appropriate variables so that XSS attacks will not be possible.

(File `other/index.php`)

Assignment 3: Form validation

1) Implement the **client-side** validation for the book add form. All constraints can be implemented by adding appropriate attributes to existing HTML form elements.
   - **Author**: allow only letters, spaces, dots and dashes; the field is required;
   - **Title**: required field, can contain arbitrary characters;
   - **Description**: may contain arbitrary characters, optional;
   - **Price**: required, should contain a non-negative number;
   - **Year**: required, should contain a number between 1500 and 2020;
   - **Quantity**: required, should contain a number greater or equal to 10.
2) Implement both the **server-side** and **client-side** validation for the book edit form. To see how server-side can be implemented, study the book add controller. The rules are similar to the add form.
   - **Author**: allow only letters, spaces, dots and dashes; the field is required;
   - **Title**: required field, can contain arbitrary characters;
   - **Description**: may contain arbitrary characters, optional;
   - **Price**: required, should contain a non-negative number;
   - **Year**: required, should contain a number between 1500 and 2020;
   - **Quantity**: required, should contain a non-negative number.
   - **Id**: required, should contain a positive number.

(Files `view/book-{add,edit}.php, controller/BookController.php`)