1. Problem: No value is being submitted since name attributes in input tags are missing

Fix: add the name attributes in the input tags

```
<label>
    Email: <input type="email" name="email">
</label>
<label>
    Password: <input type="password" name="password">
</label>
<input type="submit" name="submit"/>
```

2. Problem: No HTTP Method in form is specified, GET will be used by default (refer to https://www.w3.org/TR/html401/interact/forms.html#h-17.3). Therefore, the email and password will be shown in the url which makes the site very insecure

Fix: specify HTTP POST for the form:

```
<form method="POST">
```

3. Problem: the site is vulnerable to XSS, since the user can embed <script> tags in the input fields to make XSS

Fix: to prevent that, we convert html to text by using the PHP htmlspecialchars() function

```
$email = htmlspecialchars($_POST['email']);
$password = htmlspecialchars($_POST['password']);
loginUser($email, $password);
```

4. Problem: the site is vulnerable to SQL Injection, since user can concatenate strings to make SQL Injection (assumed that the login function simply does the login and no further validations)

Fix: there are multiple ways to prevent SQL Injection, one is by using the $mysqli->real_escape_string() function:

```
$email = htmlspecialchars($_POST['email']);
$password = htmlspecialchars($_POST['password']);

$mysqli = new mysqli("host", "user", "password", "database");

$email = $mysqli->real_escape_string($email);
$password = $mysqli->real_escape_string($password);

loginUser($email, $password);
```

5. Problem: the submitted parameters might not exist or is NULL

Fix: we check the submitted parameters by using the PHP isset() function:

```
if (isset($_POST['submit'],$_POST['email'], $_POST['password']))
```