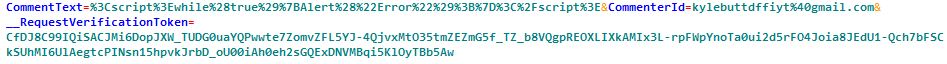
***https://github.com/boqli/SecuringAppsAssignment***  
For testing the security of the page created ZAP was used along with Modzilla.

File injection was attempted on the website, firstly by trying to add ‘DROP TABLE TASK’ {1} into the commenting section of the website. A script tag with a short piece of code was also added. For the script tag {2} the program changed the < and { symbols to other character to prevent the code from being executed. The drop table example was not changed but remained the same, still causing no effect to the database. The email of the uploader was also changed, showing no clear error. The database though did not allow the input from the ZAP software.   
The site was not made to output the exact error to the user once it is promoted, so the outputs of ZAP were misleading although the database was checked to see what made it in or not. The anti-forgery token also allowed some of the attacks to be stopped as the tokens would not match from on ZAP.

{1}

Text

Description automatically generated with medium confidence  
{2}

  
{3}

For testing broken authentication, a different email was attempted that does not exist, this was obviously not allowed, neither was trying to pass a script or code inside the email or password field {4}. Different passwords were also attempted to the same account, also generic admin emails and simple passwords too. This was to see if there was a simple [admin@admin.com](mailto:admin@admin.com) user that would have a simple password along with a lot of control on the site. The correct measures were taken to not allow any of these.

  
{4}

Two links, namely {5}, were tried to test if a user who is not logged in could access the locations and information, possibly putting something into the database that is not secure and should not be present. Luckily the website would not allow the user to view any information if it is not authenticated. The attacker would also not be able to submit files as they would not fill multiple criteria in the query to be accepted.

<https://localhost:44397/FileUpload/Upload?taskId=733eb2bd-224a-4f1b-e0ca-08d914b245ee>  
<https://localhost:44397/FileUpload/ViewComments/NzE5ZjY3Y2ItZjlmOC00NTYyLTE5YjtMDhkOTE0OGY3MDA2>   
 {5}

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Description** |
| T1 | Remote anonymous user | A user who has not yet authenticated to the website |
| T2 | Authenticated user | A registered user who has valid credentials, ability to upload assignments and view tasks. |
| T3 | Teacher | User who can do any operation the Authenticated user can and also upload tasks, download files and create students. |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Name** | **Description** | **Trust level** |
| A1 | User | Assets that relate to a website user |  |
| A1.1 | User’s login data | User’s credentials username and password. This asset needs protection because if it is stolen another user would be able to do anything which the user can do | T2 Authenticated user  T3 Database server admin |
| A1.2 | User’s personal data | User’s personal data including contact information. This needs protection because some personal data might be important such as telephone number | T2 Authenticated user  T3 Database server admin |

|  |  |
| --- | --- |
| Id | TR1 |
| Name | Adversary tries to supply malicious data when logging in |
| Description | Adversary tries to input special characters to be able pose as another user, or logs in without having an appropriate username and password. Handling of data is critical in this regards. |
| Stride | Tampering, Elevation of privilege |
| Entry Points | (E1.1) Login page |
| Assets | (A1.2) User’s personal data, (A2) Backend database |
| Mitigation Strategy | Using stored procedures or parameterized queries |

|  |  |
| --- | --- |
| Id | TR2 |
| Name | Adversary tries to supply malicious data when commenting |
| Description | Adversary tries to input special characters to be able pose as another user, or alter the database or site code in anyway. |
| Stride | Tampering, Elevation of privilege |
| Entry Points | (E1.1) Commenting Page |
| Assets | (A2) Backend database |
| Mitigation Strategy | Using stored procedures or parameterized queries |