Exercise 2

Deadline: 22.10.2015

Description

In Exercise 2, the groups will work to remedy any vulnerabilities found in the black box and white box testing phase. To help them with this, they will be given a list of all known vulnerabilties of the web application.

The groups will also be expected to add the following features:

To add revenue to their site, Superhealth Inc. would like to introduce paid doctors. Doctors are a special sort of user. The admins will have the ability to declare any user a doctor user. The doctor user can only view and answer posts that fulfill the two following conditions:

- It is posted by a user who has a bank number connected to their profile
- The user has actively chosen to pay to allow visibility to doctors.

The fact that a paid post has been answered by a doctor should be visible to all users. Thus it should be somehow distinguished from a post that has not been paid for and has not already been answered by a doctor. This distinction should be easily noticeable by both doctors and normal users alike without having to read the entire post.

The groups will not need to worry about or consider any actual transaction. The following will suffice to “prove” that a transaction has taken place:

The cost paid by a user of an answered post is $10. A user only pays for the first post answered by a doctor, and a doctor is only paid for answering posts that has not already been answered by a doctor. The company keeps $3 for each unanswered post answered by a doctor.

A paying user should have an overview of accumulated costs of their posts in their profile. A doctor should have an overview of how much money they have earned by answering posts in their profile.

Installation and setup

One group member of each group should create a GitHub-repo with the source code. The link to this repo should be included when delivering the exercise.

Each member of the group must pull from this repo and set up a local setup. See the PDF on itslearning on how to do this.

Students are strongly advised to use an IDE to program. PHPStorm by IntelliJ is available for free on an academical license. Activate this here: https://www.jetbrains.com/student/

Deliverables

The groups should have two deliverables by the end of exericise 2:

- A report that contains the following:
    - How the group ranked the risks of the already existing vulnerabilities
    - How the group fixed the already existing vulnerabilities
    - Potential risks of implementing the new features and how the group avoided these
- The mitigated web application should be delivered by adding “dansolhan” as a collaborator to the GitHub-repo. After the deadline, no change to the code is allowed.

Rules of engagement

- Do not use third party libraries. Any code that you write should be written using PHP built-in libraries or libraries available through the Twig or Slim framework.

  If you feel a library is especially important, contact the student / teaching assistants and ask to be allowed to use it. Any vulnerability mitigation or feature that is accomplished using an unverified third-party library will be considered not properly implemented.

  Note: Third-party libraries are any libraries that need to be specifically and additionally installed. Any functionality already present within php, Twig and Slim is fine.

- The application must retain all existing functionality. Securing the application by removing functionality is not allowed, and neither is securing the application by taking it offline.

- Keep in mind that in exercise 3, another group will try to break your app, so make sure they can’t.

Hints

The groups should consider the following when implementing the new features and writing the report:

- What are the business assets of the web applications?
- Do the requirements themselves pose any business risks?
- Are any of the business assets especially sensitive? How can this be mitigated?
- Imagine that you were to demonstrate potential dangers of the new features to your employers. How should your group do this?