SecureED-NODE.

*User manual*

# Abstract

SecureED. is a webapp that simulates a university registration system. It allows students to try to break the app, using various vulnerabilities and exploits. The software also resets itself to its initial settings following a session, allowing users to try various things without worrying about destroying their installation.

This document contains the manual of how to use SecureED. The rest of the document is ordered as follows: Chapter 1 describes the system requirements. Chapter 2 contains an explanation of how to start the program. Chapter 3 includes instructions for Professors to enter grades. Chapter 4 includes instructions for Students to search for and register for courses. Chapter 5 includes instructions for admins to add, search and edit non-admin user accounts. Chapter 6 describes vulnerabilities and provides explicit examples of their implementations.

This document describes functionality of SecureED. v. 1.0, dated 31 August 2022.

# Table of Contents

1. [Requirements 1](#_bookmark0)
   1. [System Requirements 1](#_bookmark1)
2. [Getting Started 2](#_bookmark2)
   1. [Installation 2](#_bookmark3)
   2. [Start-up 2](#_bookmark4)
   3. [Log In 4](#_bookmark5)
   4. [Forgot Password 4](#_bookmark6)
   5. [Log Out 6](#_bookmark7)
   6. [Closing the Program 6](#_bookmark8)
3. [Faculty 7](#_bookmark9)
   1. [Dashboard 7](#_bookmark10)
   2. [Enter Grades 7](#_bookmark11)
      1. [Description 7](#_bookmark12)
      2. [CSV Format 7](#_bookmark13)
      3. [Instructions 8](#_bookmark14)
4. [Students 9](#_bookmark15)
   1. [Dashboard 9](#_bookmark16)
   2. [Course Search 9](#_bookmark17)
      1. [Description 9](#_bookmark18)
      2. [Instructions 9](#_bookmark19)
   3. [Course Enroll 11](#_bookmark20)
      1. [Description 11](#_bookmark21)
      2. [Instructions 11](#_bookmark22)
5. [Administrators 12](#_bookmark23)
   1. [Dashboard 12](#_bookmark24)
   2. [Create Account 12](#_bookmark25)
      1. [Description 12](#_bookmark26)
      2. [Instructions 12](#_bookmark27)
   3. [User Search 14](#_bookmark28)
      1. [Description 14](#_bookmark29)
      2. [Instructions 14](#_bookmark30)
   4. [Edit Account 15](#_bookmark31)
      1. [Description 15](#_bookmark32)
      2. [Instructions 15](#_bookmark33)
6. [Vulnerabilities 17](#_bookmark34)
   1. [Improper Input Validation 17](#_bookmark35)
   2. [Path Traversal 17](#_bookmark36)
   3. [Cross-Site Scripting 18](#_bookmark37)
   4. [SQL Injection 19](#_bookmark38)
   5. [Exposure of Sensitive Information 20](#_bookmark39)
   6. [Unrestricted File Upload 21](#_bookmark40)
   7. [Improper Access Control 22](#_bookmark41)
   8. [Improper Authentication 23](#_bookmark42)
   9. [Cross-Site Request Forgery 23](#_bookmark43)
   10. [Code Injection 24](#_bookmark44)
7. [User Credentials 25](#_bookmark45)

## REQUIREMENTS

## SYSTEM REQUIREMENTS

SecureED-Node. is written in HTML, CSS, ExpressJS, sqlite3 and Node JS, and was built and tested on Google Chrome v89 – thus, Chrome is the recommended browser. Otherwise, all libraries are included in the installation folder. Screen resolutions above 1920x1080, and aspect ratios greater than 16:9, are untested, and thus unsupported.

## GETTING STARTED

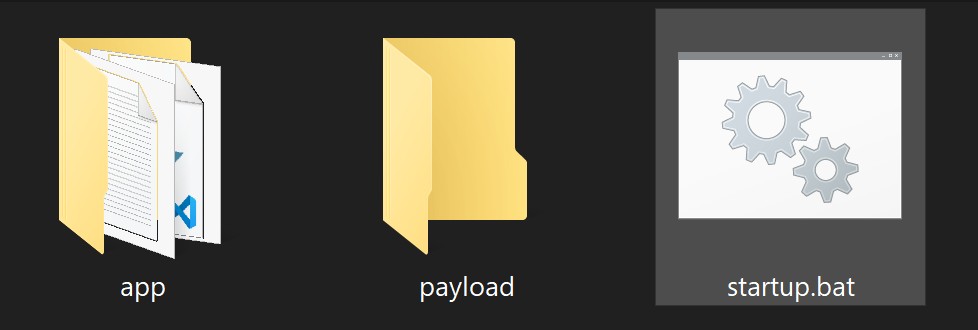
## INSTALLATION

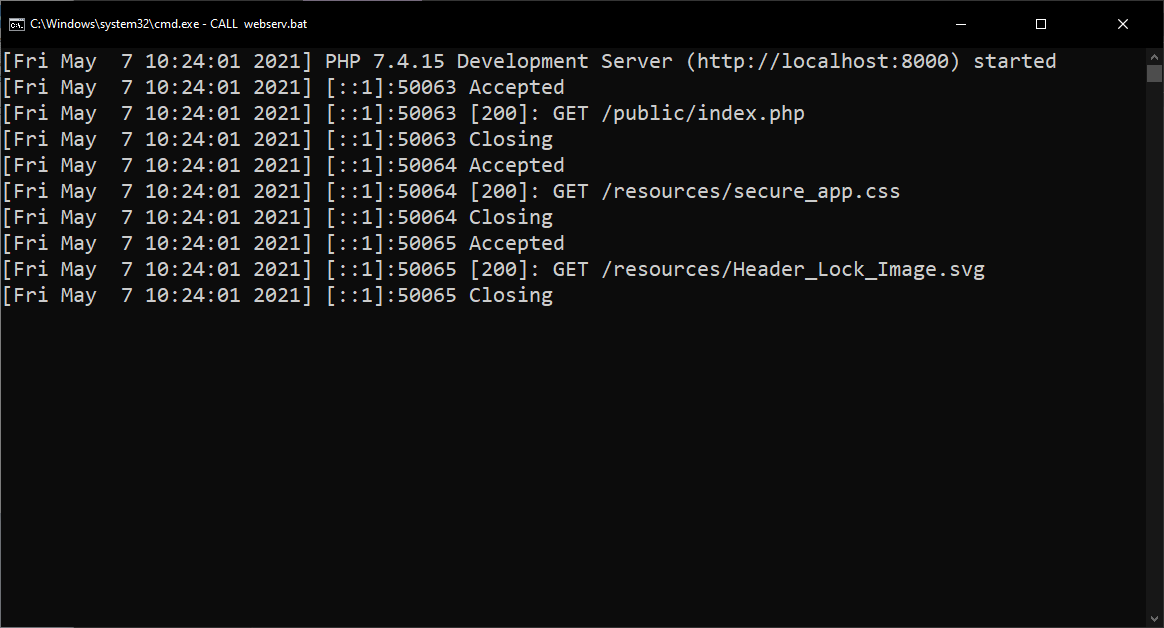
Unzip the file to the folder of your choice. All of the necessary libraries are included – no setup is required.

* 1. **START-UP**

SecureED. is portable, thus self-contained. To start the webserver, please do the following:

1. Navigate to the root folder of your installation.
2. Execute startup.bat





Your default browser will be opened, and you will be presented with a login screen. Note the command prompt opened in the background – **DO NOT CLOSE THIS** until you want to end your session! Upon closing the command prompt in the background, your session will be terminated, and all data will be reset to their default values.

* 1. **LOG IN**

Enter a username and password into the spaces provided, and then press “Submit”.



*Note: If you have forgotten your credentials, please select “Forgot password?”. See the next section for information regarding how to recover your password.*

## FORGOT PASSWORD

In order to recover a lost password:

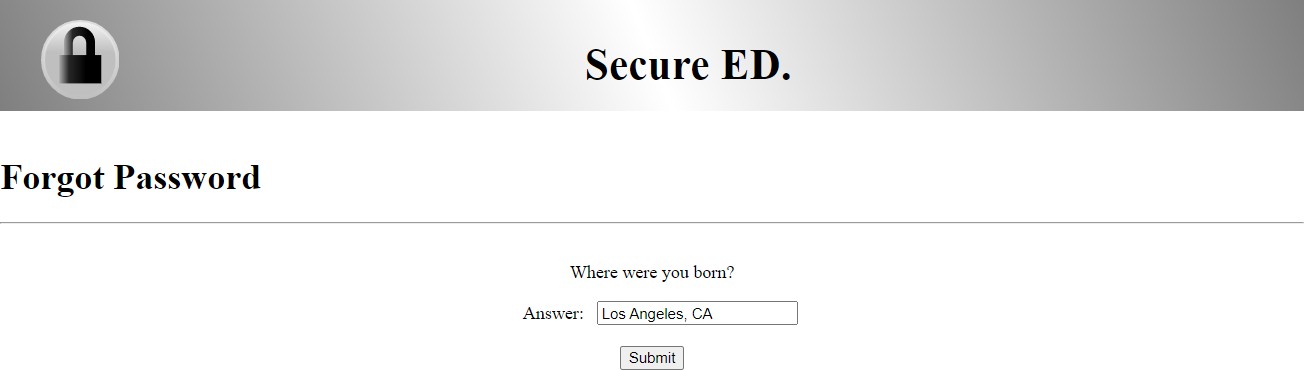
* + 1. Click “Forgot Password?” on the Log In page.



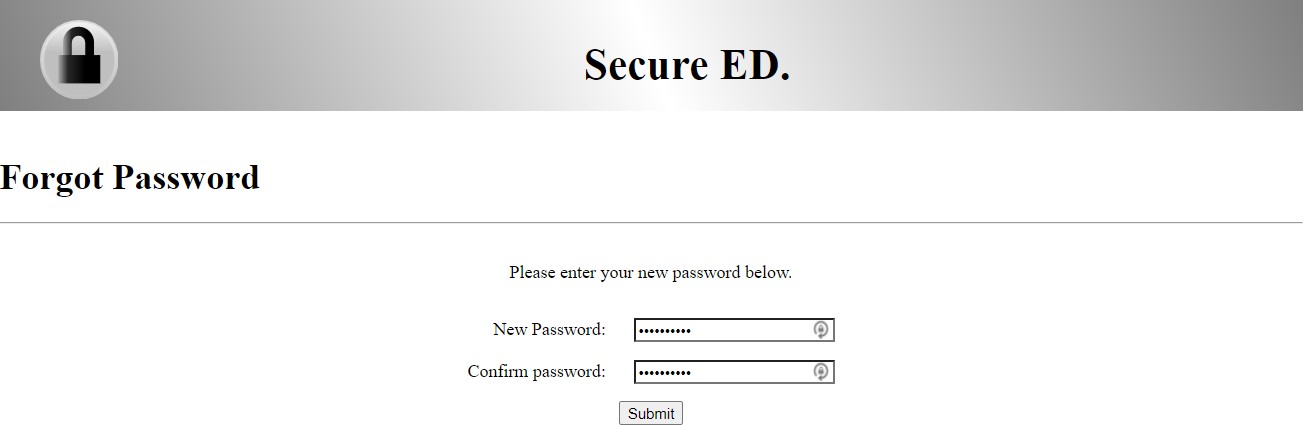
* + 1. On the next page, please enter your email in the provided field, and select “Submit”.



* + 1. You will then be asked the security question associated with that account.



* + 1. Enter the correct answer, and you will be taken to a page that will allow you to change your password.



*Note: Be sure that the contents of the New Password and the Confirm Password fields match!*

* 1. **LOG OUT**

After you have successfully logged in, you may log out at any time. To do so, click on the “Log Out” button on the top pane of any page.

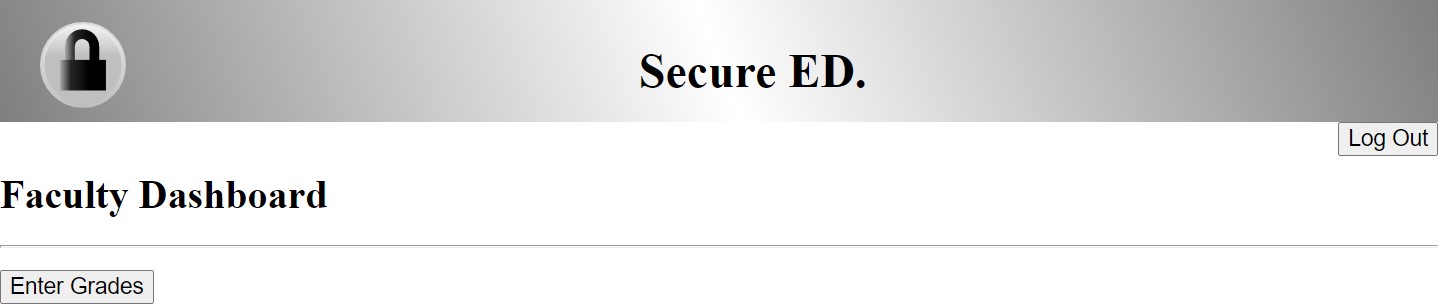


## CLOSING THE PROGRAM

To close the program, close your browser and exit the command prompt window. The program will reset itself to its default values the next time you run startup.

## FACULTY

## DASHBOARD



## ENTER GRADES

* + 1. **DESCRIPTION**

The Enter Grades functionality allows a faculty member to add final grades to a course, using a

.csv (comma-separated values) file.

* + 1. **CSV FORMAT**

Uploaded files must be comma separated text values, in the following format:

USER\_ID,GRADE USER\_ID,GRADE USER\_ID,GRADE

…

e.g.:

927066652,A

927190481,F

927201552,C

927237023,B

927369978,D

927410321,F

927512079,A

927535222,A

927565074,B

927600214,C

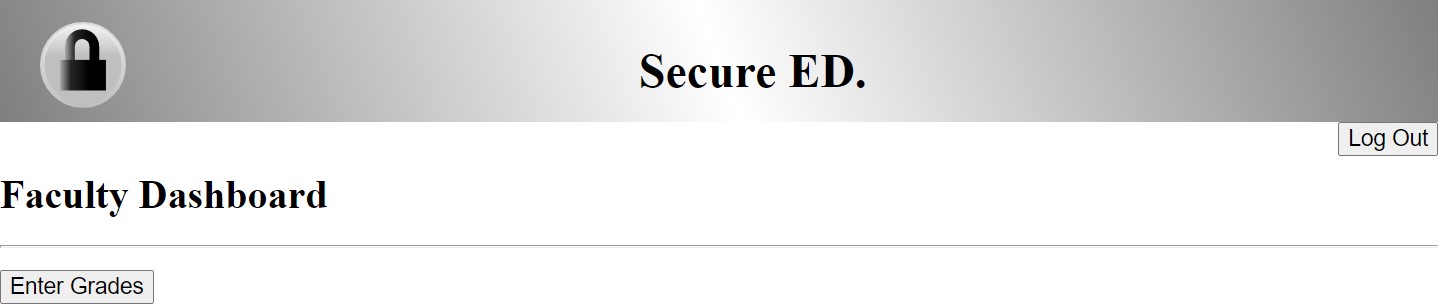
927620146,A

927671464,F

927694539,C

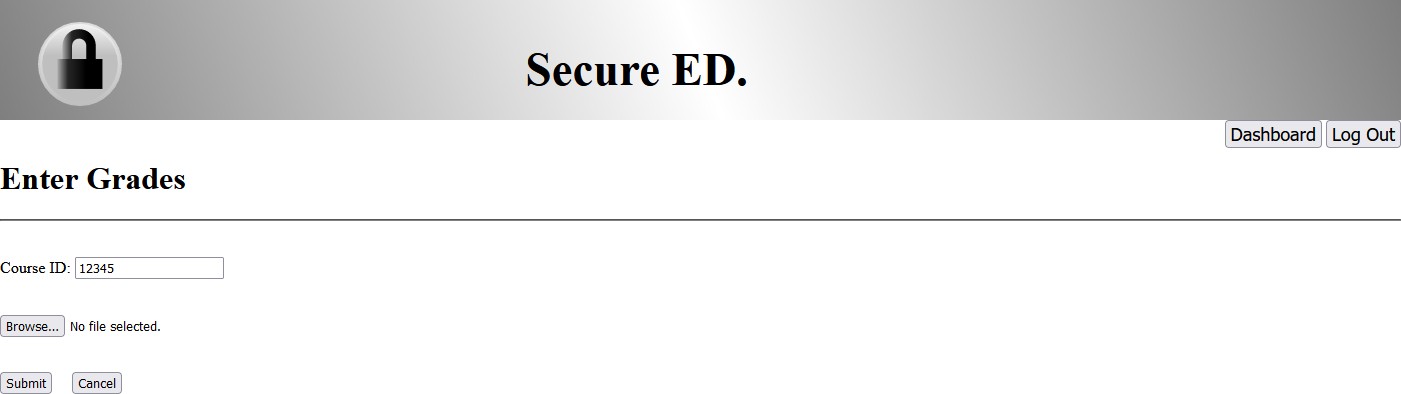
927790514,B

* + 1. **INSTRUCTIONS**
       1. From the Dashboard, select “Enter Grades”.





* + - 1. On the following page, enter the Course ID (CRN).



* + - 1. Select “Browse”.
      2. Select a .csv file from your computer (see 3.2.2 for correct formatting).
      3. Select submit to upload the file.

## STUDENTS

## DASHBOARD

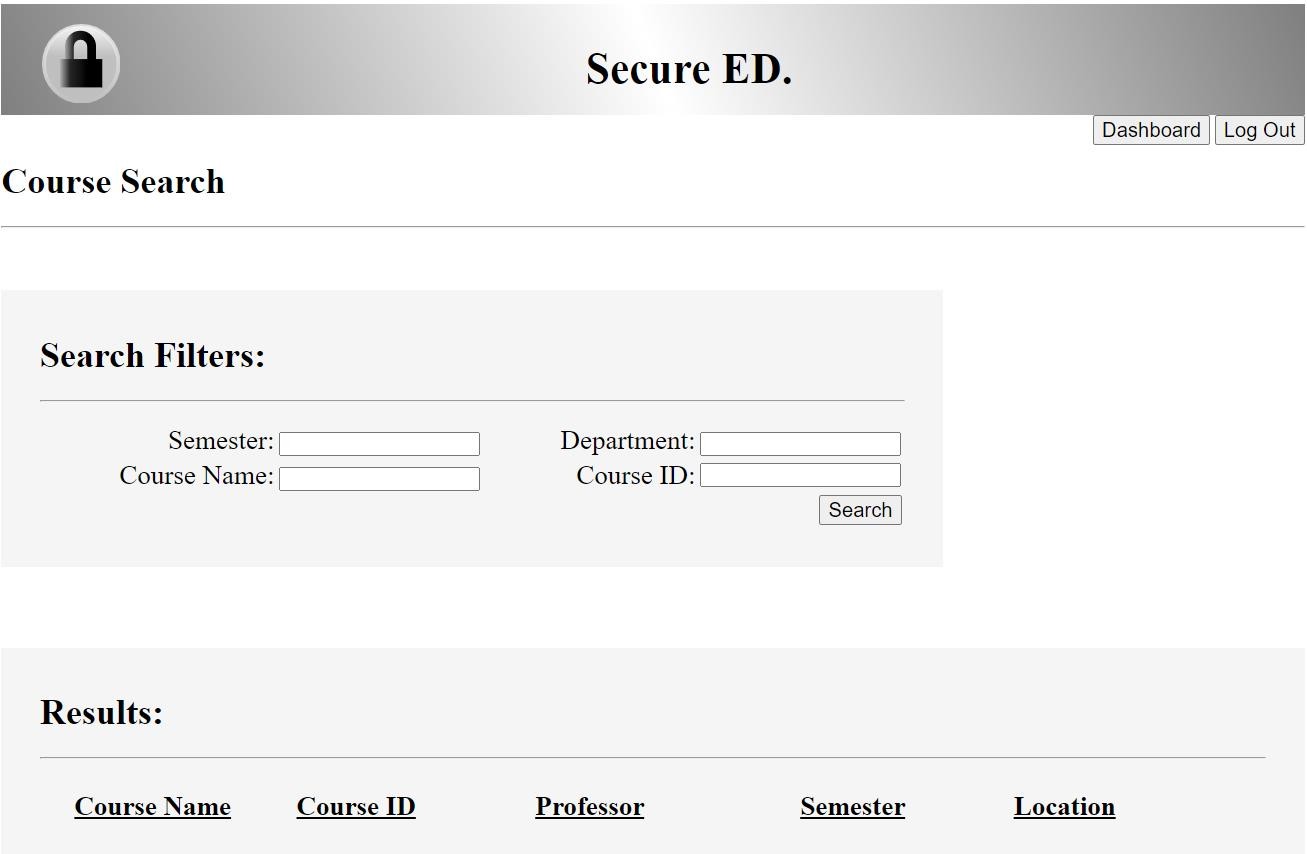
## COURSE SEARCH

* + 1. **DESCRIPTION**

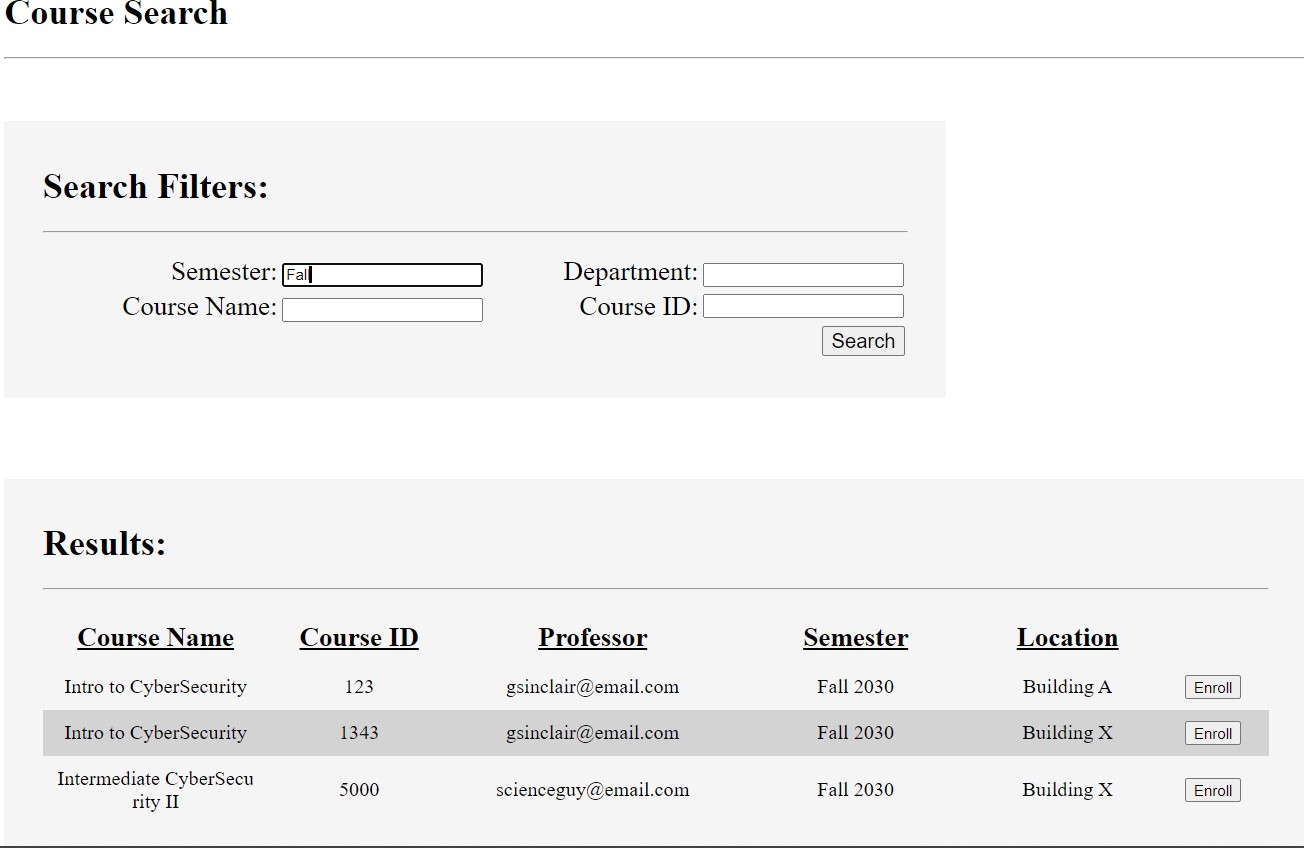
Course search allows students to find current, past and future courses.

* + 1. **INSTRUCTIONS**

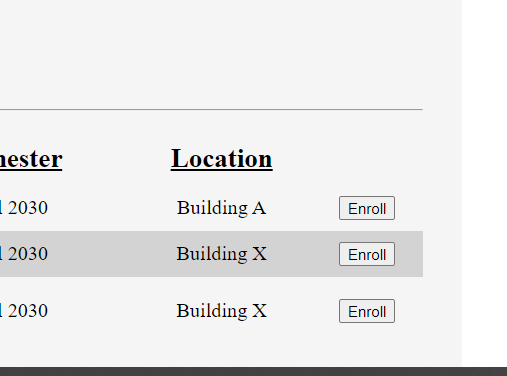
1. From the Dashboard, select “Course Search”.
2. On the following page, enter information relevant to the course you may be interested in.



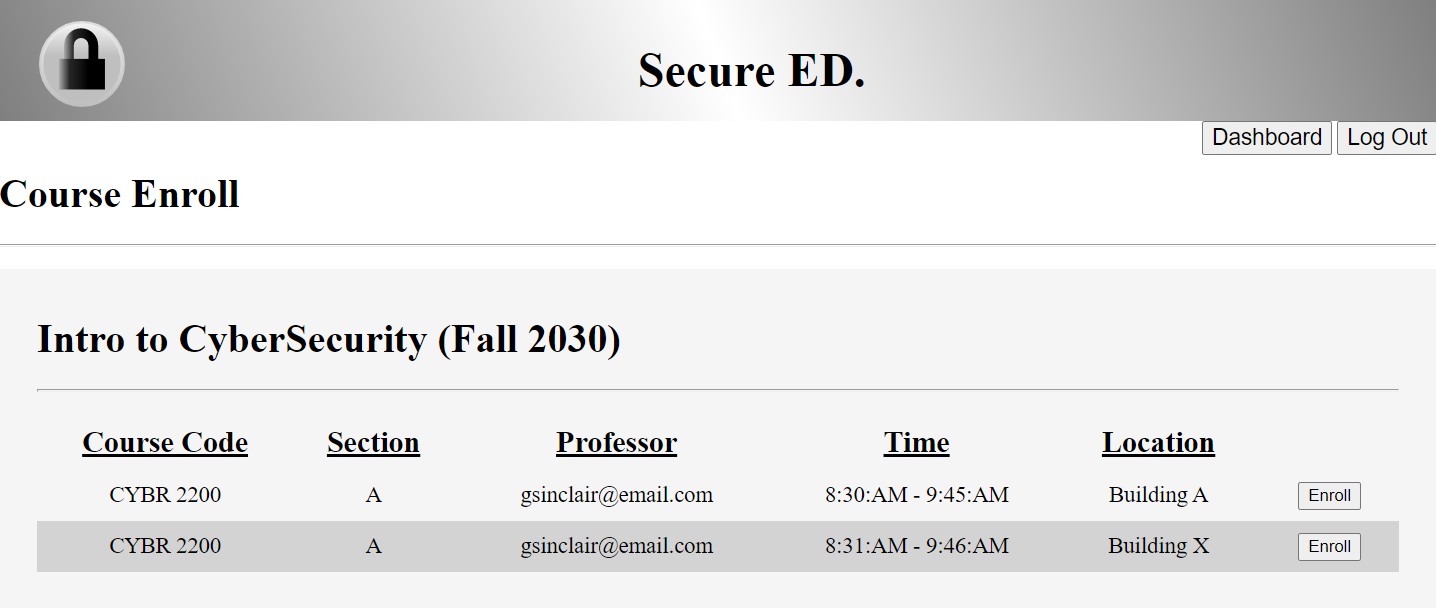
1. Select search.



1. For a list of sections, select enroll.



1. You will be presented with a list of sections to browse.



*Note: If you don’t wish to enroll at this time, you can always select “Dashboard” to return to your landing page without enrolling in a class!*

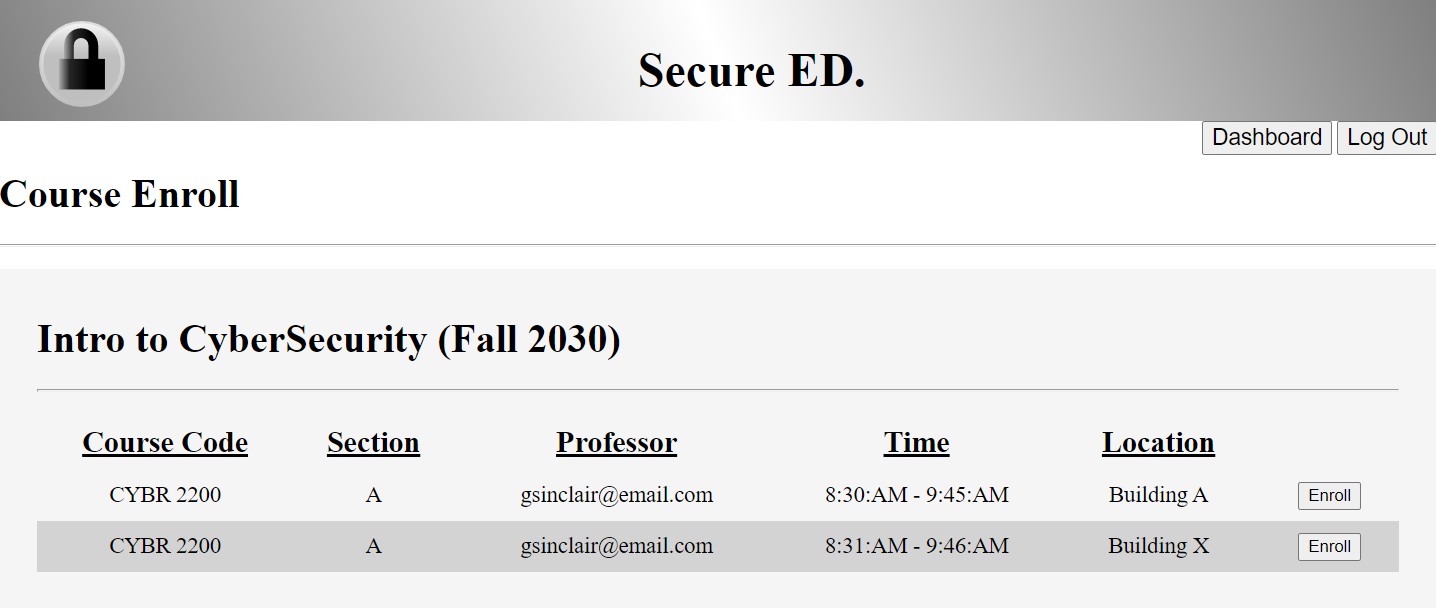
## COURSE ENROLL

* + 1. **DESCRIPTION**

After finding a course using Course Search, a student can enroll in a section of the course.

* + 1. **INSTRUCTIONS**

1. Find a course per the instructions in 4.2.2.
2. Select “Enroll” to bring up a list of sections.
3. Select “Enroll” next to the section you wish to enroll in.



You will automatically be returned to your dashboard.

## ADMINISTRATORS

## DASHBOARD

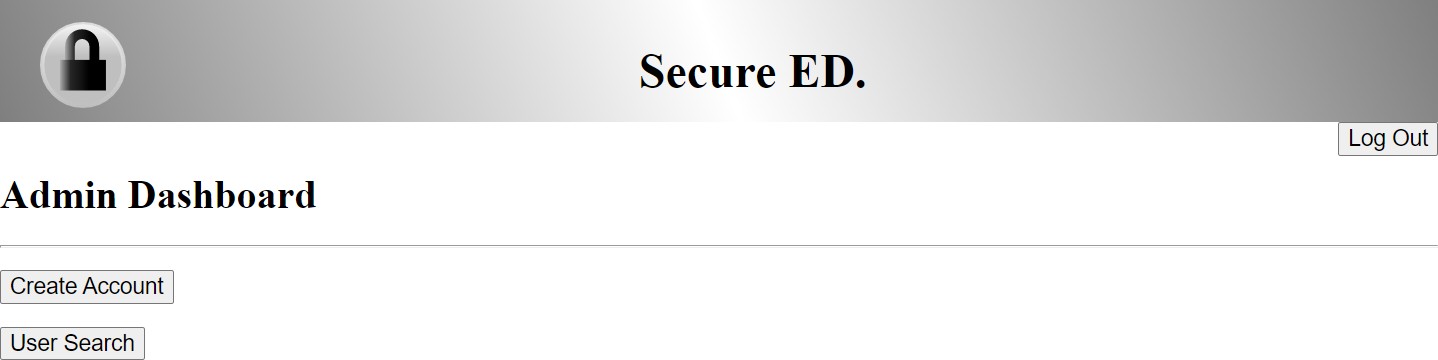
## CREATE ACCOUNT

* + 1. **DESCRIPTION**

Allows the administrator account to create new accounts for new users. This is the only way to create a new account!

* + 1. **INSTRUCTIONS**

1. From the Dashboard, select “Create Account”.



1. On the following page, select the account type.

*Note: Admin accounts cannot be created!*



1. Enter the user’s:
   * First and Last Name
   * Date of Birth
   * Rank (faculty) or Year (students)
   * Email
   * Password
   * Security Question
   * Answer (to Security Question)
2. Confirm the user’s e-mail and password, and re-enter them in their respective confirmation fields.
3. Select “Submit”.

## USER SEARCH

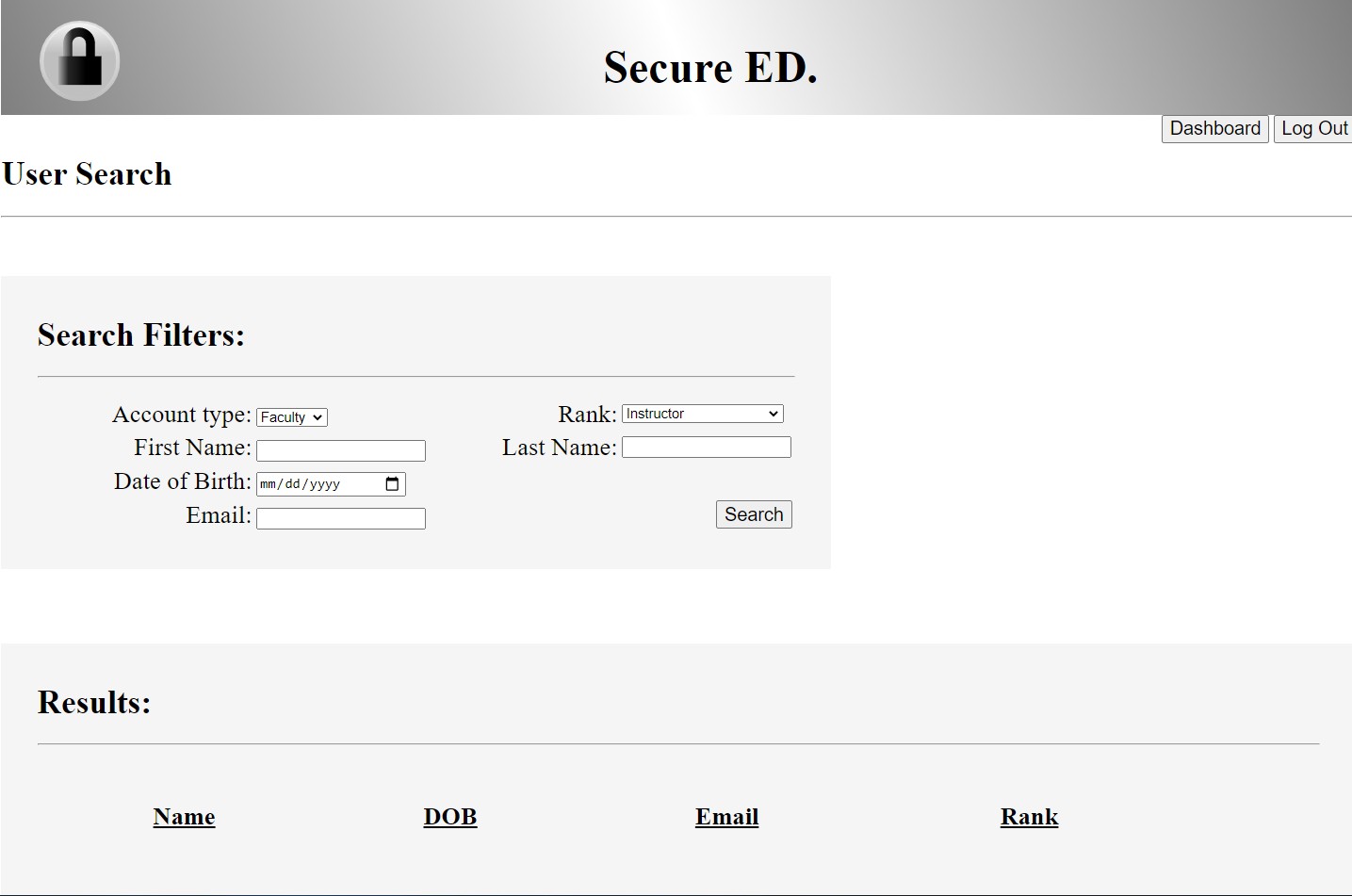
* + 1. **DESCRIPTION**

Allows administrators to search for users using filters.

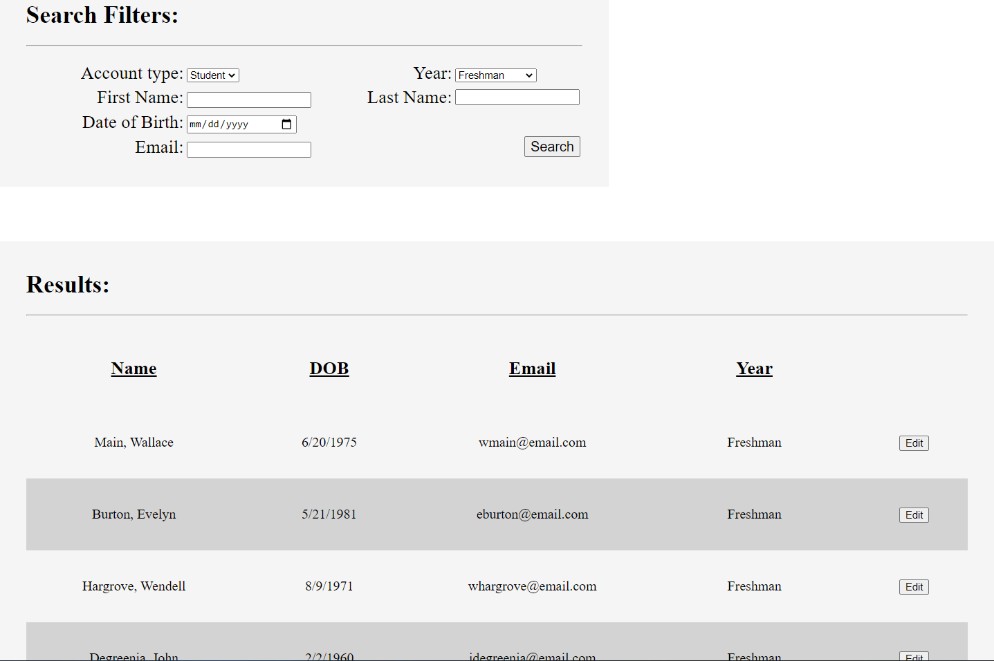
* + 1. **INSTRUCTIONS**

1. From the Dashboard, select “User Search”.
2. Select an account type, and a Rank (faculty) or Year (students).

*Note: These two fields are mandatory!*



1. Enter any more relevant information into the fields provided.
2. Select “Search”.



A list of matching accounts will appear below.

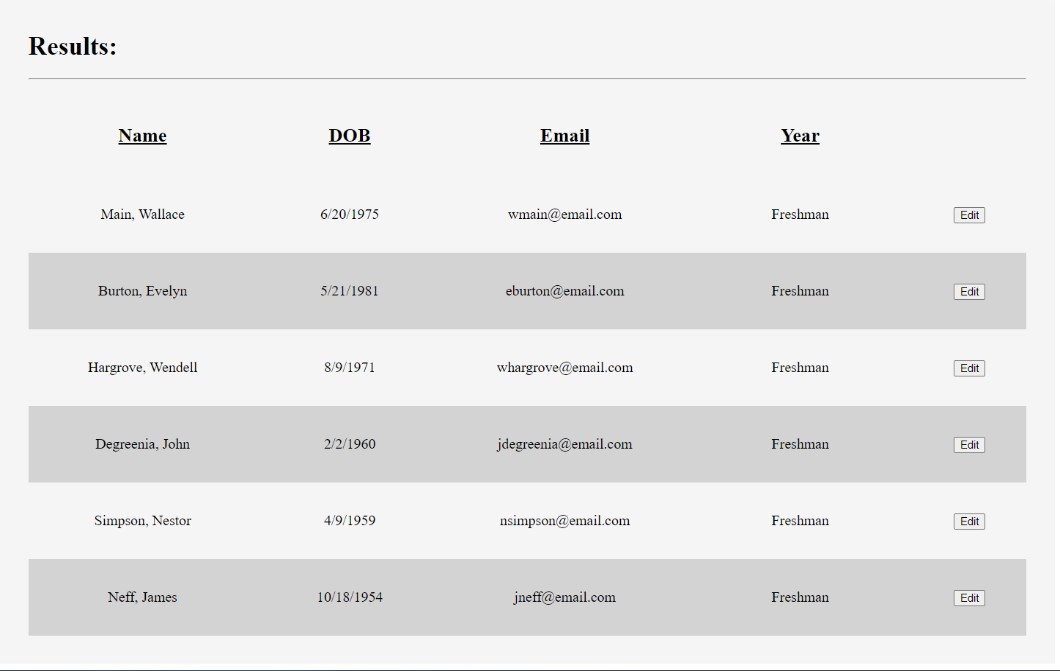
## EDIT ACCOUNT

* + 1. **DESCRIPTION**

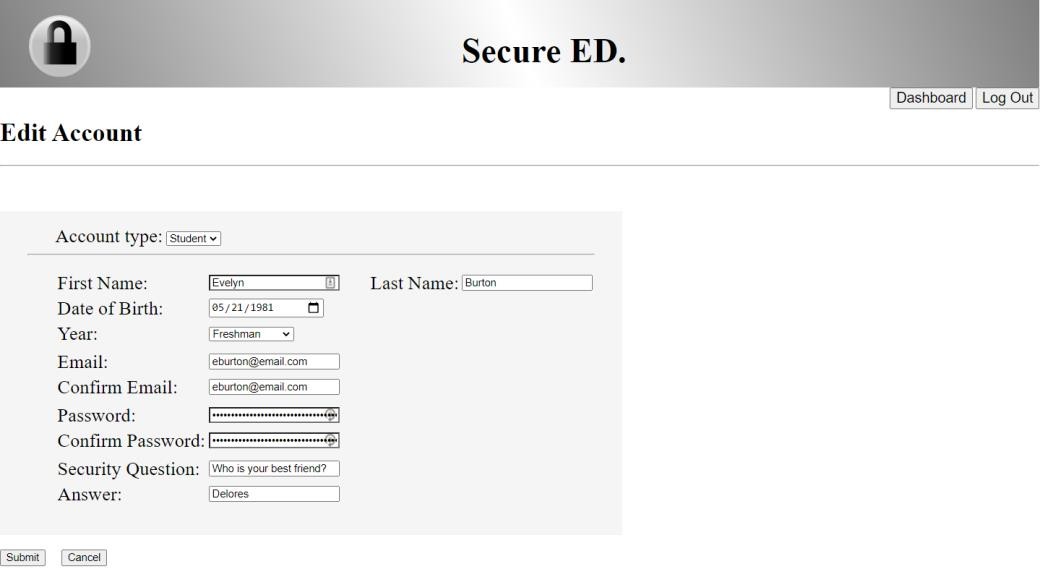
Allows an administrator to edit a non-administrator account.

* + 1. **INSTRUCTIONS**

1. Find a user per the instructions in 5.3.2.
2. Select “Edit” to the right of the user’s information.



1. Make any desired changes to the prepopulated data in the fields provided.



*Note: Don’t forget to confirm any changes to the e-mail or password!*

1. Select “Submit”.

## VULNERABILITIES

This section contains the vulnerabilities in the application. Simply follow the steps to see examples of each vulnerability. Some of these vulnerabilities will alter the behavior of the application. Restarting the application (running startup.bat) will reset the application to its original configuration.

## IMPROPER INPUT VALIDATION

|  |  |  |
| --- | --- | --- |
| *Improper Input Validation* occurs when input or data is not properly checked to ensure that it can be processed correctly. | | |
| ## | Instructions | Result |
| Ex.1 | 1. Go to the Login page: http://localhost:3000/ 2. Enter the following for username:   *' OR 1=1;*   1. Leave the *Password* field blank. 2. Click “Submit” 3. When done, click “Log Out” | Both the username and password fields are not validated  This bypasses the login using a Boolean attack, allowing you to access the admin account.  *Also SQL Injection*  *Also Improper Access Control* |

## PATH TRAVERSAL

|  |  |  |
| --- | --- | --- |
| *Path Traversal* occurs when users are able to access parts of the web app that they are not supposed to be able to access, due to a lack of proper pathname restrictions. Potentially, methods can be employed which will allow an attacker to access directories that are not normally public, such as the root folder. | | |
| ## | Instructions | Result |
| Ex.1 | 1. Navigate to http://localhost:8000/files 2. Click on the folder secured-node and navigate to db folder 3. Click to download the database persistentconndb.sqlite 4. Go to the login page http://localhost:3000/ | This will download the database, which should not be directly accessible.  *Also Improper Access Control Also Exposure of Sensitive Information* |

## CROSS-SITE SCRIPTING

|  |  |  |
| --- | --- | --- |
| *Cross-Site Scripting* is a type of attack whereby malicious script is injected into a site to be run on the client. | | |
| ## | Instructions | Result |
| Ex.1 | 1. Go to the Login page:  *http://localhost:3000/* | This puts a video on the search results. |
|  | 2. Enter the following for username: | *Also Improper Input Validation* |
|  | [*admin@email.com*](mailto:admin@email.com) |  |
|  | 3. Enter the following for password: |  |
|  | *Password1* |  |
|  | 1. Click “Submit” 2. Click “User Search” 3. For “Account Type” select “Student” 4. For “Year” select “Junior” 5. Click “Search” 6. Click “Edit” to change a student’s information. 7. For “First Name” enter the following: |  |
|  | *<iframe height="166"* [*src="http://w*](http://www.youtube.com/embed/oHg5SJYRHA0?autoplay=1)*ww.y*[*outube.com/embed/oHg5S*](http://www.youtube.com/embed/oHg5SJYRHA0?autoplay=1)*J*[*YRHA0?autoplay=1"*](http://www.youtube.com/embed/oHg5SJYRHA0?autoplay=1) *frameborder="0"></iframe>* |  |
|  | 1. Click “Submit” 2. Repeat steps 6 to 8 3. When done, click “Log Out” |  |

* 1. **SQL INJECTION**

|  |  |  |
| --- | --- | --- |
| *SQL Injection* is an insertion of an SQL query in a field or file where other inputs are expected, potentially revealing sensitive data or modifying the database. | | |
| ## | Instructions | Result |
| Ex.1 | Note: This will delete portions of the database and the application will need to be restarted.   1. Navigate to Forgot Password page:   *http://localhost:3000/forgotpassword.html*   1. Enter the following for “Email”:   [*scienceguy@email.com*](mailto:scienceguy@email.com)   1. Click “Submit” 2. Enter the following for “Answer”:   *Charity Nye*   1. Click “Submit” 2. Enter the following for “New Password” and “Confirm Password”:   *'; DROP TABLE users; --*   1. Click “Submit” 2. Enter the following for username:   [*admin@email.com*](mailto:admin@email.com)   1. Enter the following for password:   *Password1*   1. Click “Submit” (Note: you won’t be able to login) 2. Since the User Table has been deleted, restart the application (run the batch file startup) | This will drop the User table. The application will no longer recognize any users.  *Also Improper Input Validation* |
| Ex.2 | 1. Navigate to Forgot Password page:  *http://localhost:3000/forgotpassword.html* | This will add a new admin account for Hacker Man. |

|  |  |  |
| --- | --- | --- |
|  | 1. Enter the following for “Email”:   [*scienceguy@email.com*](mailto:scienceguy@email.com)   1. Click “Submit” 2. Enter the following for “Answer”:   *Charity Nye*   1. Click “Submit” 2. Enter the following for “New Password” and “Confirm Password”:   *INSERT INTO users (acctype, password, fname, lname, dob, email, studentyear, facultyrank, squestion, sanswer, isTemporary) VALUES (1, 'efa9562b824820a468a6918b3476e8e5ccbe5d02', 'Hacker', 'Man', '1901-01-01', 'hackerman@getrekt.com', NULL, NULL, 'Why am I here?', 'CSRF', 1); --*   1. Login as hacker by entering the following for username:   [*hackerman@getrekt.com*](mailto:hackerman@getrekt.com)   1. Enter the following for password:   *111*   1. Click “Submit” 2. When done, click “Log Out” | *Also Improper Input Validation* |

## EXPOSURE OF SENSITIVE INFORMATION

|  |  |  |
| --- | --- | --- |
| *Exposure of Sensitive Information*, as the name suggests, is any information in a web app’s code, pages, or file structure, which is sensitive in nature. This can include things like private user information, business information, or any other data that should otherwise be considered confidential. | | |
| ## | Instructions | Result |

|  |  |  |
| --- | --- | --- |
| Ex.1 | 1. Try to login without entering a username or password | This will dump debugging info to the screen. |
| Ex.2 | 1. Go to the Login page:   *http://localhost:3000/*   1. Enter the following for username:   [*admin@email.com*](mailto:admin@email.com)   1. Enter the following for password:   *Password1*   1. Click “Submit” 2. Click “User Search” 3. Right-click on the page and select “view source” 4. Scroll to the bottom of the page 5. copy the link below:   [*http://localhost:3000/resources/usersearchdisplay.js*](http://localhost:3000/resources/usersearchdisplay.js)   1. Paste the link into browser (navigate to page) | View detailed implementation information with  developer’s comments. |

* 1. **UNRESTRICTED FILE UPLOAD**

|  |  |  |
| --- | --- | --- |
| *Unrestricted File Upload* is an exploit whereby files of an unrestricted type can be uploaded, especially types which were not intended to be uploaded by the web designers. | | |
| ## | Instructions | Result |
| Ex.1 | 1. Go to the Login page:   *http://localhost:3000/*   1. Enter the following for username:   [*scienceguy@email.com*](mailto:scienceguy@email.com)   1. Enter the following for password: | The system uploads an incorrect (and potentially malicious) file that is accessible.  *Also Improper Input Validation Also Improper Access Control* |

|  |  |  |
| --- | --- | --- |
|  | *Password2*   1. Click “Submit” 2. Click “Enter Grades” 3. Enter anything for the Course ID 4. Click “Browse” and select the file named “HelloWorld.js”. It should be in the folder named “payloads” in the application public folder. 5. Click “Submit” 6. Navigate to the following link:   *http://localhost:3000/payloads/HelloWorld.js*   1. When done, go back to the previous page and click “Log Out” |  |

## IMPROPER ACCESS CONTROL

|  |  |  |
| --- | --- | --- |
| *Improper Access Control* is when the application does not provide access to a resource appropriate to the access level of the user. | | |
| ## | Instructions | Result |
| Ex.1 | 1. Go to the Login page:   *http://localhost:3000/*   1. Enter the following for username:   [*student@email.com*](mailto:student@email.com)   1. Enter the following for password:   *Password5*   1. Click “Submit” 2. Navigate to the following link:   [*http://localhost:3000/enter\_grades*](http://localhost:3000/enter_grades)   1. When done, go back to the previous page and click “Log Out” | A functionality that is intended for faculty user is accessible to student user |

* 1. **IMPROPER AUTHENTICATION**

|  |  |  |
| --- | --- | --- |
| *Improper Authentication* is when an app does not properly check the claimed identity of an actor. | | |
| ## | Instructions | Result |
| Ex.1 | 1. Go to the Login page:   *http://localhost:8000/public/index.php*   1. Enter the following for username:   [*scienceguy@email.com*](mailto:scienceguy@email.com)   1. Enter the following for password:   *90fa6dc0e910c9a7f5e5f3fbca1e669caeb80289*   1. Click “Submit” 2. When done, go back to the previous page and click “Log Out” | This logs in with a hashed password, instead of requiring the actual password itself. |

## CROSS-SITE REQUEST FORGERY

|  |  |  |
| --- | --- | --- |
| *Cross-Site Request Forgery* is an attack whereby a user can be made to do things on their account that they don’t want to do, often via phishing or social engineering. | | |
| ## | Instructions | Result |
| Ex.1 | Note: Requires Mozilla Firefox (Do not run private mode)   1. Go to the Login page:   *http://localhost:3000/*   1. Enter the following for username:   [*admin@email.com*](mailto:admin@email.com)   1. Enter the following for password:   *Password1*   1. Click “Submit” 2. Find the file named “cutecat.html”. It should be in the folder named “payloads” in the application folder. 3. Double-click or open the file using a web browser. 4. Go back to the SecureED application and click “Log Out” | This will create a new admin account for Hacker Woman. |

|  |  |  |
| --- | --- | --- |
|  | 1. Login as hacker by entering the following for username:   [*fake@email.com*](mailto:fake@email.com)   1. Enter the following for password:   *csrf*   1. Click “Submit” |  |

* 1. **CODE INJECTION**

|  |  |  |
| --- | --- | --- |
| *Code Injection* is any injection of malicious code, whether by Unrestricted File Upload, Cross- Site Scripting, SQL Injection, or any other method. This example demonstrates PHP code injection via unrestricted file upload. | | |
| ## | Instructions | Result |
| Ex.1 | Note: You may need to turn off your antivirus software for this example.   1. Using a text editor, create a file with the following content:   *const parameter = new URLSearchParams(window.location.search).get('parameter');*  *if (parameter) {*  *document.write(parameter);}*   1. Save the file as “Code.js” 2. Go to the Login page:   *http://localhost:3000/*   1. Enter the following for username:   [*scienceguy@email.com*](mailto:scienceguy@email.com)   1. Enter the following for password:   *Password2*   1. Click “Submit” 2. Click “Enter Grades” 3. Enter anything for the Course ID 4. Click “Browse” and select the file named “Code.js” (from step 2) 5. Click “Submit” 6. Navigate to the following link:   *http://localhost:3000/uploads/Code.js* | Code is injected that allows the attacker to run js commands |

|  |  |  |
| --- | --- | --- |
|  | 12. Navigate to the following link:  *http://localhost:3000/uploads/Code.js?parameter=value* |  |

## USER CREDENTIALS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Account Type** | **Username** | **Password** | **Security Question** | **Security Answer** |
| Faculty | [scienceguy@email.com](mailto:scienceguy@email.com) | Password2 | Favorite Relative? | Charity Nye |
| Student | [student@email.com](mailto:student@email.com) | Password5 | Where were you born? | Los Angeles, CA |
| Admin | [admin@email.com](mailto:admin@email.com) | Password1 | How many siblings do you have? | 0 |