Database Development and Class Registration

Jamain Hughes

The University of Arizona Global Campus

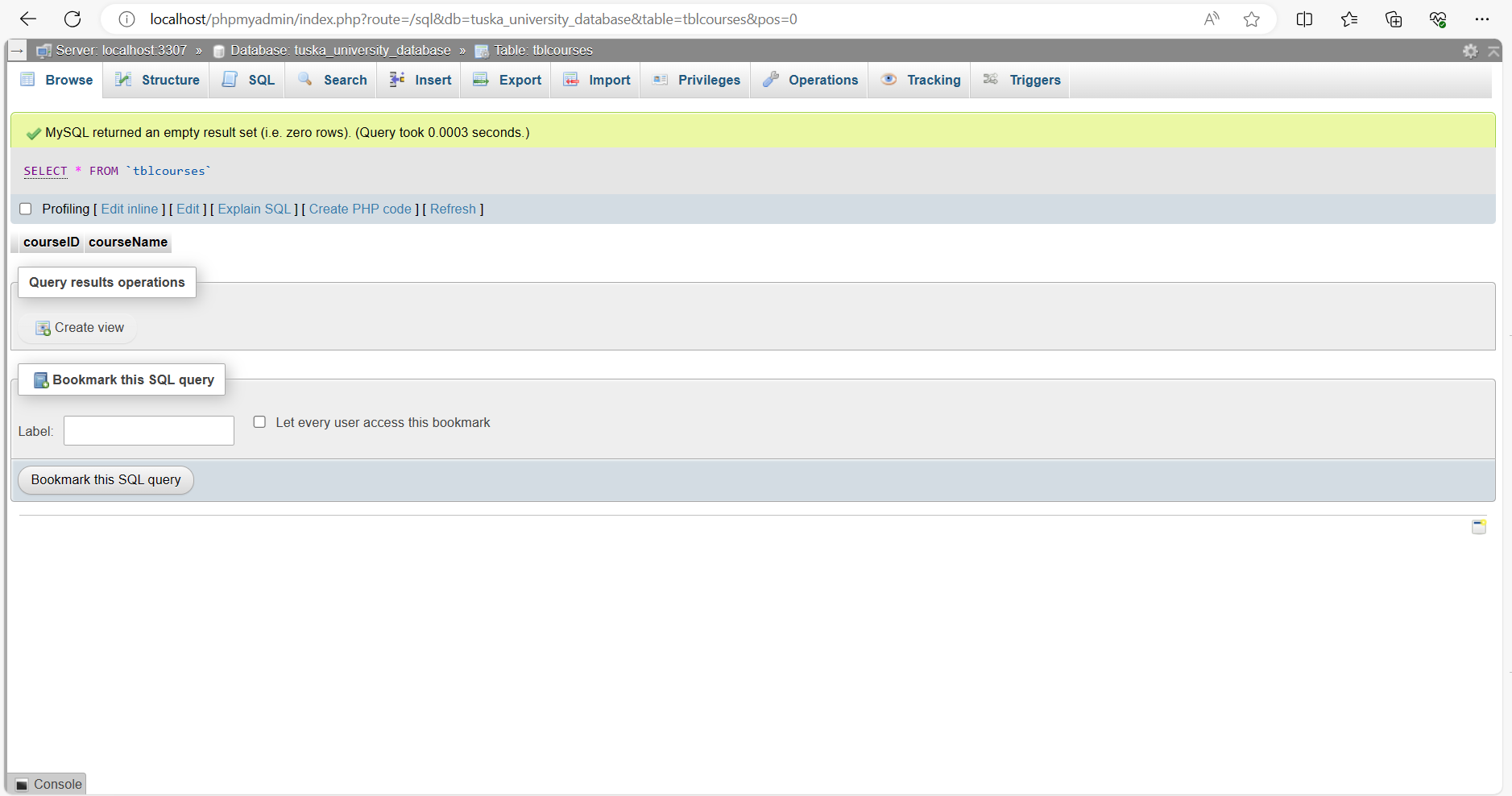
CST 499: Capstone for Computer Software Technology

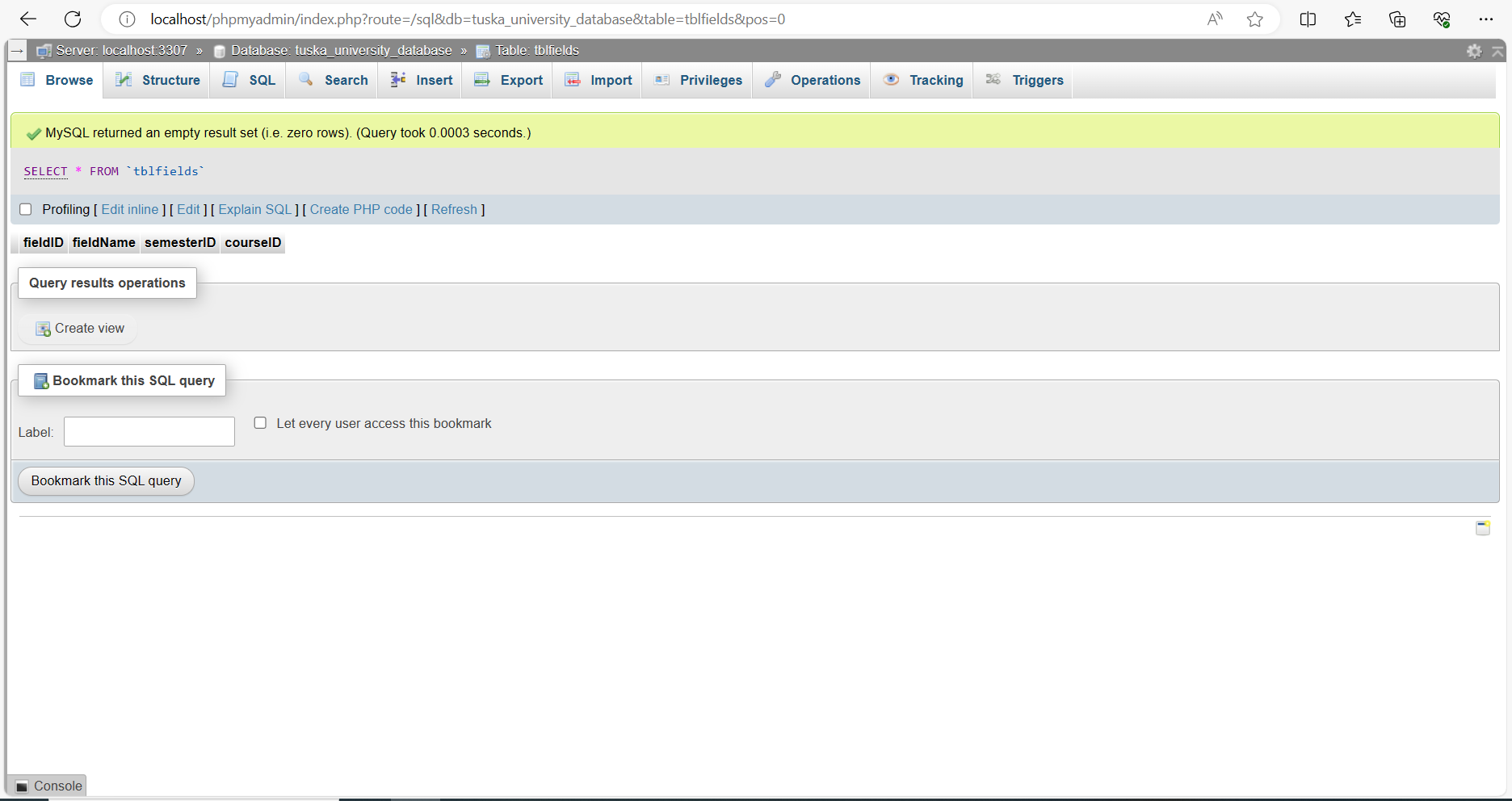
Professor Amjad Alkilani

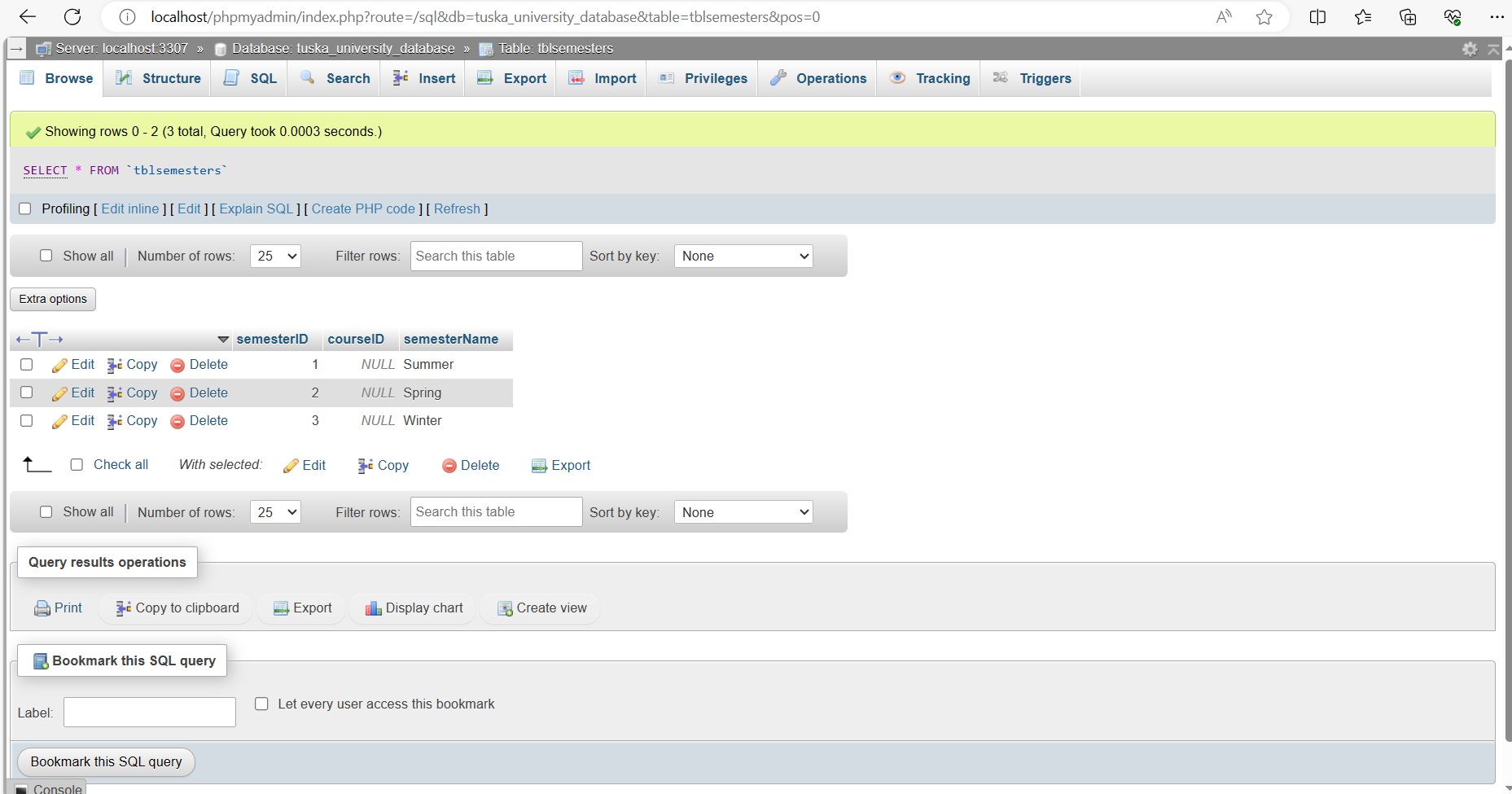
January 22, 2024

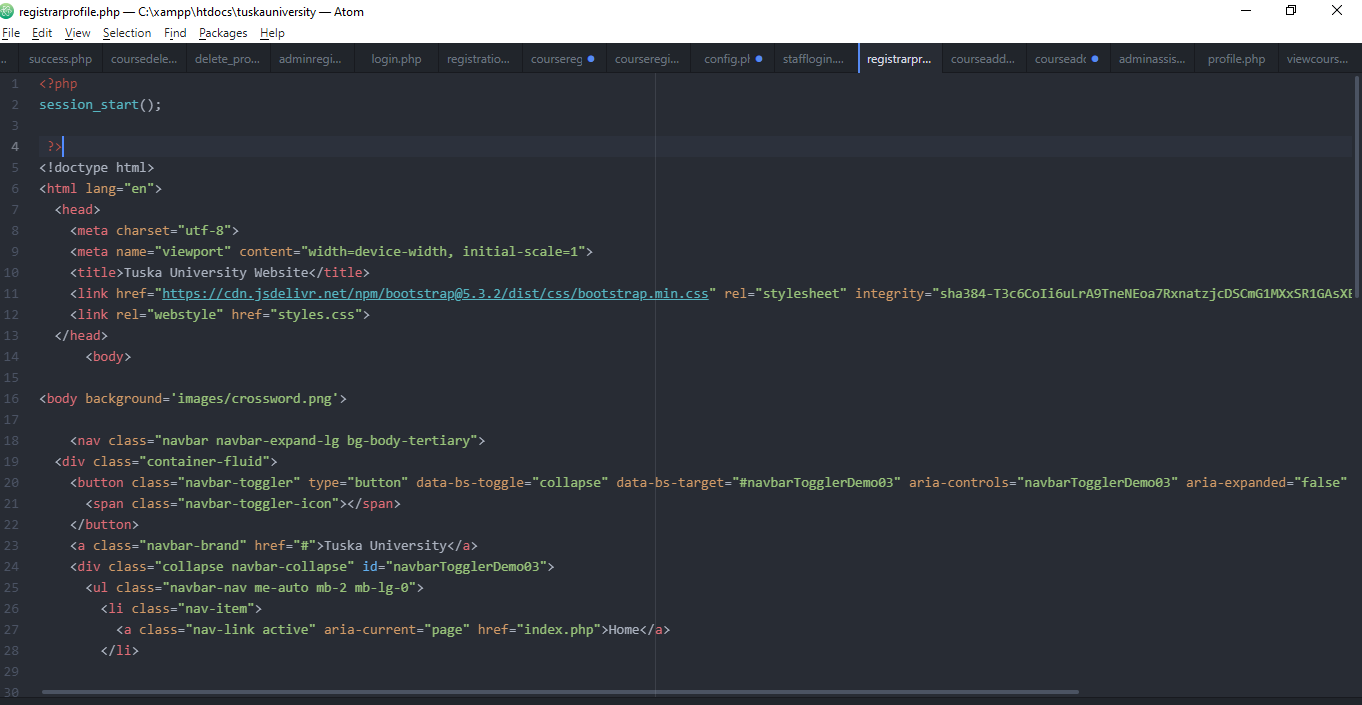
This week’s assignment is a continuation for the implementation phase that I have started during week 3. During week three I developed the landing page, user registration page, user login page, and the database tables for the student registration system. At this point, I have all the pieces installed and working together. These pieces include the functional and non-functional requirements that have been elicited during the requirements and analysis stage within this software system’s lifecycle. These requirements consist of a student registration page, registrar administrator profile page, administrator assistant profile page, and student profile page. Each user within my software system will be able to carry out their primary functions which is to register for classes, create classes, and delete classes. Throughout this paper I will state the steps I took to achieve the functionalities along with screenshots of the website to display the dynamic state of each functionality.

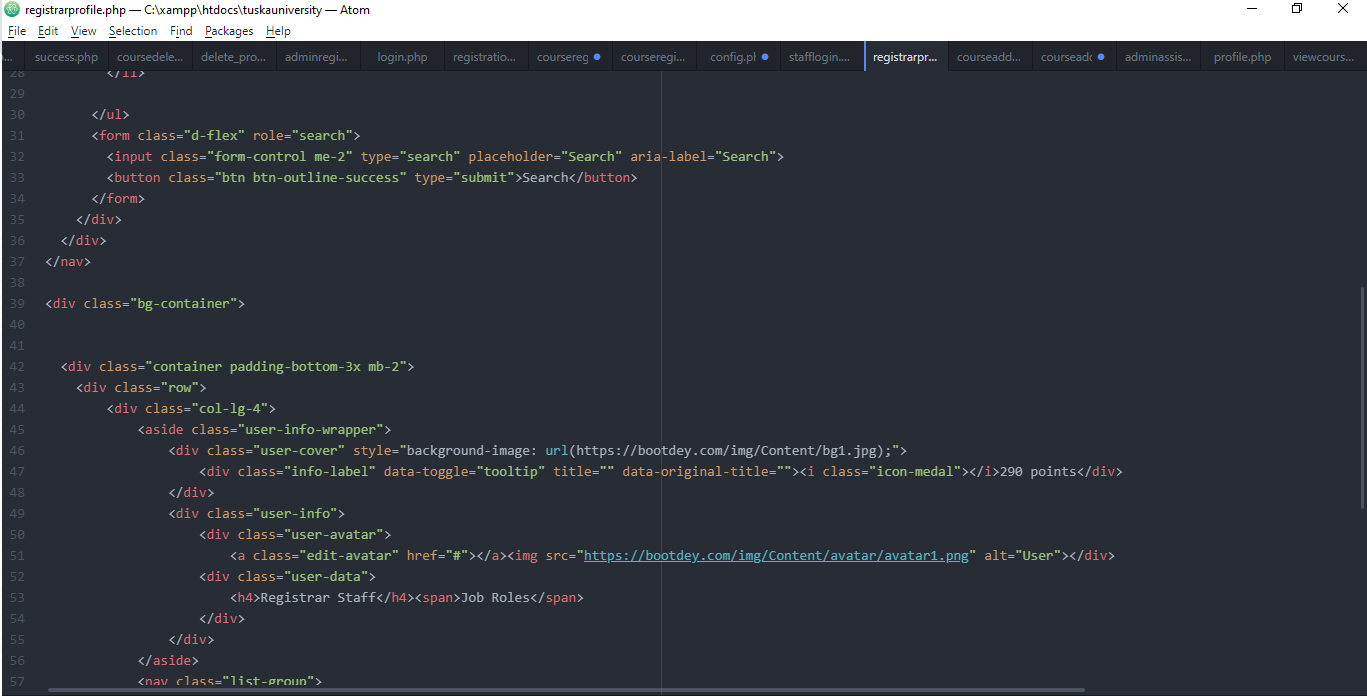
After developing the registration and login pages for both staff and students, I began to develop the profile page for the registrar staff. The Specific roles of the registrar staff is to add courses to the university’s schedule, update courses on the university’s schedule, delete courses against the university's schedule, and view all the registered student’s information. To achieve these functionalities, I began with creating the registrar staff profile account that requires the user to in their username, password, email address, and admin role. Once a registrar staff has successfully registered and logged into their account, they will be redirected to their profile page titled registrarprofile.php. This profile page is a very simplistic design that displays the user, the username, and the registrar staff’s current roles. There are clickable links below the profile picture that allow the admin to carry out their roles. Once the user clicks on the add courses button, they will be redirected to a page titled couseadd.php. This page will allow the user to add classes to the course schedule through another file that I created titled courseaddprocess.php. This page holds the logic to carry out the primary function of adding courses to the database. Within courseadd.php, the admin is presented with a simple form with drop down menus of the semester that have already been manually inputted into the database table titled tblsemesters. After choosing the desired semester they would like to add courses to, they will then type in the field of study they would like to add courses to. After they create the field of study, they will then be able to type the name of the course or courses that they would like to add. After pressing the add course button, the course or courses will be added to the database. Here are examples of the dynamic state of the add course function.

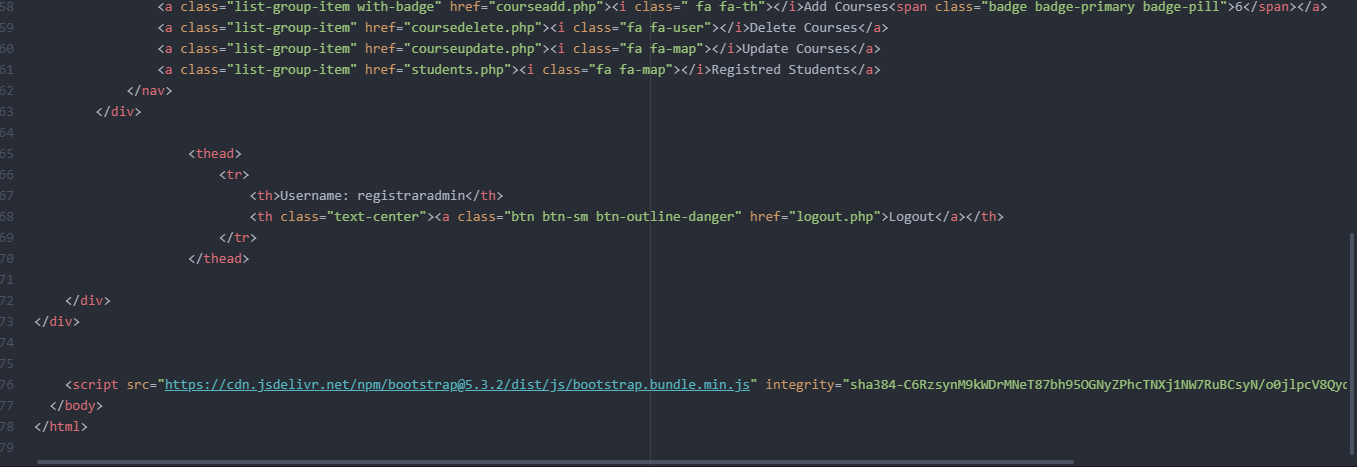
Figure 1. Tblcourses before admin add courses.

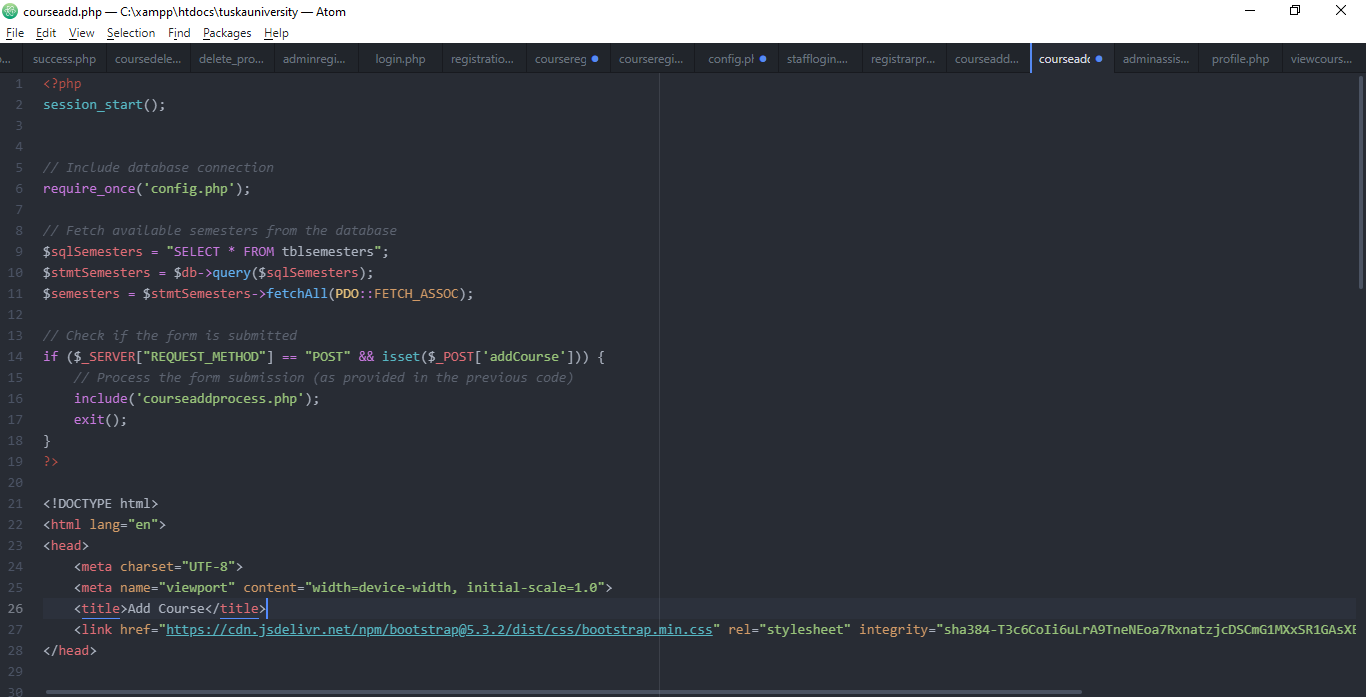
Figure 2. Tblfields database table before registrar adds courses.

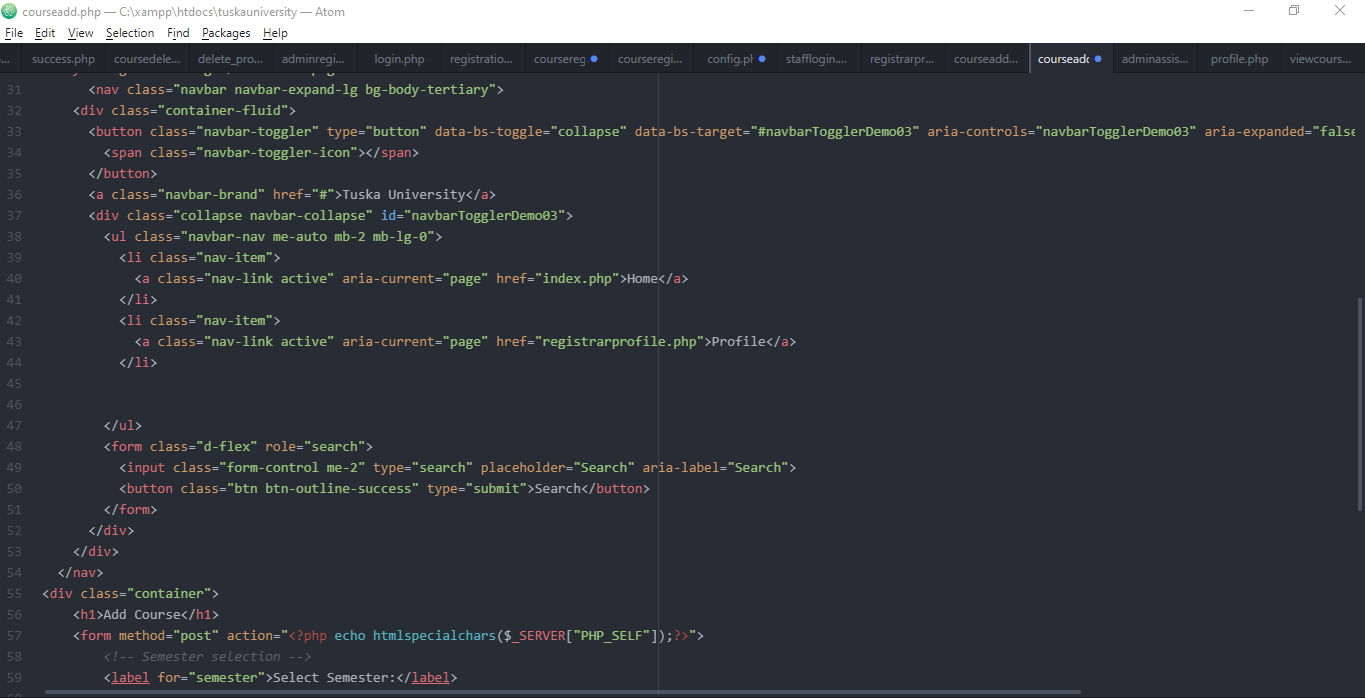
Figure 3. Tblsemester manual input.

Figure 4. Admin Registrarprofile.php file.

Figure 5. Admin Registrarprofile.php cont.

Figure 6. Admin Registrarprofile.php cont.

Figure 7. Admin Courseadd.php file.

Figure 8. Admin Courseadd.php file cont.

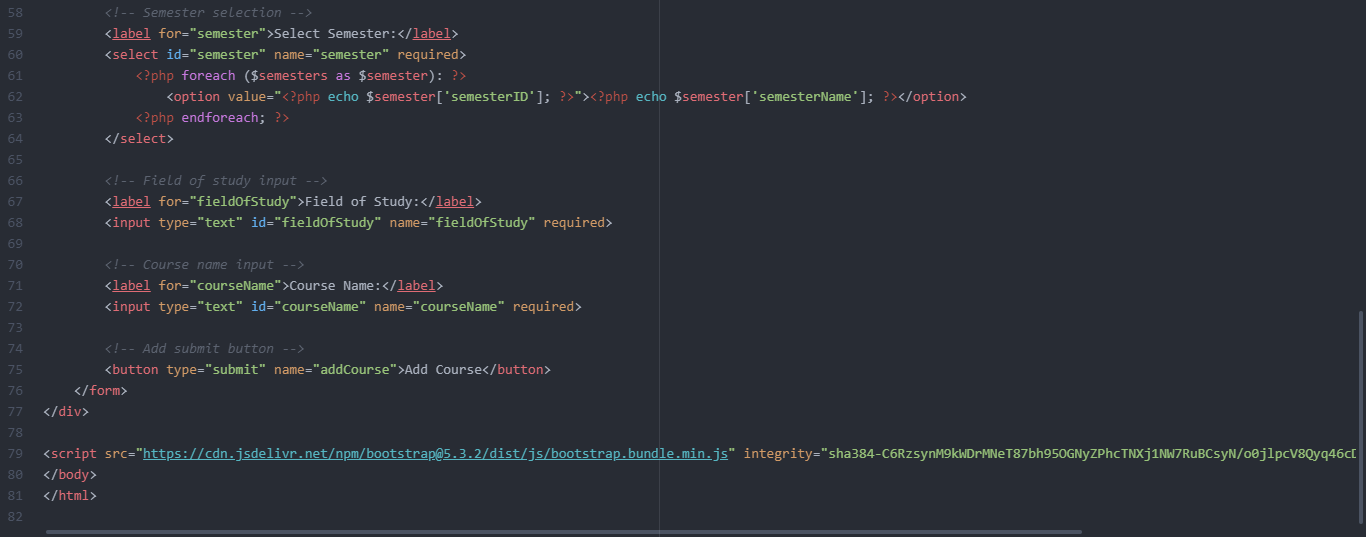
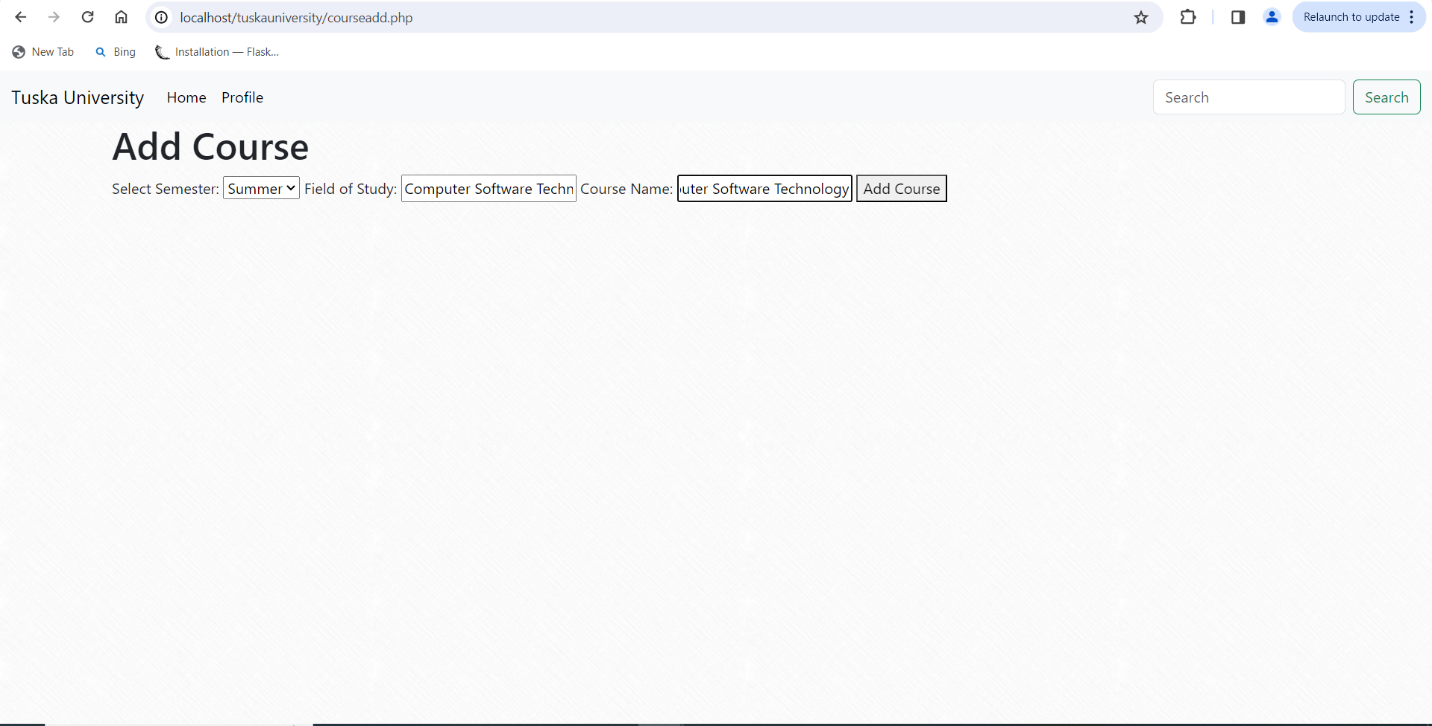
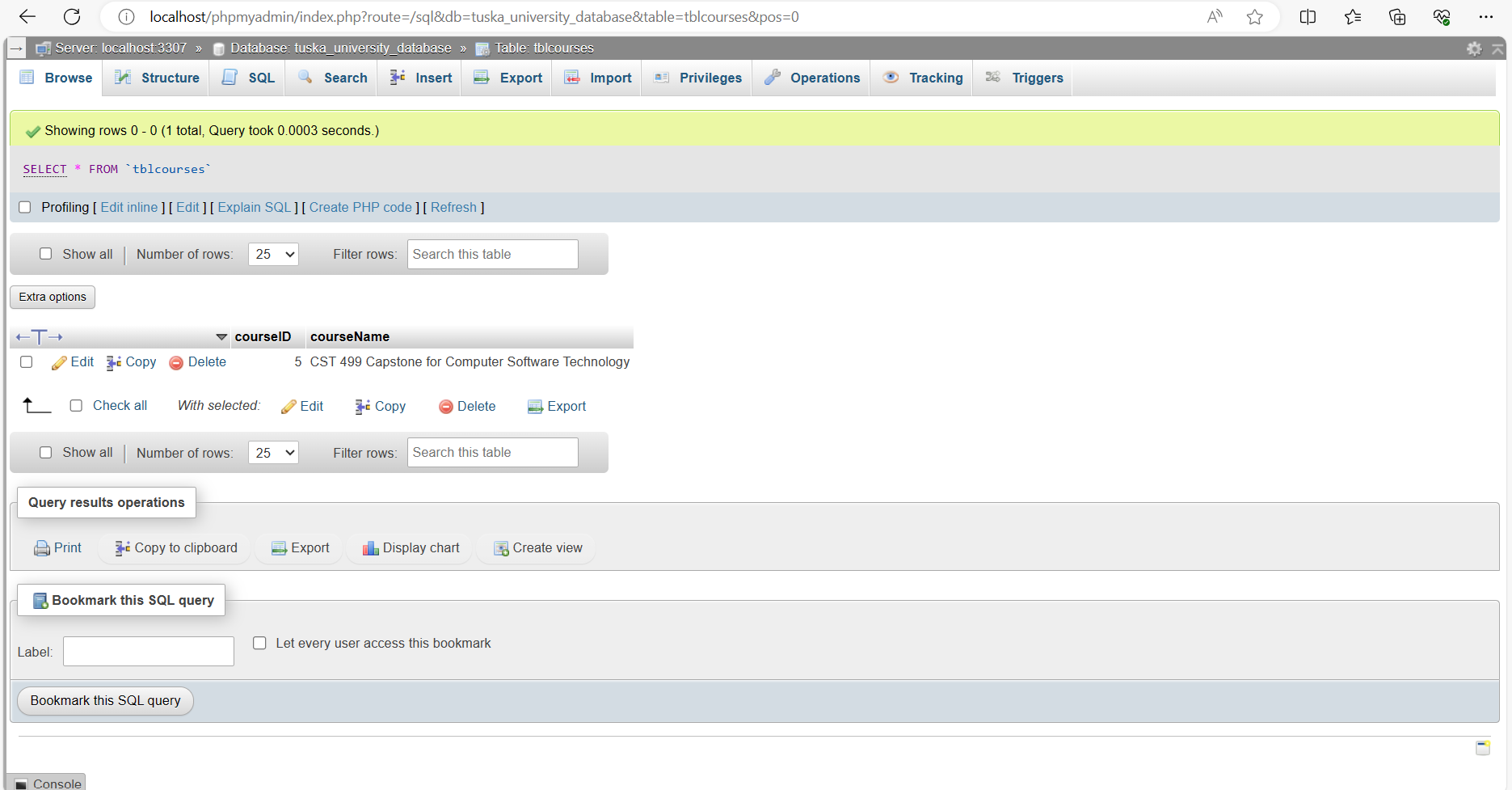
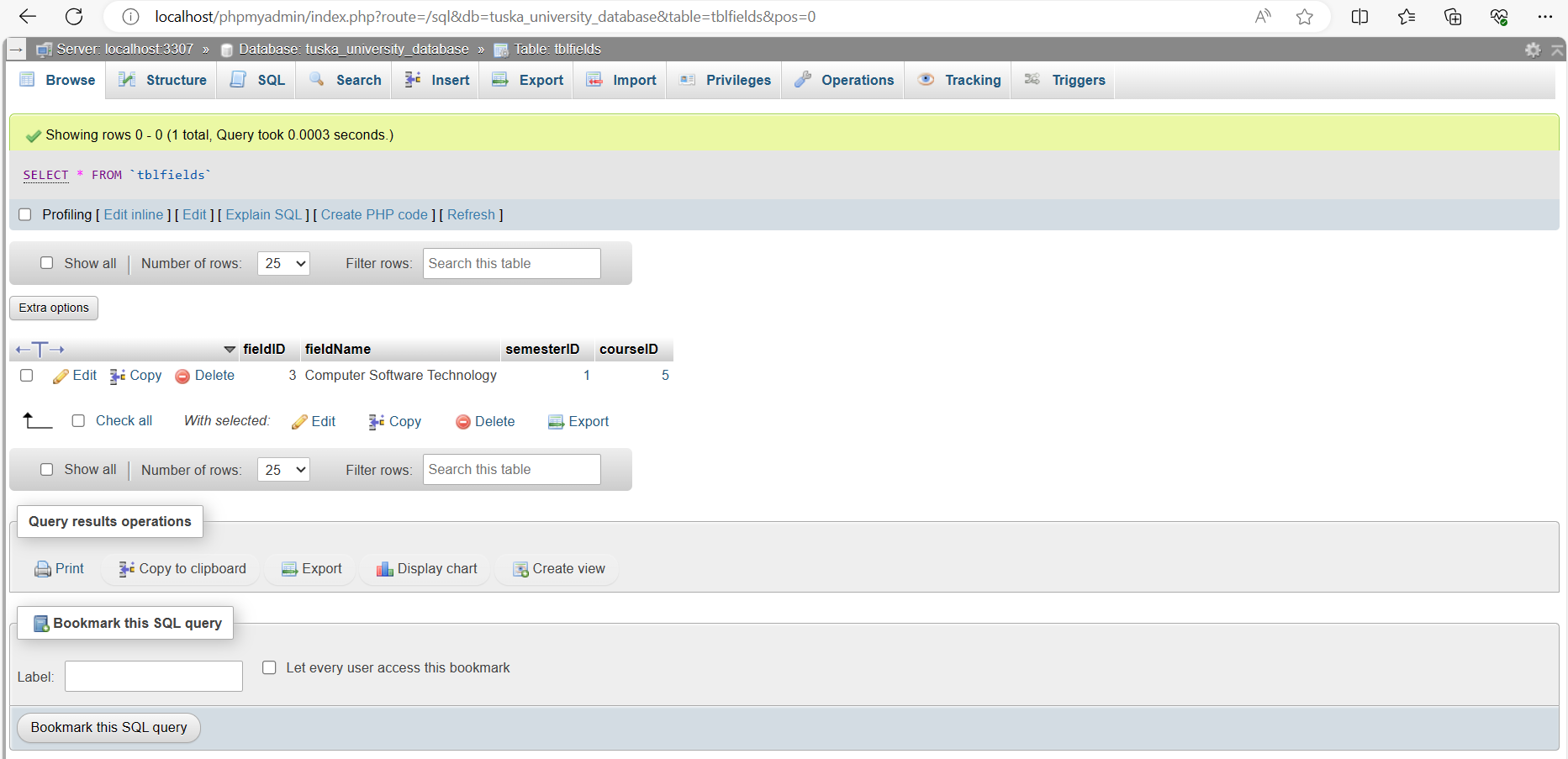
Figure 9. Admin Courseadd.php file cont.

Figure 10. Admin Courseaddprocess file.

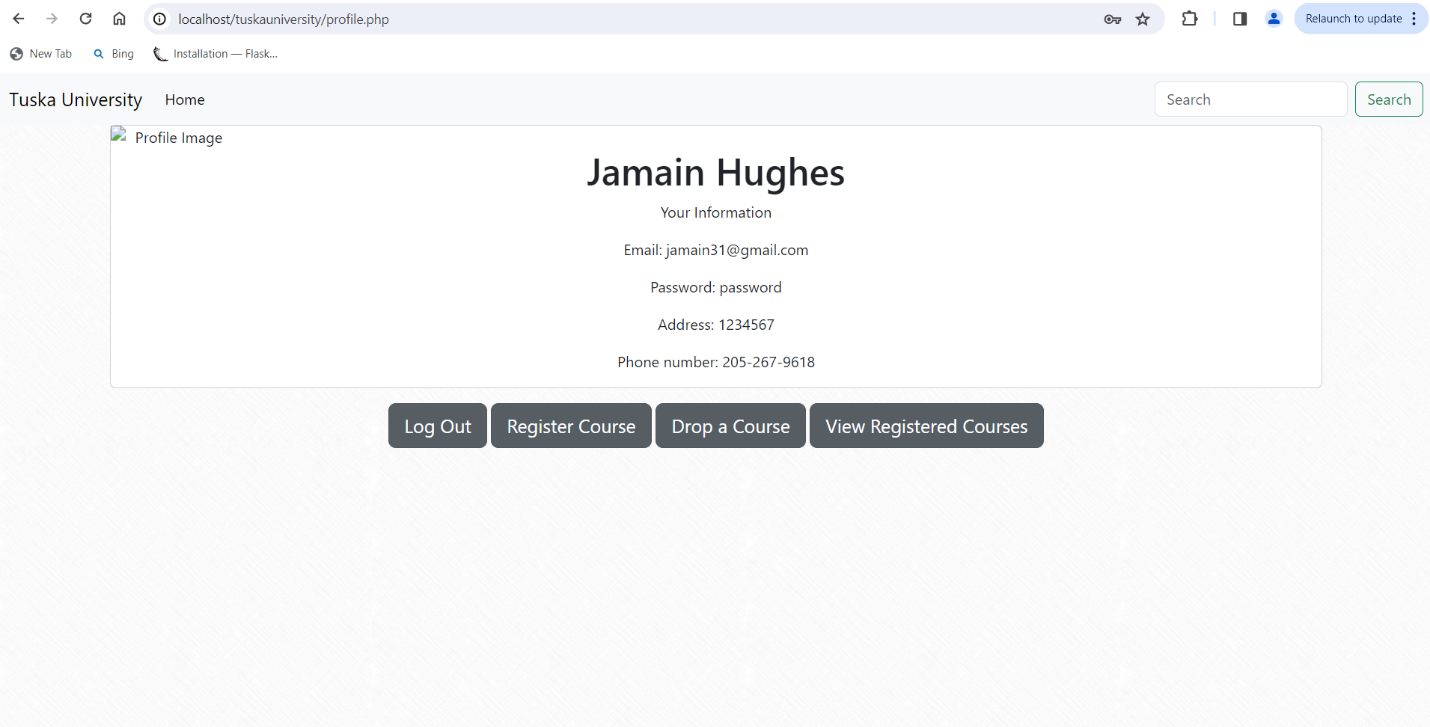
Figure 11. Admin Courseaddprocess.php file cont.

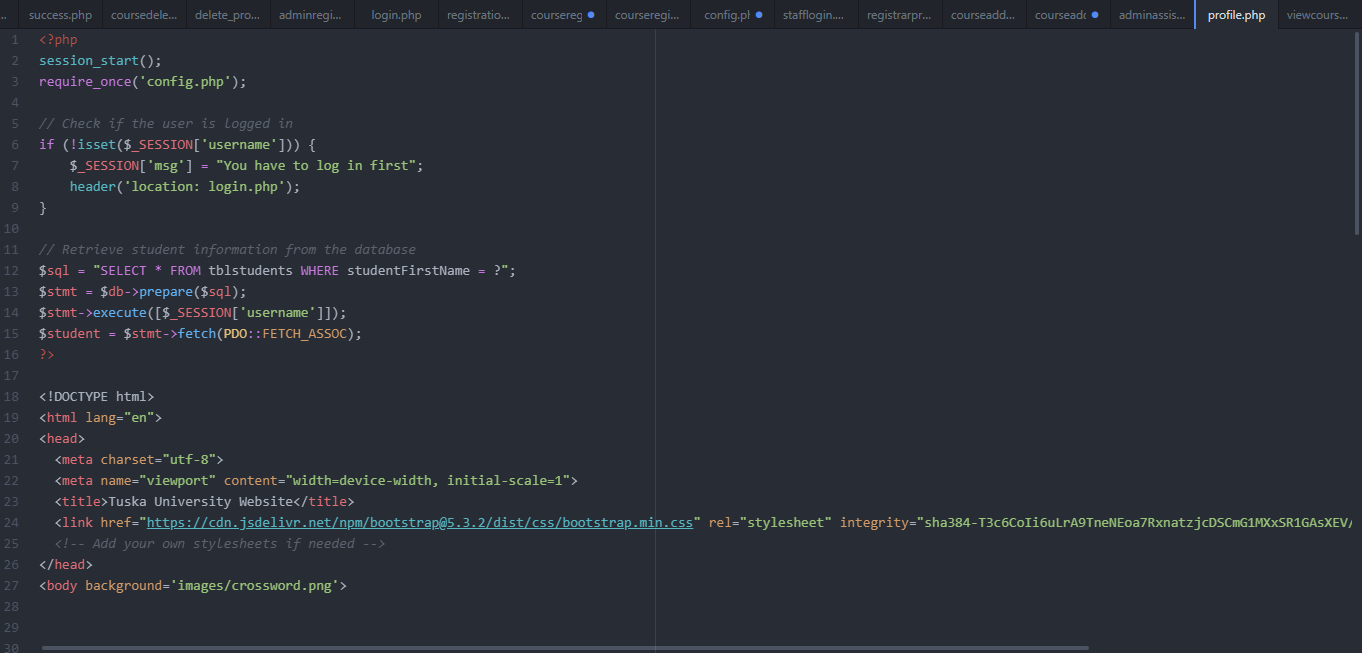
Figure 11. Admin Course add webpage

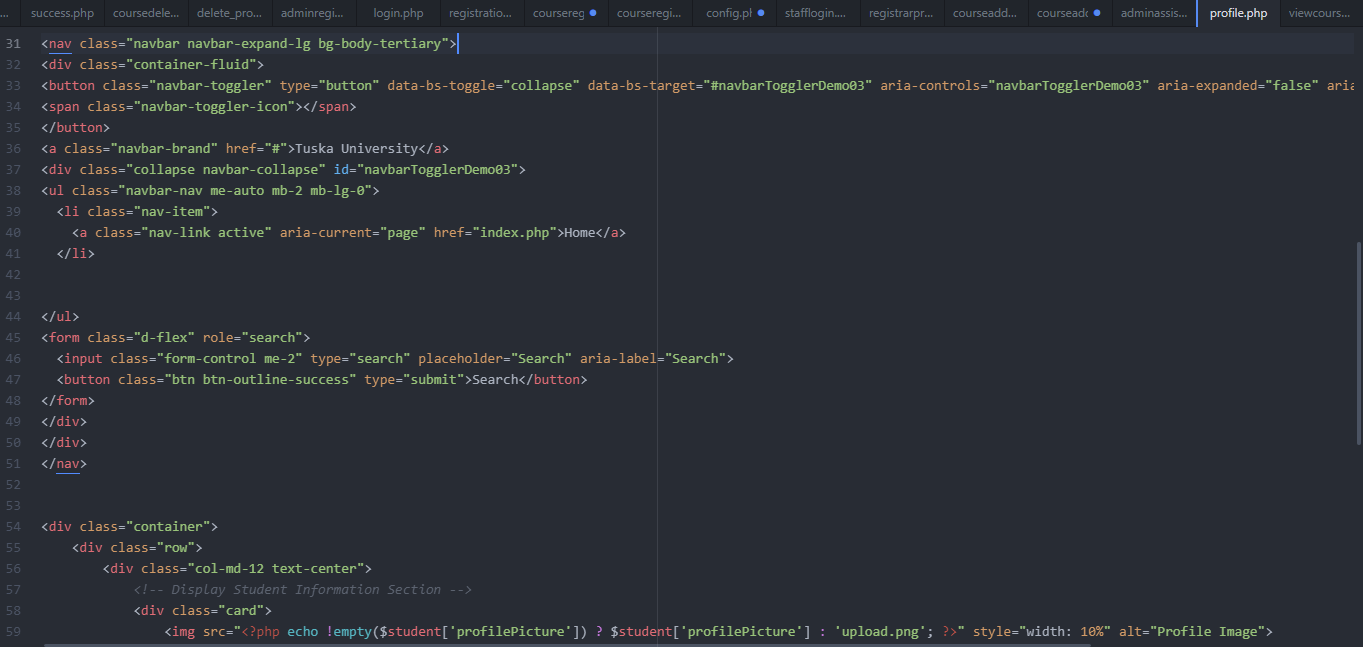
Figure 12. Tblcourses after adding a course to the schedule.

Figure 12. Tblfields after adding a course to the schedule.

After developing the course and add functionality to the website, I developed the student’s profile page. The student’s profile page is titled profile.php. This webpage is a very simplistic design that displays the student’s information. There are also four clickable buttons below the student profile card that allow the student to log out, register for courses, drop courses, and view their registered courses.

Figure 13. Student profile card.

Figure 14. Student profile.php.

Figure 15. Student profile.php cont.

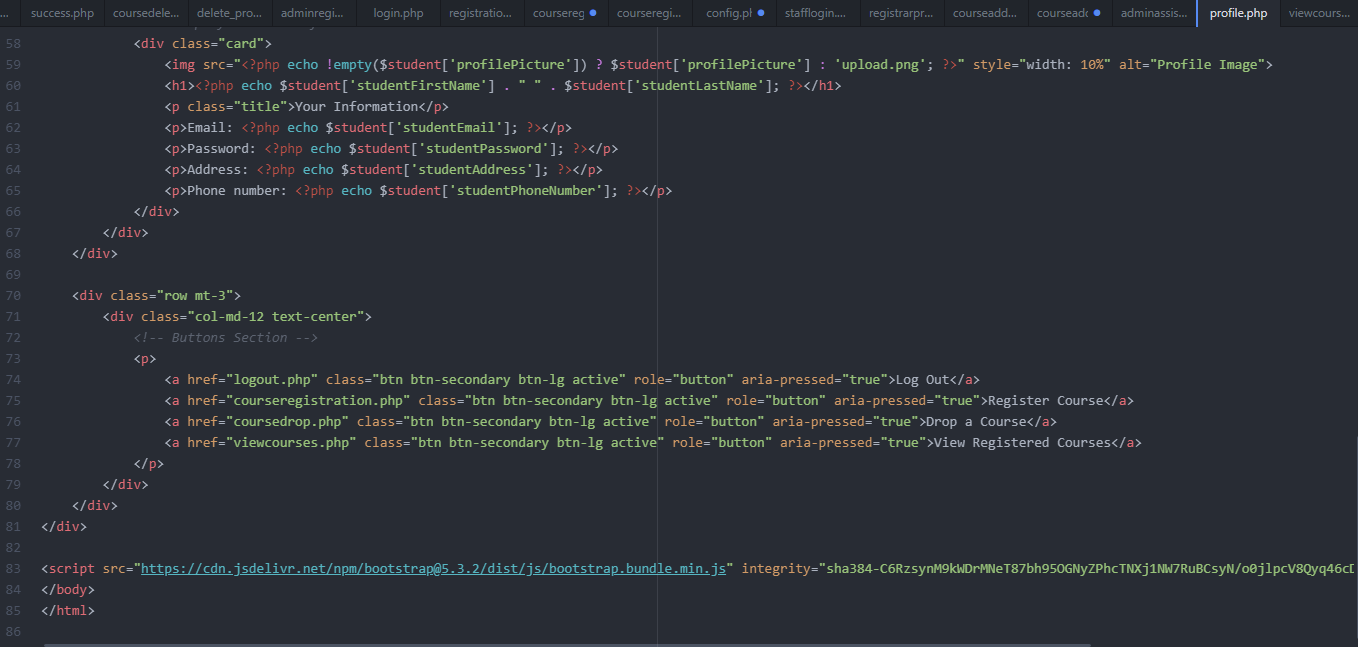


Figure 16. Student profile.php cont.

For a student to register for a class, I created a page titled courseregistration.php. This page is a simple form that has drop down menus that will allow students to select the semester they have applied to, the field of study they have applied to, and the specific course they have registered for. After the user register for a class, the changes will be implemented towards the database.

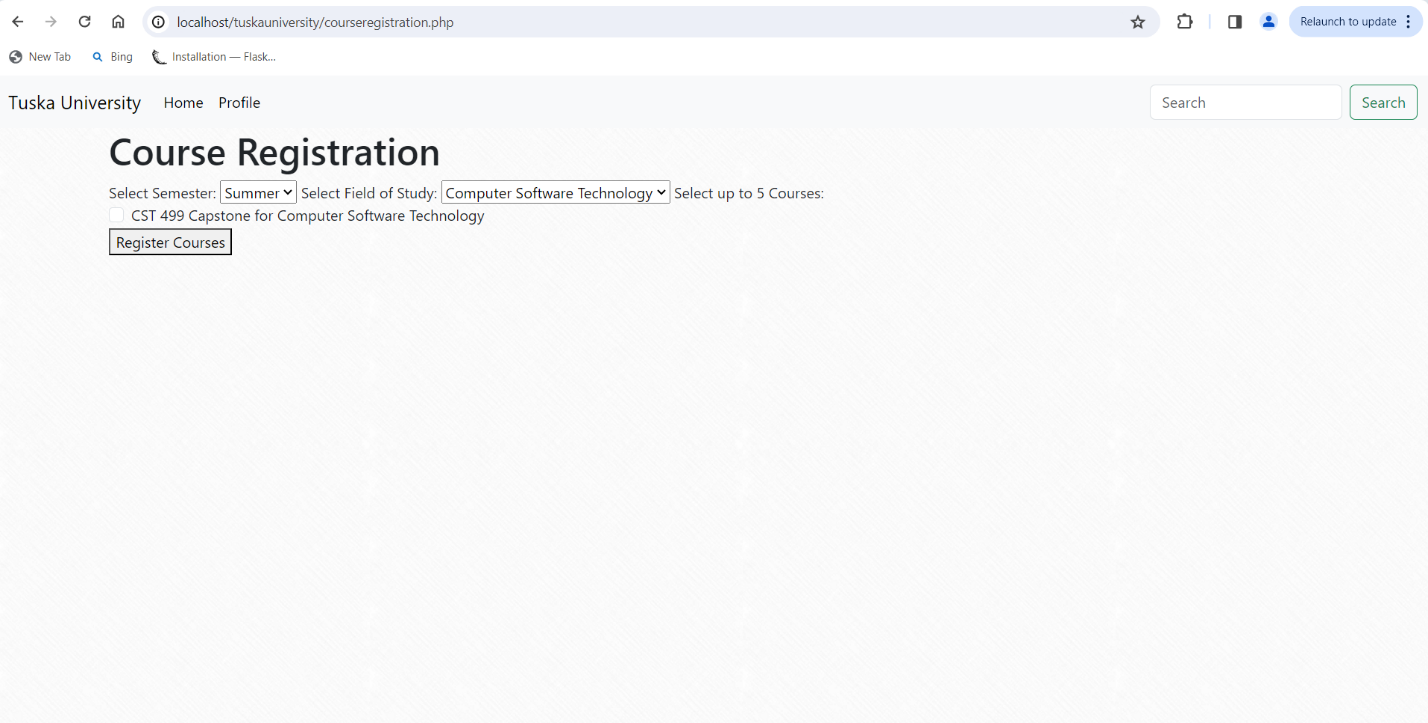
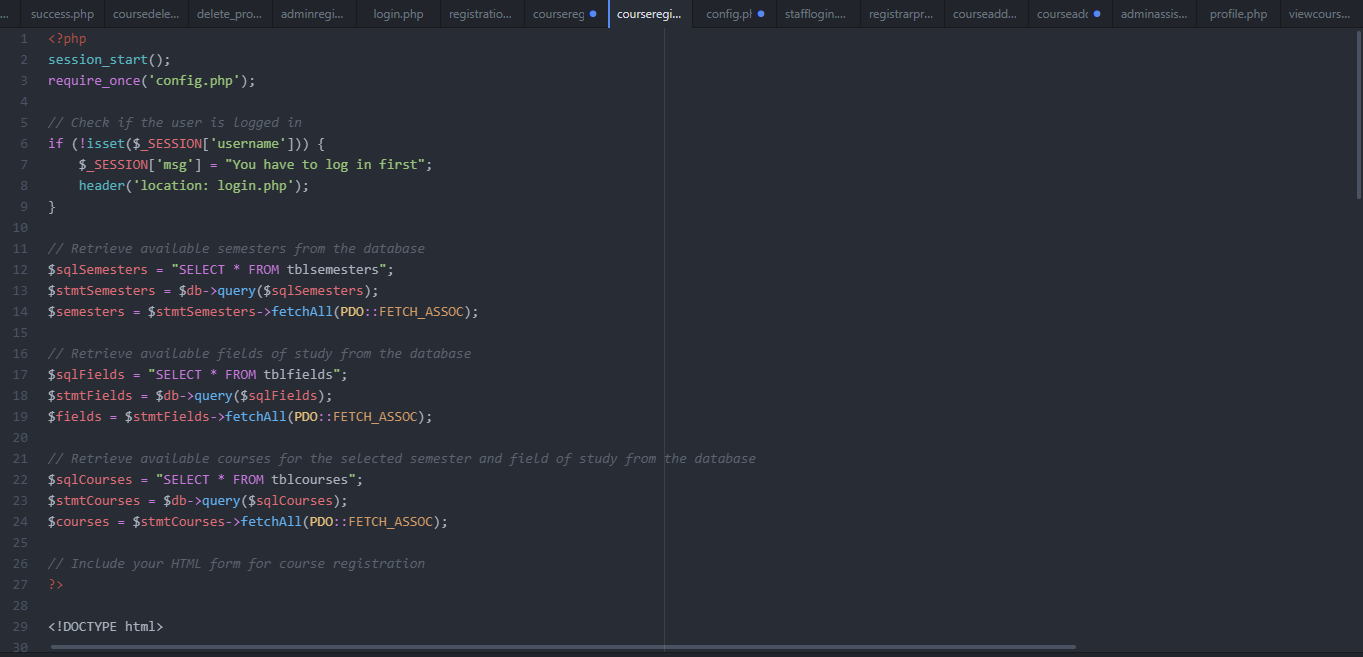
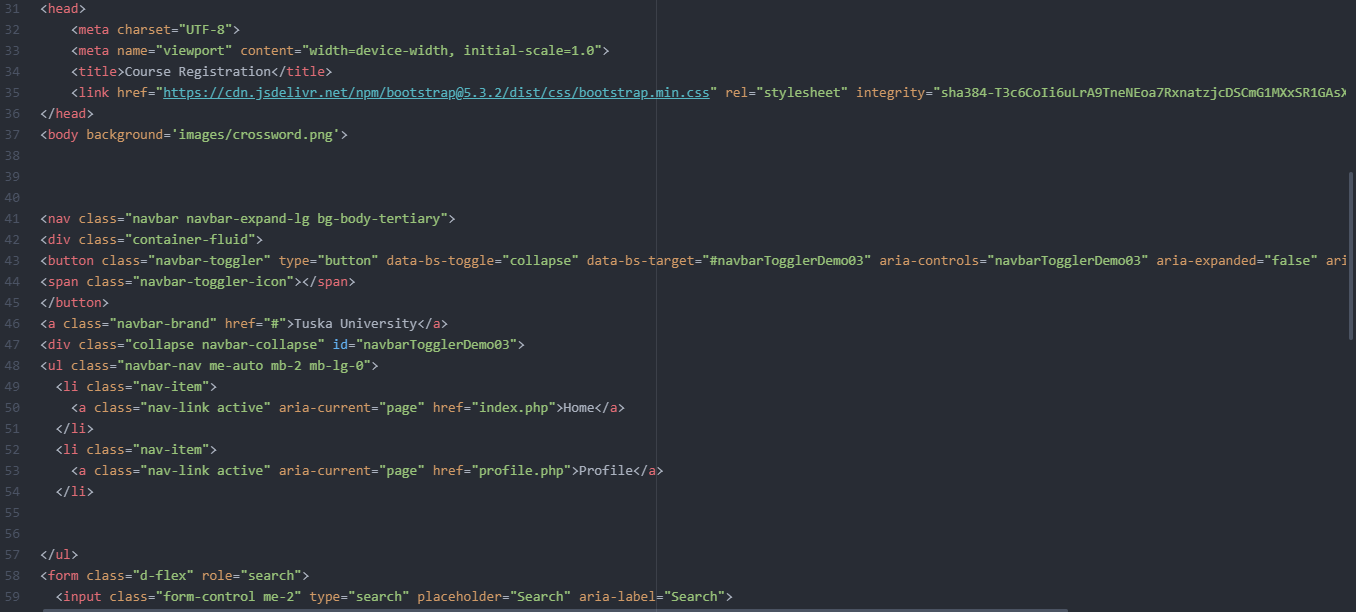
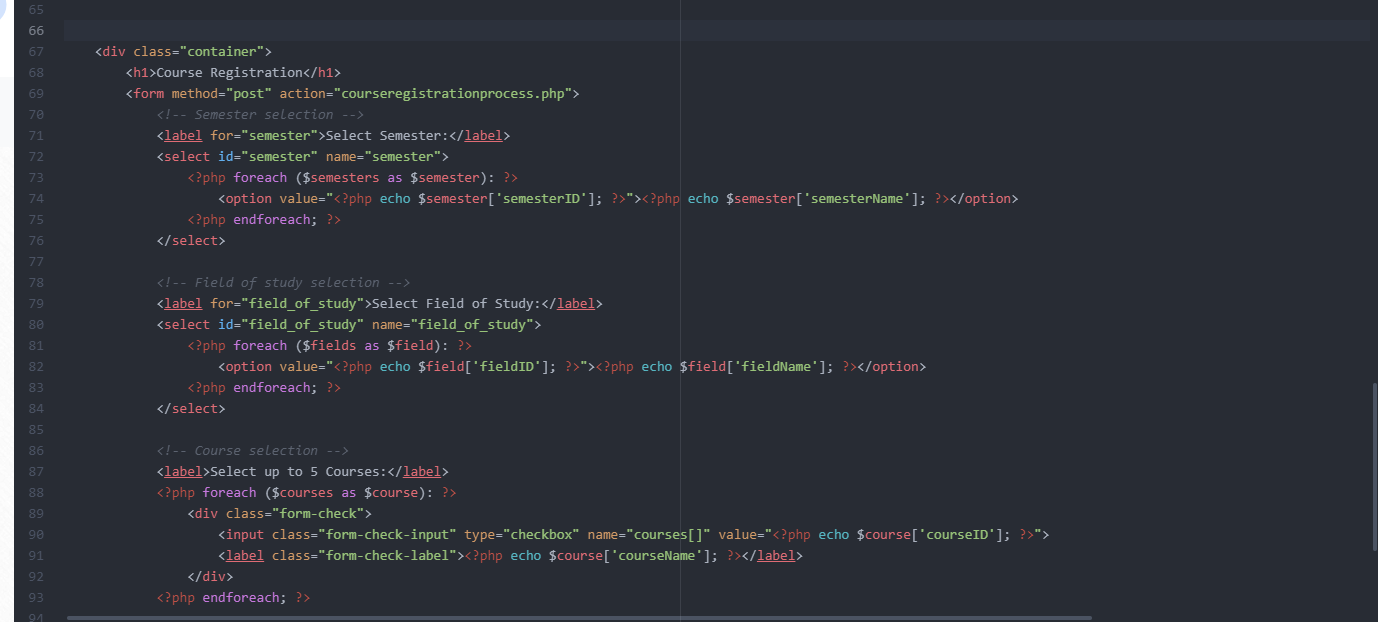


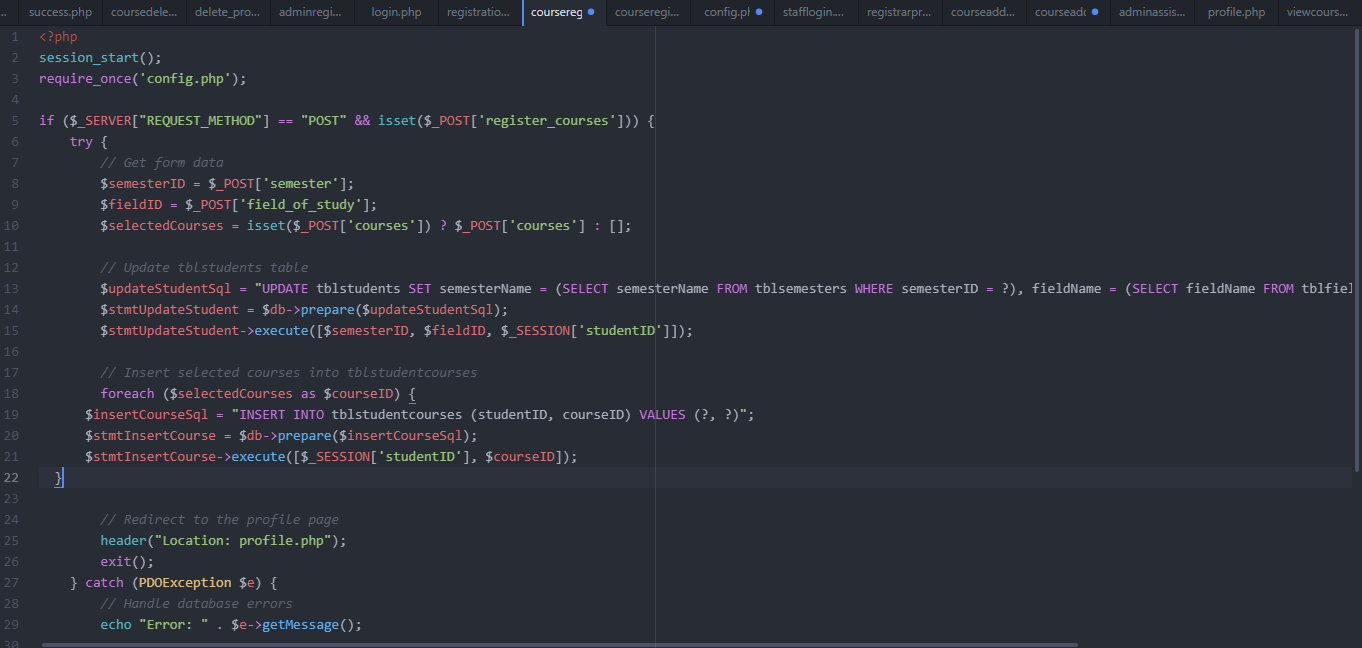
Figure 17. Student course registration webpage.

Figure 18. Student courseregistration.php file.

Figure 19. Student courseregistration.php file cont.

Figure 20. Student courseregistration.php file cont.

To handle the logic of adding a course against the database, I created a file titled courseregistrationprocess.php. This file handles the primary logic of connecting the information to the database.

Figure 21. Student courseregistartionprocess.php file.

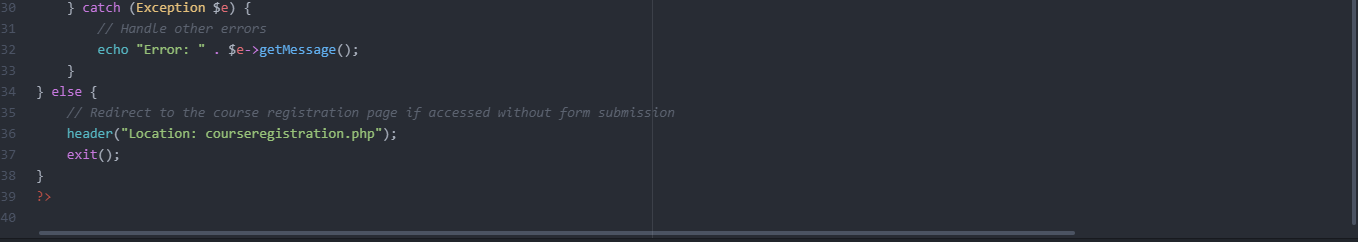
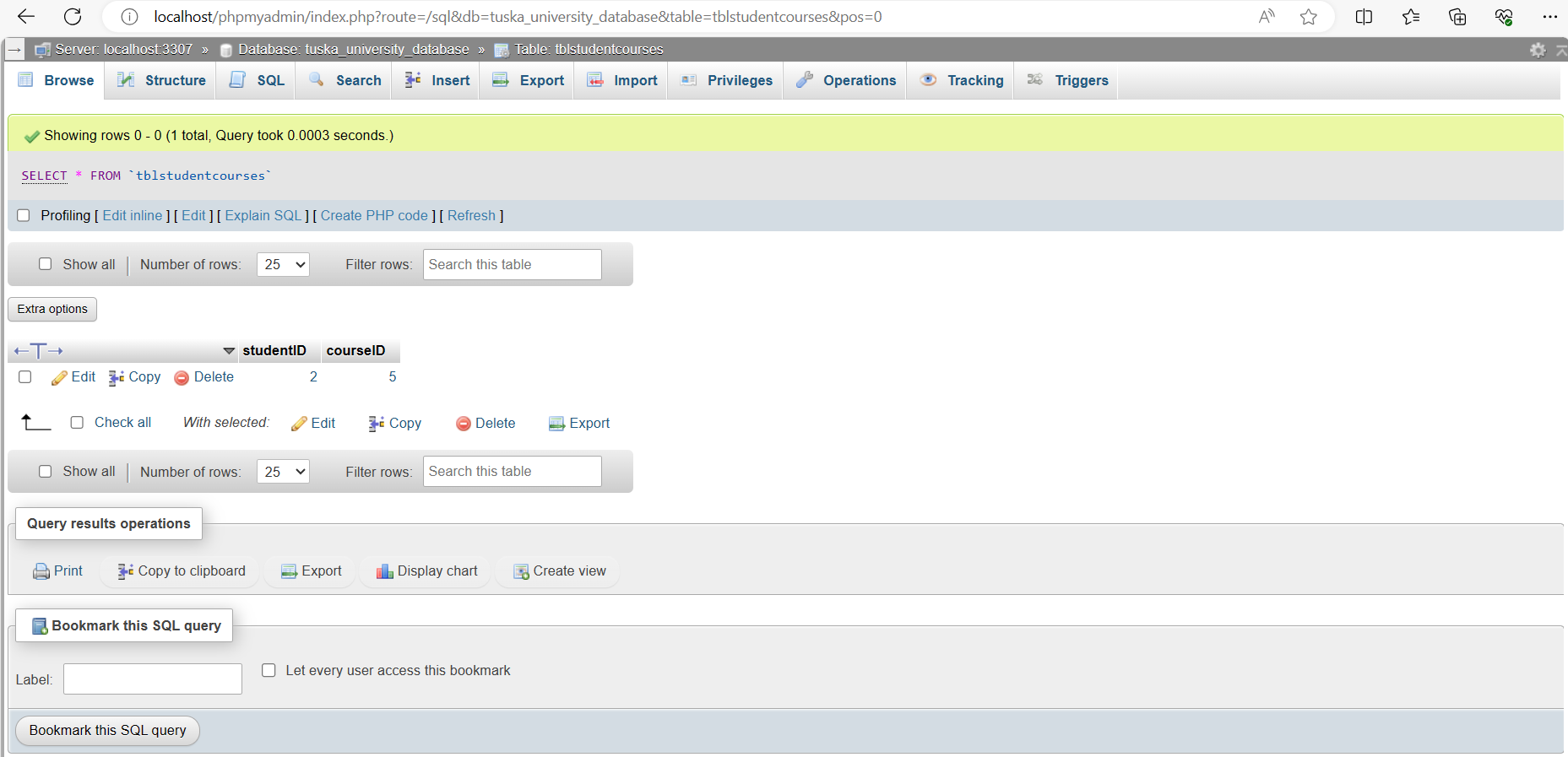
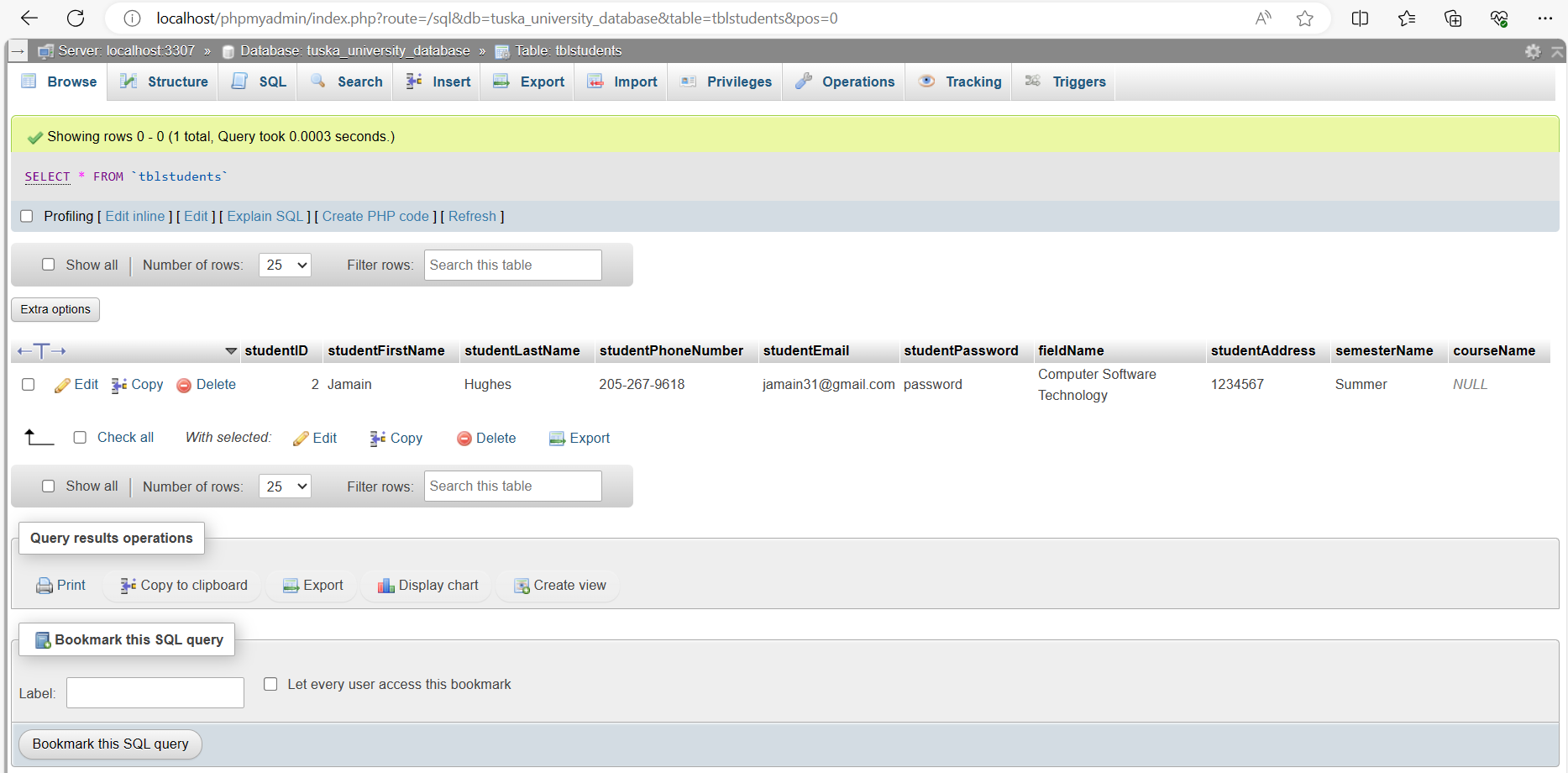
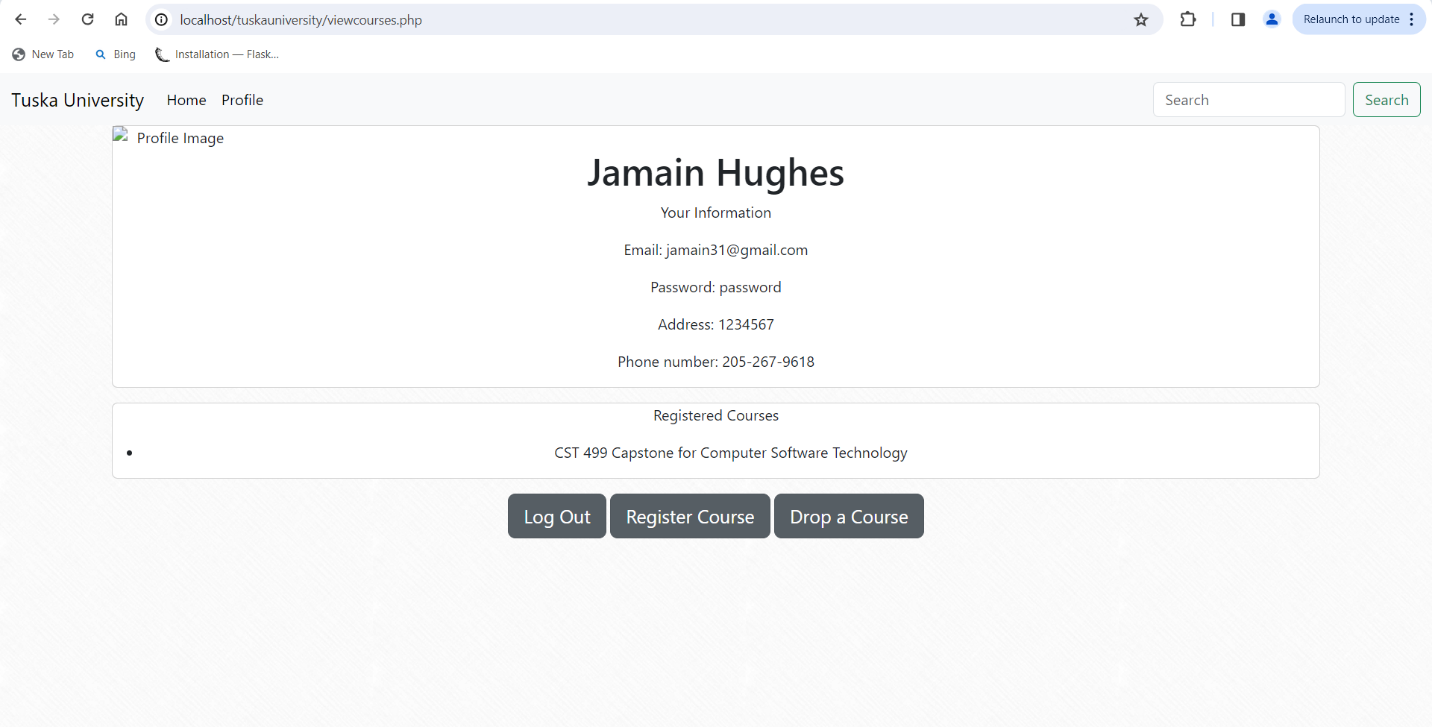


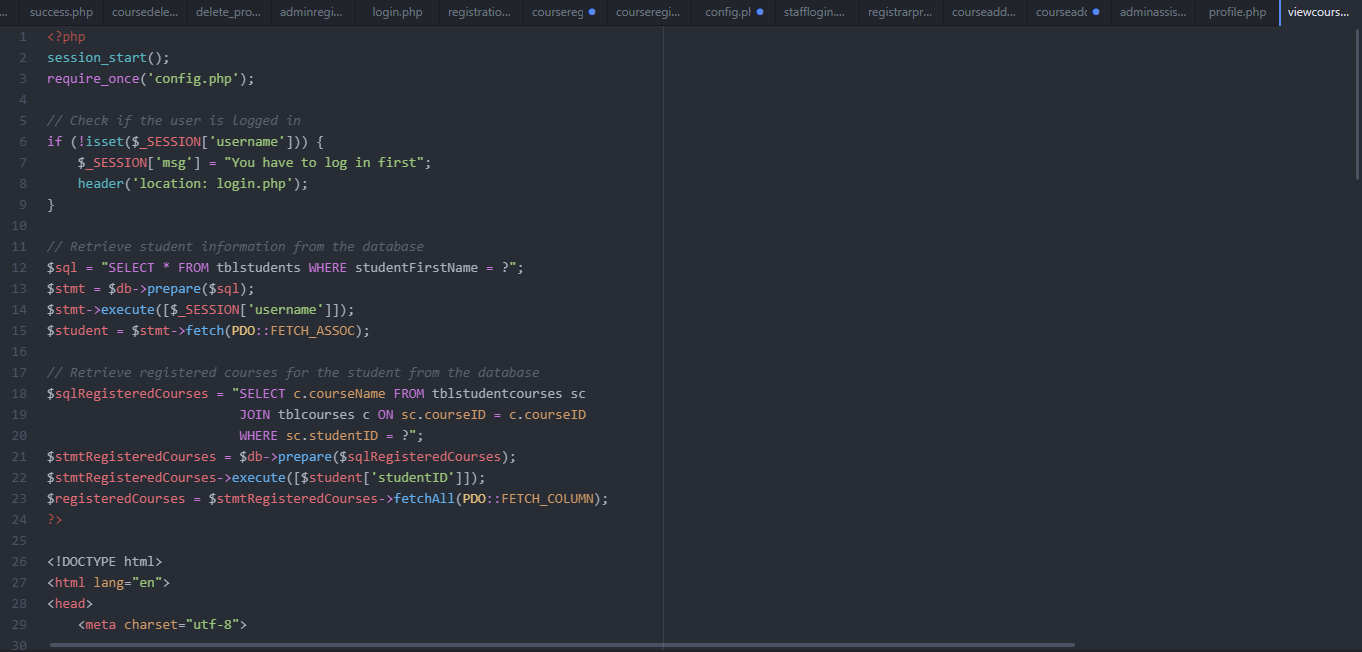
Figure 22. Student courseregistrationprocess.php file cont.

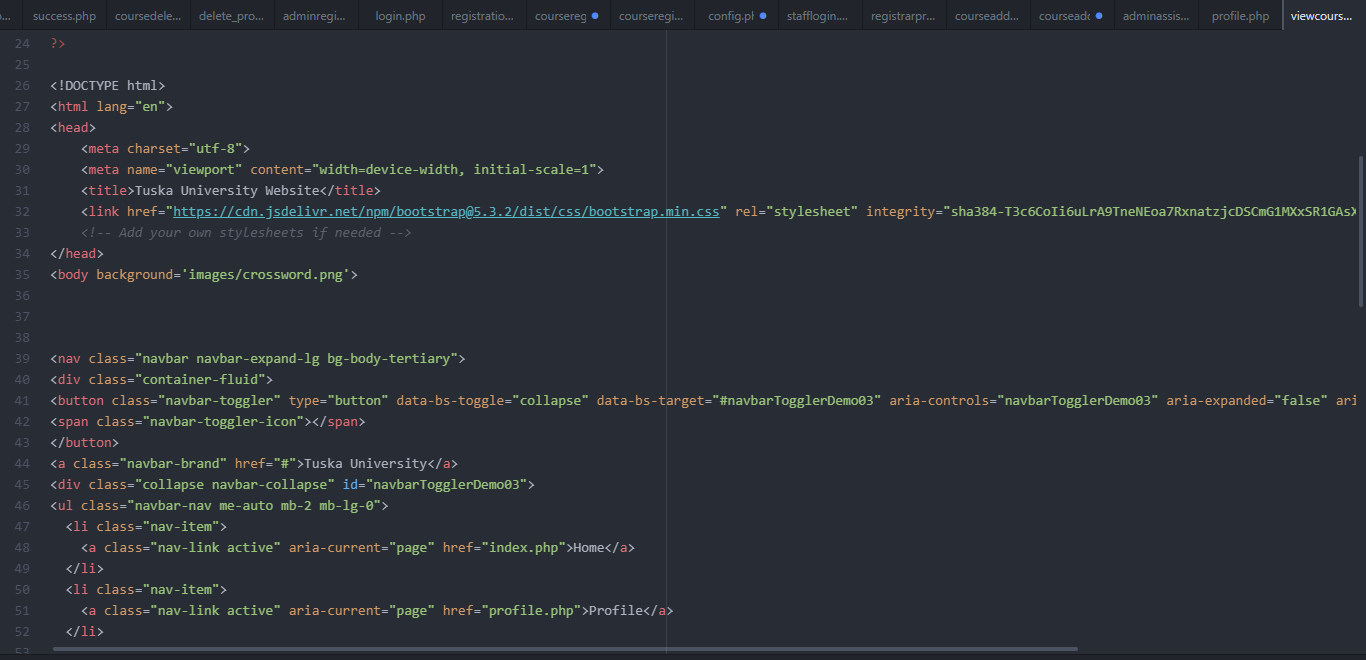
To allow the database to handle the courses that students apply to, I have created a table within my database called tblstudentcourses. This is where the data of the registered courses will be recorded. This data will also be recorded to the tblstudents database table as well.

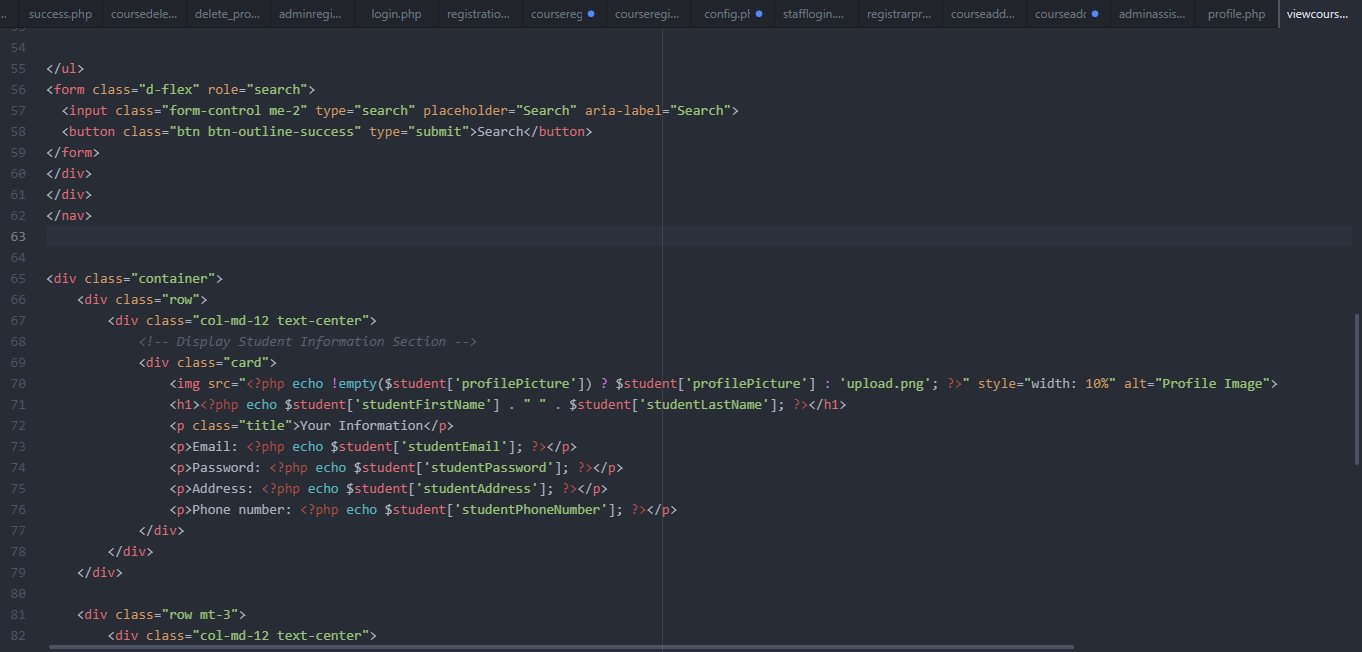
Figure 23. Tblstudentcourses after student register for class.

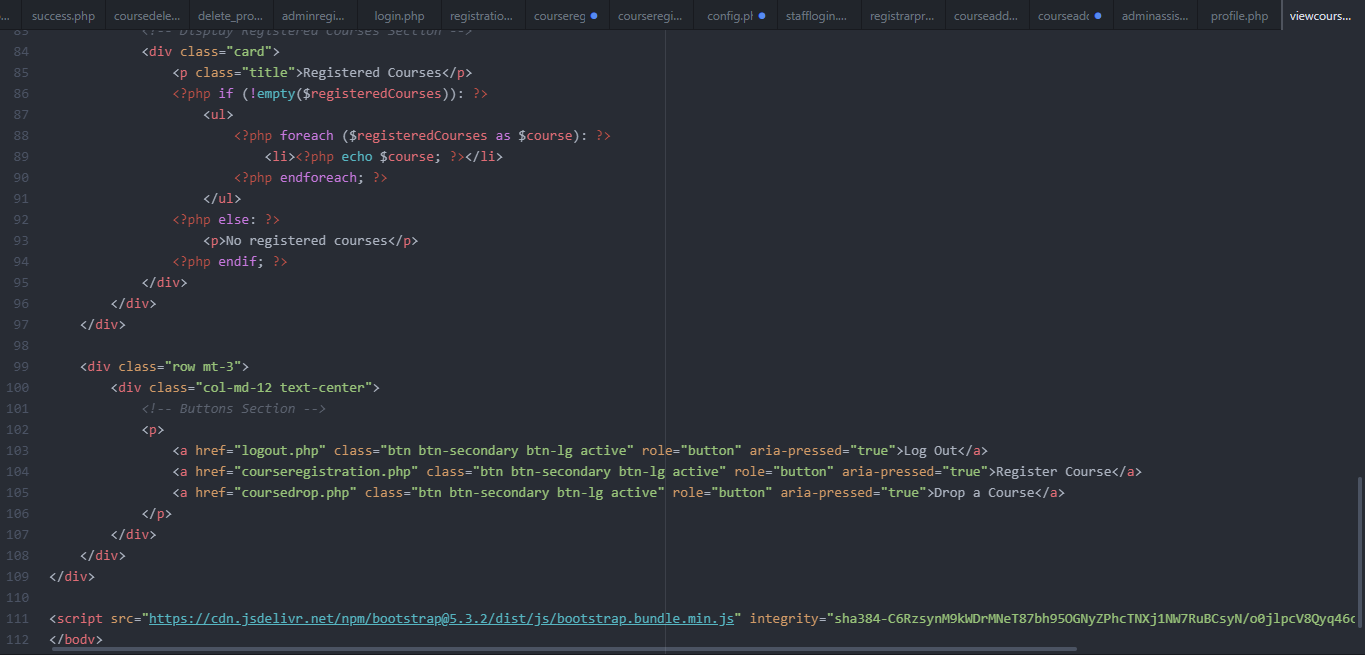
Figure 24. Tblstudents after student register for class.

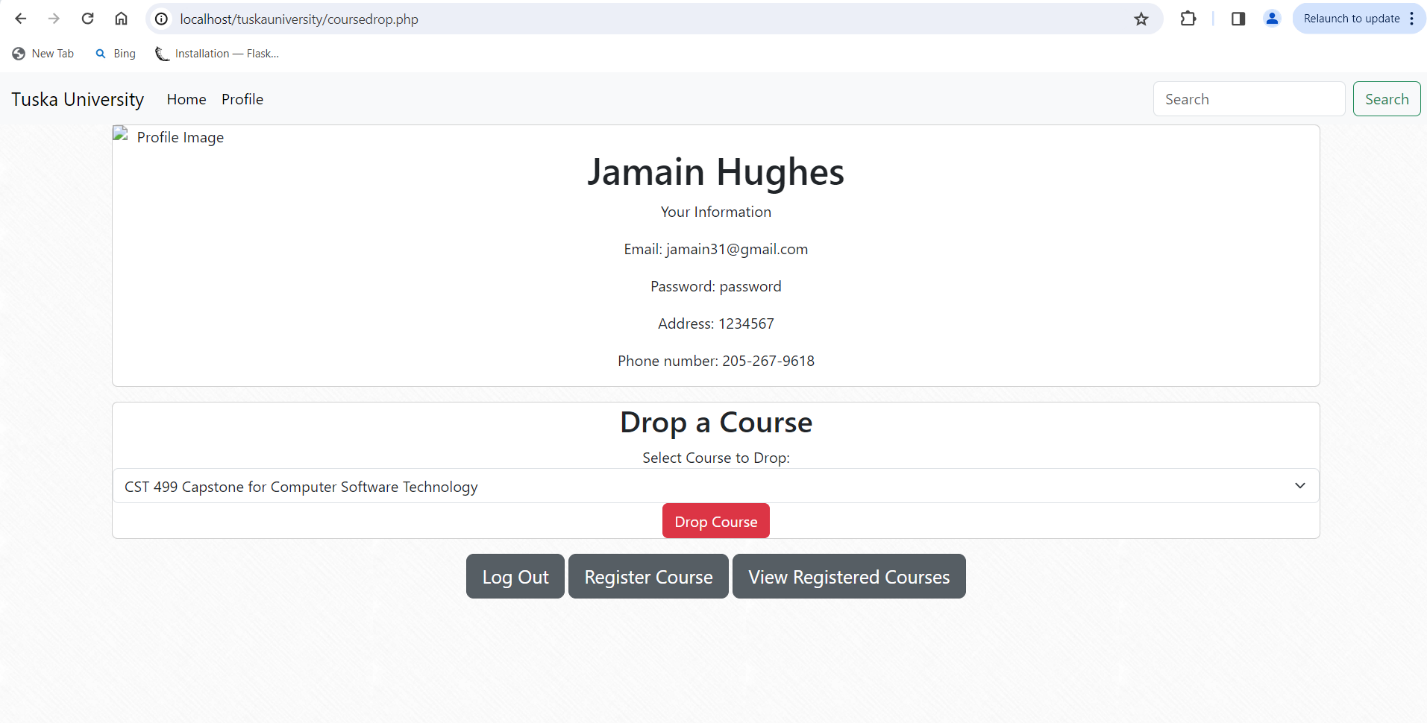
Students will also be able to view their current registered courses on their profile. Once they click on the registered courses button they will be presented with the information.Figure 25. Student course view webpage.

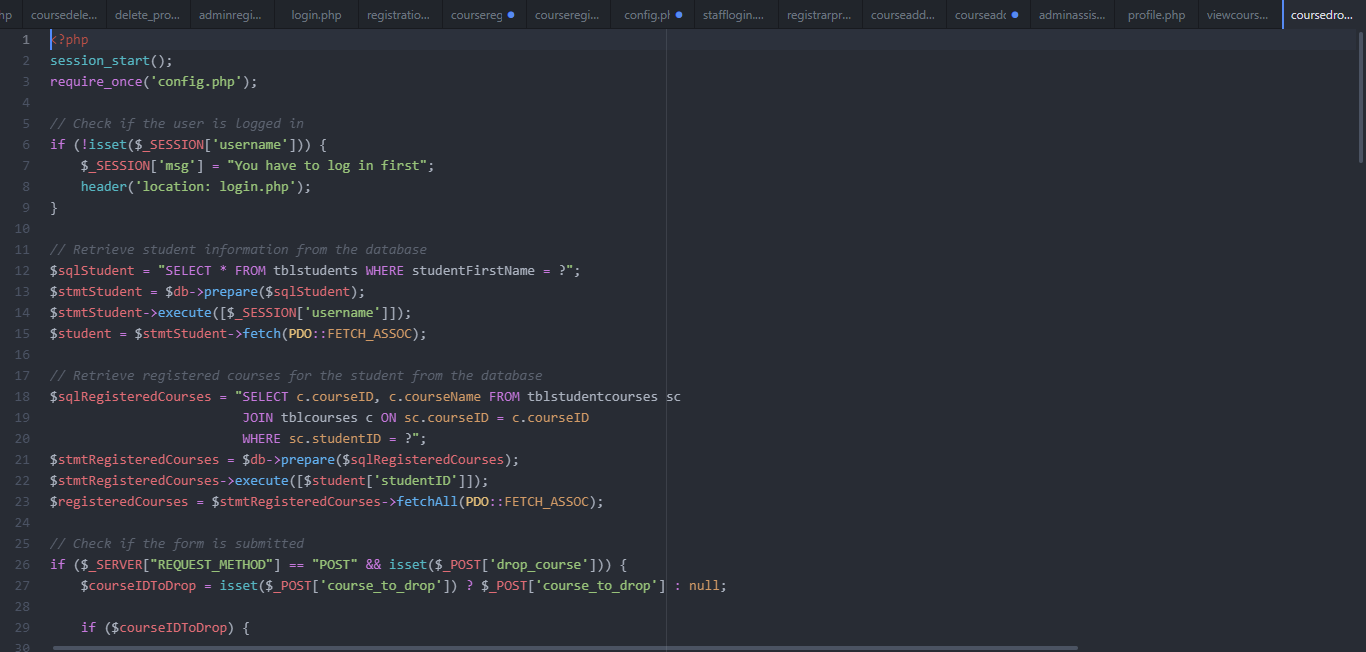
Figure 26. Student viewcourse.php file.

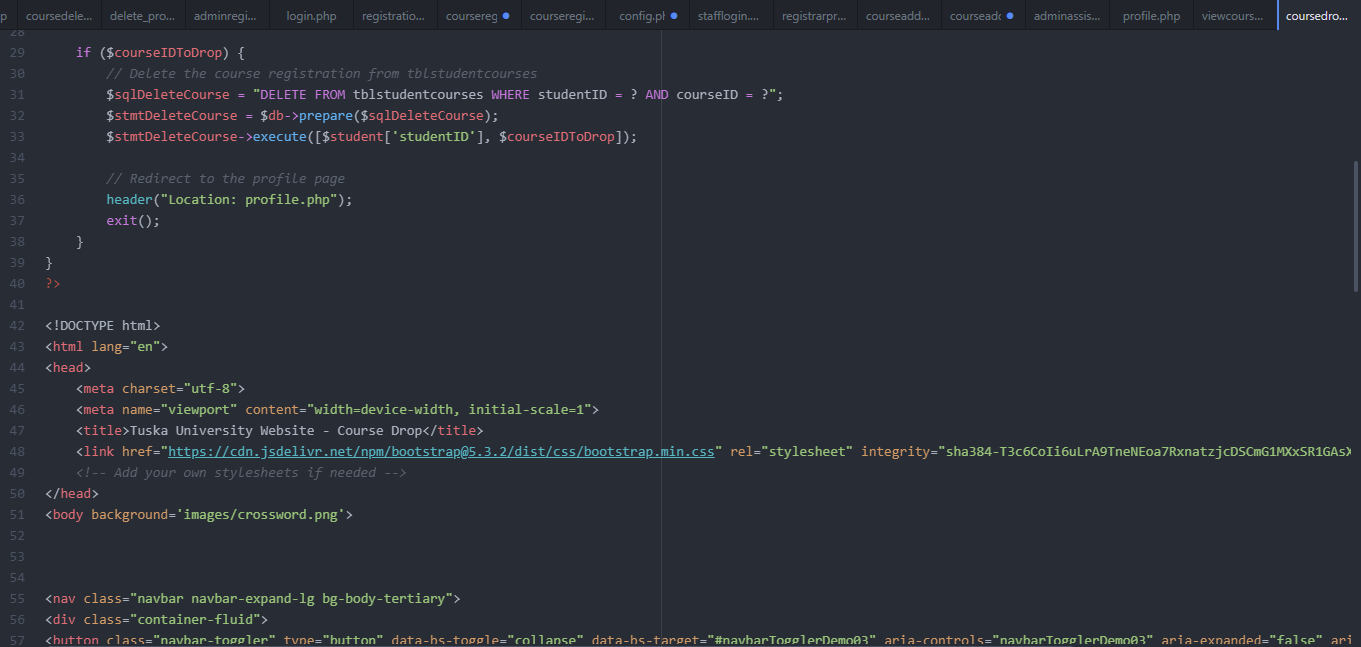
Figure 27. Student viewcourse.php file cont.

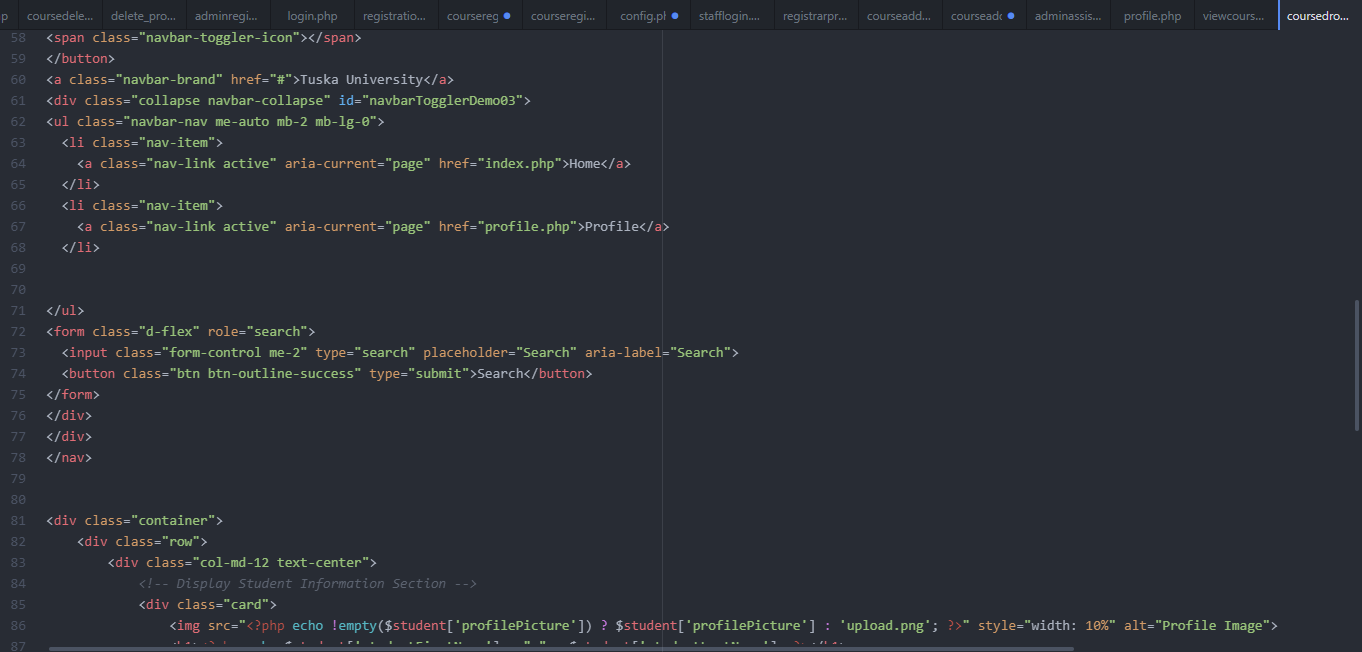
Figure 28. Student viewcourse.php file cont.

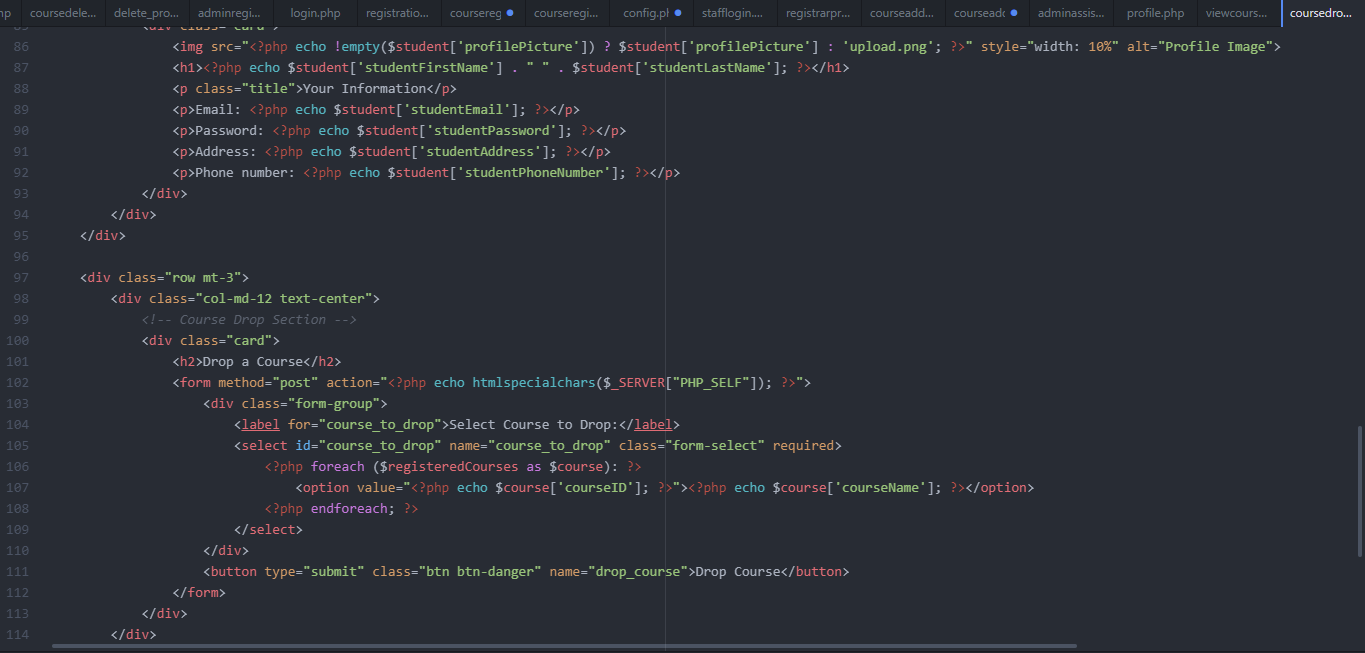
Figure 29. Student viewcourse.php file cont.

Once a student has registered for a course, they will be able to drop their course when they need to. In order to create this functionality, I created a fie titled coursedrop.php. This page is a simple page that has a dropdown menu of the different courses that the student is registered to followed by a button to confirm the course cancelation. Figure 30. Student courses drop webpage.

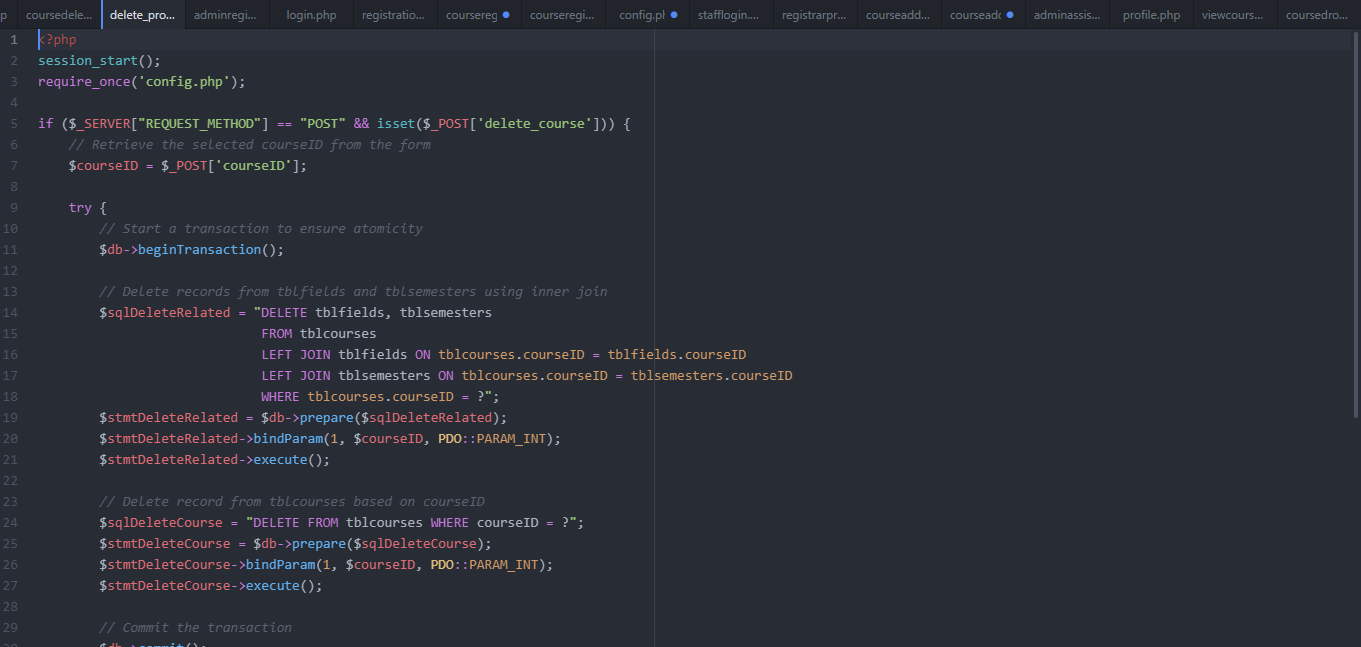
Figure 31. Student coursedrop.php file.

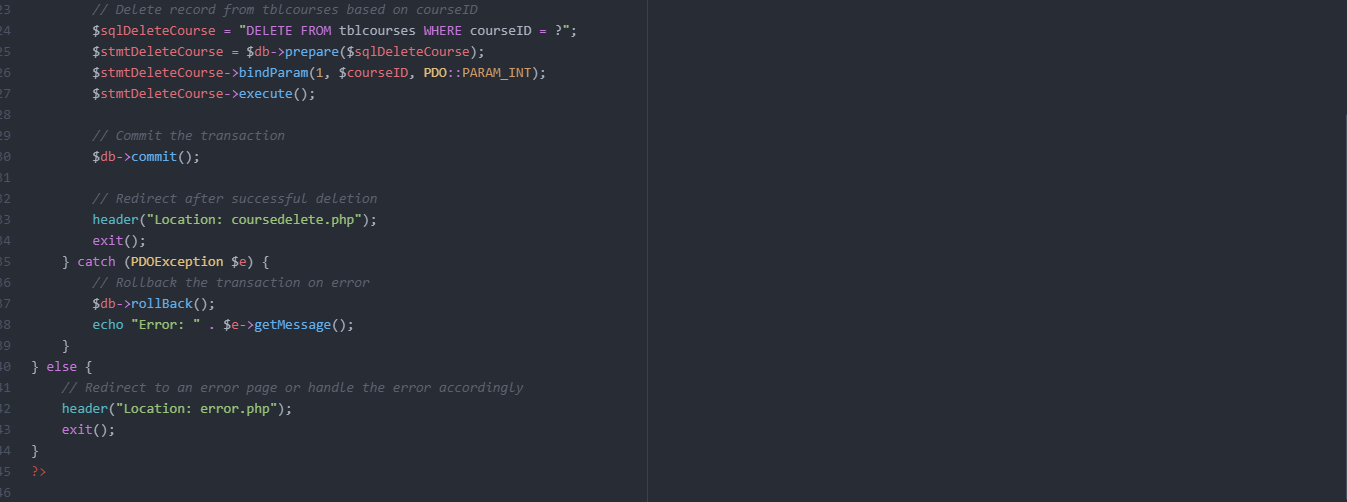
Figure 32 Student coursedrop.php file cont.

Figure 33. Student coursedrop.php file cont.

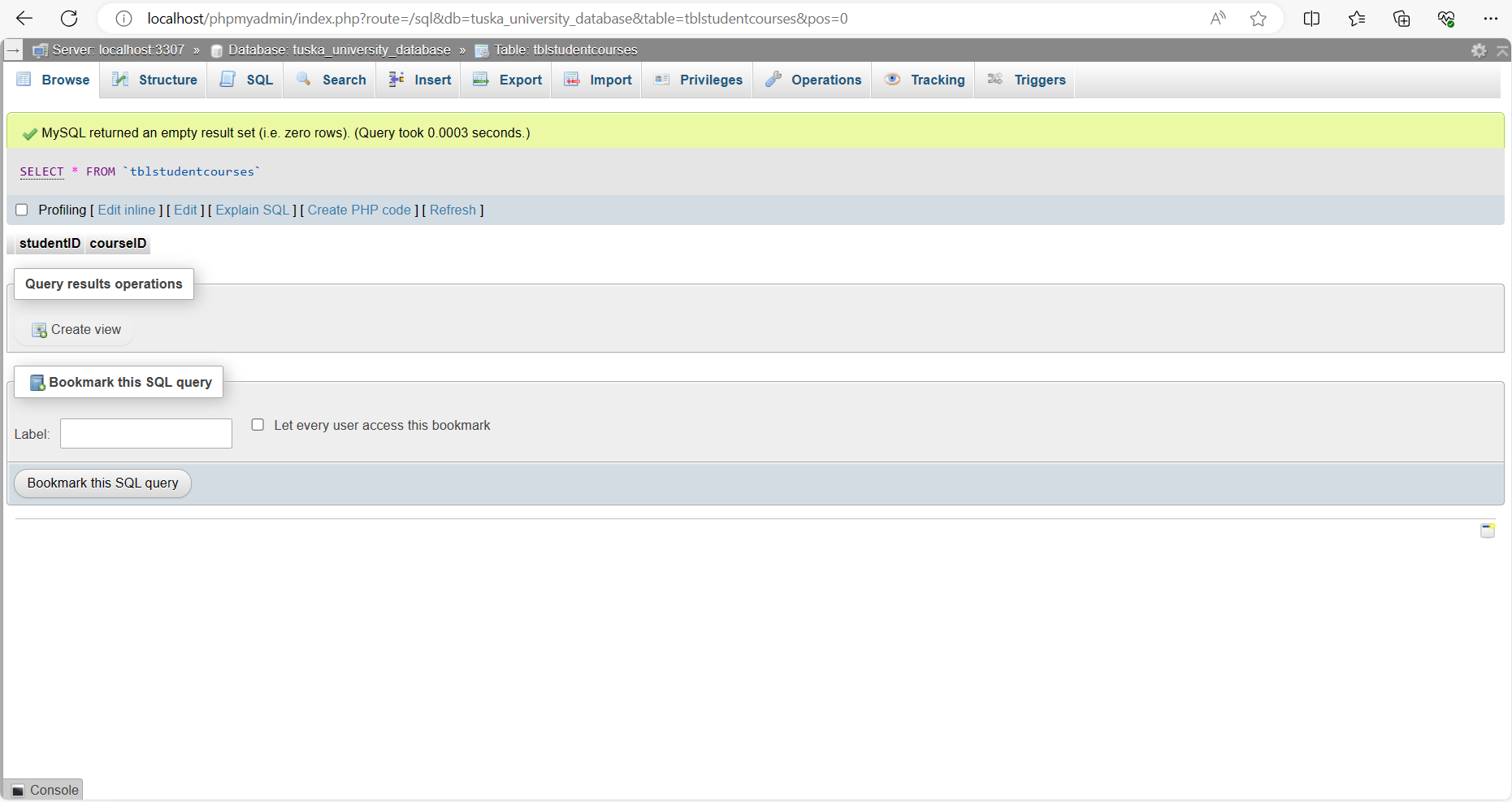
Figure 34. Student coursedrop.php file cont.

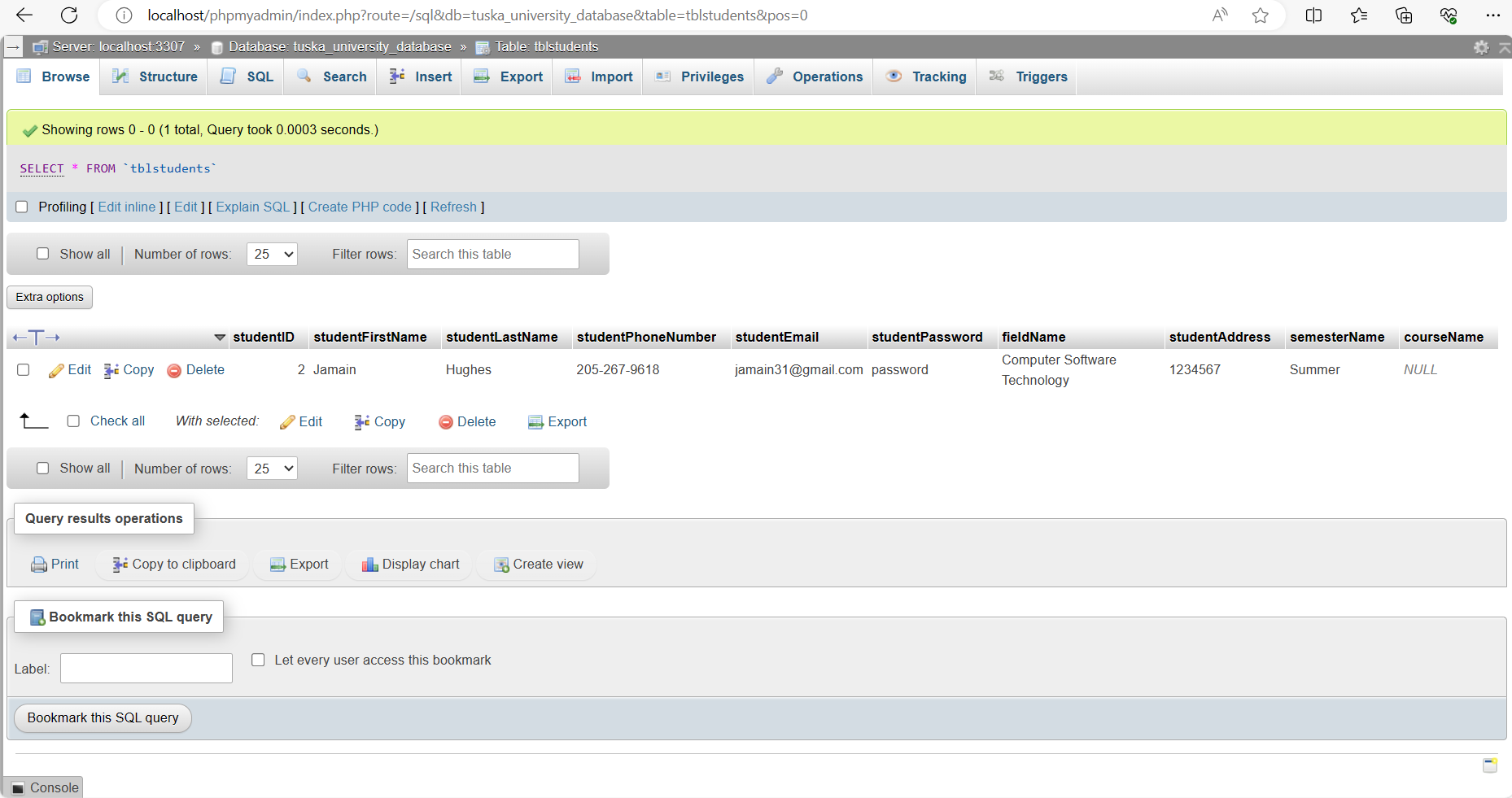
In order to allow for this functionality, I have created a file titled delete\_process.php. This file holds the logic to allow students to drop courses.

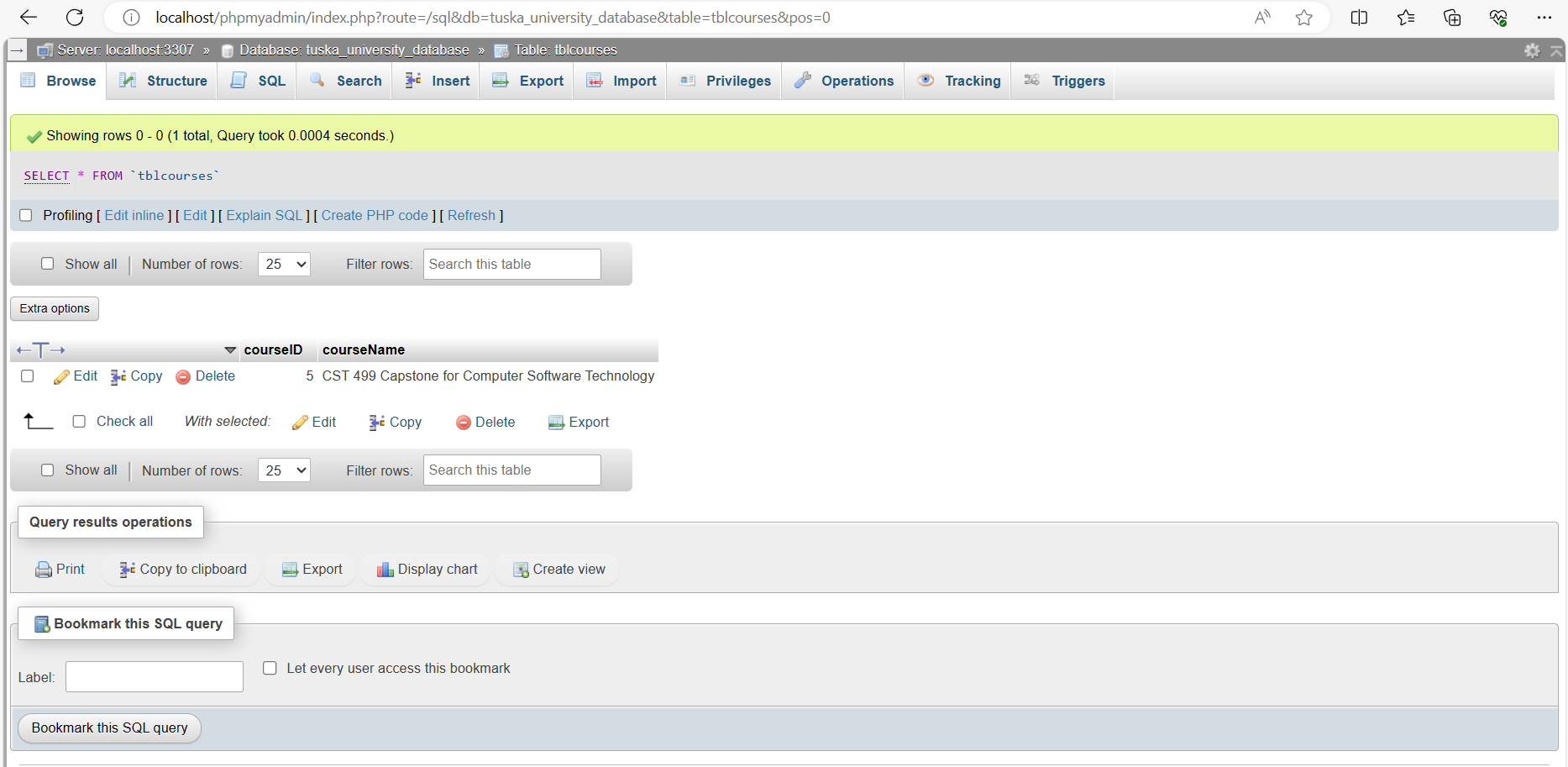
Figure 35. Student courses delete\_process.php file.

Figure 36. Student courses delete\_process.php file.

Once the student has dropped their course, the changes will be implemented within the database The course that have been dropped will then become available for any student who wish to register for that class.

Figure 37. Tblstudentcourses after student have dropped a course.

Figure 38. Tblstudents after student dropped a course.

Figure 39. Tblcourses after student dropped course.

My experience throughout the implementation stage was difficult in some parts but the overall result of having a functioning software system was much worth the effort. There were some features that I wanted to include in the software system to enhance the usability of the software system like mobile compatibility, exception handling, and a more visually appealing layout but, I didn’t have the time or skillset to do so. I am very satisfied with my results. The hardest part of the implementation stage was coming up with the logic to process all the data so that it can effectively communicate with my database. Now that my student registration system has met its functioning primary functions, I will now begin to work on the usability, maintainability, and security of this website.

References

DeBarros, A. (2022). Practical SQL: A beginner’s guide to storytelling with data. No Starch Press.

Duckett, John. (2014). HTML & CSS: Design and build websites. John Wiley and Sons.

Duckett, Jon, Ullman, C., & Stone, E. (2022). PHP & MySQL: Server-side web development. John Wiley & Sons.

Tsui, F., Karam, O., & Bernal, B. (2018). [Essentials of software engineering](https://uagc.instructure.com/courses/126521/modules/items/6439323) (4th ed.). Jones & Barlett Learning.