**CS6314 Project Part2** (update: 2025-03-26)

|  |  |  |  |
| --- | --- | --- | --- |
| Member of Team | NetID | First Name | Last Name |
| 1. | mxb240023 | Mahima | Bhushan |

Submission requirements.

Submit (1) a word document (this file with your answers and listing of any program & its run [log or screenshots]) and (2) a zip file of a folder which will contain all the codes (all the source codes, data and any other items related to this assignment and your work done).

This word document (this file) is your documentation (as a basis to add your answers) to be included here - all your answers, all your program-listings, instructions to compile and run [screen-shots, terminal text, or session log] to show your work done, including how to compile and run for each cases.

The word file should also have (1) in header with the course & section number, your name (Name: last name, first name), your netID (email), and Assignment #, (2) in footer with page number, (3) line number (restart from each page in left margin. Page Layout => Line Numbers). (4) Your program should have some comments (minimal or reasonable), to tell the code-reader what a segment of code is doing, and with the comment in the front/head of the program about you, this course and assignment, etc. (you may copy and paste some out of this document for your comment).

\*\* Your executable codes (that you submit) should run in cs1 without any change or recompilation.

\*\* Bring this cover sheet (this page) to TA for your demo.

Upload (attach) this document (with your answers) and a zip file (containing all the codes [source and data, etc.]).

Score-sheet (please specify what you have done by each team member has done below).

|  |  |  |  |
| --- | --- | --- | --- |
| Member Name | 1. | Demo (as TA needs) | Documentation  Required |
| Task5  Course Authentication:  userid & password, login & logout in PHP  40% | Done 100% of Task5  Login, Logout, Admin Main Page, Manage Users Page, View Login/Logout logs, Ordinary User Course Management Page and took care of revoked conditions. | Done 100% | Done 100% |
| Task8  Documentation – this document (& Demo as TA needs). Deduction if not done (up to 100%). | Done 100% | Done 100% | Done 100% |

|  |
| --- |
| Deduction - Documentation (this .doc file) and upload  Max -70% if not done or poorly prepared |
| Deduction for Demo (Demo schedule or arrangement will be scheduled by TA, for your demo).  Note. You do the demo only for the part(s) that you have done and submitted.  Max -50% if not done or poorly prepared. |
| Deduction (To use of xampp - all work/project to be done using xampp frame)  Do all your work/project done using/via xampp frame. If not, there will be severe penalty (-70%). |

CLO - After successful completion of this course, the student should be able to:

1. Ability to understand web architecture, standards, protocols, tools, and technologies

2. Ability to understand HTML, HTML5 and CSS.

3. Ability to understand JavaScript, JQuery, AJAX, XML, JSON

4. Ability to understand Database Technologies and SQL

5. Ability to understand Server-side programming with PHP

6. Ability to understand Web Services SOAP and RESTful Web Services

7. Ability to understand Web Security Protocols & Standards Semantic Web

# 

# Table of Contents

1. [Part 2](#_Part_2)
   1. [Task 5](#_Task_5.)
   2. [Task 8](#_Task_8.)
2. [Part2 Solution & Document](#_Part2_solution_&)
3. [Part2 Task5 Solution & Document](#_Part2_Task5_solution)
   1. File Name: setusers.php
   2. File Name: Login.php
   3. SQL listing of each table in planner database.
      1. DB Table Name: user
      2. DB Table Name: userlog
   4. Screenshots of degree planner & run
      1. log in
      2. Main Page
      3. log out
   5. Each part/task provide Screenshots for testcases
      1. Logging in as admin

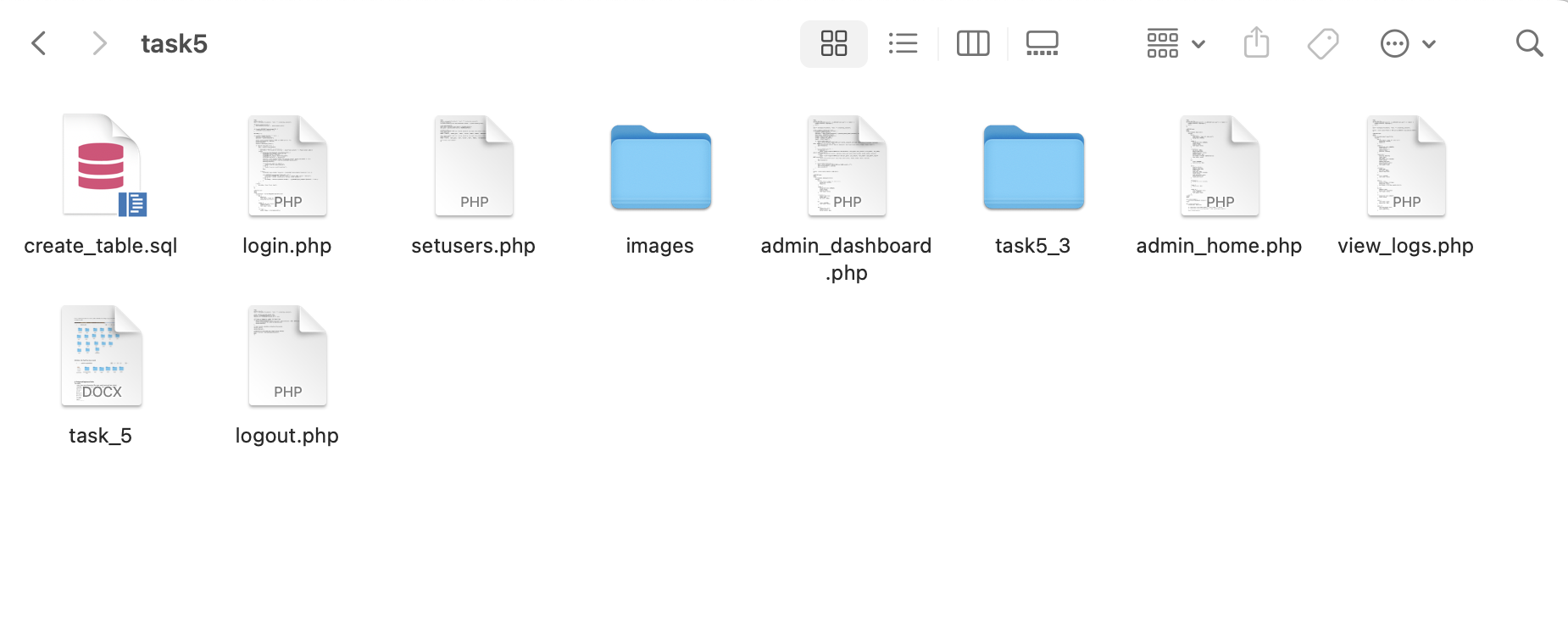
* Adding Users
* Updating Users
* Deleting Users
  + 1. Logging in as Ordinary User
* Updating course
* Adding course
* Deleting course
  + 1. When user failed its log-in step more than 3 times

# Part 2

Create a folder (cs6314) for this course, and its subfolder for each part (e.g., cs6314/task5) in Xampp for this assignment (task). For your submission, upload the zip file of this directory (with all its contents – all the files and all subdirectories) and this document (as your project report). You will also have a few more subfolders as needed. Note that this folder should contain also a folder for all sql statements (so that all tables can be created using these sql for the project, by exporting the mysql database of all the tables for this project in sql format.

A group of blue folders

Description automatically generated



Run Login.php to start

A screenshot of a computer

Description automatically generated

### Task 5.

The task is to manage users by administrator and to implement login process & logout process of Project Part1.

You may use www13 for your base code for this part.

(1) Design and implement DB tables: user and userlog. (a) User table contains user information: id# (auto), userid, password (encrypted), user type (admin or user), user status (active, inactive, revoked, deleted), first name, last name, email, create-date&time, last-update date&time. (b) Userlog table contains user log information (each time of login): id# (auto), userid, login date-time, logout date-time, session-id.

(2) Administrator runs an initial php script (e.g., setusers.php) to add one admin user and one normal user (as shown below) and use the following users for your test cases. The encrypted password is kept in the table.

Administrator will run an initial php script (setusers.php) to add initial users.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ID# | User Type | User Status | User name | UserID | Password | Email ID | created - Date & Time |
| 1 | admin | active | Admin Name1 | admin1 | admin1 | admin1@utdallas.edu | 2025-03-10 15:30:00 |
| 2 | user | active | User Name2 | user2 | user2 | user2@utdallas.edu | 2035-03-11 09:10:01 |

(3) Provide a web page for admin user (once log-in) to add, update, delete user(s) to add the following users:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 3 | User | active | User Name3 | user3 | user3 | user3@utdallas.edu |  |
| 4 | User | active | User Name4 | user4 | user4 | user4@utdallas.edu |  |
| 5 | Admin | active | Admin Name5 | user5 | user5 | admin5@utdallas.edu |  |

(4) Log-in & Log-out: Provide the log-in and log-out functionalities for a user (active), along with database table for user information. One may log in as an ordinary user to see the main menu of course management (list, search, add, delete, update). One may log in as an administrator to see the administrator main-menu to manage users (to add, delete, update user info). Here an administrator can update user status to be one of active, inactive, deleted, revoked (when failing log-in three times or more), or to be initial status (when user signed up for first time).

(5) After the task is done, the user is to log out. For log-out, each web page should provide a button for a user to log out. For each user login and logout, the system will keep track of the record of user log-in & log-out saved in DB table (userlog), to keep the necessary record of log-in and log-out for each user: userid, time of log-in, and time of log-out and session information (session-id).

Userlog table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Logid  (Auto) | UserID | Login  Date & Time | Logout  Date & Time | SessionID |

(6) Provide a web page for admin when log-in, to show current log-in/log-out status of users. Provide a main menu for admin (when log-in) to select and work with user or with userlog.

(7) When user failed its log-in step, let user try it for second or third time. If it is failed more than third time, make the userid to be blocked (status to be revoked).

### Task 8.

Documentation.

Create a folder for each task (cs6314/…, to be submitted as a zip files for all of your work, codes, sql, react, node documentation.

(1) In this word document: Prepare and submit a technical document of all your work (for each part) with (a) table of contents (data/program dictionary – list of each file, web-page, DB table/entry, … and its description), (b) source listing of code, sql, json, …, (c) web pages (screenshots – make sure that it is readable [without zoom-in or 50% penalty].

(2) Prepare and submit two video tutorials (3-5min in MP4) for your project work (a walk-through and overview of your web system, including web pages in workflow and backend systems – codes, DB, …) - one video tutorial (Task8A) for the php-based system (LAMP, WAMP, or MAMP) and the other video tutorial (Task8B) for MERN based system. If the video files gets too big, please compress it or reduce its size with resolution/quality to a reasonable level to be viewable.

Note. If you have screenshots in the document, make sure that it is readable (font size & no need to zoom).

# Part2 solution & document

Provide a data/program dictionary of each file, program, DB table, web page, … and what this file is about.

|  |  |  |
| --- | --- | --- |
| Directory (e.g., cs6314/task5) | File Name (e.g., index.php) | Description |
| MySQL Database | user (table) | Stores users' login info: id, userid, encrypted password, status, email, etc. |
| MySQL Database | userlog (table) | Tracks each login and logout event with session ID and timestamps. |
| MySQL Database | course1 (table) | Contains all course details: number, title, instructor, time, and location. |
| cs6314/task5 | login.php | Login page for both admin and users. Starts a session and redirects accordingly. |
| cs6314/task5 | logout.php | Ends user session, updates logout time in userlog table, and redirects. |
| cs6314/task5 | setusers.php | Initializes DB with one admin and one normal user with encrypted passwords. |
| cs6314/task5 | create\_table.sql | SQL script to create user, userlog, and possibly course1 tables. |
| cs6314/task5 | admin\_dashboard.php | Admin-only page to add, update, delete users and manage user statuses. |
| cs6314/task5 | admin\_home.php | Welcome page or redirector for admin after login. |
| cs6314/task5 | view\_logs.php | Admin page showing user login/logout history with session IDs. |
| cs6314/task5/images/ | utd-logo.png | UTD logo used in the course management page header. |
| cs6314/task5/task5\_3 | task3.php | Reusing the course management page from task3.  Main interface for ordinary users to list, search, add, update, and delete courses. |
| cs6314/task5/task5\_3 | task3.js | JavaScript file for course operations using AJAX (add, update, delete). |
| cs6314/task5/task5\_3 | |  | | --- | |  |  |  | | --- | | task3.css | | Custom CSS for layout, headers, buttons, UTD colors, and responsiveness. |
| cs6314/task5/task5\_3 | task3\_addcourse.php | PHP backend to insert a new course record into the course1 table. |
| cs6314/task5/task5\_3 | task3\_updatecourse.php | PHP backend to update an existing course using AJAX. Returns JSON response. |
| cs6314/task5/task5\_3 | task3\_deletecourse.php | PHP script to delete a course from course1 table using AJAX. |

Program Listing in project folder, including all PHP, HTML, CSS, Javascript, SQL, MongoDB entry, React, and any others. Make sure to provide a proper heading for each file name.

# Part2 Task5 solution & document

1. File Name: setusers.php (To initially add in 2 users)

|  |
| --- |
| <?php $conn = new mysqli("localhost", "root", "", "university\_courses"); // Check connection if ($conn->connect\_error) die("Connection failed: " . $conn->connect\_error); // Pre-hash passwords $admin\_pass = password\_hash("admin1", PASSWORD\_DEFAULT); $user\_pass = password\_hash("user2", PASSWORD\_DEFAULT); // Insert users $conn->query("INSERT INTO user (userid, password, user\_type, user\_status, first\_name, last\_name, email, create\_datetime) VALUES ('admin1', '$admin\_pass', 'admin', 'active', 'Admin', 'Name1', 'admin1@utdallas.edu', '2025-03-10 15:30:00')"); $conn->query("INSERT INTO user (userid, password, user\_type, user\_status, first\_name, last\_name, email, create\_datetime) VALUES ('user2', '$user\_pass', 'user', 'active', 'User', 'Name2', 'user2@utdallas.edu', '2025-03-11 09:10:01')"); echo "Initial users added!"; ?> |

2. File Name: Login.php (To login as a user or admin)

|  |
| --- |
| <?php session\_start(); $conn = new mysqli("localhost", "root", "", "university\_courses");  if ($conn->connect\_error) { die("Connection failed: " . $conn->connect\_error); }  if (!isset($\_SESSION["login\_attempts"])) { $\_SESSION["login\_attempts"] = []; }  $errorMsg = "";  if ($\_SERVER["REQUEST\_METHOD"] == "POST") { $userid = $\_POST["userid"]; $password = $\_POST["password"];  $stmt = $conn->prepare("SELECT \* FROM user WHERE userid = ?"); $stmt->bind\_param("s", $userid); $stmt->execute(); $result = $stmt->get\_result();  if ($result->num\_rows === 1) { $user = $result->fetch\_assoc();  if ($user["user\_status"] !== 'active') { $errorMsg = "Your account is currently " . $user["user\_status"] . ". Please contact admin."; } else { if (password\_verify($password, $user["password"])) { $\_SESSION["login\_attempts"][$userid] = 0; $\_SESSION["userid"] = $userid; $\_SESSION["user\_type"] = $user["user\_type"]; $\_SESSION["session\_id"] = session\_id();  $stmtLog = $conn->prepare("INSERT INTO userlog (userid, session\_id) VALUES (?, ?)"); $stmtLog->bind\_param("ss", $userid, $\_SESSION["session\_id"]); $stmtLog->execute();  if ($user["user\_type"] == "admin") { header("Location: admin\_home.php"); } else { header("Location: task5\_3/task3.php"); } exit;  } else { $\_SESSION["login\_attempts"][$userid] = ($\_SESSION["login\_attempts"][$userid] ?? 0) + 1;  if ($\_SESSION["login\_attempts"][$userid] >= 3) { $conn->query("UPDATE user SET user\_status = 'revoked' WHERE userid = '$userid'"); $errorMsg = "Account revoked due to too many failed attempts."; } else { $errorMsg = "Incorrect password. Attempt " . $\_SESSION["login\_attempts"][$userid] . " of 3."; } } } } else { $errorMsg = "User ID not found."; } } ?>  <!DOCTYPE html> <html> <head> <title>Login - Course Management System</title> <style> body { margin: 0; font-family: "Segoe UI", sans-serif; background-color: #f7f7f7; }  .header { background-color: #154734; /\* UTD Green \*/ padding: 30px 0; text-align: center; }  .utd-logo { width: 120px; /\* increased width \*/ max-width: 100%; height: auto; margin-bottom: 10px; }   .header img { width: 150px; }  .header h1 { margin: 10px 0 0 0; font-size: 24px; color:rgb(255, 255, 255); }  .login-container { margin: 50px auto; width: 400px; background-color: white; border-radius: 8px; padding: 30px; box-shadow: 0 0 15px rgba(0,0,0,0.1); }  .login-container h2 { text-align: center; color: #004225; margin-bottom: 25px; }  input[type="text"], input[type="password"] { width: 100%; padding: 12px; margin: 8px 0 16px; border: 1px solid #ccc; border-radius: 5px; }  button { width: 100%; padding: 12px; background-color: #b94600; color: white; border: none; border-radius: 5px; font-weight: bold; font-size: 16px; cursor: pointer; }  button:hover { background-color: #933600; }  .error { color: #c0392b; background: #fcecec; padding: 10px; border: 1px solid #e74c3c; border-radius: 4px; margin-bottom: 15px; } </style> </head> <body>  <div class="header"> <img src="images/utd-logo.png" alt="UT Dallas Logo" height="140"> <h1>Course Management System</h1> </div>  <div class="login-container"> <h2>Login</h2>  <?php if ($errorMsg): ?> <div class="error"><?= $errorMsg ?></div> <?php endif; ?>  <form method="POST"> <label for="userid">User ID:</label> <input type="text" name="userid" required>  <label for="password">Password:</label> <input type="password" name="password" required>  <button type="submit">Login</button> </form> </div>   </body> </html> |

SQL listing of each table in planner database. Make sure to have a proper heading for each table for a comment.

1. DB Table Name: user

|  |
| --- |
| CREATE TABLE user (  id INT AUTO\_INCREMENT PRIMARY KEY,  userid VARCHAR(50) NOT NULL UNIQUE,  password VARCHAR(255) NOT NULL,  user\_type ENUM('admin', 'user') NOT NULL,  user\_status ENUM('active', 'inactive', 'revoked', 'deleted') NOT NULL,  first\_name VARCHAR(50),  last\_name VARCHAR(50),  email VARCHAR(100),  create\_datetime DATETIME DEFAULT CURRENT\_TIMESTAMP,  last\_update\_datetime DATETIME DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP  ); |

2. DB Table Name: userlog

|  |
| --- |
| CREATE TABLE userlog (  id INT AUTO\_INCREMENT PRIMARY KEY,  userid VARCHAR(50),  login\_datetime DATETIME DEFAULT CURRENT\_TIMESTAMP,  logout\_datetime DATETIME,  session\_id VARCHAR(255)  ); |

Screenshots of degree planner & run

1. log in

|  |
| --- |
| A screenshot of a computer  Description automatically generated  Login either as admin or ordinary user: |

2. Main Page:

|  |
| --- |
| **If logged in as Admin:**  Provides the admin to see the administrator main-menu which will have two options 1) **manage users** (to add, delete, update user info) and 2) to view the current **log-in/log-out status** of users.    A screenshot of a computer  Description automatically generated  On Clicking Manage Users shows the Manage Users Page:  A screenshot of a computer  Description automatically generated |
| On Clicking View Login/Logout Logs shows the User Login/Logout Activity  A screenshot of a computer  Description automatically generated |
| After the work is done the admin can logout through any page (Main page, Manage Users Page or the User Login/Logout Activity Page  On clicking Logout it brings the user back to the Login Page.  **If Logged in as an Ordinary User:**  The user will see the main menu of course management where he can list, search, add, delete and update the courses.    A screen shot of a computer  Description automatically generated  A screenshot of a computer  Description automatically generated |
| On clicking Logout, it brings the Ordinary user back to the Login Page. |

3. log out

|  |
| --- |
| After Login, all the pages have an option of Logout. On clicking it, the user will be redirected back to the login page and the session will end.  For each user login and logout, the system will keep track of the record of user log-in & log-out saved in DB table (userlog), to keep the necessary record of log-in and log-out for each user: userid, time of log-in, and time of log-out and session information (session-id).  As shown below all the pages after login are provided with a logout button:  **Admin User Main Page:**  A screenshot of a computer  Description automatically generated  Admin User (User Login/Logout Activity Page):  A screenshot of a computer  Description automatically generated  Admin User (Manage User Page):  A screenshot of a computer  Description automatically generated  Ordinary User Course Management Systems Page:  A screen shot of a computer  Description automatically generated  On clicking the logout it bring the user back to the login page:  A screenshot of a login screen  Description automatically generated |

4. For each part/task, user input check: provide some input check (screenshots) for each field for add or update.

|  |
| --- |
| **Logging in as admin:**  A screenshot of a computer  Description automatically generated  It goes to admin\_dashboard.php    A screenshot of a computer  Description automatically generated  Go to Manage Users:  A screenshot of a computer  Description automatically generated  Adding a User3:  A screenshot of a computer  Description automatically generated  Under All users we can now see user3  A screenshot of a computer  Description automatically generated  After adding 2 more users A screenshot of a computer  Description automatically generated  Updating user5 to inactive  A screenshot of a computer  Description automatically generated  A screenshot of a computer  Description automatically generated  Deleting user5  A screenshot of a computer  Description automatically generated  User5 is no longer seen under all users  A screenshot of a computer  Description automatically generated  When Admin clicks the View Login/Logout Logs on the Main Page:  He will see the below.  A screenshot of a computer  Description automatically generated  **When an ordinary user logs in correctly this is the page that will show up:**  A screen shot of a computer  Description automatically generated  A screenshot of a computer  Description automatically generated  Updating a course:  Updating the course timings  A close-up of a screen  Description automatically generated  A computer screen shot of a pencil  Description automatically generated  Changed the time  A screen shot of a course  Description automatically generated  You can see that the time is changed:  A white and orange rectangle  Description automatically generated  Adding a course:  A screenshot of a computer  Description automatically generated  A screenshot of a computer  Description automatically generated  Deleting a Course:  Deleting this course  A close-up of a red and white rectangle  Description automatically generated  Press on the delete icon  A screenshot of a cellphone  Description automatically generated  Asks for confirmation  A screenshot of a message  Description automatically generated  A screenshot of a computer  Description automatically generated  Course has been deleted  A screen shot of a cellphone  Description automatically generated  Ordinary user Logging out:  The user id is user2  A green and white sign  Description automatically generated  It comes back to the login page:  A screenshot of a computer  Description automatically generated  So if an admin logs in and checks we can see that the user2 has been logged out  A screenshot of a computer  Description automatically generated  **When user failed its log-in step, let user try it for second or third time. If it is failed more than third time, make the userid to be blocked (status to be revoked).**  There is an active admin2 user let us verify that if he failed more than 3 times what happens:  A screenshot of a computer  Description automatically generated  A screenshot of a login screen  Description automatically generated  A screenshot of a course management system  Description automatically generated  The 3rd time the user fails, the account is revoked:  A screenshot of a login screen  Description automatically generated  So when logged in as admin1 and checked the status of admin2 we can see it has been revoked.  A screenshot of a computer  Description automatically generated |