2022 U.S. Information Technology Collegiate Conference Regionals - Application Development Competition

**Do not put your name(s) or your school’s name on anything that you submit, including comments, file metadata, videos, and images.**

**Doing so will result in disqualification of your team.**

**The only identifying information you should use is your team number.**

**Programming Challenge: Application Development for Real Time Sports Management Agency**

Your team has been tasked with developing a 3-tier tennis grand slam winners’ database management application for Real Time Sports Management Agency. Your application must allow both admin users and regular users (tennis players) to login first. Once an admin logs in, they should be able to do two things: 1) view all of the past grand slam winners, and 2) add / delete / update grand slam winners’ information in the database. Once a regular user logs in, they are supposed to see only their own grand slam wins from the database. They should not be able to view the grand slam wins of other players. Both admin users and regular users must be able to sort / filter / count the data.

The requirements for each tier in the 3-tier application are as follows:

**Tier 3: (Data Tier)**

**All data must be stored in databases or text files. If you use databases, then your application must work without the need for the competition judges to install any database client. You must include the database files or text files that your application requires along with the rest of your code in ZIP file format. Applications which store and retrieve data using databases, XML, or JSON format will earn more points than applications which store information as comma (or tab) delimited text files.**

As a part of this data tier, add the following tables to your database:

* Credentials
	+ Design: Add user name and password fields to this table
	+ Data: Add the following 2 records to this table:
		- Username: rfederer, password: gsw20
		- Username: rnadal, password: gsw22
* AdminCredentials
	+ Design: Add user name, password, and admin password fields to this table
	+ Data: Add the following record to this table:
		- Username: admin, password: nimda, admin password: superuser
* GrandSlamInformation
	+ Design: Add grand slam id, grand slam name, winning player id, winning player name, winning player earnings in $
	+ Data: Add the following 11 records to this table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Grand Slam ID** | **Grand Slam Name** | **Winning Player ID** | **Winning Player Name** | **Winning Player Earnings** |
| 2022\_1 | Australian Open | P002 | Rafael Nadal | $ 2,275,000\* |
| 2022\_2 | French Open | P002 | Rafael Nadal | $ 1,925,000\* |
| 2020\_2 | French Open | P002 | Rafael Nadal | $ 1,775,000\* |
| 2019\_2 | French Open | P002 | Rafael Nadal | $ 1,655,000\* |
| 2019\_4 | U.S. Open | P002 | Rafael Nadal | $ 2,125,000\* |
| 2018\_1 | Australian Open | P001 | Roger Federer | $ 2,000,000\* |
| 2018\_2 | French Open | P002 | Rafael Nadal | $ 1,500,000\* |
| 2017\_1 | Australian Open | P001 | Roger Federer | $ 1,875,000\* |
| 2017\_2 | French Open | P002 | Rafael Nadal | $ 1,375,000\* |
| 2017\_3 | Wimbledon | P001 | Roger Federer | $ 2,250,000\* |
| 2017\_4 | U.S. Open | P002 | Rafael Nadal | $ 1.925,000\* |
| \* - Please note that these are not the actual prize monies awarded.These number have been made up for the purposes of this challenge. |

Set the primary keys for each table appropriately. Use the appropriate data types and field sizes for each field in the table. You do not need to create relationships between the tables.

**Tier 1: (User-Interface Tier)**

The application must use multiple forms. The first form must be a login form which allows two types of users to login. Create a new logo for Real Time Sports Management Agency using any image editing software and include that logo on the form. Add two messages on the login form. The first should be a welcome message (Welcome to Real Time Sports Management Agency) and the second should be “Please login to continue …”. The form should then have two boxes, one to enter the username and the other to enter the password. Then, there should be a button to login. The first type of user is the admin user. Once the word “admin” is entered in the username field, an additional admin password box must appear on the login form. In other words, there must be two passwords for admin users. Make sure that both password boxes are masked, meaning the passwords should not appear in plain text to the user. For unsuccessful logins, display an appropriate message. Display an appropriate message after 3 straight unsuccessful login attempts and close the application.

The second form should display a list of past grand slam winners. All past winners should be displayed on the same screen as a snapshot. Admin users should be able to view information about ALL past grand slam winners. Regular users must be able to view only their own past grand slam wins. They cannot view past grand slam wins of other tennis players. Add buttons on the second form to let both admin and regular users to do the following:

* Sort the data by year starting with the most recent year first
* Filter the data by earnings that are above a custom amount entered by the user using a textbox. Make sure that the earnings are displayed on the form using the comma-separated currency format.
* Count the data by grand slam name (Australian Open, French Open, etc.) entered by the user using a textbox. Users must be able to enter part of the grand slam name (example: Open) and the application must retrieve all grand slams which have that in their names.
* Logout of the application

On the second form, there must be an additional button only for admin users to allow admins to view the data from the GrandSlamInformation table using the third form.

On the third form, which only admin users must be able to see, display the data one record at a time. On this form, add buttons to let admin users do the following:

* Move to the previous record
* Move to the next record
* Insert a new record about a grand slam winner into the database
* Delete an existing record about a grand slam winner from the database
* Save changes to an existing record about a grand slam winner in the database

Note that all changes made to the database (additions / deletions / updates) must be persistent, meaning that once the application is closed and reopened, the changes must remain intact from the last time the application was used.

**Tier 2: (Business Logic Tier)**

In this tier, implement Object-Oriented Programming (OOP) concepts such as encapsulation, inheritance, and polymorphism. This will require creation of classes, objects, properties, constructor methods, getter methods, setter methods, user-defined methods, user-defined functions, etc. All business logic will need to be implemented in this tier. If you are unable to do so, then implement all of the business logic in tier 1 (the form’s code).

**Requirements:**

1. OOP approaches are preferred and will receive a bonus over non-OOP submissions.
2. All forms, UI control names, classes, objects, variables, methods, functions, must use meaningful names. The project name and file names must also be meaningful.
3. All data must be stored in databases or text files. Data stored and retrieved using databases, XML or JSON format will earn more points than regular text data. Data files must be stored and accessed from their own folder (Data).
	1. If you are using text files to store data, then do not use an absolute file path(i.e., do not specify a path like c:\data\data.txt) in your program code. Rather, use a relative path (such as data\data.txt).
	2. If you are using text files to store data, then please create a constant at the top of your code which is used to access the file(s). This will allow the judges to easily change the location, if needed.
4. Exception handling must be included. Throw exceptions for missing files (if you are using text files to store data) and possible user entry errors (such as invalid username, password, earnings amount, grand slam type, etc.).
5. Strong security must be provided. Examples include: Limited use of textboxes (use drop downs, lists, radio buttons, check boxes, etc. when possible). When using textboxes, check for security intrusion (such as the user trying the enter SQL or HTML information).
6. Project Documentation – Must provide internal (comments) document for the application, functions, and classes to include: IPO (Input-Process-Output) information.
7. Screen Shots: There is no required number of screen shots. However, screen shots must show ALL working parts of the application. For example, you should have shots demonstrating successful regular user login, successful admin user login, unsuccessful login for either user type, second form for a regular user, second form for an admin user, results of the sort operation, results of the filter operation, results of the count operation, third form only for admin users, successful addition of new grand slam record, successful deletion of an existing grand slam record, successful update of an existing grand slam record, etc.

If a screen shot is not included for a part of the application, it is assumed that part of the application does not work. **All screen shots must be placed in one file, such as a Word or PDF document**. Reminder: Applications without screen shots will not be judged.

1. Efficient/Professional: Efficient code will be given more points than other code that accomplishes the requirements, but is not efficient. Code must be professional, well organized, and designed.
2. Assumptions: As is the case when gathering requirements from users, many program details are not included in the provided description. There are cases were you must make reasonable assumptions about how the application should function. Document these assumptions in a Word, PDF, or text file and include them with your solution.
3. **Note for Java submissions**: If you are using Java, in addition to installing Java 11, you may need to install Java11FX (now a separate install) from the following site: [https://openjfx.io/.](https://openjfx.io/) The judges will be compiling the Java submissions using Java11 with Java11FX. Any additional tools you use (such as form builders) must use only the libraries contain in these installs. If the program does not compile, it will not be judged.

Tie Breaker (work on if you have time):

1. Database / XML / JSON implementation
2. Full OOP implementation (tier 2 programming was appropriate)
3. Adequate security implementation
4. User documentation

 **What to turn in:**

**Create separate subfolders (under a folder with your team name – i.e., APP\_##) for each of the following. Application folder which includes all code including source code and compiled (executable) code. Data folder which includes all database files, XML files, or JSON data files (with example data). Image folder which include any images (example: logo file) used in the application. Documentation folder which includes screen shots (all in one file) and an assumptions file. Then compress all items into one ZIP file (do not use any other compression technique). Non-zip files will NOT be judged!**

**You MUST submit screen shots of your working code. Failure to provide screen shots will cause your application to not be judged. Even if you do not complete all requirements, you still need to include screen shots of what does work.**

**Code that has syntax errors and code that produces runtime errors when executed will not be judged. You should comment out code that ‘almost works’.**

**Good luck with the competition!!!**