Assessment Instrument

CREATE DATABASE ACCESS FOR A COMPUTER APPLICATION USING STRUCTURED QUERY LANGUAGE

Unit Standard ID: 114048

NQF Level: 5

Credits: 9

Notional Hours: 90

# Assessment

## Section A: Formative Assessment

*Answer the following questions;*

### Question 1 (SO 1, AC 1, AC 2)

1. Identify the requirements for database access for a computer application using SQL. (6)

*Clarification: Identify what requirements for data access might motivate for selecting SQL as an access mechanism for an application? i.e. why would you choose SQL over other data storage abstractions (flat file, XML … etc.).*

* Control of redundant data
* No violation of data integrity
* No relying on human memory to store and to search data
* Controlled data access
* Data security
* Data sharing

1. Explain the feasibility of the requirements. (2)*Clarification: Explain how SQL meets these requirements for data access i.e. in what way does it facilitate the requirements.*

Databases can be organized in many different ways by using different models. The data model of a database is the logical structure of data items and their relationships. Data is organized into tables (or relations).  Each table has a set of fields which define the structure of the data stored in the table.  A record is one instance of a set of fields in a table

### Question 2 (SO 1, AC 2)

Identify the database access objectives and critical performance factors. (6)

*Clarification: Over and above data access requirements, a database design team might have certain objectives they wish to meet in their database design. What are some of these objectives, and how can the design of a database help to meet them?*

Normalization helps the database design to meet the database objectives. Normalization is a way to efficiently organize data store in the database.

Normalization ensures data redundant by eliminating same data stored in more one table and storing related data in table.

Another database objective is data integrity and normalization helps to meet data integrity by ensuring that data is the accurate and consistent, having relationships between the tables, so that they share data

Another database objective is reliability, performance, and normalization helps ensure fast searching, sorting, and creation of indexes, since tables are narrower, and more rows fit on a data page.

### Question 3 (SO 1, AC 4)

Explain how a review procedure can be adopted in order to ensure that database access outcomes/results meets the database access requirements. (6)

*Clarification: Given a specific set of access requirements, how can a review process help with meeting these access requirements (in terms of ease of development, security, hardware requirements, operational effort, or other factors)?*

**The Conceptual Design Review** - this review help to validate the application involves a presentation of the statement of purpose as well as a general overview of the desired functionality that will be provided by the application. This review should be conducted as early as possible to determine the overall feasibility of a project, this checks if projects provide duplicate or inadequate functionality.

**The Logical Design Review** -it deals with a thorough review of all data elements, descriptions, and relationships. This is review is done when first cut of the logical data model has been completed.

**The Physical Design Review:**  This is where the database is reviewed in detail to ensure that all of the proper design choices were made.

**The Organization Design Review:**  This review ensures that enterprise-wide concerns of the organization with respect to the application being reviewed. It can done through answering questions such as how does this system interact with other systems in the organization.

Review processes help with the following

* A review process can be adopted to help identifying bugs early before they cost too much.
* A review process can be adopted to build powerful database with better performance
* A review process can be adopted to improve productivity, ease of use.
* A review process can be adopted to ensure data integrity and enforce database policy without relying on an external program to do so

### Question 4 (SO 1, AC 3)

Explain how a developer can estimate the development effort for a database access project so that costs can be estimated. (5)

The developer projects can find a good estimate for a database access by using similar projects. A developer can start by identify all of the work that needs to the done within the project, and list of the activities in the order in which they to happen and who must me responsible and estimate the time and cost using a pert chart.

## Section B: Summative Assessment

### Question 1 (SO 2, AC 1, AC 2, AC 3, AC 4, AC 6)

1. You have just started a new company. It is time to hire some employees. You will need to create a table that will contain the following information about your new employees: firstname, lastname, title, age, and salary. After you create the table, you should receive a small form on the screen with the appropriate column names. If you are missing any columns, you need to double check your SQL statement and recreate the table. (8)

Create database SMCompany

create table Employees(

EmployeeId int IDENTITY(1,1) PRIMARY key,

FirstName varchar(225),

LastName varchar(225),

Tittle varchar(225),

Age int,

Salary DECIMAL(10,2)

);

SELECT \* from Employees;

1. It is time to insert data into your new employee table. Your first three employees are the following:
   * Jonie Weber, Secretary, 28, 19500.00
   * Potsy Weber, Programmer, 32, 45300.00
   * Dirk Smith, Programmer II, 45, 75020.00

Enter these employees into your table first, and then insert at least 5 more of your own list of employees in the table.

After they're inserted into the table, enter select statements to:

1. Select all columns for everyone in your employee table.

select \* from employees;

1. Select all columns for everyone with a salary over 30000.

select \* from employees where salary > 30000;

1. Select first and last names for everyone that's under 30 years old.

select FirstName, LastName from employees where age < 30;

1. Select first name, last name, and salary for anyone with "Programmer" in their title.

select FirstName, LastName, Salary from employees

  where tittle like '%Programmer%';

1. Select all columns for everyone whose last name contains "ebe".

select \* from employees where LastName like '%ebe%';

1. Select the first name for everyone whose first name equals "Potsy".

select \* from employees where firstName = 'Potsy';

1. Select all columns for everyone over 80 years old.

select \* from employees where age >80;

1. Select all columns for everyone whose last name ends in "ith".

select \* from employees where lastname like '%ith';

Create at least 5 of your own select statements based on specific information that you'd like to retrieve.

* SELECT \* FROM EMPLOYEES order by Firstname asc;
* SELECT DISTINCT Firstname FROM EMPLOYEES;
* SELECT  DISTINCT Firstname FROM EMPLOYEES

where tittle like '%Programmer' or tittle like 'Programmer%';

* SELECT \* from Employees where lastname ='Potsy'and lastname ='Weber';
* SELECT \* from Employees where age BETWEEN 20 and 50;

(6)

1. After each update, issue a select statement to verify your changes.
2. Jonie Weber just got married to Bob Williams. She has requested that her last name be updated to Weber-Williams.

* update Employees set LastName ='Weber-Williams'

where FirstName ='Jonie' and LastName = 'Weber';

1. Dirk Smith's birthday is today, add 1 to his age.

* update Employees set Age = age + 1

where FirstName ='Dirk' and LastName = 'Smith';

1. All secretaries are now called "Administrative Assistant". Update all titles accordingly.

* update Employees set tittle ='Administrative Assistant'

where tittle = 'Secretary';

1. Everyone that's making under 30000 are to receive a 3500 a year raise.

* update Employees set salary =salary +3500

where salary <30000;

1. Everyone that's making over 33500 are to receive a 4500 a year raise.

* update Employees set salary =salary +4500

where salary >33500;

1. All "Programmer II" titles are now promoted to "Programmer III".

* update Employees set tittle ='Programmer 111'

where tittle = 'Programmer 11';

Create at least 5 of your own select statements based on specific information that you'd like to retrieve. (6)

* SELECT \* FROM EMPLOYEES order by Lastname asc;
* SELECT \* from Employees where lastname ='Potsy'and lastname ='Weber';
* SELECT  DISTINCT Firstname FROM EMPLOYEES

where tittle like '%Programmer' or tittle like 'Programmer%';

* SELECT \* from Employees where age BETWEEN 25 and 50;
* SELECT DISTINCT Firstname FROM EMPLOYEES;

1. After each update, issue a select statement to verify your changes.
2. Jonie Weber just got married to Bob Williams. She has requested that her last name be updated to Weber-Williams.

* update Employees set LastName ='Weber-Williams'

where FirstName ='Jonie' and LastName = 'Weber';

1. Dirk Smith's birthday is today, add 1 to his age.

* update Employees set Age = age + 1

where FirstName ='Dirk' and LastName = 'Smith';

1. All secretaries are now called "Administrative Assistant". Update all titles accordingly.

* update Employees set tittle ='Administrative Assistant'

where tittle = 'Secretary';

1. Everyone that's making under 30000 are to receive a 3500 a year raise.

* update Employees set salary =salary +3500

where salary <30000;

1. Everyone that's making over 33500 are to receive a 4500 a year raise.

* update Employees set salary =salary +4500

where salary >33500;

1. All "Programmer II" titles are now promoted to "Programmer III".

* update Employees set tittle ='Programmer 111'

where tittle = 'Programmer 11';

1. All "Programmer" titles are now promoted to "Programmer II".

* update Employees set tittle ='Programmer 11'

where tittle = 'Programmer';

Create at least 5 of your own update statements and submit them. (8)

* update Employees set tittle ='Programmer 11'

where tittle = 'Programmer' and age > 40;

* update Employees set tittle ='Software Developer'

where tittle like '%Programmer%';

1. Perform the following deletes (Use the select statement to verify your deletes):
2. Jonie Weber-Williams just quit, remove her record from the table.

* delete from employees

 where FirstName = 'Jonie' and lastName = 'Weber-Williams';

1. It's time for budget cuts. Remove all employees who are making over 70000 dollars.

* delete from employees where salary > 70000;

Create at least two of your own delete statements, and then issue a command to delete all records from the table.

* delete from employees where Firstname like '%eber%';
* delete from employees where age > 55;
* delete from employees where age BETWEEN 60 and 80;
* delete from employees where tittle ='programmer';

(4)

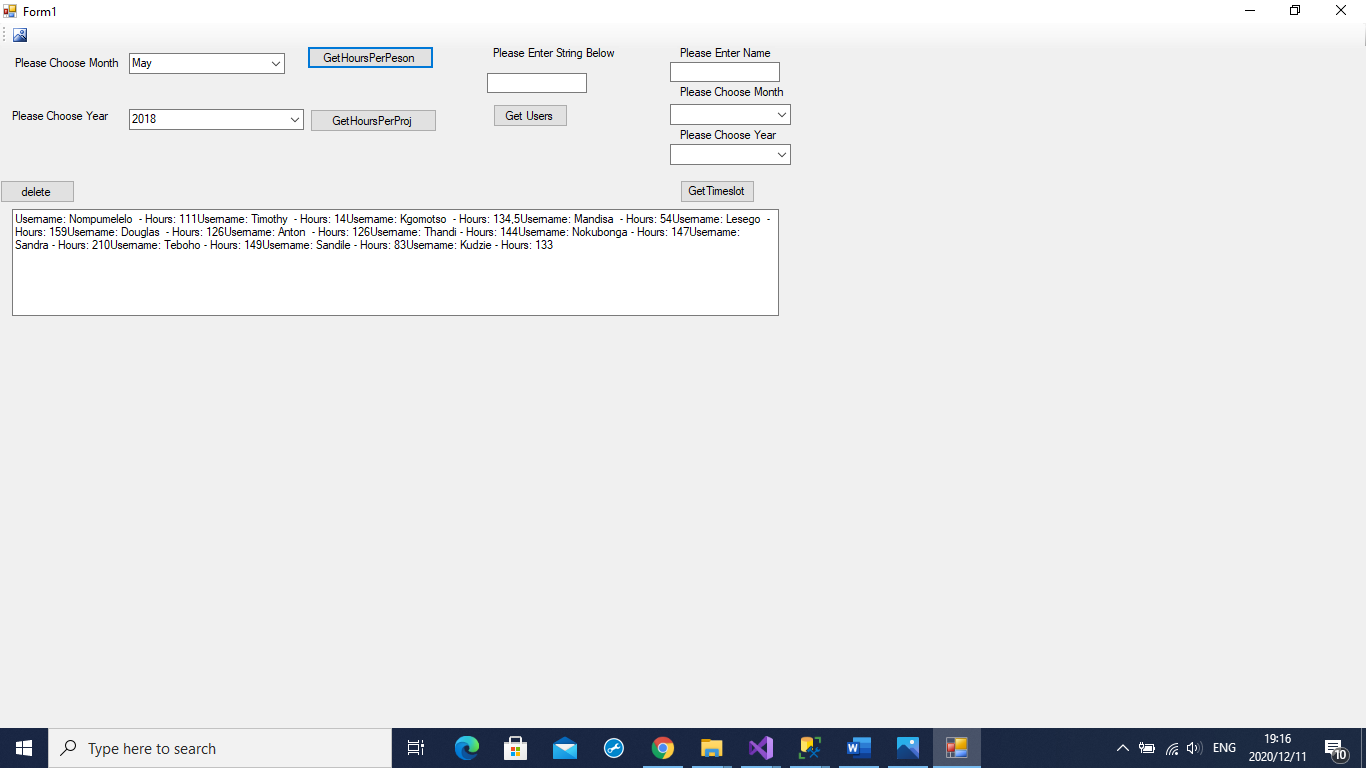
1. Drop your employee table. (6)

Drop table  employees ;

### Question 2 (SO 3, AC 1, AC 2, AC 3, AC 4, AC 5, AC 6, AC 7, AC 8)

Complete the following activity. The purpose of the activity is to connect to a SQL database using a C# program.

1. Get the file timesheets.sql from your facilitator. Use this file to create a database using SQL Server.
2. Create a Visual Studio solution (console application or Windows application).
3. Within this solution, write code that asks the user for month (e.g. June 2019) and then connects to the Timesheets database and outputs the following:
4. The total hours recorded per person for the month entered.
5. The total hours recorded per project for the month entered.
6. Within this solution, also write code that takes as input a string, and lists all the users in the Timesheets database's User table whose user name contains that string.
7. Finally, write code that takes as input a name (string) and a month, and lists the full Timeslots recorded by the first matching User for that month.

**Include the code and screenshots for your solution in your POE as evidence.**

private void button1\_Click\_1(object sender, EventArgs e)

{

var months = comboBox1.SelectedItem.ToString();

var year = comboBox2.SelectedItem.ToString();

using (var conn = CreateConnection())

{

var getHours = conn.Query<Users>(@"SELECT u.Username, t.TotalHoursCaptured FROM Users AS u

JOIN (SELECT UserId,SUM(HoursCaptured) AS TotalHoursCaptured FROM Timeslots

WHERE DATENAME(month, Date)=@month AND YEAR(Date) = @year

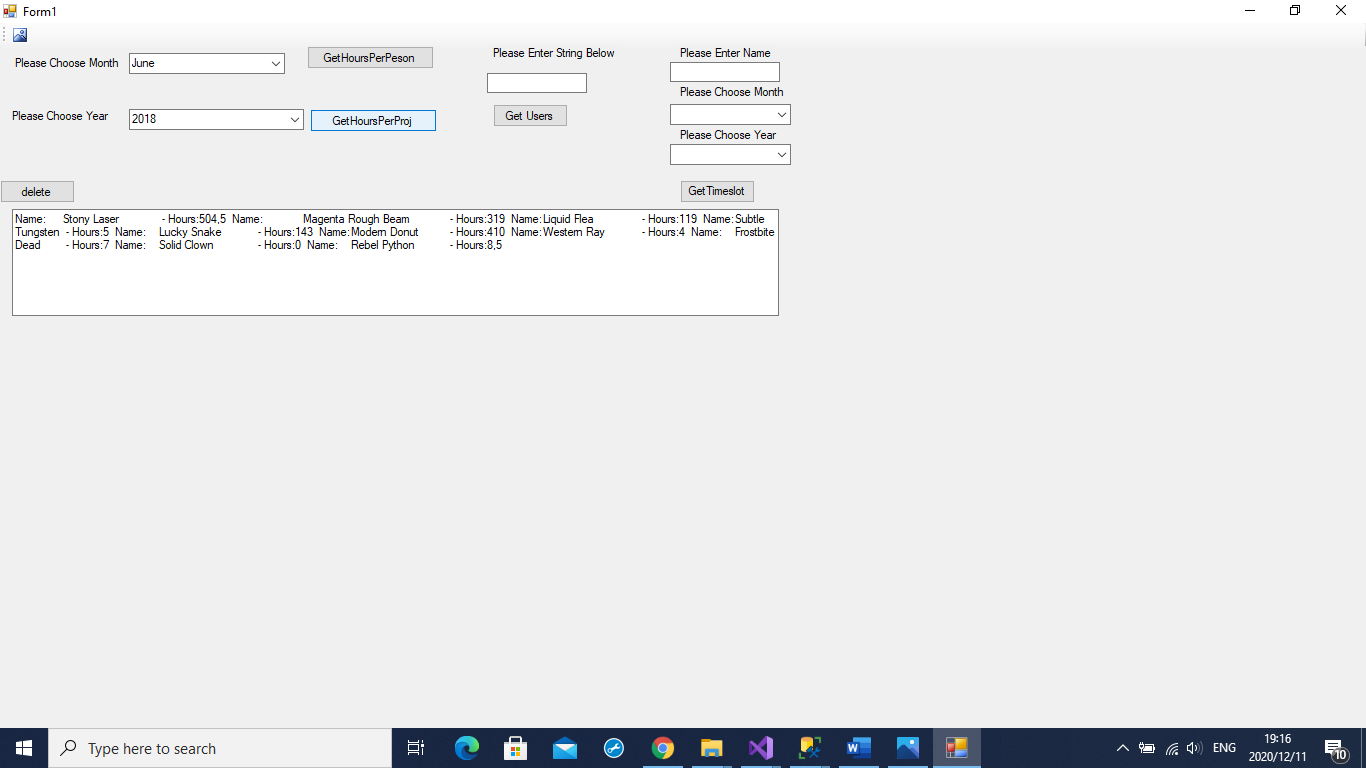
GROUP BY UserId

) AS t

ON t.UserId = u.UserId"

, new { month = months, year = year });

updateoutput(getHours);



private void button2\_Click(object sender, EventArgs e)

{

var months = comboBox1.SelectedItem.ToString();

var year = comboBox2.SelectedItem.ToString();

using (var conn = CreateConnection())

{

var getProHours = conn.Query<Projects>(@"SELECT u.Name, t.TotalHoursCaptured FROM Projects AS u

JOIN (SELECT ProjectId,SUM(HoursCaptured) AS TotalHoursCaptured FROM Timeslots

WHERE DATENAME(month, Date)=@month AND YEAR(Date) = @year

GROUP BY ProjectId

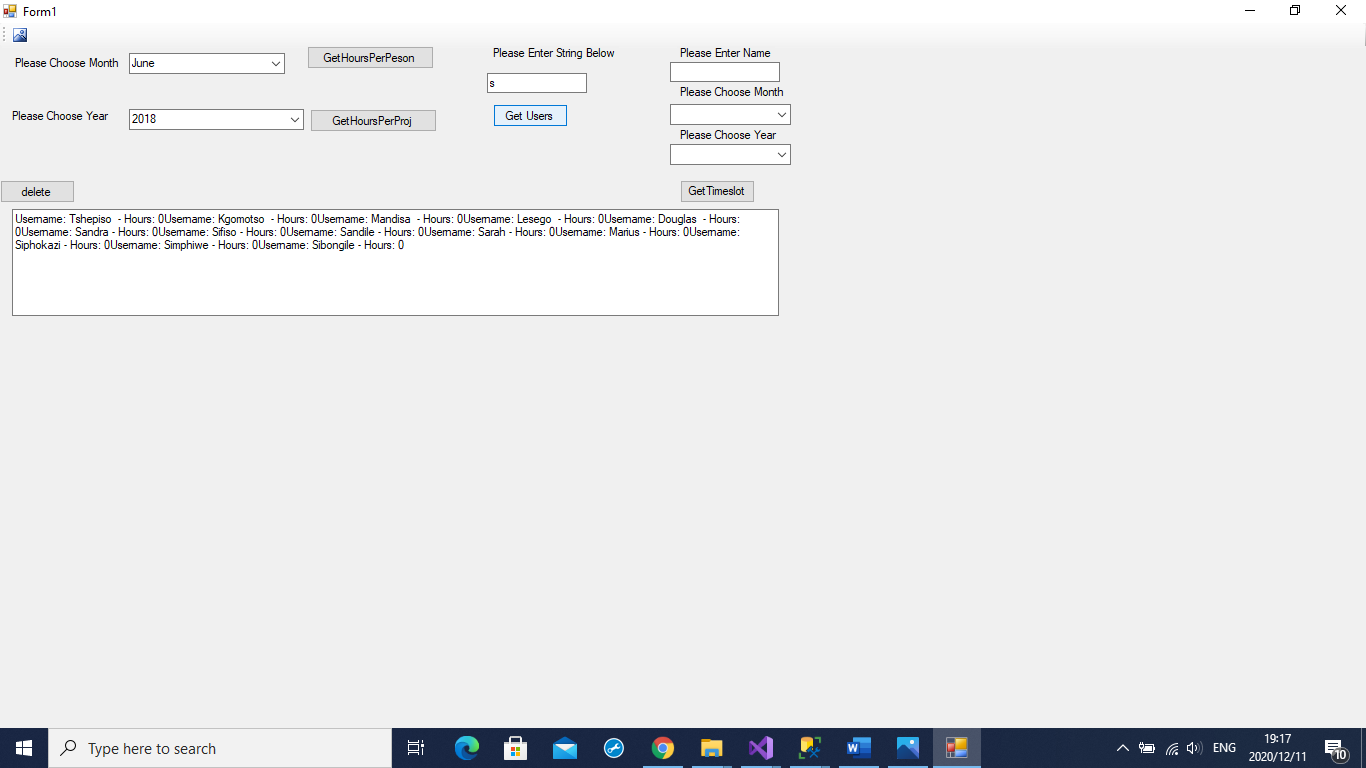
) AS t

ON t.ProjectId = u.ProjectId"

, new { month = months, year = year });

updateoutput(getProHours);

}



private void GetUsers\_Click(object sender, EventArgs e)

{

using (var conn = CreateConnection())

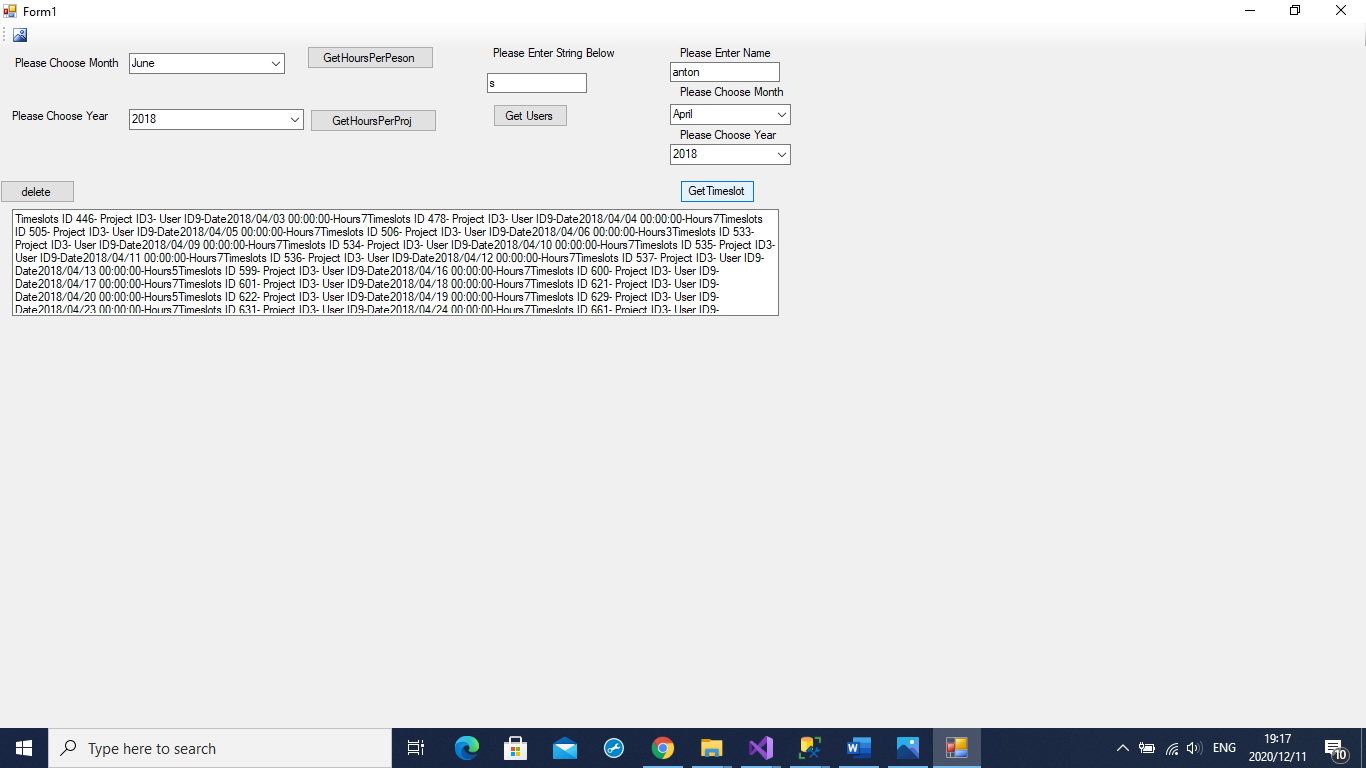
{

var getString = (@"SELECT \* FROM Users

WHERE Username LIKE @name");

updateoutput(conn.Query<Users>(getString, new { name = "%" + usersTxt.Text + "%" }));

}



private void getTimeslots\_Click(object sender, EventArgs e)

{

using (var conn = CreateConnection())

{

{

var sqlGetTopMatching = (@"SELECT t.\* FROM Timeslots AS t

JOIN (SELECT TOP (1) \* FROM Users

WHERE Username LIKE @name) AS u ON t.UserId=u.UserId

WHERE DATENAME(month, t.Date)=@month AND YEAR(t.Date) = @year");

updateoutput(conn.Query<Timeslots>(sqlGetTopMatching, new { name = "%" + nameTxt.Text + "%", month = comboBox3.SelectedItem.ToString(), year = comboBox4.SelectedItem.ToString() }));

}

private void delete\_Click(object sender, EventArgs e)

{

resultTxt.Clear();

}

### Question 3 (SO 4, AC 1, AC 2, AC 3, AC 4)

You are required to test the program that you created in question 2. Follow the specifications below;

1. The testing checks all program logic paths.
2. The testing corrects program code to eliminate errors identified through testing.
3. The testing verifies that the database access functions in the required environment.
4. The testing verifies that the database access performs according to the design requirements.

**Include your test plan and test results in your POE.**

**Please also provide a link to a zip file of the application that you developed (use any file sharing product that will provide a link that does not expire – ie. Dropbox, Google Drive, etc). This should include instructions for installation. An evaluation checklist will be completed by the facilitator after running and verifying your application.**

### Question 4 (SO 5, AC 1, AC 2)

Prepare documentation for the program code that you developed in question 2 above. The following must be produced:

* Technical document
* End user document

(10)

|  |  |
| --- | --- |
| **STUDENT NAME:** ………..…………………………….  **ID:** ……………………………………………..  **COMPANY:** ………………….……………………….. | **EVALUATION CHECKLIST**  DATE:  TIME:  Question 3 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **EVALUATION CRITERIONS** | | **Yes** | **No** | **COMMENTS OR ACTION REQUIRED** | |
|  | |  |  |  | |
| The testing checks all program logic paths. | |  |  |  | |
| The testing corrects program code to eliminate errors identified through testing. | |  |  |  | |
| The testing verifies that the database access functions in the required environment. | |  |  |  | |
| The testing verifies that the database access performs according to the design requirements. | |  |  |  | |
| **GENERAL COMMENTS:** | | | | | |
|  | | | | | |
| Date…………………….. | Time started……………….. | | | | Time completed………………. |
| **MENTOR/SUPERVISOR NAME**  **………………………………** | **MENTOR SIGNATURE**  **……………………………………** | | | | **ASSESSOR ENDORSEMENT**  **(SIGNATURE)**  **…………………………………** |