Visual Studio Windows Form Application #5
DataGrid & Transaction Views

Dr. Thomas E. Hicks
Computer Science Department
Trinity University

Use LibraryApp5 Project

Visual Studio Windows Form Application #1
Basic Form Properties

http://carme.cs.trinity.edu/thicks/3321/Handouts/VS-1-SE-Basic-Form-Tutorial.pdf

Visual Studio Windows Form Application #2
Button & TabFrame Properties

http://carme.cs.trinity.edu/thicks/3321/Handouts/VS-2-Buttons-TabFrame-Form-Tutorial.pdf

Visual Studio Windows Form Application #3
ComboBox, CheckBox, MDI Containers

http://carme.cs.trinity.edu/thicks/3321/Handouts/VS-3-ComboBox-CheckBox-MDI-Containers-Tutorial.pdf

Visual Studio Windows Form Application #4
Views, ViewMode, & EditMode

http://carme.cs.trinity.edu/thicks/3321/Handouts/VS-4-Views-ViewMode-EditMode-Tutorial.pdf

Purpose

1] The purpose of this tutorial is to show how to create, and use, a basic Windows Application with Visual Studio. Buttons and TabFrames will be added to the TextBoxes & Labels used in the first tutorial. Additional basic form Guidelines will be introduced.

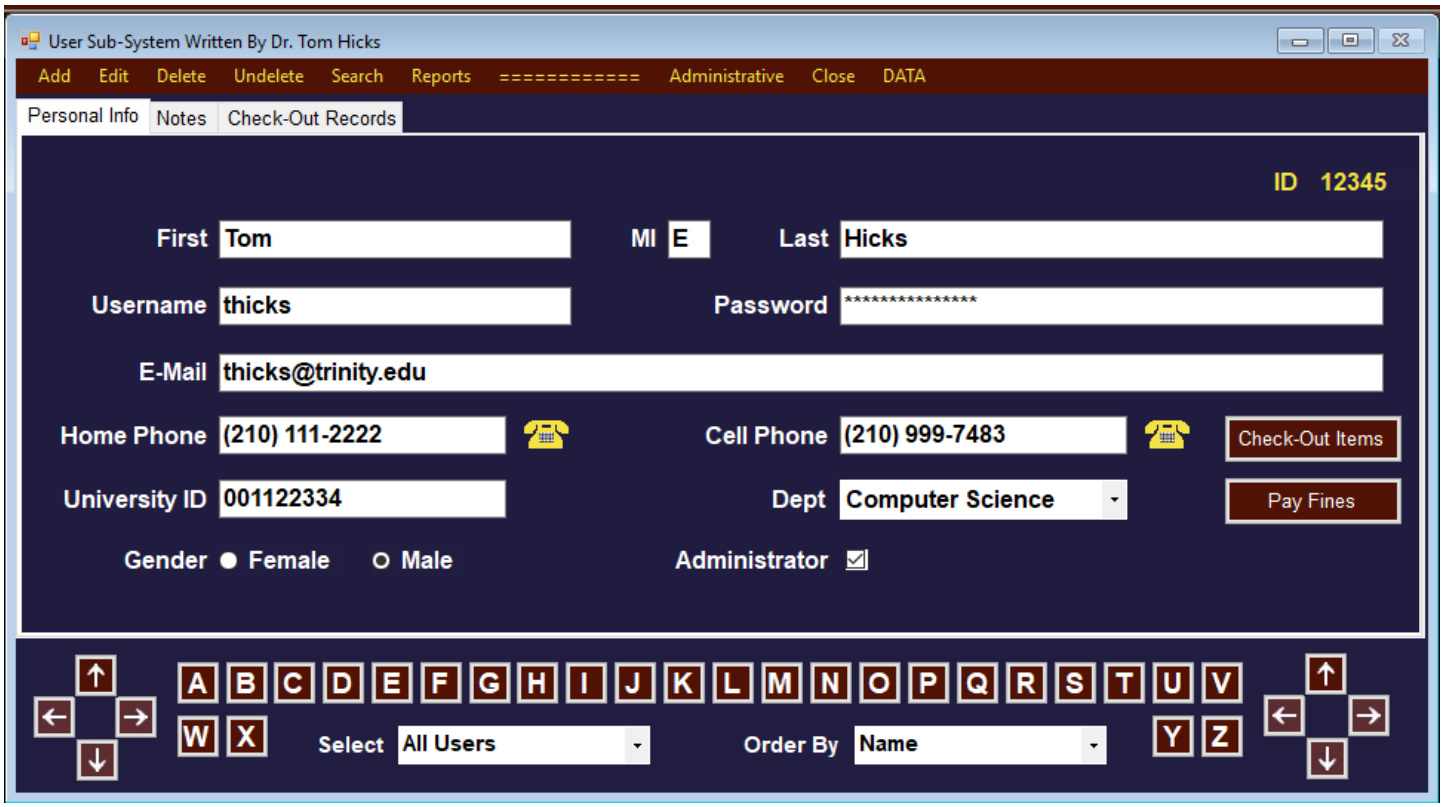
2] Topics included are:

A) Make The Temporary Backups
B) Rename The Application
C) Alter FillFormVariables – Use Your User Record
D) Check-Out
E) DataGridView

F) DataGridView In Prototype
G) Check-Out Detail
H) ViewMode Revisited ➔ Testing = false
I) EditMode Revisited ➔ Testing = false
J) AddMode Revisited ➔ Testing = false
K) Make A Backup 6

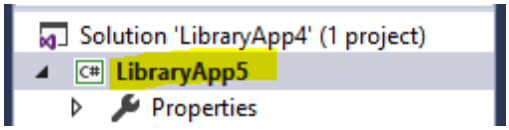
Last Form

1] Our form, at the endo of LibraryApp4, looked like the following:

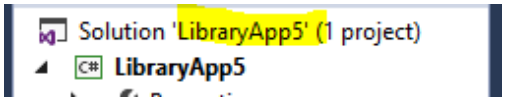


Rename The Application

1] The Folder Should Be Called LibraryApp5. Open Project → **LibraryApp5**. Right-Mouse Click on the LibraryApp4 ➔ Select Rename. Change the name to LibraryApp5. (See Below)



2] Select Solution 'LibraryApp4' ➔ change to LibraryApp5 (See Below)



Alter FillFormVariables – Use Your User Record

1] In the next two tutorials, we are going to simulate some simple searching; we will include searching by Name (FullName). We have practiced doing database searches were we might want to display all of the information about all of the users whose FullName started with **Hicks, T**.

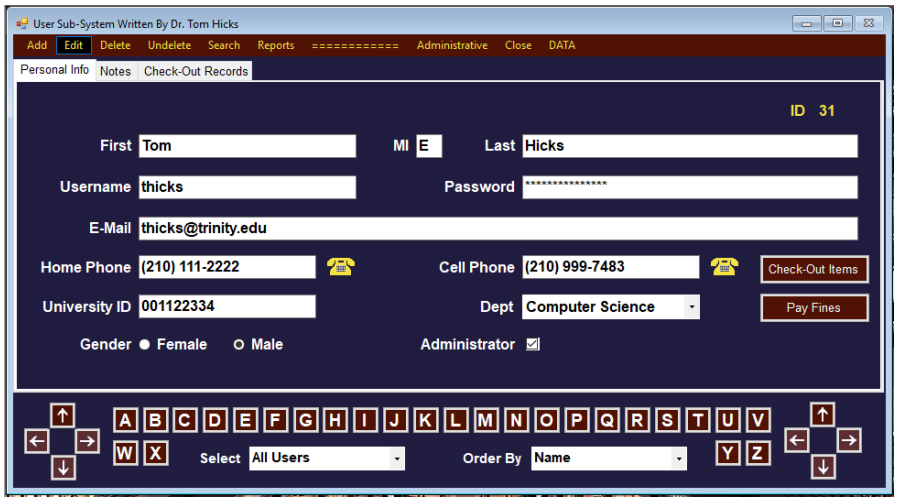
```
SELECT *
FROM User
WHERE (FullName LIKE "Hicks, T%");
```

Message | Result1 | Profile | Status

ID	First	MI	Last	FullName	DeptID	GenderID	UniversityID	Email	UserName	Password
31	Tom	E	Hicks	Hicks, Tom E	1	2	13131	thicks@)trinity.edu	thicks	trinity31

2] I am going to ask you to simulate searching for yourself. Please make sure that all of the information in your view form is about you → you can see my information in the form below.

3] Alter FillFormVariables() to use the the data in your record. Select an ID of your choice. If this were a database design course, I would teach you to fill the form from the database user table.



4] Edit your database. Change the data in record ID = 10 to match that in your form.

Check-Out Transactions → Database

1] We are about to start work on our first Transactional view. We are going to simulate the Check-Out records for the user.

2] Creating these records with a database are quite simple. We are not going to simulate the fill of the datagrid manually so that the user will see what it would look like, but the database queries to do this in our final application are not all that difficult. We shall review that below.

3] The check out transactions involve three tables : Users, CheckOut, and Media. We would not want to include deleted records in our queries.

4] Using your database, write, and execute, a query to display the **CheckOut** table layout (list all of the fields)

```
DESCRIBE CheckOut;
```

Message | Result1 | Profile | Status

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	(Null)	auto_increment
userid	int(11)	YES		(Null)	
dayout	int(11)	YES		(Null)	
monthout	int(11)	YES		(Null)	
yearout	int(11)	YES		(Null)	
dateno	int(11)	YES		(Null)	
deleted	char(1)	YES		(Null)	

5] Write, and execute, a query to display all of the information about the first ten check-out transactions. We definitely do not want to include any deleted transactions,

```
SELECT *
FROM CheckOut
WHERE (Deleted = "F")
Limit 0,10;
```

Message | Result1 | Profile | Status

id	userid	dayout	monthout	yearout	dateno	deleted
1	23	10	1	2018	20180110	F
2	98	5	10	2018	20181005	F
3	45	5	6	2017	20170605	F
4	75	27	6	2018	20180627	F
5	88	22	10	2018	20181022	F
6	72	21	1	2018	20180121	F
7	5	7	6	2017	20170607	F
8	42	21	12	2017	20171221	F
9	84	10	12	2017	20171210	F
10	47	23	4	2018	20180423	F

6] Write, and execute, a query that will display the number of Check-Out Transactions you have had. I will do it for me.

```
SELECT COUNT(*) AS NoCheckOutTransactions
FROM CheckOut
WHERE (Deleted = "F") AND (UserID = 31);
```

Message | Result1 | Profile | Status

NoCheckOutTransactions
8

7] Write, and execute, a query that will display all of the information about your last three Check-Out Transactions listing the most recent first. I will do it for me. You will be using User.ID = 10.

```
SELECT *
FROM CheckOut
WHERE (Deleted = "F") AND (UserID = 31)
Order By DateNo DESC
Limit 0,3;
```

Message | Result1 | Profile | Status

id	userid	dayout	monthout	yearout	dateno	deleted
100	31	22	12	2018	20181222	F
262	31	25	8	2018	20180825	F
736	31	17	4	2018	20180417	F

8] Adjust the query above → to display the following for UserID = 10

User ID	Transaction ID	Date
31	100	12/22/2018
31	262	8/25/2018
31	736	4/17/2018

9] For the query above, you should have something like the following → using your User.ID = 10.

```
1  SELECT UserID AS "User ID", ID AS "Transaction ID", CONCAT(MonthOut, "/", DayOut, "/", YearOut) AS "Date"
2  FROM CheckOut
3  WHERE (Deleted = "F") AND (UserID = 31)
4  Order By DateNo DESC
5  Limit 0,3;
```

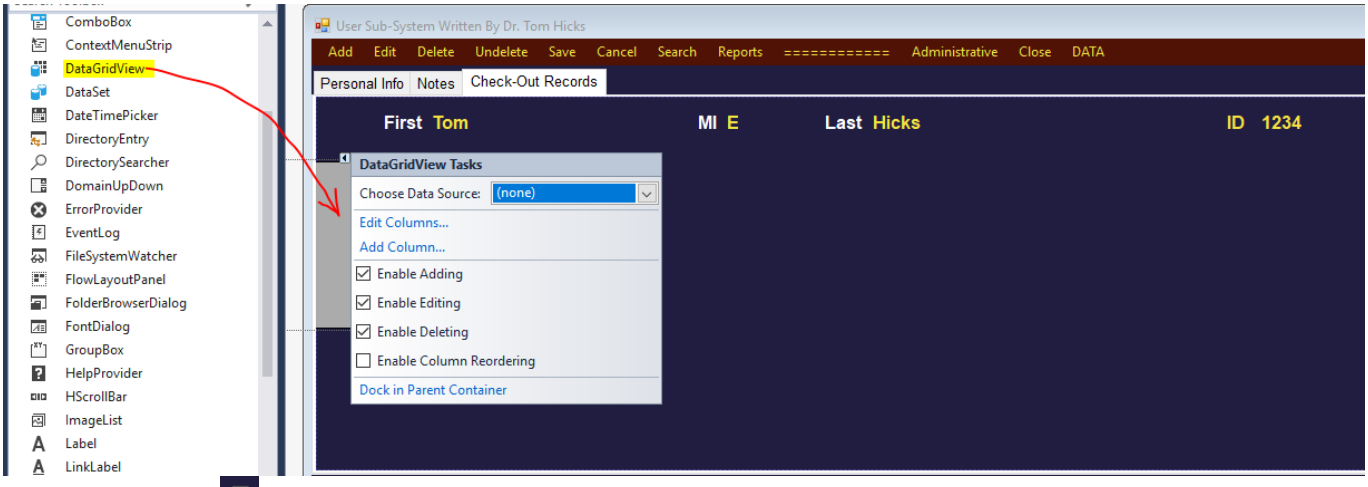
Message | Result1 | Profile | Status

User ID	Transaction ID	Date
31	100	12/22/2018
31	262	8/25/2018
31	736	4/17/2018

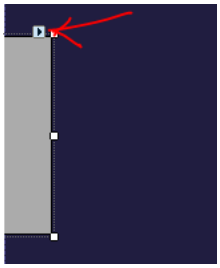
10] The purpose of the database work above is to show you how easy it is to form a database query which will collect the CheckOut transactional information for the current user, This query can be loaded directly into the DataGridView object.

DataGridView

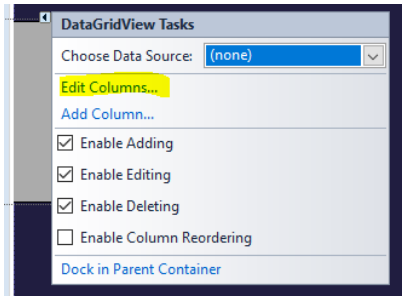
1] Drag a DataGridView from the ToolBox to the Check-Out Records tab.



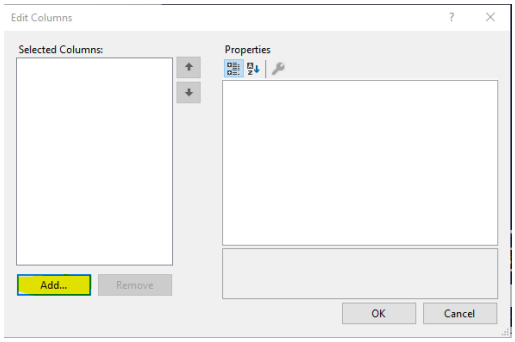
2] Click on the  to expand the options.



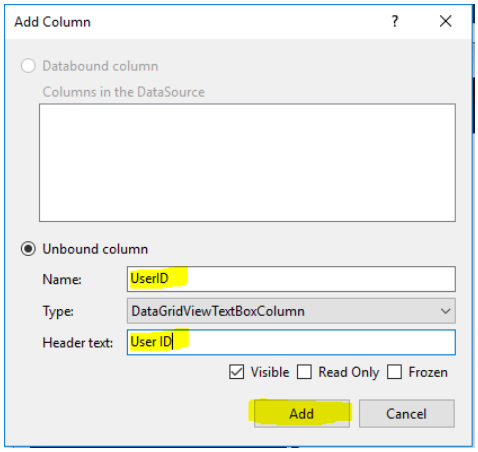
3] Push the Edit Columns button.



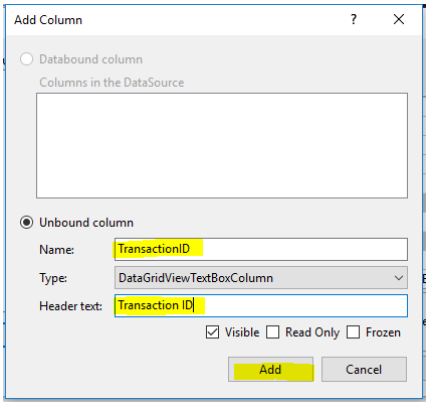
4] Push the Add button.



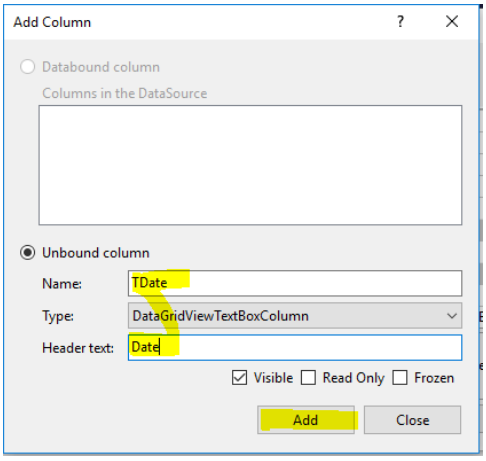
5] Add a TextBox column UserID with header "User ID"



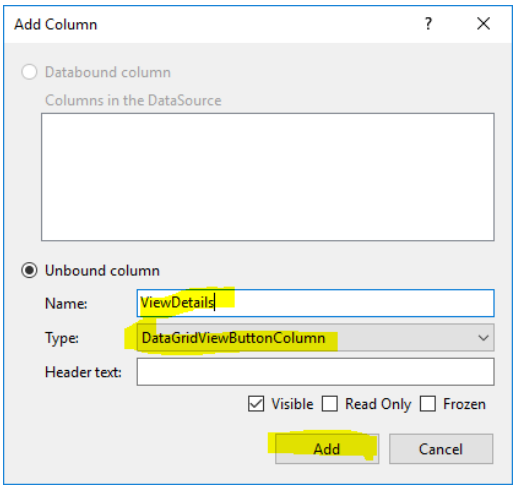
6] Add a TextBox column TransactionID with header "Transaction ID"



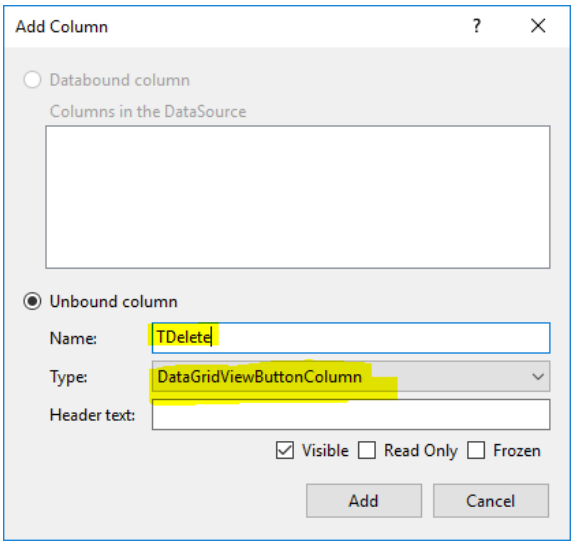
7] Add a TextBox column TDateD with header "Date"



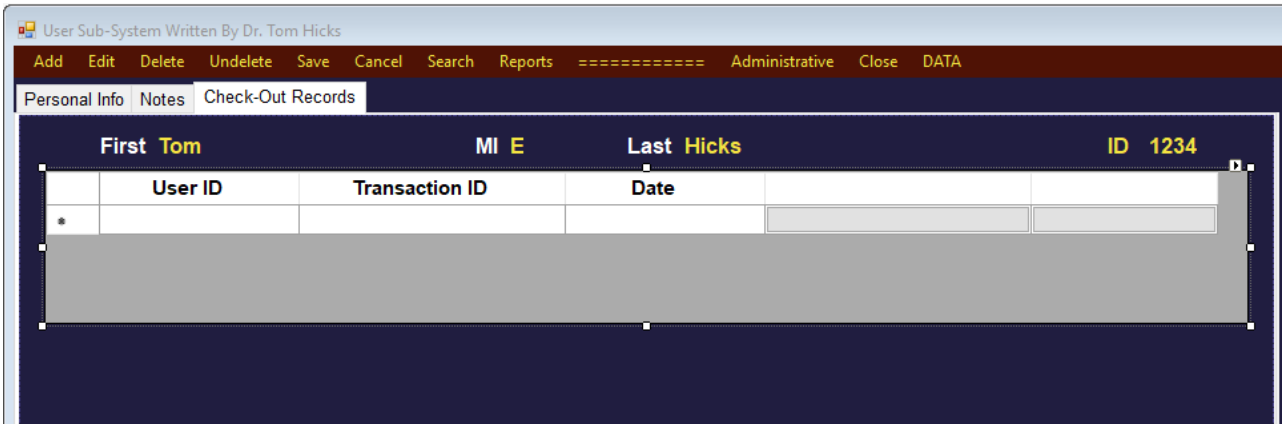
8] Add a Button column ViewDetails with a blank header ➔ When implement the database query, much like the one above, I would place "View Details" on the button.



8] Add a Button column TDelete with a blank header ➔ When implement the database query, much like the one above, I would place "Delete" on the button.

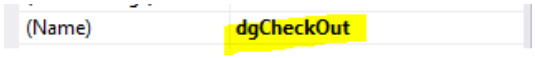


8] Play with the controls until you get a layout that looks like the following: (Use Arial Font). I would even suggest coloring the buttons once you add the query to the database.



9] When formatting the grid data, one of the controls you will want to change will e ColumnHeaderDefaultCellStyle.

10] Name the DataGrid as shown below.



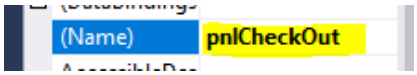
Simulating The DataGridView In The Prototype

1] Providing the necessary data and hooking up the database stuff is way too much work to do for a prototype. This is work that will be done during the development stage when implementing the project.

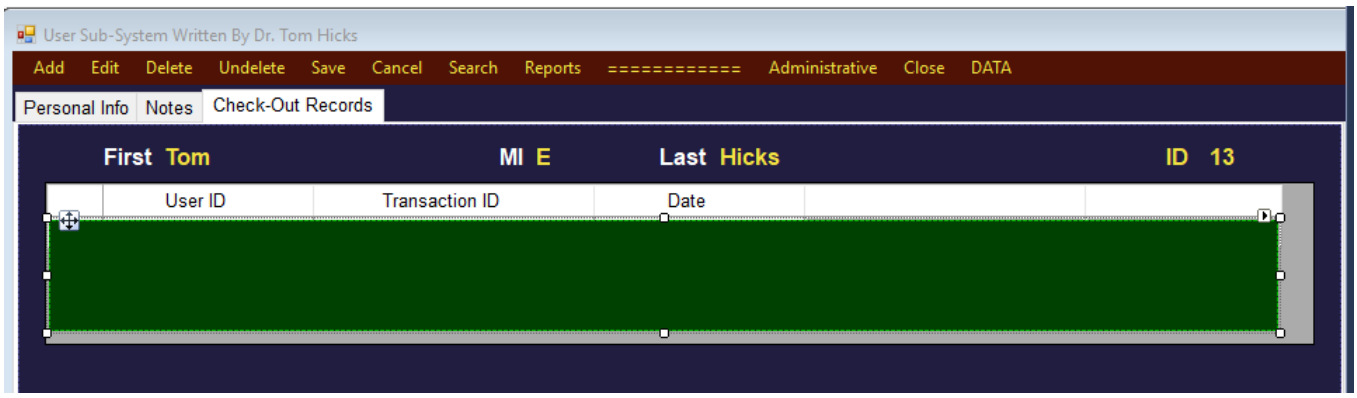
2] That said, leaving the blank grid, in the graphic above is not something the stakeholders can visualize. What we can do is simulate "what will happen" with maybe 2 or 3 transactions.

3] Since I can easily hide, or show a panel, I will use a panel for the simulated data.

4] Drag a panel to the Check-Out Records tab. I would place it just below the Grid header line. I have colored my panel green → there is no need for you to do so. Name the panel pnlCheckOut.



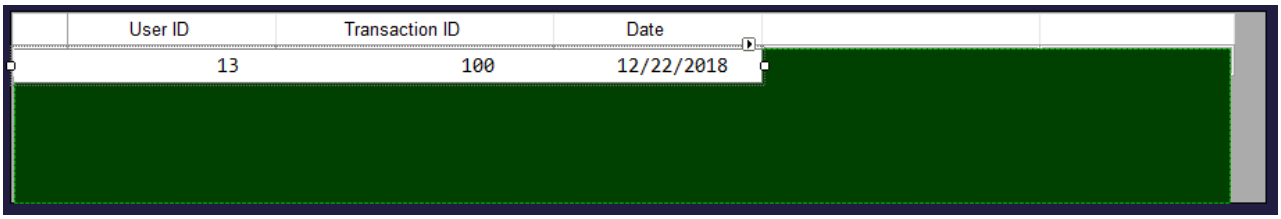
5] You can see my panel below.



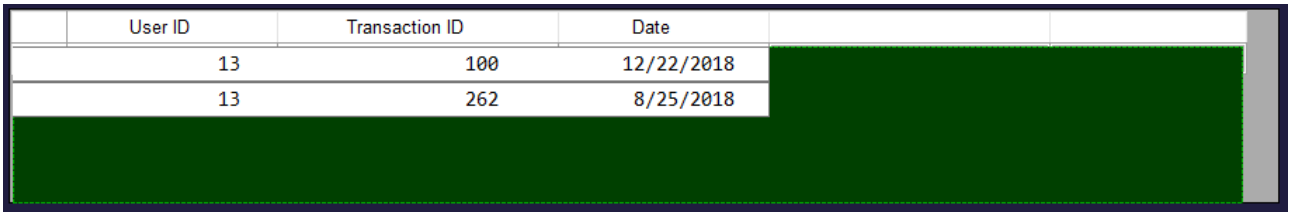
6] I am going to try to model the two most recent transactions for this user.

User ID	Transaction ID	Date
31	100	12/22/2018
31	262	8/25/2018
31	736	4/17/2018

7] Drag a TextBox on top of your form. Use Consolas font → because it will line up well in columns. Place the data from your most recent check-out transaction at the top as shown below (use your data); if our database does not include at least two-three transactions for you, make up your own data but do not use mine.

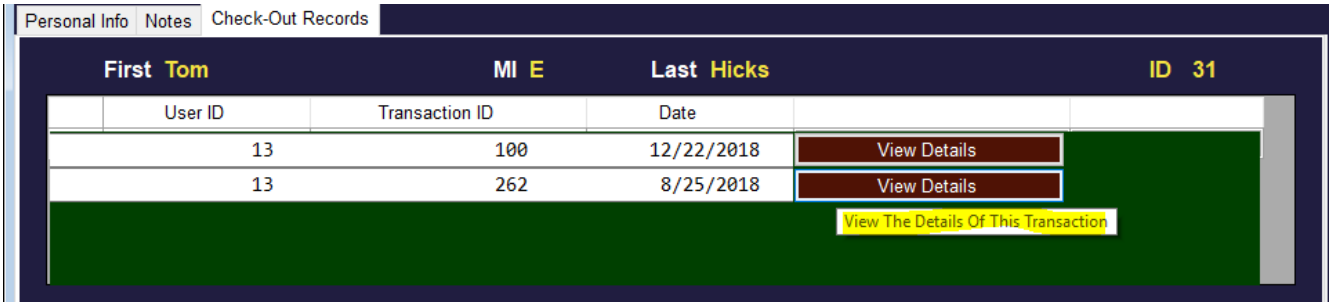


8] Copy/Paste another copy of your TextBox to your form. Place the data from your second most recent check-out transaction at the top as shown below (use your data). Notice how this font forms nicely formatted columns.

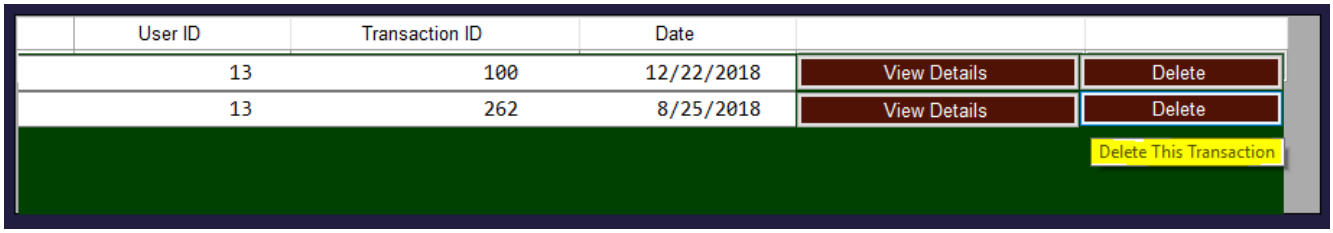


GUIDELINE # 37➔ DataGrids will often be used to implement transactions. The prototype should allow the user to visualize the project yet to come.

9] Add **btnViewDetails1** and **btnViewDetails2** to your form ➔ make the tool tips "View The Details Of This Transaction". Use the color scheme of your form.



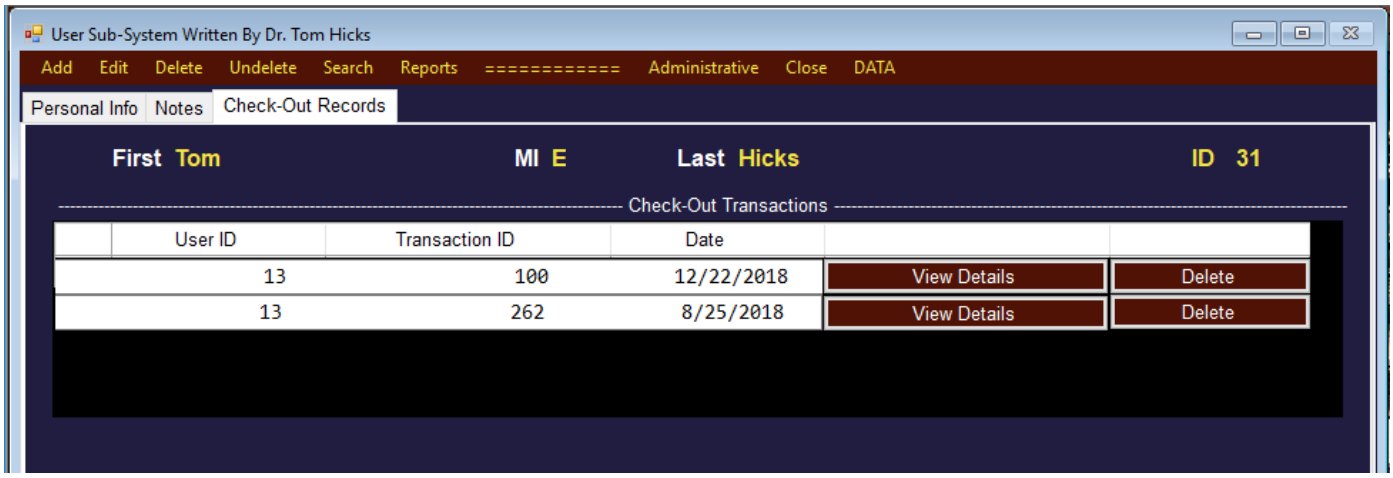
10] Add **btnDelete1** and **btnDelete2** to your form ➔ make the tool tips "Delete This Transaction". Use the color scheme of your form. You may make the Delete button a different color if you like?



11] Make sure that you have at least enough room for 4 records in the dgCheckOut. When there are more than 4 transctions, when filling the DataGrid from a database, scroll bars will be automatically placed on the right. Make sure that you leave room for them. There have been times when I have pasted a graphic, of scroll bars, in the panel so that my stakeholders can better visualize the final project.

12] You need not insert the graphic on your prototypes, but you should tell the user something like:
(1) the 4 most recent records will always be shown on top
(2) when they have more than 4 records, there will be scroll bars to the right
(3) they will be able to use those scroll bars to access older records

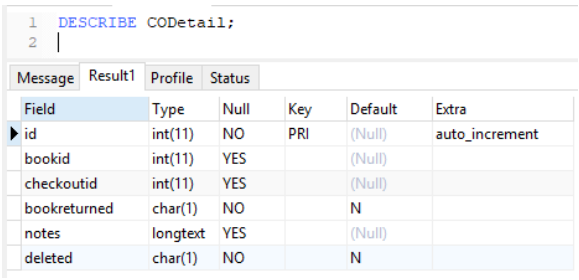
13] We want things to be very clear to our users. Place a very small **"Check-Out Transactions"** label above your DataGrid as shown below. Note that I also made the DataGrid and Panel background Black to hide it a bit.



GUIDELINE # 38➔ As a general rule, each of the buttons on your Protoype, with the exception of Reports, should do something or show something.

Check-Out Detail → Database

1] Write, and execute, a query to display the CODetail table layout (list all of the fields)



```
1  DESCRIBE CODetail;
2
```

Message | Result1 | Profile | Status

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	(Null)	auto_increment
bookid	int(11)	YES		(Null)	
checkoutid	int(11)	YES		(Null)	
bookreturned	char(1)	NO		N	
notes	longtext	YES		(Null)	
deleted	char(1)	NO		N	

2] Write, and execute, a query to display the number of records in CODetails. There are 10,000 records

```
SELECT COUNT(*) AS NoCODetailRecords
FROM CODetail;
```

Message | Result1 | Profile | Status

NoCODetailRecords
10000

3] Write, and execute, a query to display all of the information about the first ten records in CODetails.

```
SELECT *
FROM CODetail
LIMIT 0,10
```

Message | Result1 | Profile | Status

id	bookid	checkoutid	bookreturned	notes	deleted
1	37	805	F		F
2	93	196	F		F
3	22	474	F		F
4	74	242	F		F
5	2	334	F		F
6	64	160	F		F
7	91	34	F		F
8	47	206	F		F
9	65	609	F		F
10	11	702	F		F

4] My most recent check-out transaction is #100. Write, and execute, a query to display the number of valid records associated with Check-Out Transaction 100.

```
SELECT  COUNT(*) AS NoBooksCheckedOutTransction100
FROM CODetail
WHERE (codetail.Deleted = "F") AND (CheckOutID = 100)
```

Message | Result1 | Profile | Status

NoBooksCheckedOutTransction100
11

5] My most recent check-out transaction is #100. Write, and execute, a query to display all of the valid check-out details about that transaction. Note that there are 11.

```
SELECT  *
FROM CODetail
WHERE (codetail.Deleted = "F") AND (CheckOutID = 100)
```

Message | Result1 | Profile | Status

id	bookid	checkoutid	bookreturned	notes	deleted
656	100	100	F		F
699	38	100	F		F
1252	38	100	F		F
2295	78	100	F		F
4253	84	100	F		F
4612	18	100	F		F
5327	27	100	F		F
5952	17	100	F		F
6306	53	100	F		F
6412	50	100	F		F
7174	86	100	F		F

6] If we wanted to create a table that includes the User ID and all of the detail associated with Transaction 100, we would have to join CheckOut and CODetail. Look at the results of the following query.

```
SELECT UserID AS "CheckOut.UserID", CheckOut.ID AS "CheckOut.ID",
       CODetail.CheckOutID  AS "CODetail.CheckOutID", BookID, BookReturned
FROM CheckOut, CODetail
WHERE (codetail.Deleted = "F") AND (CheckOut.ID = codetail.CheckOutID)
      AND (checkout.ID = 100)
```

Message | Result1 | Profile | Status

CheckOut.UserID	CheckOut.ID	CODetail.CheckOutID	BookID	BookReturned
31	100	100	100	F
31	100	100	38	F
31	100	100	38	F
31	100	100	78	F
31	100	100	84	F
31	100	100	18	F
31	100	100	27	F
31	100	100	17	F
31	100	100	53	F
31	100	100	50	F
31	100	100	86	F

7] The query above emphasizes the join associated with the two tables ➔ {not include deleted CODetail}
`(CheckOut.ID = codetail.CheckOutID)`

8] Write a query which will display the Transction ID, the Book ID, Book Returned, and Book Title for those items checked out in Transction 100. Note that I checked out two copies of Differential Equations → one for myself and one for my student assistant.

```
SELECT CheckOutID AS "Transaction ID", BookID AS "Book ID", BookReturned AS "Book Returned",
       Title AS "Book Title"
FROM CODetail, Book
WHERE (codetail.Deleted = "F") AND (CheckOutID = 100)
      AND (BookID = book.ID)
ORDER BY Title
```

Message | Result1 | Profile | Status

Transaction ID	Book ID	Book Returned	Book Title
100	38	F	A First Course in Partial Differential Equations
100	38	F	A First Course in Partial Differential Equations
100	27	F	A Transition to Advanced Math
100	86	F	Beginning Software Engineering
100	53	F	Biology
100	50	F	Computer Organization and Design
100	100	F	Data Communications and Networking
100	17	F	Optics
100	18	F	Quantum Mechanics
100	84	F	Software Testing Foundations, 4th Edition: A Study Guide for the Certified Tester Exam
100	78	F	Testing Computer Software

9] Write a query which will display the Transction ID, the Book ID, Book Returned, and Book Title for those items checked out in Transction 262. I want to use this query data in my prototype.

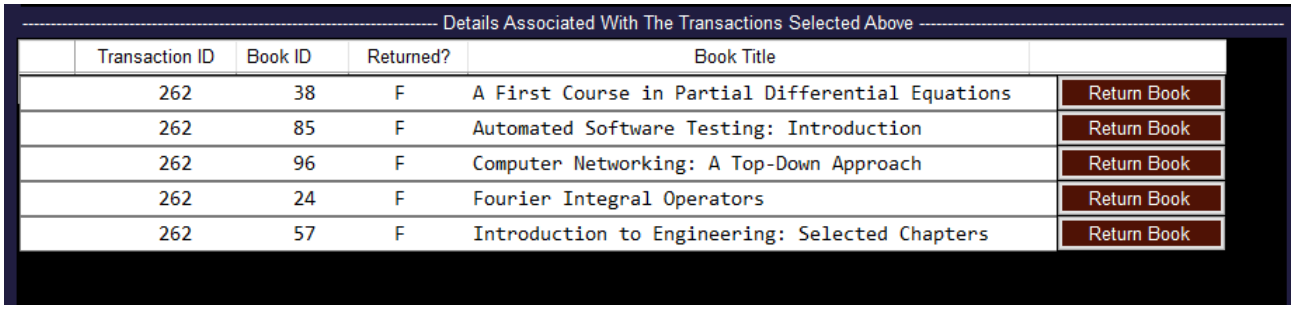
```
SELECT CheckOutID AS "Transaction ID", BookID AS "Book ID", BookReturned AS "Book Returned",
       Title AS "Book Title"
FROM CODetail, Book
WHERE (codetail.Deleted = "F") AND (CheckOutID = 262)
      AND (BookID = book.ID)
ORDER BY Title
```

Message | Result1 | Profile | Status

Transaction ID	Book ID	Book Returned	Book Title
262	38	F	A First Course in Partial Differential Equations
262	85	F	Automated Software Testing: Introduction, Management, and Performance
262	96	F	Computer Networking: A Top-Down Approach
262	24	F	Fourier Integral Operators
262	57	F	Introduction to Engineering: Selected Chapters
262	75	F	Lessons Learned in Software Testing: A Context-Driven Approach
262	34	F	Methods of Applied Mathematics
262	55	F	Organic Chemistry
262	61	F	Organic Chemistry Solutions manual
262	4	F	Physic Problem Solver
262	30	F	Precalculus and It's Applications
262	36	F	Principles of Mathematical Analysis
262	94	F	Software Engineering: The Current Practice
262	81	F	Testing and Quality Assurance for Component-Based Software

10] We are about to add a second DataGrid to our form. It is to have at least 5 rows. You may have to increase the height of your form.

11] Create a second DataGrid ➔ place it on the bottom of Check-Out Records Use at least 4-5 of the check-out detail from one of your recent Transactions. Make up the data if there is not sufficient data in the database. Do not use my records.

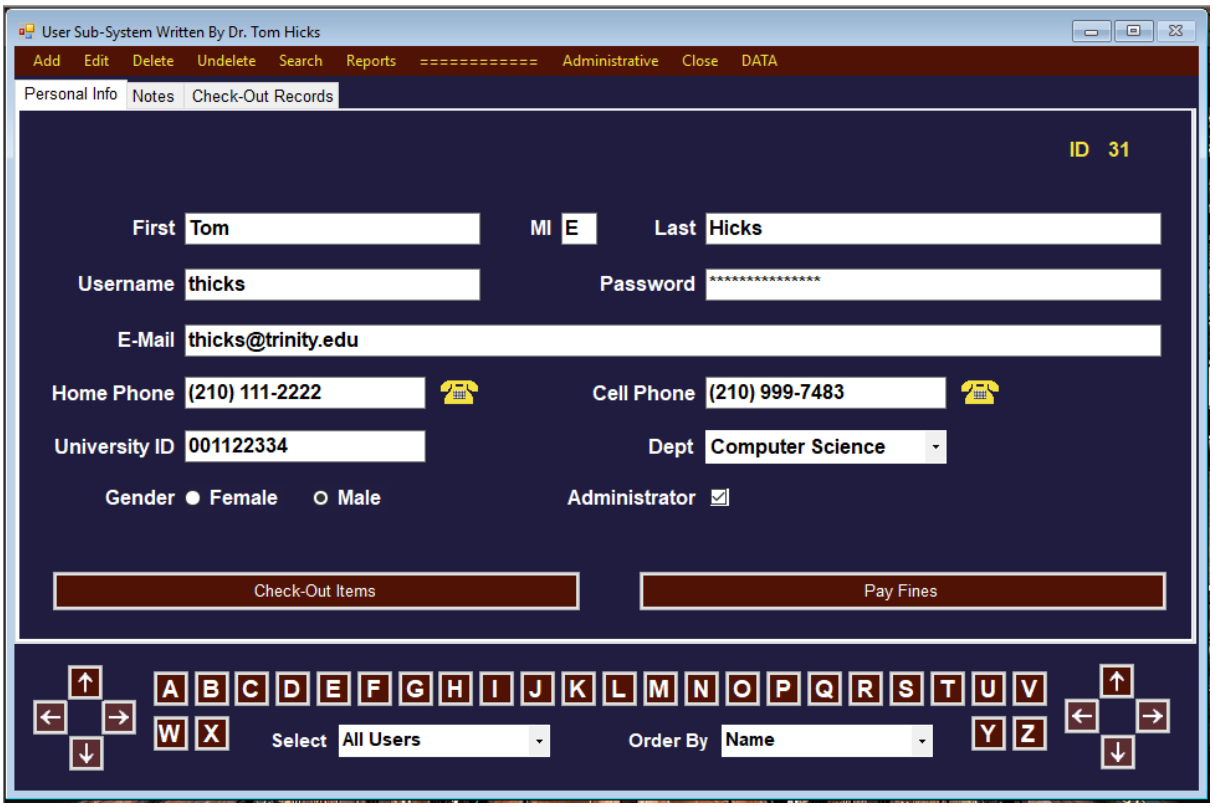


12] We want things to be very clear to our users. Place a very small **"Details Associated With The Transactions Selected Above"** label above your second DataGrid as shown below. Note that I also made the DataGrid and Panel background Black to hide it a bit.

ViewMode Revisited ➔ Testing = false

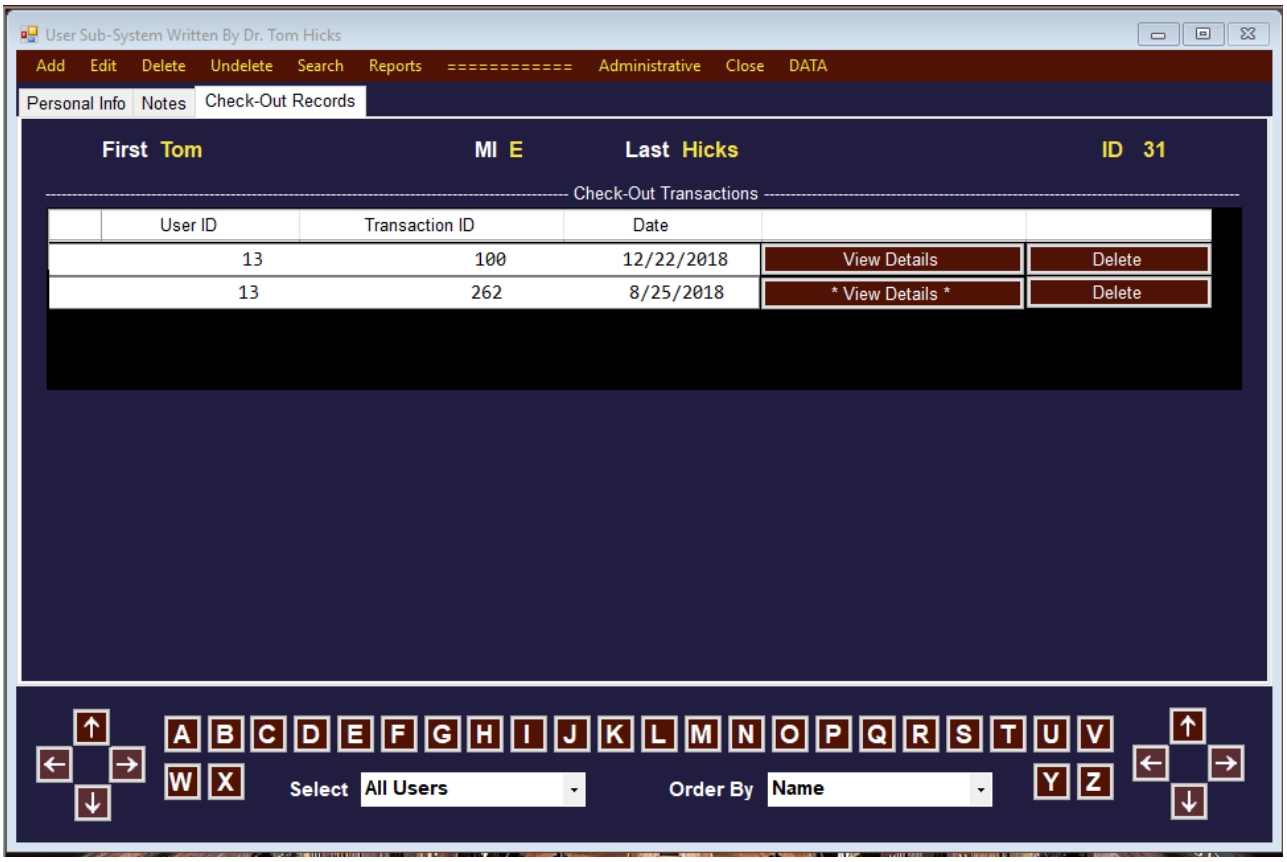
1] I did have to increase the form height in order to accommodate both of the DataGrid containers. I had to move the things around a bit to fill the space better. Were this a real world application, I would have many more items to place on the Personal Info Tab ➔ filling the space would not be a problem.

2] Adjust your form so that the Personal Info Tab looks much like the following:

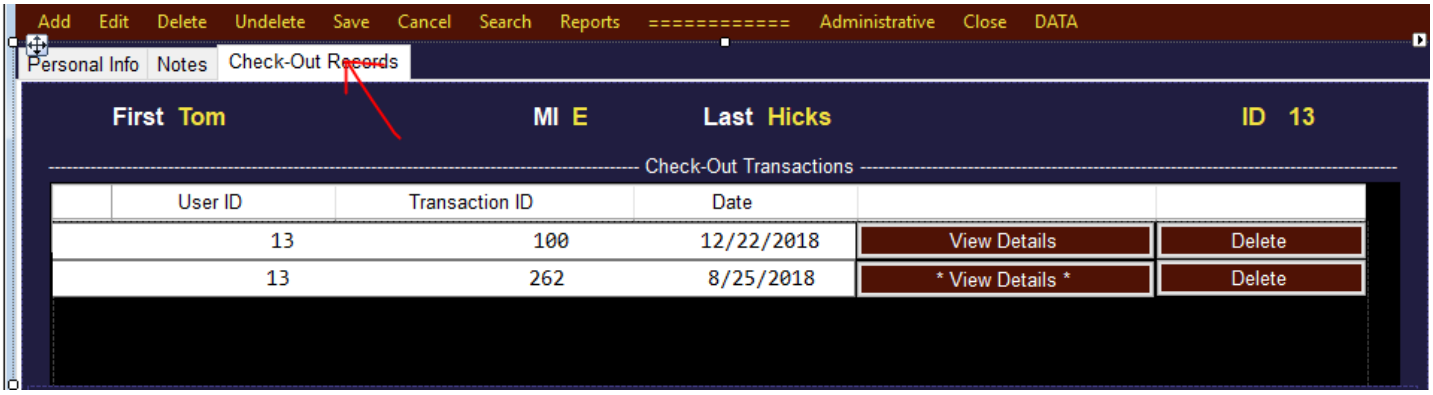


3] Adjust your form so that the Notes Tab looks much like the following:

4] Adjust your form so that the Check-Out Records Tab begins with data that looks much like the following:



5] Double-Click on the Check-Out-Records Tab to create a Selected Event for the tab. Hide the second DataGrid, and all Data.



6] Create an on-click event for one of your View Details buttons. Make the text ➔ *** View Details *** ➔ This should make it easier to remember which button is special during your prototype demonstrations.

7] When this button is pushed, reveal the bottom DataGrid on your form. It should look much like the following with your detail. Note that I am pushing the button beside Transaction ID = 262 ➔ Not that all 5 of my details (at the bottom) are associated with Transaction ID = 262.

8] The idea is to let the Stakeholder get a brief glimpse of what they are going to get and how it will work. You need not make all of your Transactions (at the top) work.

9] If you hide the second DataGrid with the Enter event as recommended earlier, you can push the Personal Info Tab and then immediately push the Check-Out Records Tab to make the second DataGrid go away again during the demo ➔ I often have to show this to the users at least two or three times to get them to understand.

10] During the Prototype Demo, it is important to emphasize that most Library Applications would require the librarian to

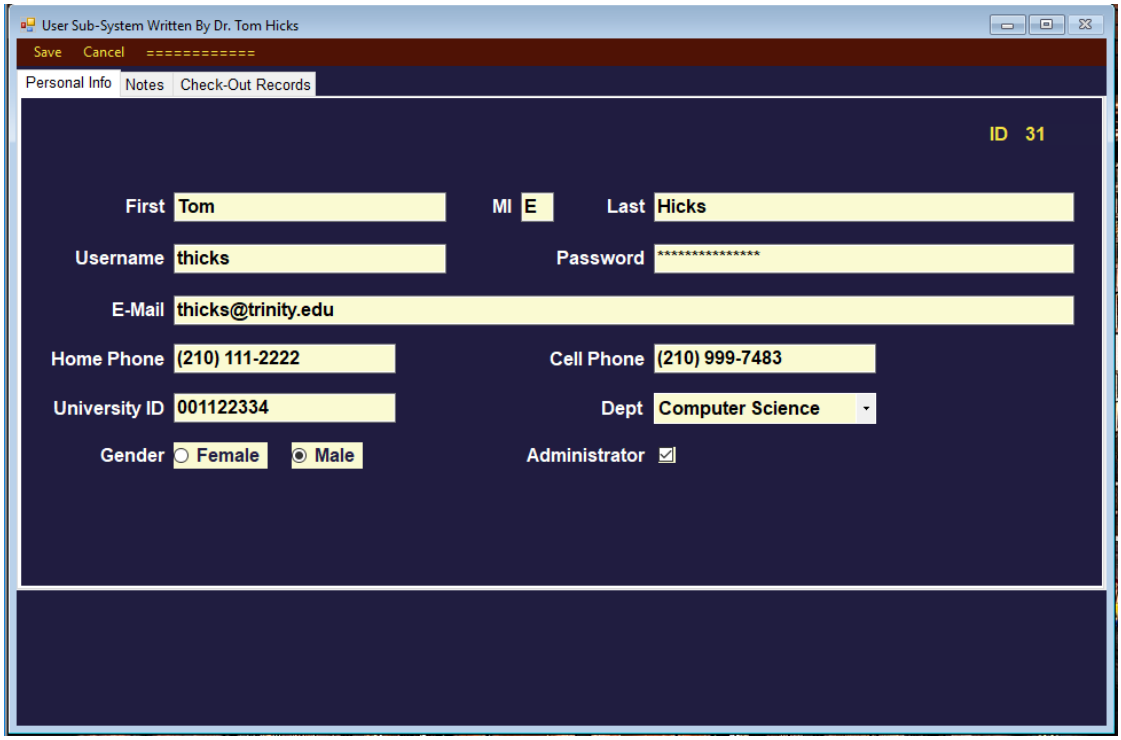
(1) Go to the Check-Out Sub-System
(2) Know, and Enter the User ID ➔ to search for Transactions for this User
(3) Find the Transaction ID
(4) Know, and Enter the Transaction ID ➔ to search the Details for this Transaction

(5) AND By The Way → You would still have load additional screens and enter additional data to Delete the Transaction
(6) AND By The Way → You would still have load additional screens and enter additional data to Return the Book

11] Make sure that you use the Prototype Demo to emphasize the many things that your solution does well. A good application should be designed to save the user steps/time.

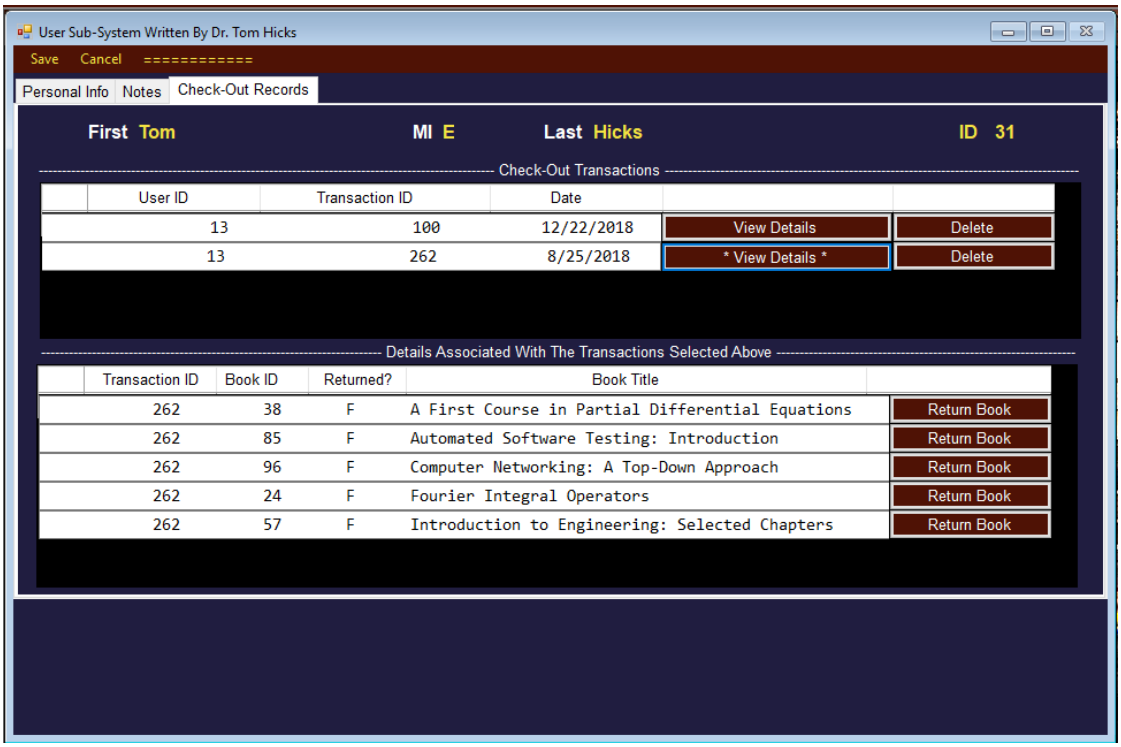
EditMode Revisited ➔ Testing = false

1] Make any changes to make your application consistent with the following EditMode processing. Adjust your form so that the Personal Info Records Tab begins with data that looks much like the following:



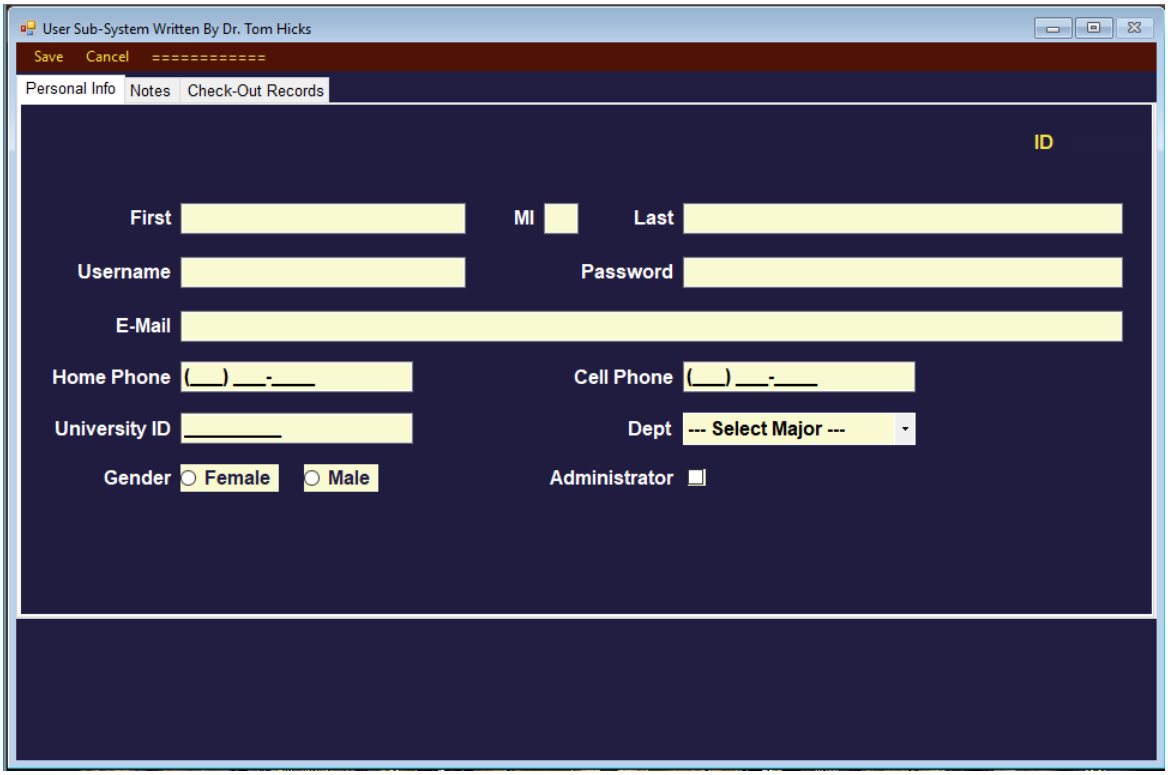
2] Adjust your form so that the Notes Records Tab begins with data that looks much like the following:

3] When Editing, you need do nothing special for on the Check-Out Records Tab

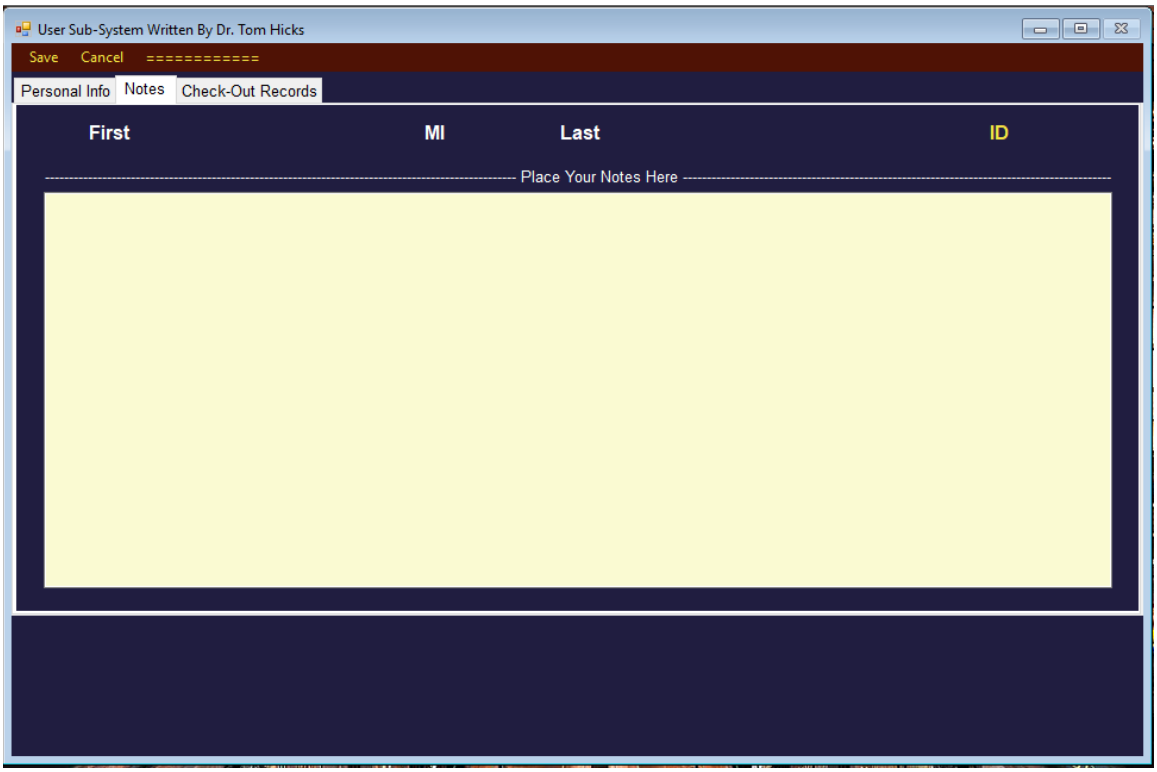


AddMode Revisited ➔ Testing = false

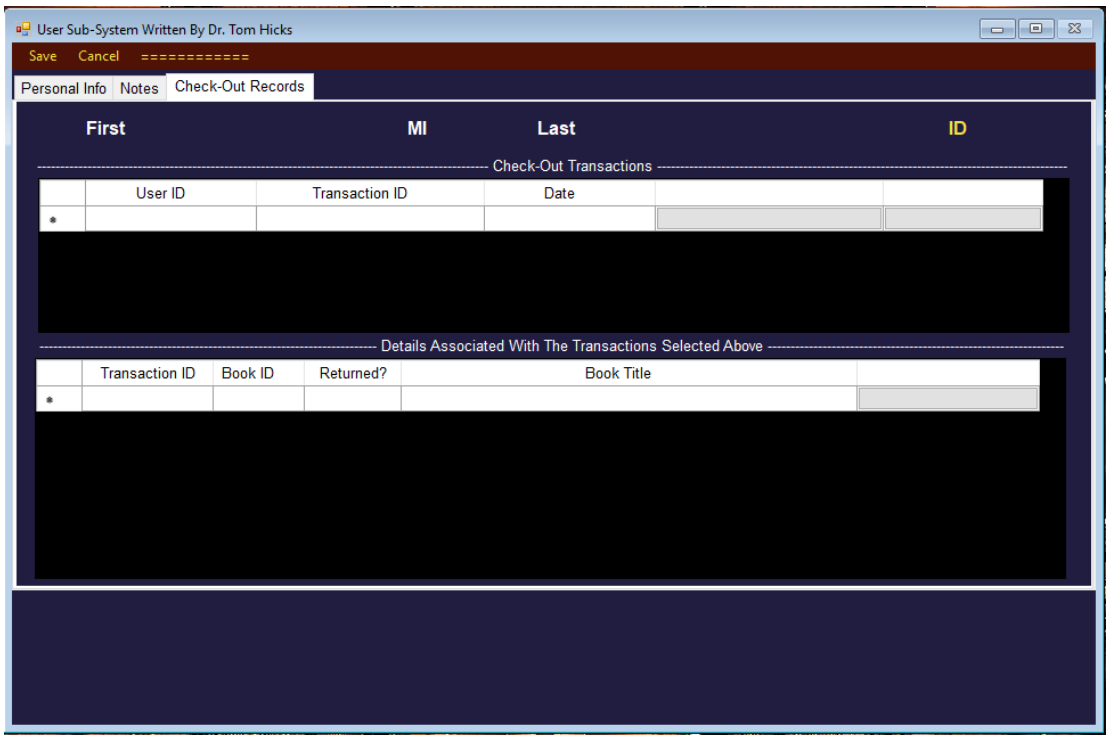
1] Make any changes to make your application consistent with the following AddMode processing. Adjust your form so that the Personal Info Records Tab begins with data that looks much like the following:



2] Adjust your form so that the Notes Tab begins with data that looks much like the following:



3] Adjust your form so that the Check-Out Records Tab begins with data that looks much like the following:



4] You might want to add something like the following when in ViewMode()

```
// --------------------- Transaction Panels --------------------------
if (pf.UserInAddMode)
{
    pnlCheckOut.Show();
    pnlDetails.Show();
}
```

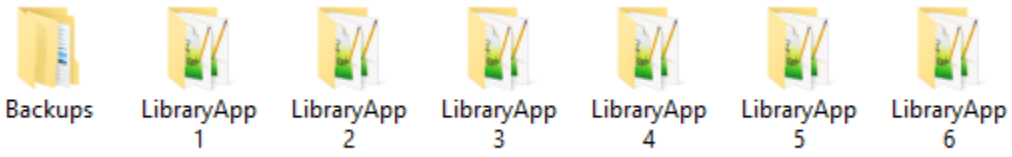
5] You might want to add something like the following when in EditMode()

```
// --------------------- Transaction Panels --------------------------
if (pf.UserInAddMode)
{
    pnlCheckOut.Hide();
    pnlDetails.Hide();
}
```

Make A Backup 6

1] It is always a good idea to make periodic backups about every half hour, or every hour, in case you mess things up so badly that recovery is difficult!

2] We would hate to lose all of the work above. Exit Visual Studio Make a backup of LibraryApp5 ➔ Call it **LibraryApp6**



3] Copy LibraryApp1, LibraryApp2, LibraryApp3, LibraryApp4, LibraryApp5 & LibraryApp6 to your personal computer.

4] Copy LibraryApp1, LibraryApp2, LibraryApp3, LibraryApp4, LibraryApp5 & LibraryApp6 to your flash drive.