Restaurant Point of Sale (RPOS) Application

By

Luke Humbard and Thomas Baum

Submitted to the
Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Engineering Technology

Restaurant Point of Sale (RPOS) Application

By

Luke Humbard and Thomas Baum

Submitted to the
Faculty of the Information Engineering Technology Program
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Engineering Technology

Luke Humbard	Date
Thomas Baum	Date
Dr. Hazem Said, Faculty Advisor	Date
James F. Sullivan, Department Head	Date

Table of Contents

List of Figures

List of Tables

Abstract

RPOS is an application for Bender's restaurant located in Indianapolis Indiana. Bender's currently has no restaurant system except a cash register and credit card transaction machine. The owner of Bender's has commissioned for a restaurant point of sale system. The system they would like to see is one with kitchen and server communication and capable of printing reports. We created this application to help enhance Bender's restaurant. The program was designed using Visual Studio C# and Access 2003. The application is easy to run and communicates both with servers and kitchen positions. The application can run across several of Microsoft's operating systems.

1. Description and Intended Use

1.1 Introduction

The Restaurant Point of Sale (RPOS) allows users to build restaurant menus and inventory/sales reports. The core of this application uses a database that retains all menu items and sales information. The data that is archived can be recalled for viewing and for printing sale/inventory reports. The restaurant point of sale application runs in a multi user environment.

There are two main parts that comprise this application, the GUI interface and the database. The GUI interface allows the user to enter data that describes the products. Once the database is filled with data, restaurant personnel will access the data and select items which will be recorded as orders. Food items will be sent to the kitchen position and drink items will be sent to the bar position. The database records the sale of the items for calculating the total sales and the food usage.

1.2 Intended Use

RPOS is specifically designed for Bender's, a restaurant located in Indianapolis, Indiana. By providing a system for servers and food preparers to communicate, Bender's will be able to offer more efficient customer service. Bender's does not have any system in place.

- ***Transmission of food orders by servers***

  Servers will take orders regularly by pen and paper. Then they will enter the order in a nearby touch screen station. The food orders will enter the database. The food items on the order will transmit to the kitchen touch screen. The drink order items will transmit to the bar touch screen.

- ***Order acknowledgement***

  Food preparers and barmaids will acknowledge the order items through the touch screen ready function. The servers will either hear an inconspicuous beep and/or notice a flashing beacon on the server touch screen for the order items that are ready.

- ***Reporting food sales***

  The orders are all stored permanently in the Access database. The total sales will be queried either by item/items or by period of time/times. The queries will be printed out.

1.3 User Profiles

There are three types of users for the application: servers, food preparers, and managers.

1.3.1Servers

The servers will take orders and enter them on a touch screen server station. They are responsible for correct order taking and entering. They have to observe the station for order acknowledgements. They will retrieve the order items for delivery. They should have basic personal computer knowledge, such as basic keyboard experience.

1.3.2 Food Preparers

The food preparer is usually the kitchen cook. He/she will view the orders on his/her touch screen and prepare the order. They will acknowledge it is ready by pressing a touch screen icon button. This will activate a ready icon on the server station screen. The food preparer needs basic computer keyboarding skills.

1.3.3 Managers

Managers are responsible for entering users and their passwords for access to the application. They also can retrieve reports on food usage and sales totals. These users must be

sophisticated in their knowledge of high level computer skills. They must be able to secure the access information, advanced keyboarding skills, and accurate data entry.

1.4 Project Design

The development of RPOS will involve knowledge of three main areas: database, networking, and programming. The programming will be done in C#. Graphical forms will be used by users to enter data to the database, on a server computer, over a network connection. Figure 1 below illustrates the three areas of knowledge. The presentation layer consist of forms with object tools. The business logic handles the input data. The data layer holds the data tables for information retrieval and manipulation.

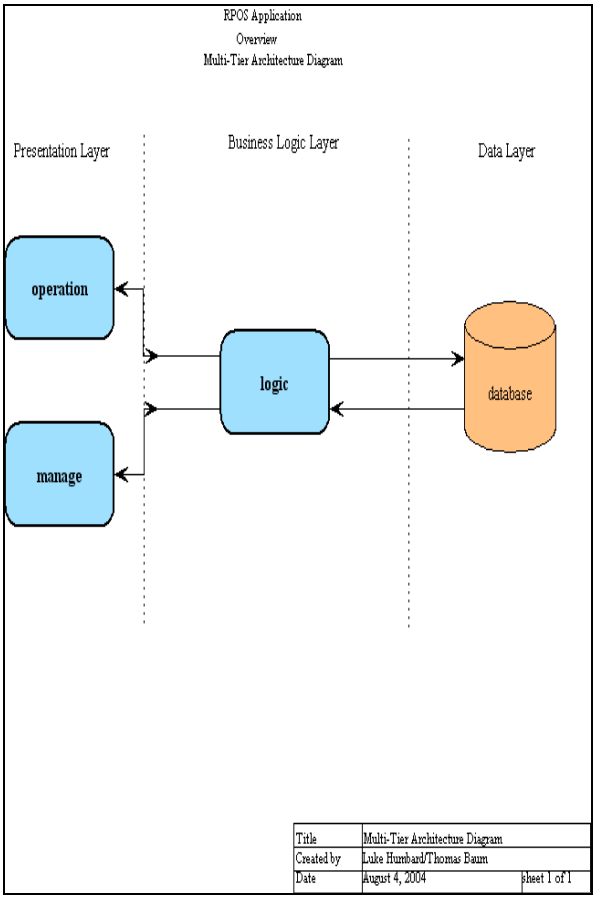


Figure 1. Multi-Tier Architecture

The following Figure 2. RPOS Logic Overview illustrates the logic flow from the programs point of entry, the password entry. From the password entry, depending on the access level, the user will enter a run mode or a manage mode. The run mode allows the user to enter order information or view orders for preparation. The manage mode allows the user to build the menu, enter users and assign passwords, and printout reports.

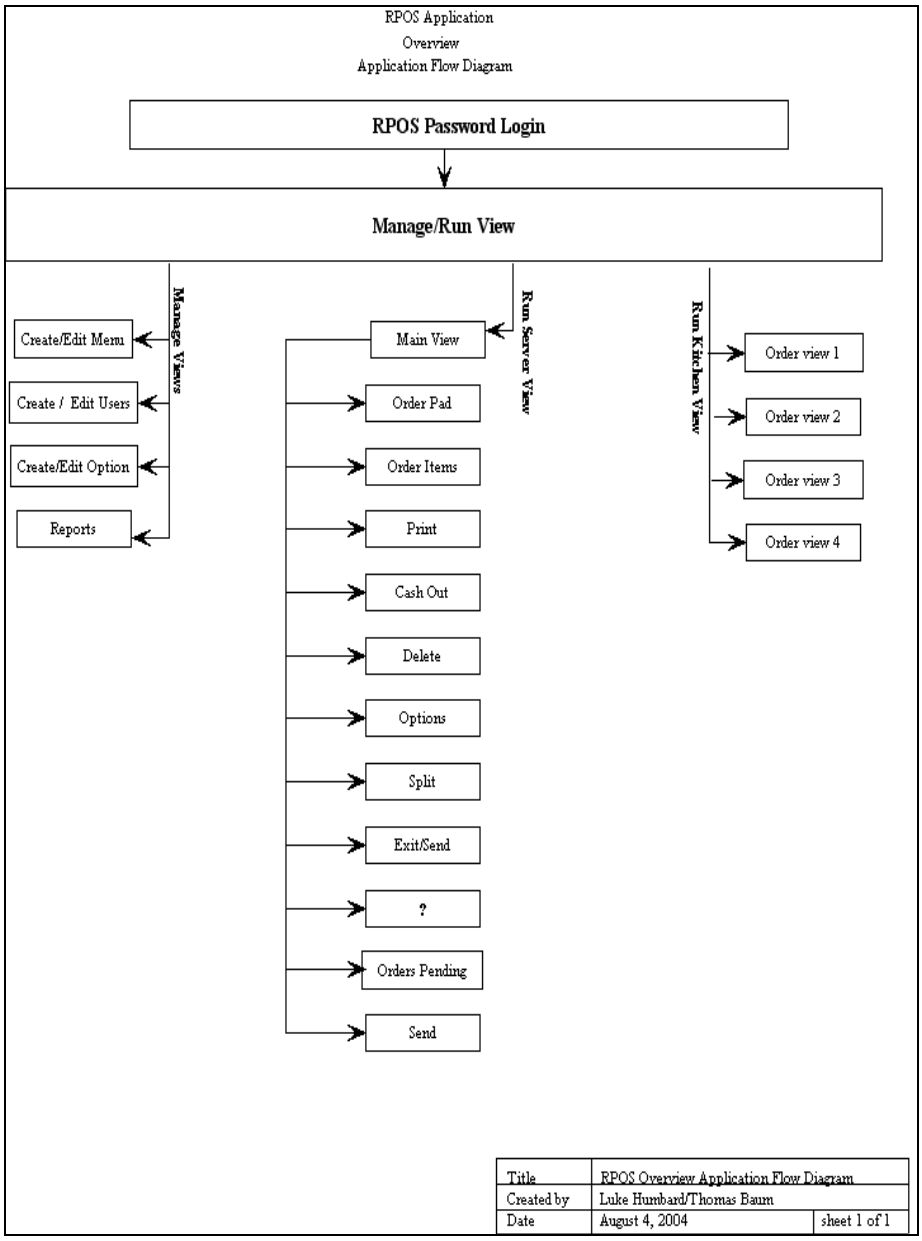
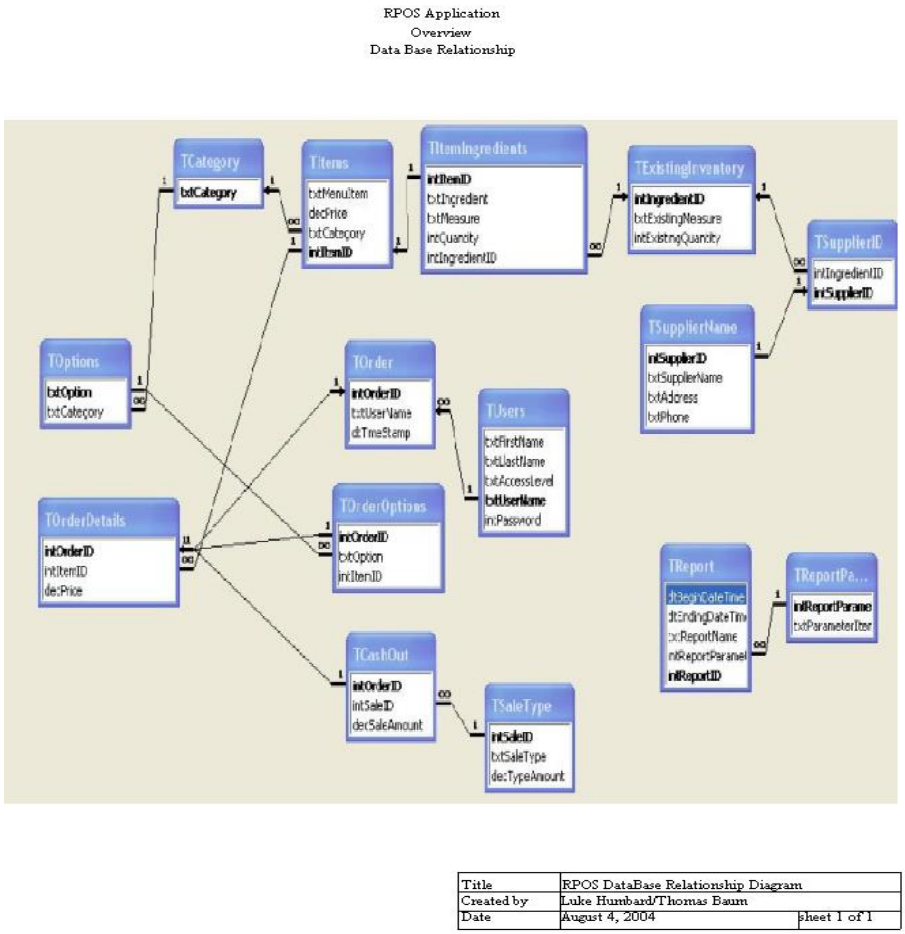


Figure 2. RPOS Logic Overview

1.4.1 Database

The database will be created with relational tables for multi users. The database will only be accessible by the program. Transmission security is not an issue with the customer. The server will be secured in a locked room, connected to a private network. User passwords will allow for limited access. Figure 3. RPOS Database Structure below illustrates the database structure.



Title	RPOS DataBase Relationship Diagram	
Created by	Luke Humbard/Thomas Baum	
Date	August 4, 2004	sheet 1 of 1

Figure 3. RPOS Database Structure

1.4.2 Networking

The networking function is important for connect ability and preservation of data. The database will receive its updates as long as there is connectivity. By actively updating the database the recovery of data is assured. Each system will use a network channel to connect to the server who will retain the database. Figure 4 below illustrates the client/server network.

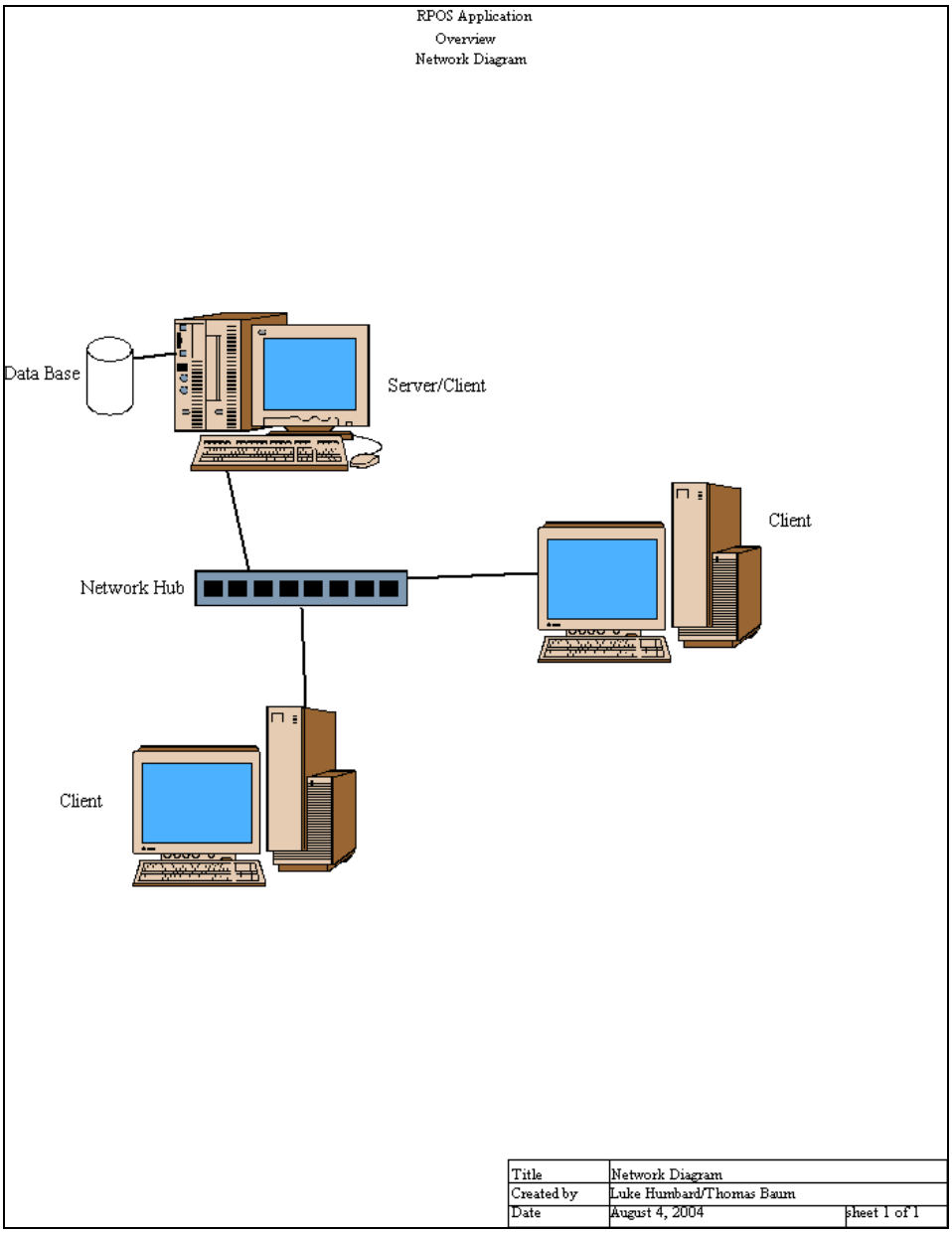


Figure 4. Client/Server Hardware

The network channel exists as a .Net proxy that executes a single call for data reads or writes. The data is written to the database immediately on send commands. The data is then retrieved for processing.

1.4.3 Programming

RPOS uses configuration, class, and object protocols to communicate with the database over the network. We are going to use Visual Studio.Net to create the user interface. Specifically we are using C#.

2.0 User Interface

The user interface is divided between three user roles of the RPOS application. The kitchen role views orders and acknowledges them for pick up. The server role enters orders and view status of orders. The manager generates reports on sales and inventory.

2.1 Kitchen Role User Interface (View Orders)

The Kitchen user will not need password entry since the only view they have is the present orders. They will be in a read only environment. The only functions they will have access to are ***order ready*** and ***order question***. Backward, forward, and exit navigation are also available. The kitchen user will view incoming orders in one color, identifying them as new orders. When they have completed an order they will press ready for that order. The order will print out two copies onto a dot matrix printer. One copy will go with the order and one for kitchen records. The order will also exit the screen and transmit to the database. The kitchen user also has the option to question an entry. In that case, they will touch the item in question and press ***order question.*** The order will be transmitted back to the server for clarification, and returned by the server. This screen will also be applied to the server at the bar for beverage order viewing. Figure 5. below is a sample Kitchen view display.

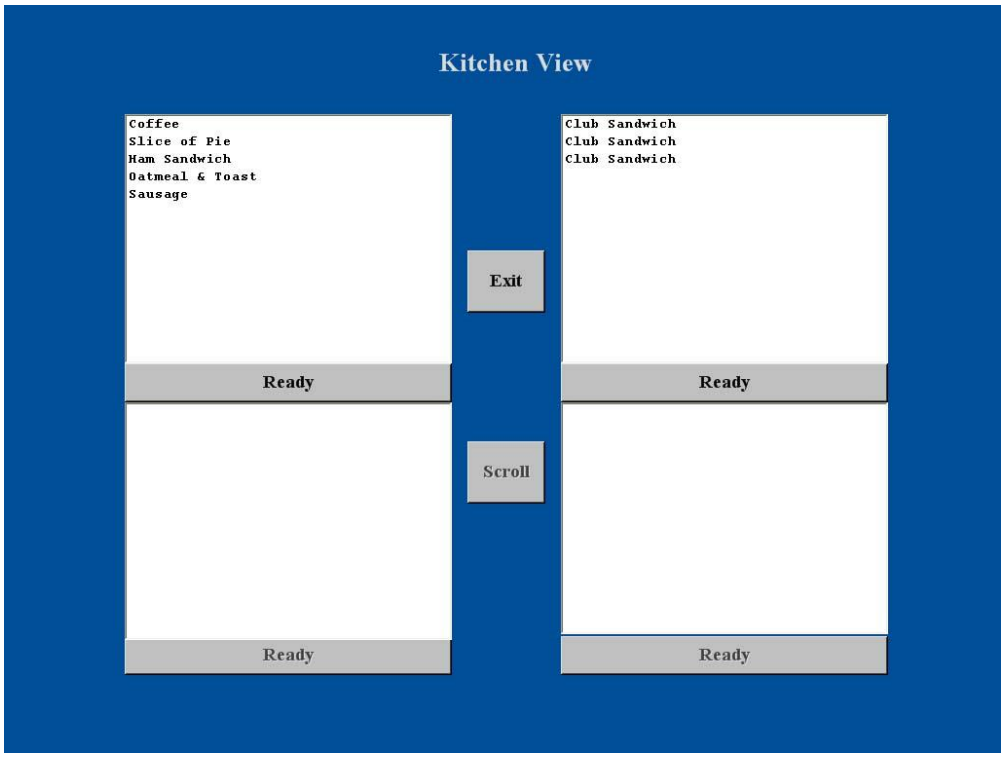


Figure 5. Kitchen Role User Interface

2.2 Server Role User Interface (View Order Input)

The server screen view has a pad for order item listings, individual item buttons, and navigation/function buttons. The server enters the order by items and then sends the order to the kitchen. The server has the following options:

- Print
- Orders Pending
- Cash Out
- Delete
- Options
- Ready

- Send
- Split
- Exit/Send
- ? (Representing an Order Question)

Once the server completes their task the exit will take them out to the password screen for authentication reentry. Figure 6. is a sample Server view.

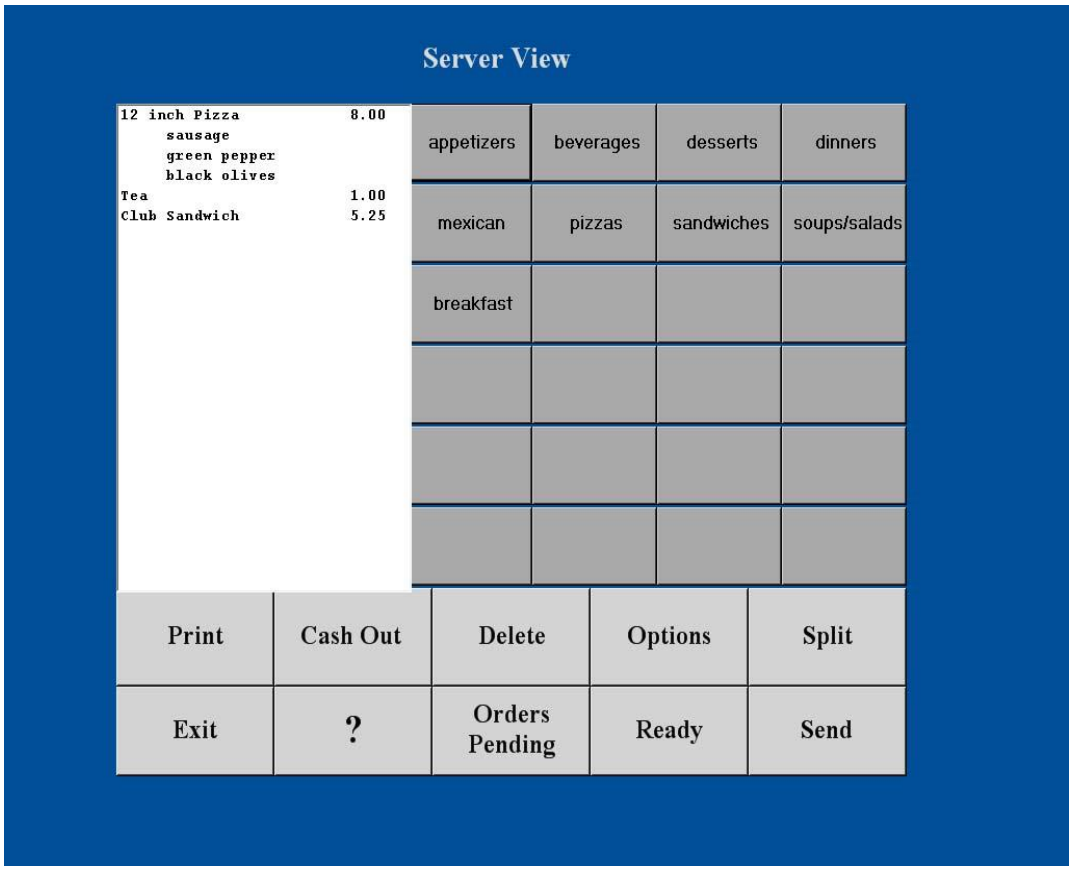


Figure 6. Server Role User Interface

2.3 Manage Role User Interface (Manage View)

Once the manager enters a password and has an authorization access level of high, the manager view contains four entry options and an exit option. At this point the manager may build menus, edit users, and obtain reports. The following options are displayed in manage view.

- Create/Edit Menu Items
- Create/Edit Users
- Create/Edit Options
- Reports
- Exit

Figure 7. Manage View illustrates the manage view.

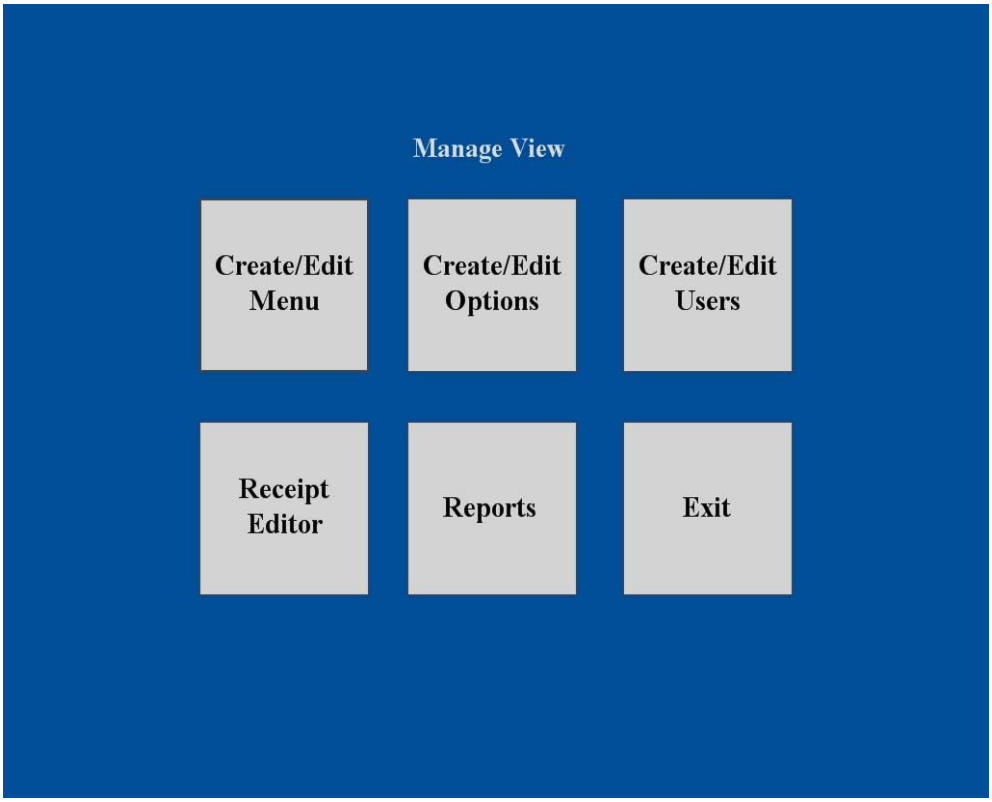


Figure 7. (Manage View)

3. RPOS Design Details

The RPOS is designed as a client/server network system. A shared database resides on the server and the clients talk to the database over a network. The clients send and receive information depending on what user interface forms are active. The user enters the application through a password view. Depending on the type of user, the user enters his/her user interface view. The user enters or selects data and executes the required sequence of actions. The client/server network system is a multi-tier architecture.

3.1 Data Layer

The POS application uses Microsoft Access to handle the data information. The database schema was determined from the information being processed at Bender's restaurant.

- A manager builds menus with menu items.
- The manager enters inventory data
- The manager enters users and assigns them access levels.
- Users enter orders.
- Food preparers view orders.
- Sales orders are totaled.
- Food usage is recorded.
- Reports obtained from archived data.

The database is the centralized organizer of the application data. The forms provide textbox, list box, and other object tools to transfer in user selections and inputs. The following describes the table fields and their purpose.

Categories are given to groups of menu items, see Figure 6. The menu items are stored in a table along with the item price. A unique identification value is assigned to the item. Ingredients of the menu item can be entered and recorded for each menu item. The ingredients have unique identification values. The unique values allow the ingredients to be recorded as a measure of quantity. From this measure of quantity a total of the ingredient usage is calculated. The ingredient identification and a supplier identification value are stored in a table. The amounts of the menu item ingredients are calculated on cash out transaction. When ingredients values get low the report function will list the ingredient item and the supplier for the manager to contact.

An option table is built on menu item options. For example, for steak dinners the option of medium rare, medium, medium well, and well done are recorded in a table with a category value. From this table when an order item is selected to the order list an options selection for that menu item will appear for the server to enter the option request.

The order table holds the menu items and their prices, along with the option selections for each item. Each order has a unique identifier value. The value links the order with a time stamp along with a server. The categories allow order items to be viewed at this point by the food preparer (kitchen view). The category beverages allow the server at the bar position to view the beverage orders.

The cash out table selects which type of payment is made and keeps totals of the sale amount.

The report table provides the means to select a time period from the time stamps and a parameter, which can be any field, or combination of fields, throughout the database.

The reporting is viewed as a list of results about the parameters that were selected. They are presented in a datagrid format. Depending on what table, field and how many of each parameter is selected, the datagrid displays the corresponding format.

3.2.1 Data Access Statements

The RPOS application uses sql statements for accessing data to and from the access database. The sql statements pass into a remote object for transport over the network to the server for execution. The data access commands are stored in a configuration file and passed into the forms as string variables for execution.

The Dataset class is used extensively for data retrieval. In C# the dataset is a powerful tool for retrieving data tables and for sending data. The dataset makes arranging and code construction a lot easier since the format and structure are built into the collection. The dataset is actually a copy of the data from the database and can be manipulated and then sent back for archiving.

The following is one example of the programming code that is used.

Figure 8. shows the configuration file data access command and how it is implemented. Keys are created to hold selected text to be passed into the class forms. Configuration files make it easy for developers to change and add commands to the existing code.

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
<appSettings>
<add key="Key1" value="SELECT * FROM TUsers" />
</appSettings>
</configuration>
```

Figure 8. Configuration File Example

Figure 9. references the keys to variables in the class forms.

```
public class frmUser : System.Windows.Forms.Form
        {
                public string strGetUsers =
System.Configuration.ConfigurationSettings.AppSettings
["Key1"];
        }
```

Figure 9. Keys represented by Variables

Figure 10. passes the variable, referenced to a key, to the remote object for execution.

```
private void loadfrmUsers()
{
DbConnect dbcUsers = new DbConnect();
ds22 =    dbcUsers.ExecuteQuery(strGetUsers);
}
```

Figure 10. Variable Key sent to Remote Object

3.3 Business Logic Layer

The business logic between the presentation layer and the data layer performs the user access, data selection and data archiving functions. The application starts with a password login form that allows a user to access the program after entering the user name and password. Once in the application, the user, depending on their access level, enters a run view. The food preparers enter the kitchen view, the servers enter the server view, and the manager enters the manage view. The logic layer handles the type of data access and the data to present to the user. After datasets are filled with data they are applied to the form's components by class objects. The CMainPassword class (Figure 11.) is an example of the encapsulation of the information to be

used as logic functionality. A method gets the entered password and compares it to the database password. After authentication it resets.

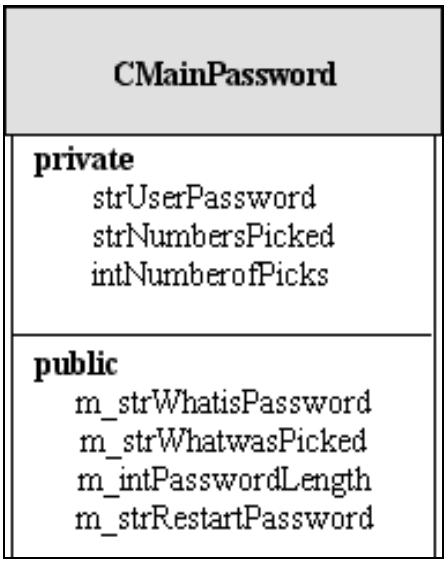


Figure 11. Class Example

Constructors initialize the object properties. The class methods return, compare, add, and delete. The logic layer compartmentalizes the decision of data to be displayed and what data to send to the database.

The data sets hold information for each form from the database. The form activity is short. Once the data is used by class objects and array lists, the database updates and the form activity state is refreshed. A configuration file is used for passing in variables and data access statements. The configuration file is used to help simplify coding. By having one file to pass variables and text into the logic layer, functionality and expandability, of the application, is enhanced.

Some of the forms have unique logic that takes place on the form. These unique logic actions are handled with components local to the active form. As each form comes into focus the

objects perform their actions and the form exits disposing of the local objects. Four diagrams were selected to show the core application logic flow.

Figure 12. is the password logic view. Figures 13, 14, and 15 are logic views of each user path, create/edit menu, kitchen, and server. The create/edit menu logic view is part of the manage view. For more detailed logic views of each logic path see appendix A.

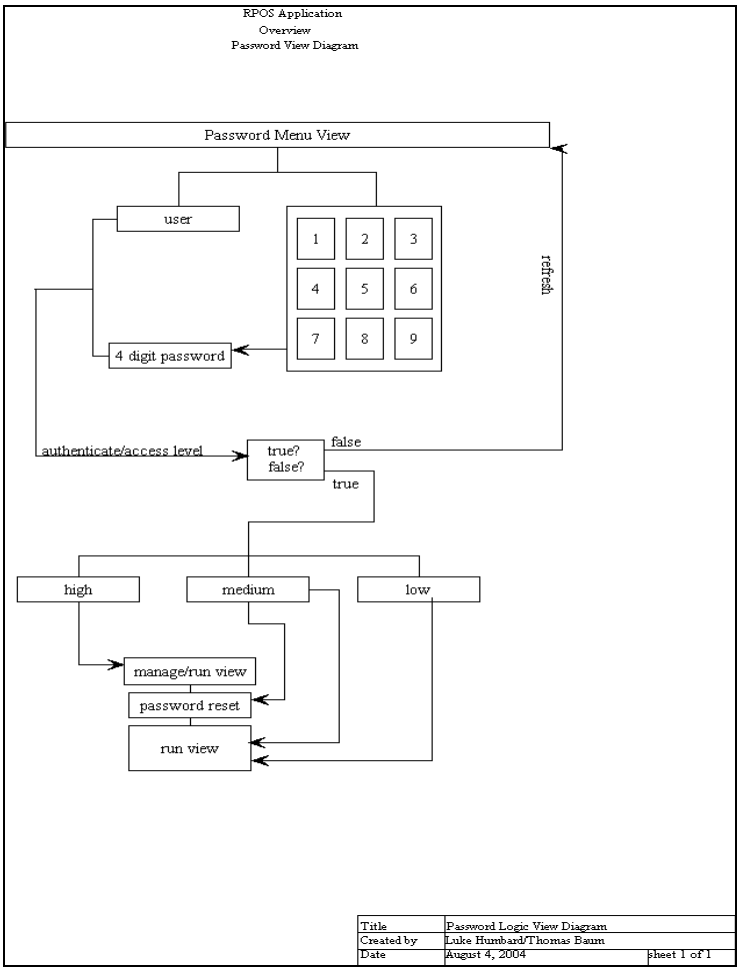


Figure 12. RPOS Password Logic View

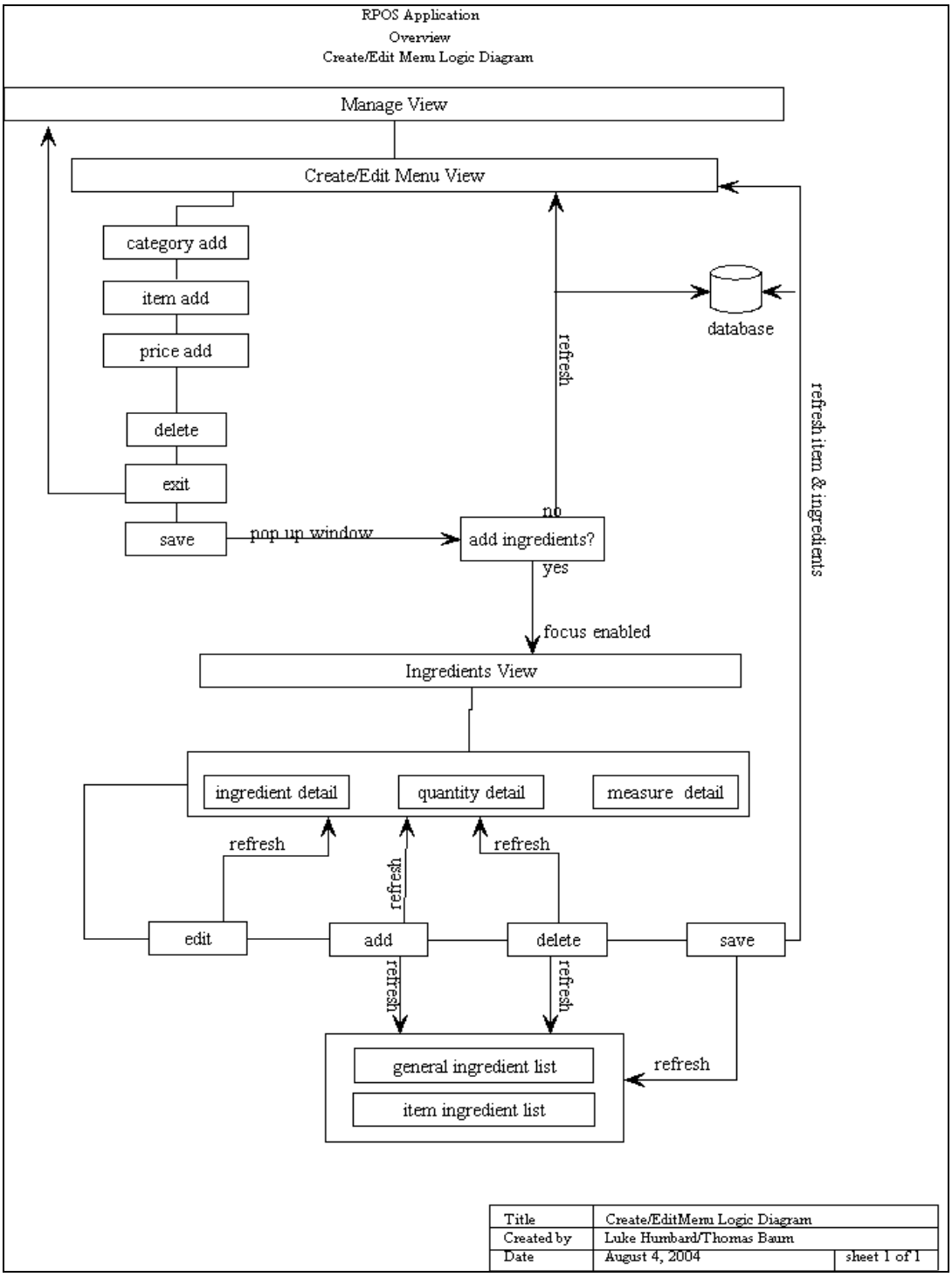


Figure 13. RPOS Create/Edit Menu Logic View

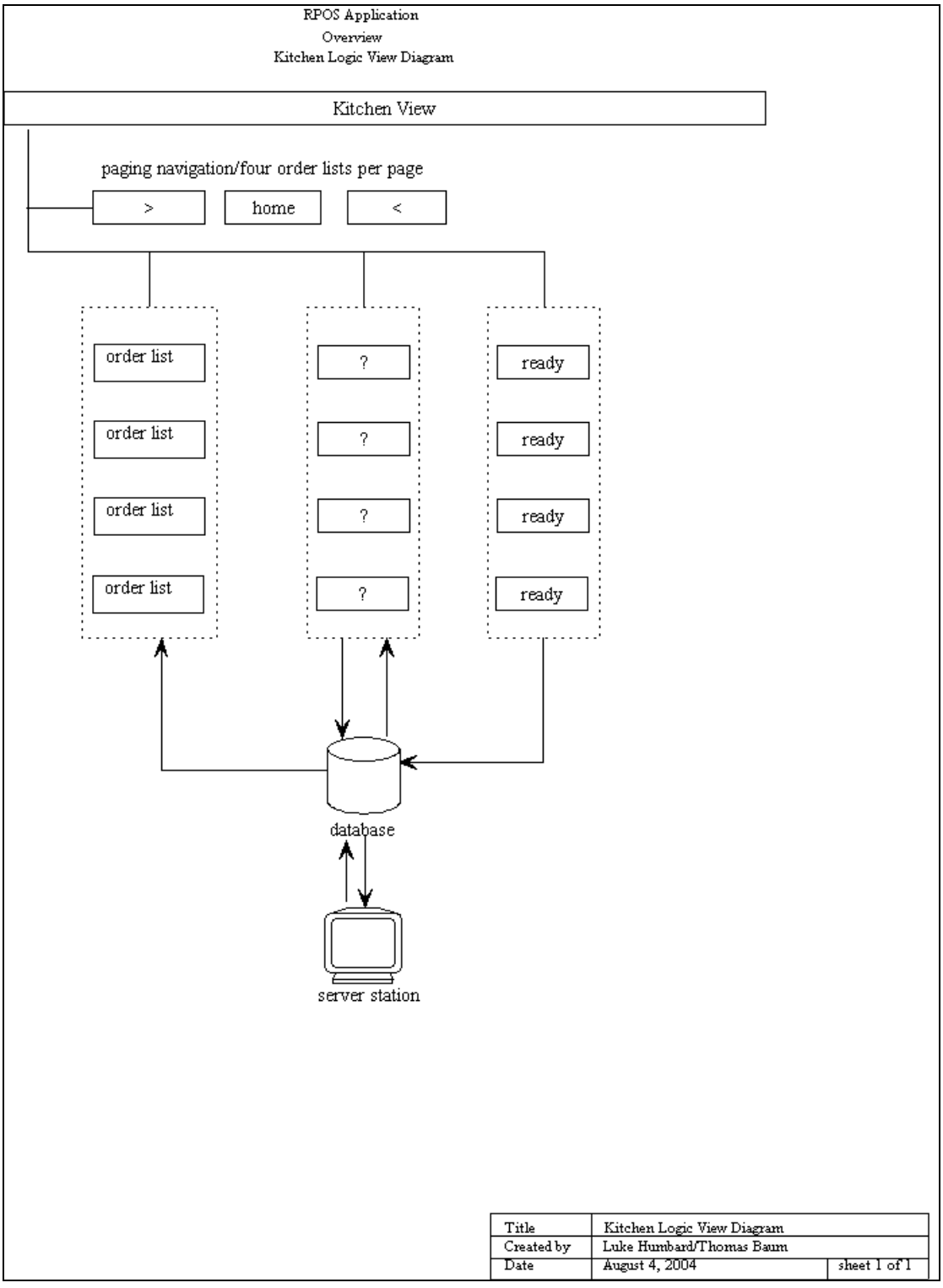


Figure 14. RPOS Kitchen Logic Overview

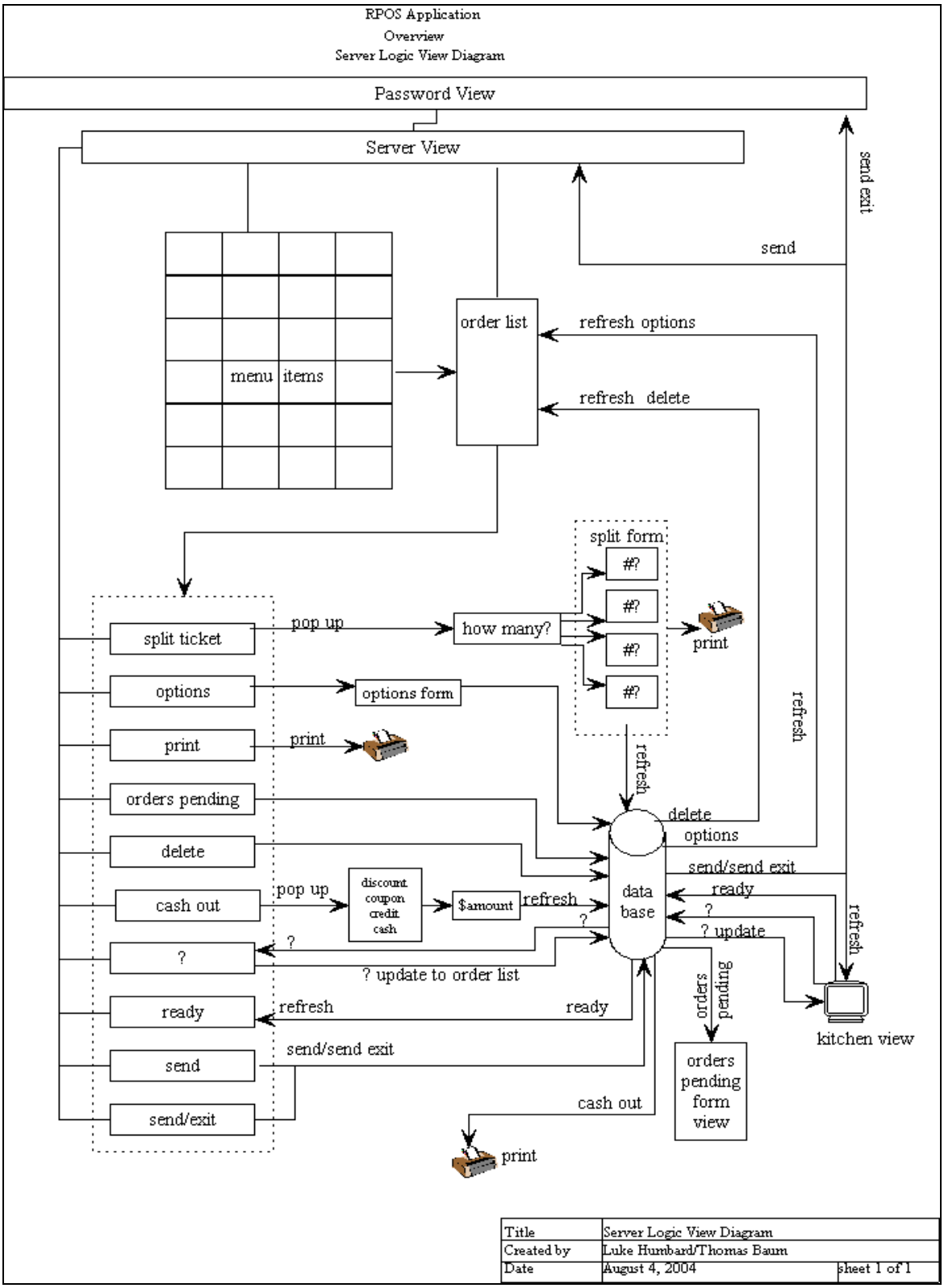


Figure 15. RPOS Server Logic Overview

3.3.1 Network Connection Logic Layer

The network connection logic is a marshal by reference object (MBRO), in Figure 16., that executes a single call over transmission control protocol (TCP) channel. Specific data access commands are passed to the MBRO. The connection is opened. The data command is executed. The connection is closed. The MBRO is a .Net component. Using the MBRO reduces the amount of code needed to connect and maintain the data access. The MBRO class library component sets the provider and data source. The MBRO opens the connection and passes a data access statement over the TCP channel to the host with the database. A dataset is used to receive data.

```
//Marshal-By-Reference remote object

        public class DbConnectRPOS : MarshalByRefObject
        {
                private OleDbConnection oleConn;

                //database Connection constructor
                public DbConnectRPOS()
                {
                        oleConn = new OleDbConnection("Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=posNewTestdb.mdb");
                }

                // Execute method sets the DataAdapter and DataSet
                public DataSet ExecuteQuery(string strQuery, string strTableName)
                {
                        DataSet returnQuery;

                        OleDbCommand olecmd = oleConn.CreateCommand();
                        olecmd.CommandType = CommandType.Text;
                        olecmd.CommandText = strQuery;

                        OleDbDataAdapter oleDa = new OleDbDataAdapter();
                        oleDa.SelectCommand = olecmd;
                        DataSet ds = new DataSet();

                        oleConn.Open();
                        oleDa.Fill(ds, strTableName);
                        oleConn.Close();
                        returnQuery = ds;
                        return returnQuery;
                }
```

Figure 16. Marshal by Reference Object

The server system uses the object in Figure 17. to allow client systems to connect to the host server.

```
ServerPOS class object

public static void Main(string[] args)

        {
                TcpServerChannel ServerDbChannel = new TcpServerChannel(54321);
                ChannelServices.RegisterChannel(ServerDbChannel);

    RemotingConfiguration.RegisterWellKnownServiceType(typeof(DbConnectRPOS),"DBCo
nnectRPOS", WellKnownObjectMode.SingleCall);
         }
```

Figure 17. Server TCP Channel Reference

The object in Figure 18. is used by the client system to connect to the server for query execution.

```
ClientPOS class object

static void Main()
          {
                TcpClientChannel ClientDbChannel = new TcpClientChannel();
                ChannelServices.RegisterChannel(ClientDbChannel);

    RemotingConfiguration.RegisterWellKnownClientType(typeof(DbConnectRPOS),"tcp://on
e:54321/DBConnectRPOS");

                Application.Run(new Form1());
          }
```

Figure 18. Client TCP Channel Reference

3.3.2 Security Logic

The RPOS database will reside on a computer that is in a locked room. The access levels listed in the database and defined by the logic layer provide the users application entry. The access levels are defined in 3 levels.

- Low
- Medium
- High

Low level access provides the user to enter orders and print out receipts. Entry into managed activities is not allowed. The activity resetting passwords is also not allowed.

Medium level access allows the user the above function and also allows the user to reset passwords for other users who have forgotten their password.

High level access allows the user all managed features , resetting of passwords and executing order entry and printing. The following chart illustrates the access levels and their functional capabilities.

Table 1. is an illustration of the Access level logic.

AccessLevel	High	Medium	Low
Manage	X		
Reset Password	X	X	
Order Entry	X	X	X
Print	X	X	X

Table 1. Access level Table

3.4 Presentation Layer

The following pages and illustrations best describe the application's proof of design. The first entry into the application is done through the password view. Below Figure 19. shows a scrollable listbox with names. Selecting your user name and entering a four digit password you obtain entry to the application.

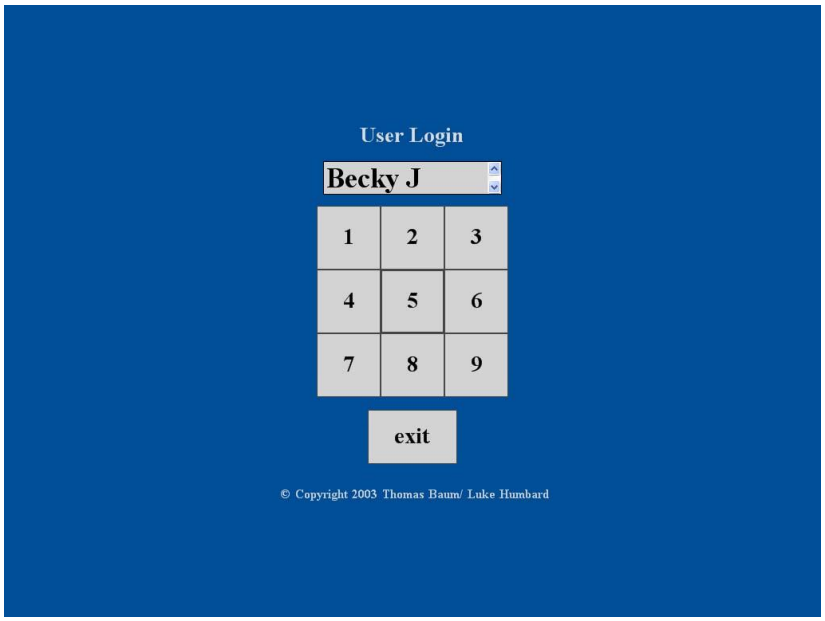


Figure 19. Password View

Low access levels will navigate the user to the next presentation view the server view. From this view the server has many functions and options to choose from. The category buttons are populated first with the menu item categories. Each category has menu items listed under them. The view consists of a notepad for order entry items and their prices, dark grey buttons showing category selections, and function buttons. The print button is for order printouts. The split button allows the server to print out separate receipts for customers from one large order. Options allows the server to add options to menu items. The delete button allows the server to delete unwanted entries. The delete function is not allowed on sent orders.

Cashout button ends the order and takes the server to a receipt printout. Orders pending takes the server to a view of all orders before cashout. The question and ready buttons are communication buttons from the kitchen to the server. They allow the kitchen to communicate to the server when the orders are ready, and if there are any questions about the order items. The send button is used when the whole order is entered. The send function connects to the server database and updates the order and order detail tables for the kitchen to view from their kitchen view screen. Figure 20. on next page illustrates the above description.

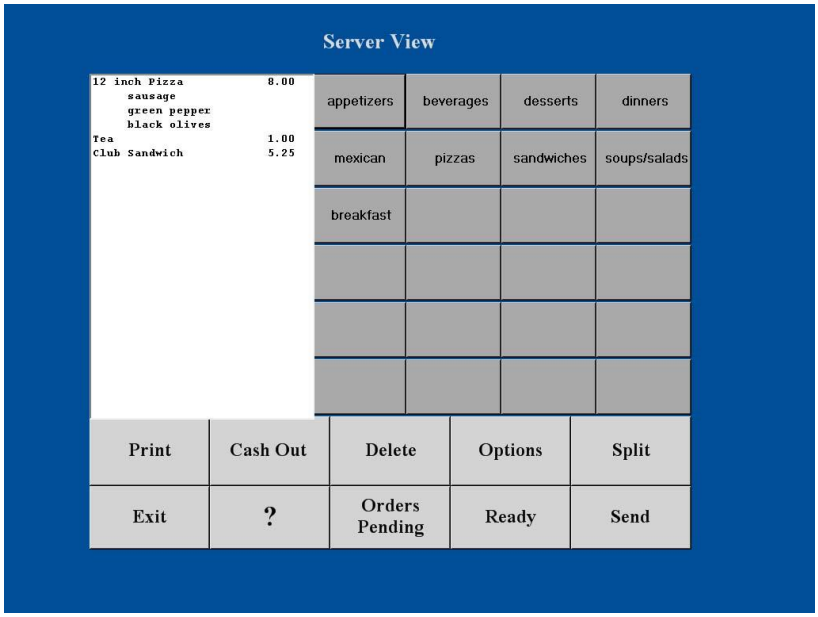


Figure 20. Server View

The split view takes the order list and allows the server to separate the order items into smaller individual receipts shown in the following Figure 21.

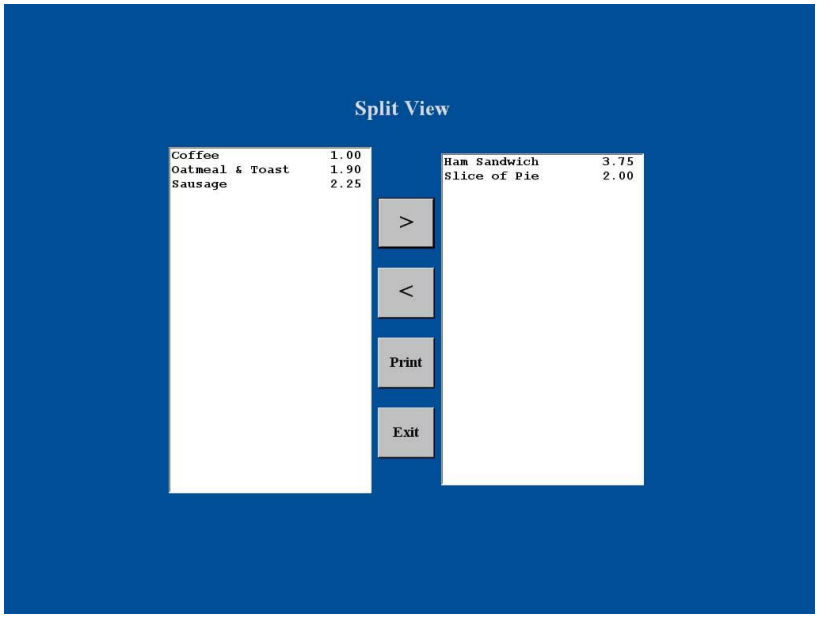


Figure 21. Split View

Options function on the server view allows the server to add options to individual menu items. When the options button is pressed the following figure 22. is enabled. Here the server can choose options to add to the order list.

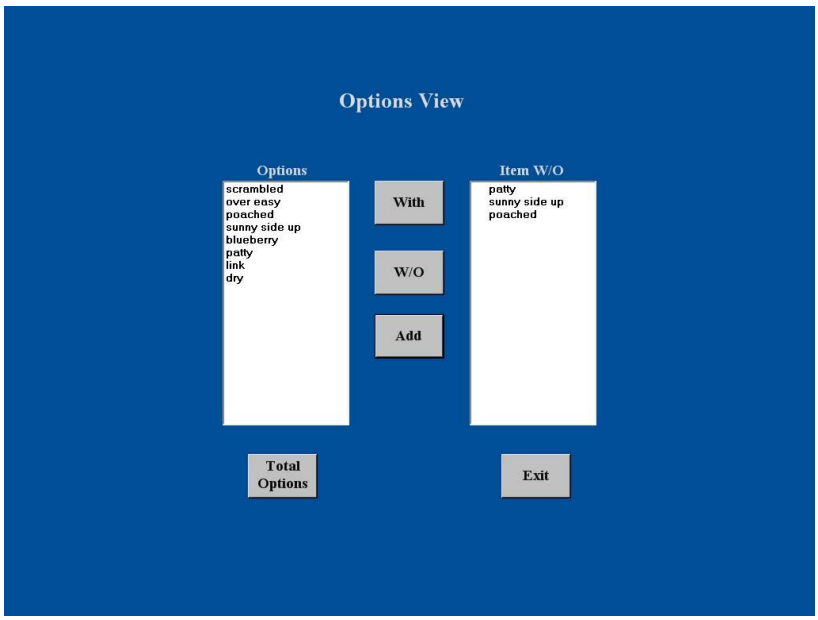


Figure 22. Options View

The orders pending button takes the server to a view of all the orders they presently have that are not cashed out. The orders pending gives the server the capability of adding to an existing order such as desert items. Figure 23. shows the orders pending view.

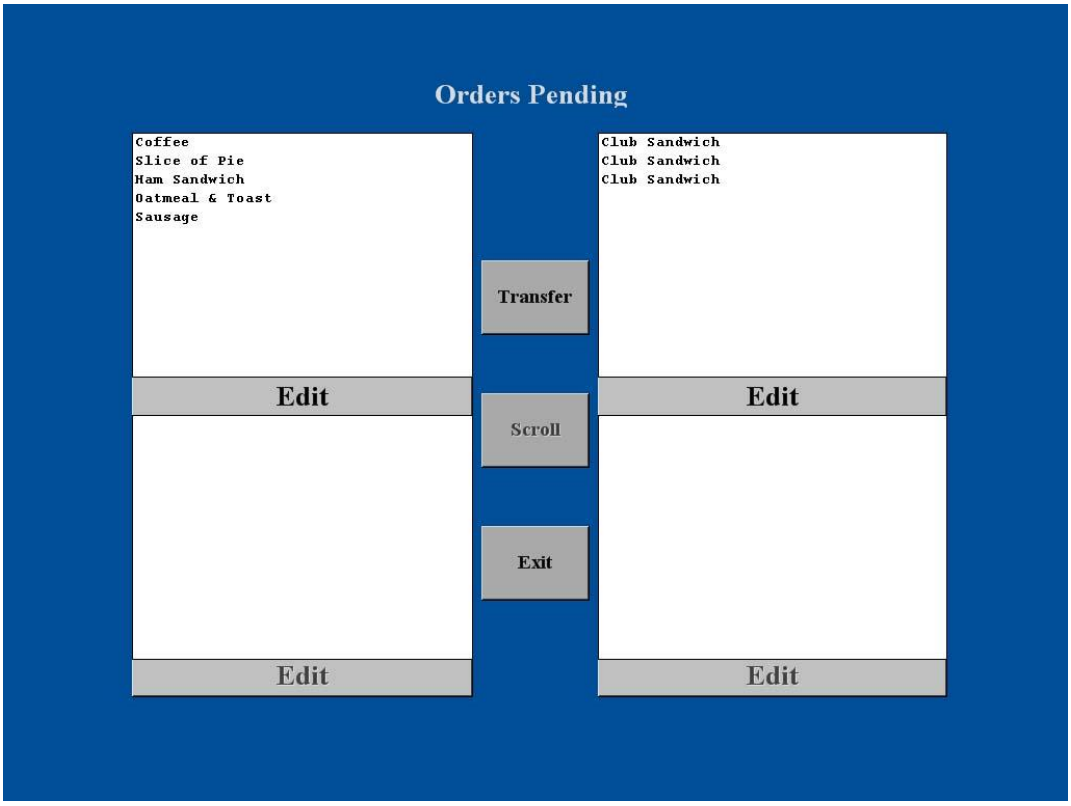


Figure 23. Orders Pending View

The cashout button displays a popup window asking for the type of adjustment , if any is to be applied, and then opens up a view for cash amount entry. Figure 24. below is the popup and Figure 25. is the entry for the amount.

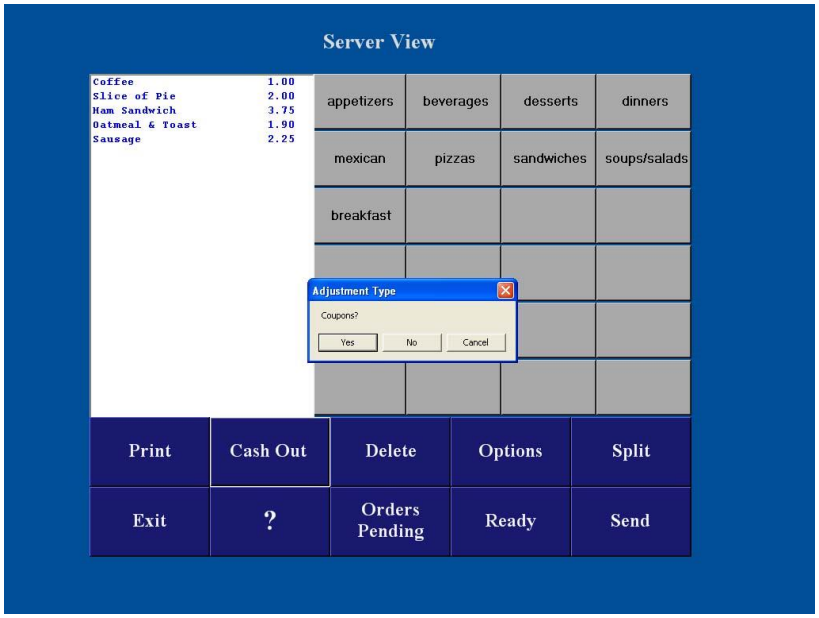


Figure 24. Cashout Popup View

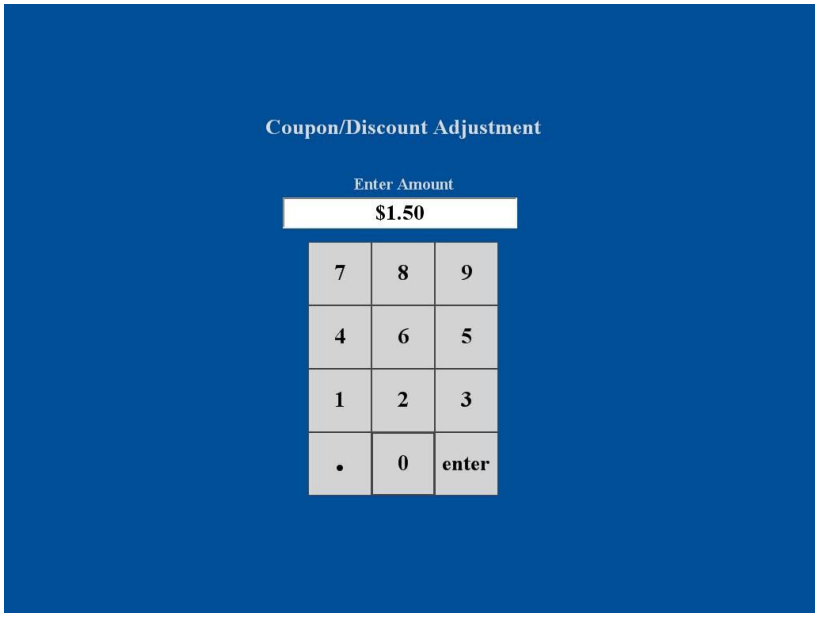


Figure 25. Adjustment Amount View

After the amount is entered the receipt is printed with the adjustment.

The question and ready buttons provide the server with a means of communication from the kitchen service. If the kitchen has a question about a menu item the food preparer signals by

simply touching the item in question, initiating a change in status for that item. The question button turns red and updates how many questions there are. The status is changed back to normal when the server issues an option change for the item in question.

The ready button simply alerts the server that an order or orders are ready for delivery. The ready button turns green. The button is returned to normal when the server presses it. Figure 26. below illustrates the conditions described above.

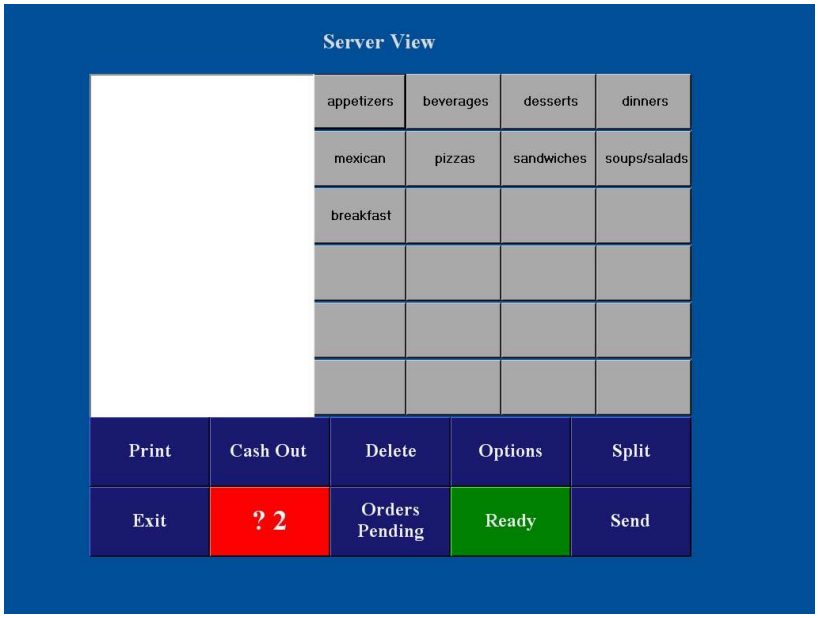


Figure 26. Server Question/Ready View

Entering the menu information and obtaining reports is all done in the manage menu views. The manage views are only seen with high priority access. The point of entry to manage views starts at the first view a high priority user comes to, Manage/Run view Figure 27. below.

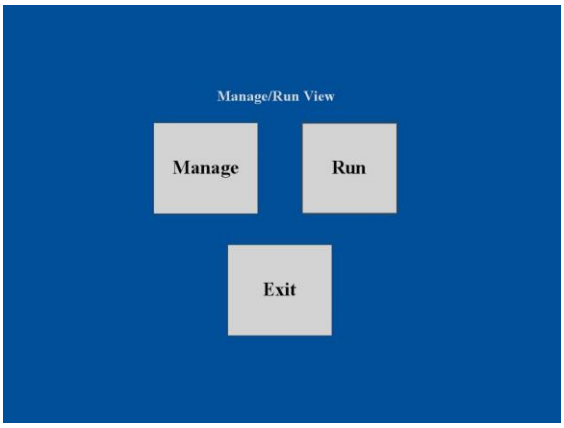


Figure 27. Manage/Run View

The manage button takes the user to a selection of managed menu views, called Manage View. Figure 28. below shows the Manage View.

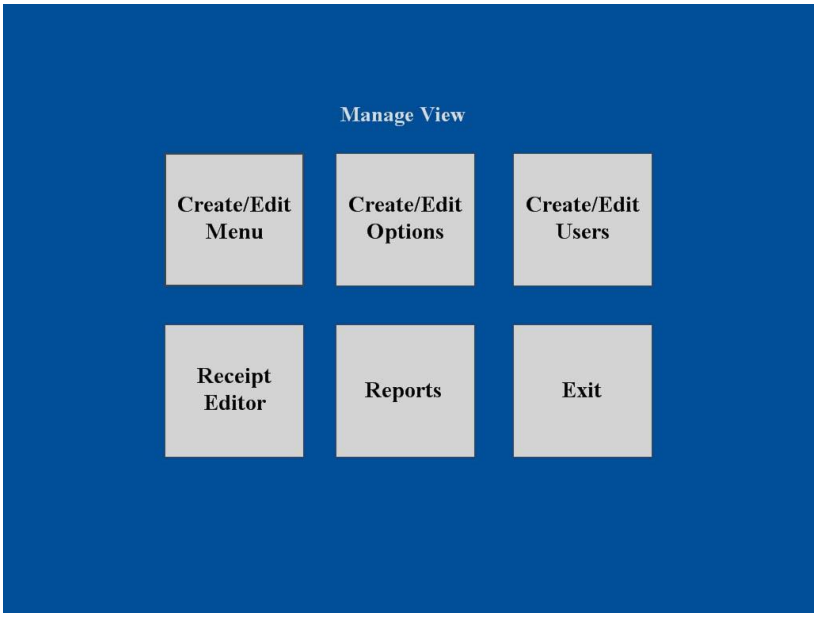


Figure 28. Manage View

Create/Edit Menu button allows the user to build the menu. Figure 29. shows details of menu item entries and the ingredients needed for the sale and inventory reports.

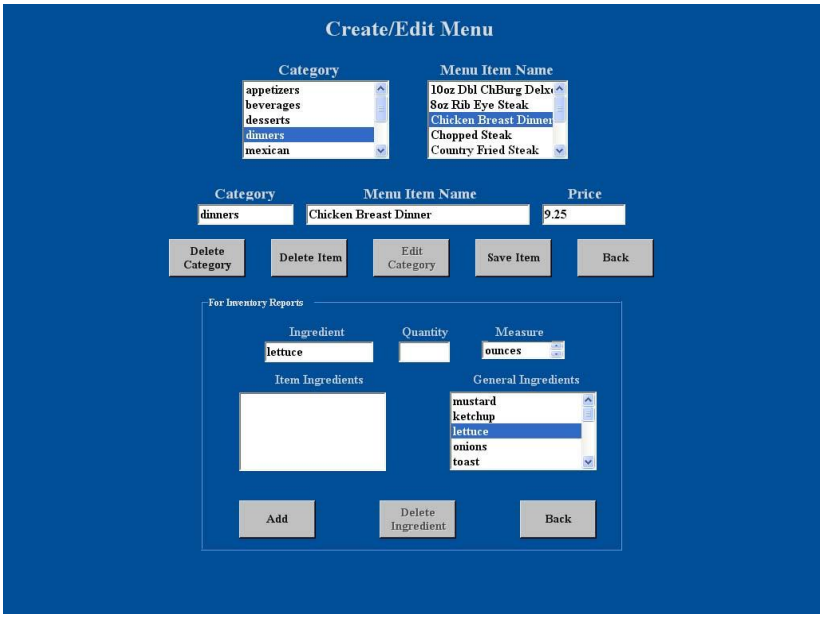


Figure 29. Create/Edit Menu View

Create/Edit Options View allows the user to enter the options for the individual menu items. The user enters options for the categories and selects add to write it to the database. When the menu item is selected and the options button is selected, on the server view, the options for that category is displayed. Figure 30. below illustrates Create/Edit Options View.

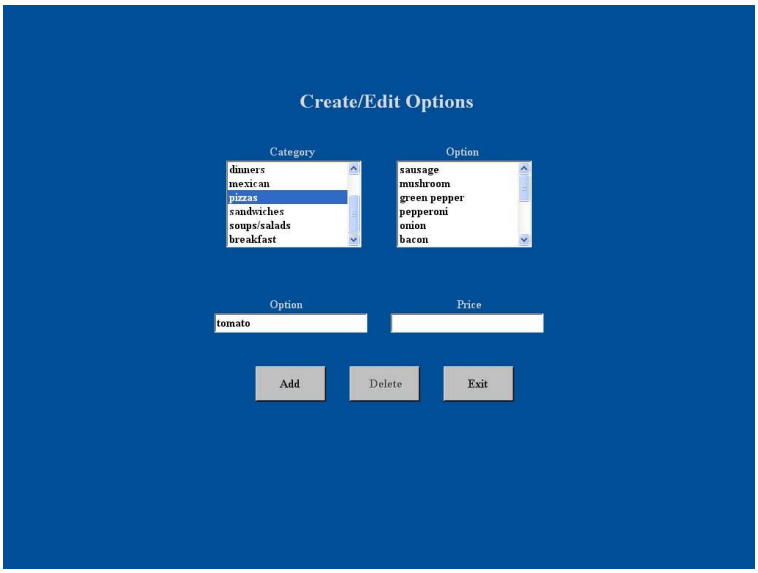


Figure 30. Create/Edit Options View

Create/Edit Users allows the manager to enter user names, passwords and their level of access. Figure 31. illustrates Create/Edit Users.

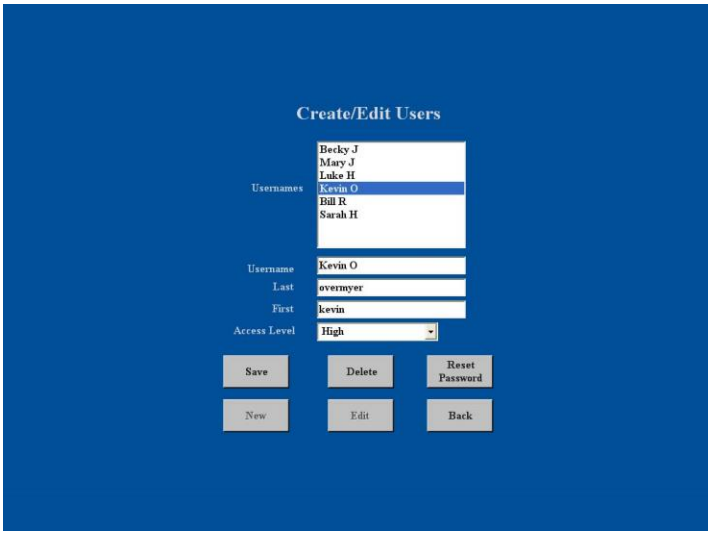


Figure 31. Create/Edit Users View

The Report Editor allows the user to enter a header, footer, tax value for eat in orders. Also the user has the options to check which items are to be taxed when they are included in to go orders. Activate checkbox provides the user the capability to remove menu items from the menu temporarily. Figure 32. below is the Receipt Editor View.

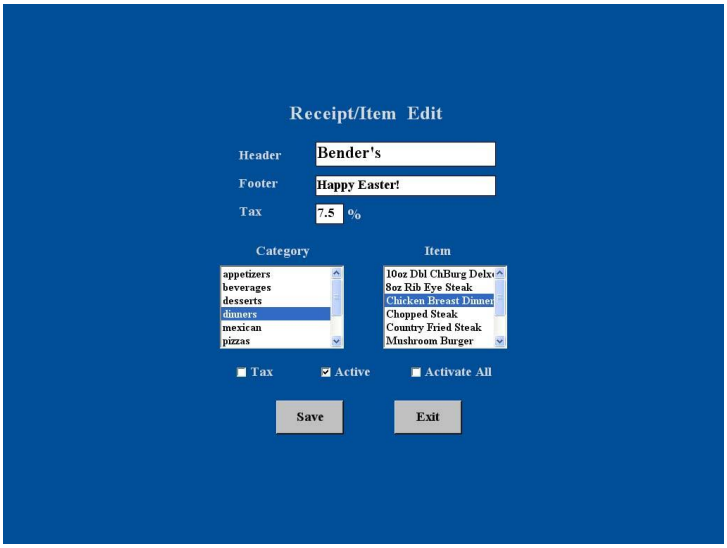


Figure 32. Receipt Editor View

The report View is used to print up a sale report or inventory report. The button brings up a popup window asking which type of report do you want to print. Figure 33 shows the window.

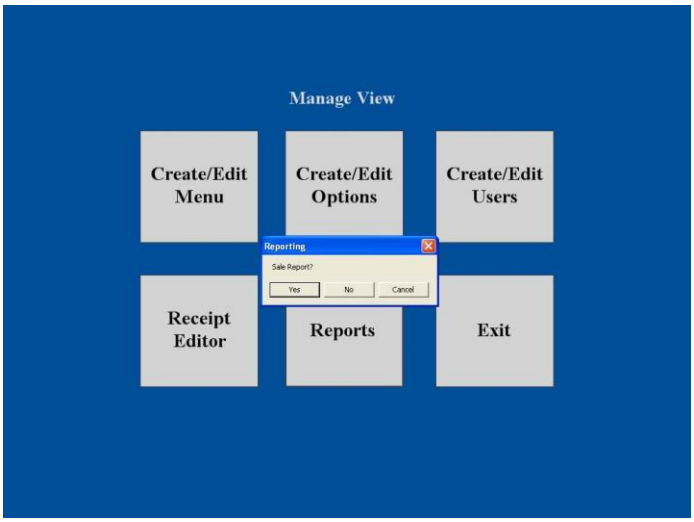


Figure 33. Report Pop Up View

If the user answers yes to the pop up window they will get the Sales Report view. The options are for total sales and/or individual menu item sales. The User is able to select a range of dates and print the sales. The printout itemizes the categories and then gives the total. Figure 34. shows the Sales Report View.

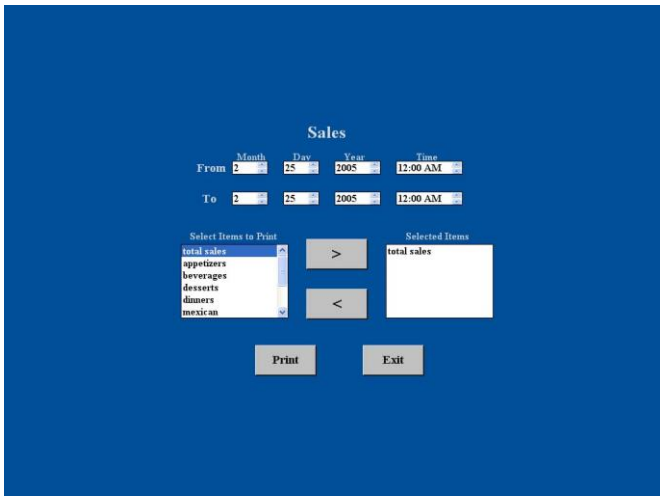


Figure 34. Sales Report View

The inventory report gives the present day and result quantities of the ingredients. The results are calculated by cashed out orders. Figure 35 shows the Inventory Report View.

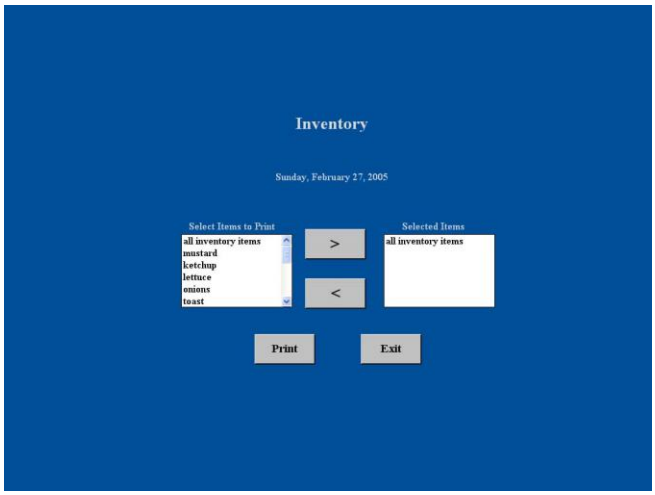


Figure 35. Inventory Report View

The kitchen view allows the food preparer to view the orders sent by the servers. The kitchen user has the capability to communicate to the server with ready orders and questions on individual menu items. Figure 36. below shows the kitchen view. Four orders are shown at once with a scroll function for paging through orders.

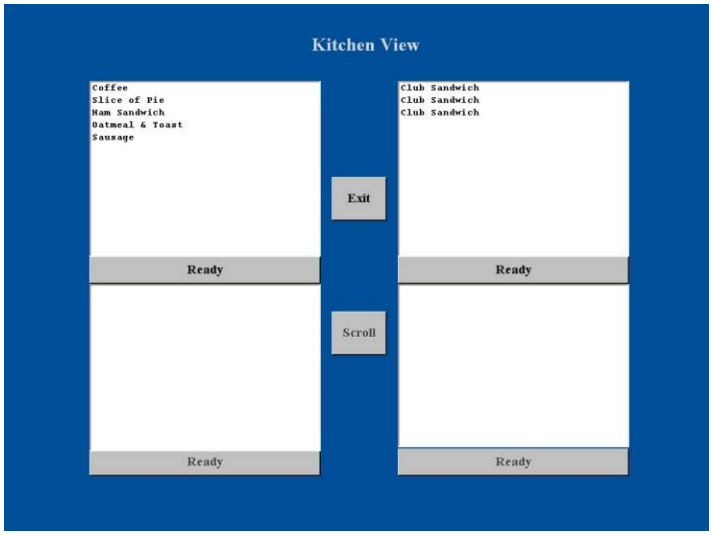


Figure 36. Kitchen View

4. Timeline

week	SD1/Spring04	SDII/Summer04	SDIII/Winter05
1	Research	Design View Diagrams	
2	Research	Design User Interface	Complete Deliverables
3	Progress Report	Draft of Project	Complete Deliverables
4	Problem/Area of Inquiry	Program Remote Object	Complete Deliverables
5	Problem/Area of Inquiry	Program Logic	Complete Testing and Troubleshooting
6	Proposal Draft	Design Database	Complete Testing and Troubleshooting
7	Proposal Draft	Program Code to Connect Database to Presentation	Complete Testing and Troubleshooting
8	Proposal Draft	Documentation	Submit Preliminary Documentation.
9	Proposal Presentation	Demonstrate Working Prototype (August 23, 2004)	Submit Final Documentation
10	Proposal Presentation	Demo	Demonstrate Final Project

Table 2. TimeLine

5. Software and Hardware Requirements

5.1 Software Requirements

- Microsoft C# SDK
- Microsoft Access 2003
- Windows 98/2K/XP
- Microsoft MDAC (Data Access Component)
- Microsoft .NET Framework 1.1
- TSP Printer Driver
- SCEPTRE USB Touch Screen Driver

- POS Application Software

5.2 Hardware Requirements

- 3 computers
- 3 Touch Screen Monitors
- 2 Thermal Printers
- 1 Dot Matrix Printer
- Network Hub
- Category 5 or 6 Ethernet Cable
- Battery Backup (Optional)

6. Budget

Item	Cost
Developmental Budget	
3 Computers	$1500.00
3 Touch Screen Monitors	$1400.00
2 Thermal printers	$ 800.00
1 Dot Matrix Printer	$ 400.00
1 Network Hub	$ 30.00
Microsoft C# SDK	$ 100.00
Microsoft Access 2003	$ 200.00
3 Microsoft 98/2K/XP OS	$ 450.00
1 Computer Server	$1500.00
Developmental Total	**$6380.00**
Production Budget	
3 Computers	$1500.00
3 Touch Screen Monitors	$1400.00
2 Thermal printers	$ 800.00
1 Dot Matrix Printer	$ 400.00
1 Network Hub	$ 30.00
3 Microsoft 98/2K/XP OS	$ 450.00
Production Total	**$4580.00**
Developmental Total	**$6380.00**
Production Total	**Recovered Cost - $4580.00**
PROJECT TOTAL	**$1800.00**

Table 3. Budget

The funds for the project is coming from the customer and ourselves. The customer is providing 25% of the cost and we are providing the rest for the hardware purchase during customer testing.

7. Deliverables

RPOS Windows Application with Order, Report, and Communication Management:

1. A back-end database that facilitates the development of a dynamic front-end interface. The database is developed on Access 2003.
2. A dynamic front-end interface developed in Visual Studio C#.
3. Access levels for users.
4. Easy navigation of the application.
5. Ability for authenticated users to:
    - Add order items, inventories, and employees into the database.
    - View reports and their associative information.
    - Update orders, inventories, and employees.
    - Expand order items and their respective ingredients to the database.
    - Reset user passwords.
    - View orders and their status
6. The application provides network communication.
    - The remote object provides the mechanism for data recording
7 The application provides information on all aspects of restaurant and bar sales including the following
    - Sales totals
    - Inventory
    - Periodical Reporting
    - List of Authenticated Users
8. The application provides user interface forms for application navigation and selection

of functions and features.

- Touch screen tools used to execute user interface functions.
- The user interface executes a remote object, sending and retrieving information to the data base.

9. The application provides a GUI interface that is user-friendly.
10. The application provide a password process for user entry.
    - access level determining the access level of entry.

8. Testing

The testing sequence occurred throughout the development of the application. As each member completed code it was induced into the whole program for testing and debugging. The following were some of the testing issues.

- Application operational on Windows 98
- Application operational on Windows 2000
- Application operational on Windows XP
- Server Identification
- Network Connectivity

Several computers were set up with each of the Microsoft operating systems and tested the applications functionality. If the server was not identified or the network connection failed the application would resort to standalone mode for operation. This took several days to verify. We found that on the older versions of Microsoft's operating systems you needed Microsoft Data Access Components (MDAC) and Microsoft .Net Framework. These two software packages are

redistributable and are needed to run applications developed from Visual Studios .Net software development kits.

Testing continued extensively throughout the development process. Debugging was performed at each step of the build process. Errors and bugs showed up at different times throughout the project development. These errors and bugs included the following categories:

- Syntax
- Logic
- Function
- Failsafe

Syntax errors were quickly identified during build sequences, since you could not run the failed application build. The debugger pointed out the error type and location in the assembly. Most of the syntax errors occurred early on in the project when Luke and I were not accustomed to the language format.

Logic errors occurred throughout the development because of its complexity. We had to think slowly and carefully of each aspect of the program. Using predetermined data in the data tables of Microsoft's Access 2003, we were able to test and verify the logic results for the program modules. Some of these errors included inserting and updating the correct order of variables in the statements.

Function errors involved the form controls and their operational functionalities, such as what happens at the on-click of the control. The right sequence of events has to occur for the application to flow in a logical sense. The following error description is one of many errors encountered. The split ticket function was activated. No items were selected to print and the print

button was activated. The program halted because we did not set the control to ignore a null request. A null request is when an action is to take place with no variables defined. Each of the controls had to be examined for a multiple of actions taken by the user. If the action was not defined then the controls would reset to a neutral state without causing program interruption. Failsafe errors had to be addressed to allow the application to recover from system interruptions. The server might go down or a break in the network might occur. If users are logged in and entering data, there has to be a way to gracefully close and notify the users of a problem. Failsafe problems coincided with function errors. On control activation and form navigation sequential actions had to make sense for the user. If the user forgot to select or enter values and variables that the application needed to act on, the program would ignore the action or provide a message to help the user to work through the process.

References

1. Mills, S. (2001). Technology Clicks with Restaurateurs. Restaurants USA. http://www.restaurant.org/research (10 January 2004)

2. Microsoft (2003). Point of Sale Terminals Overview. Windows Embedded Home. http://www.microsoft.com/windows/embedded/devices/pos/default.asp (22 December 2003)

3. Jean-Marc Bovis - Christophe Robinet. BillPro6. Vers. 6. Computer software. Billpro, 2002. Microsoft Windows XP, 6.634KB, download.

4. Client, Manager. Personal interview. 15 September 2003.

5. Retail, User. Personal interview. 12 January 2004.

6. 45 East Bar and Pub, User. Personal interview. 23 January 2004.

7. Great Clips, User. Personal interview. 24 January 2004.

8. Golden Dragon, User. Personal interview. 20 January 2004.

9. Tek-Tips Forums for computer professionals. Point of Sale Systems Forum. http://www.tek-tips.com/gthreadminder.cfm/lev2/3/lev3/90/pid/693 (19 January 2004)

10. Mac & Joe's, User. Personal interview. 23 January 2004.

Appendix A. Detail Logic Diagrams

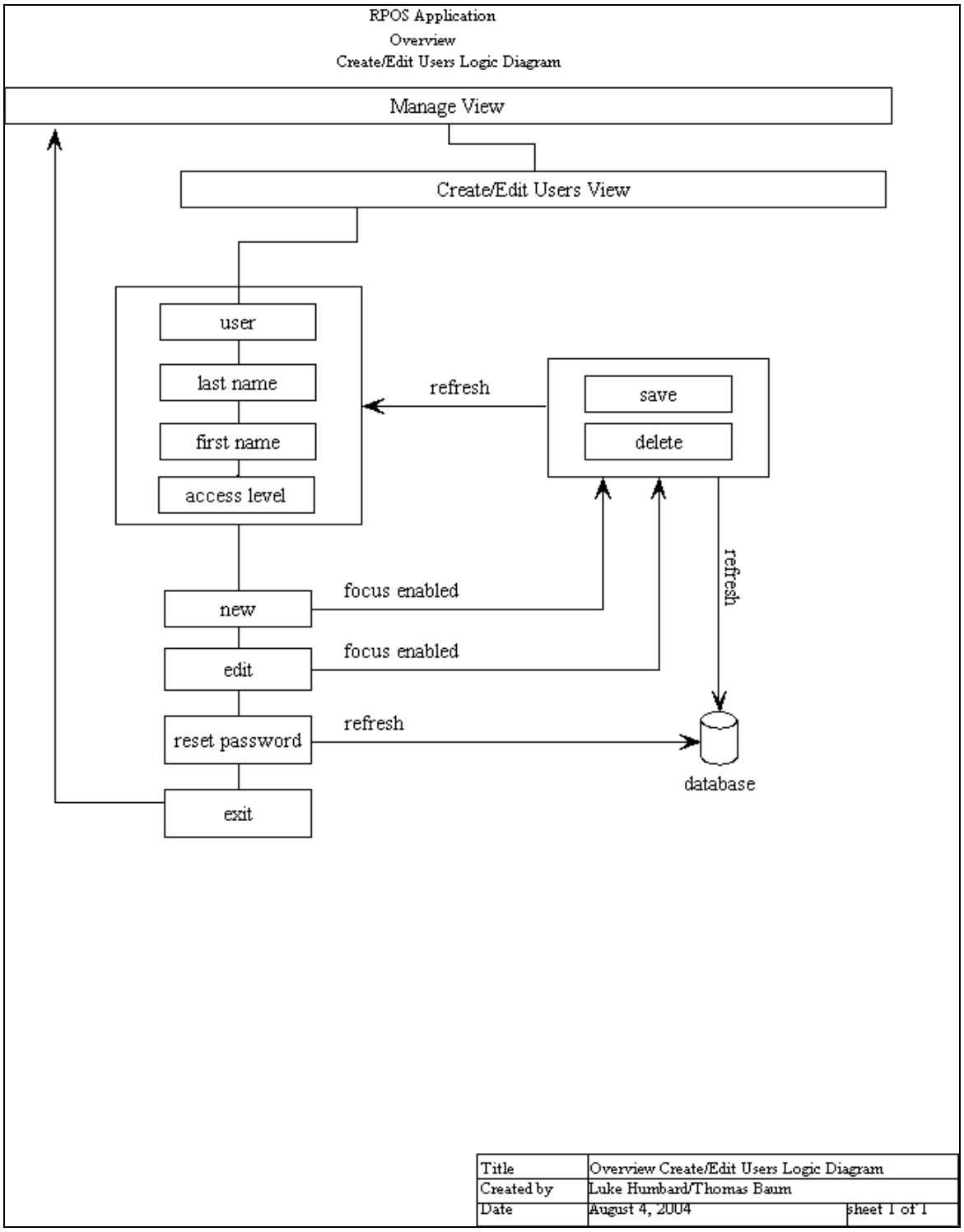


Figure A 1. (Create/Edit Users Logic Diagram)

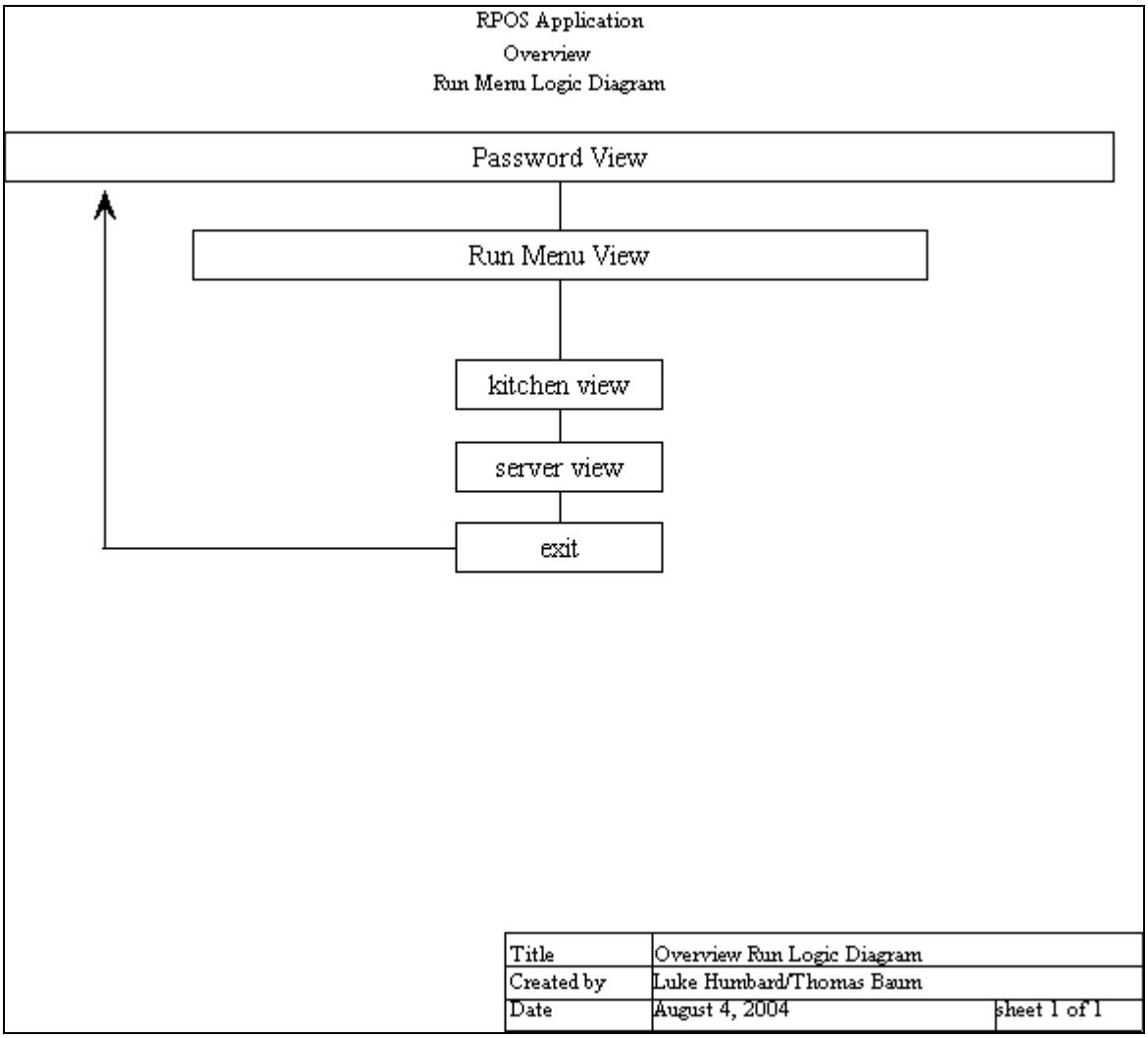


Figure A 2. (Run View Logic Diagram)

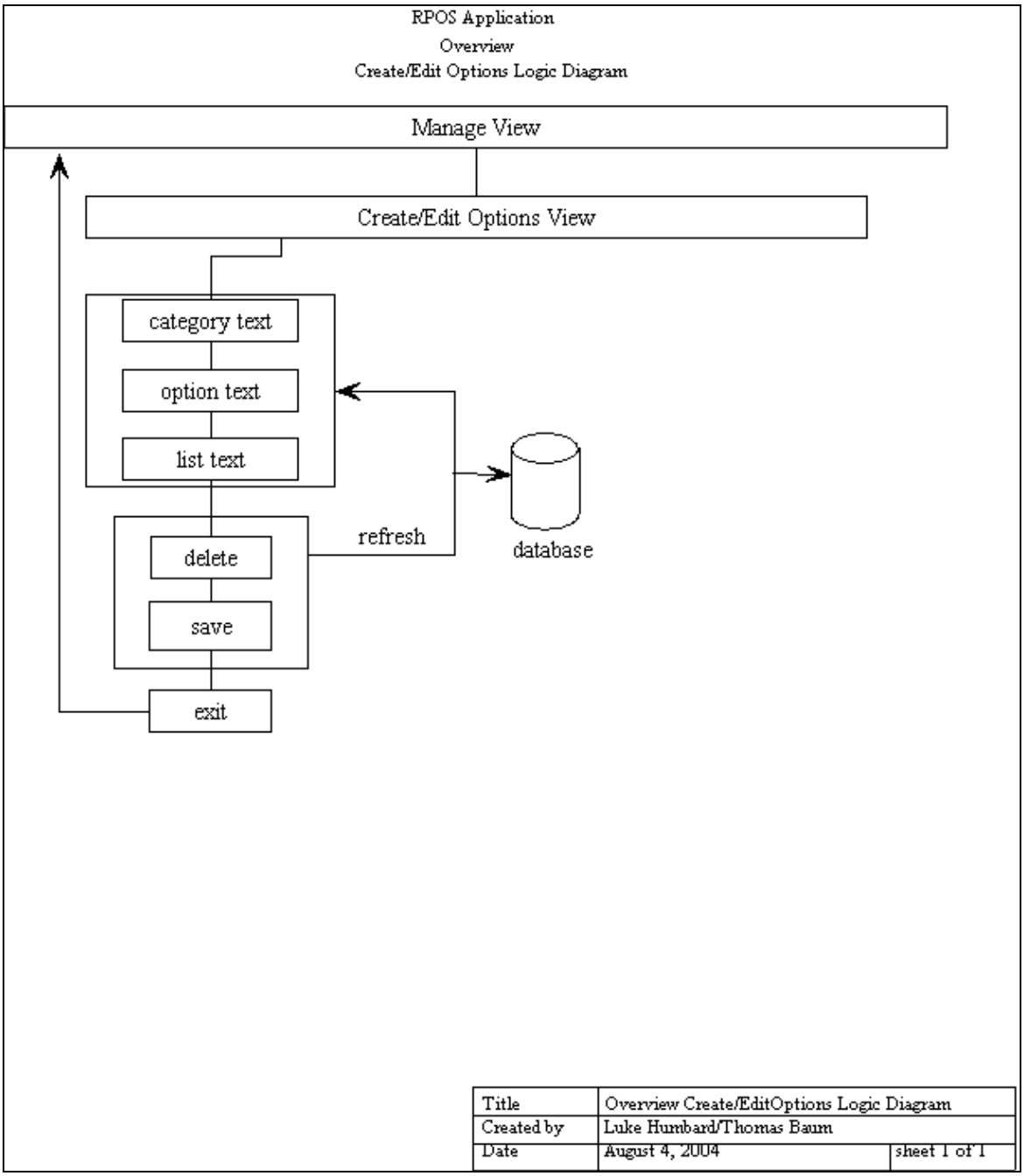


Figure A 3. (Create/Edit Options Logic Diagram)

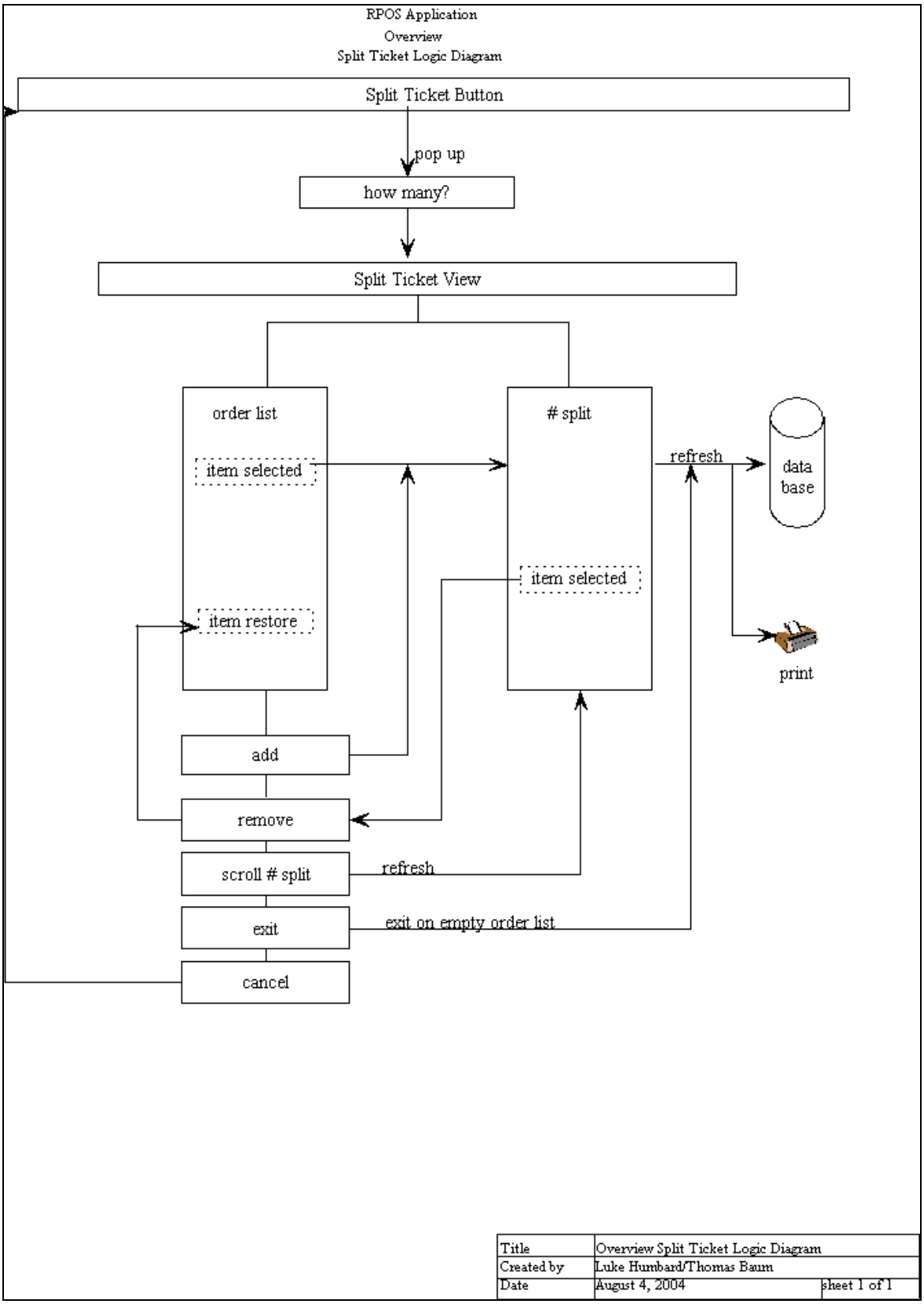


Figure A 4. (Split Order logic View Diagram)

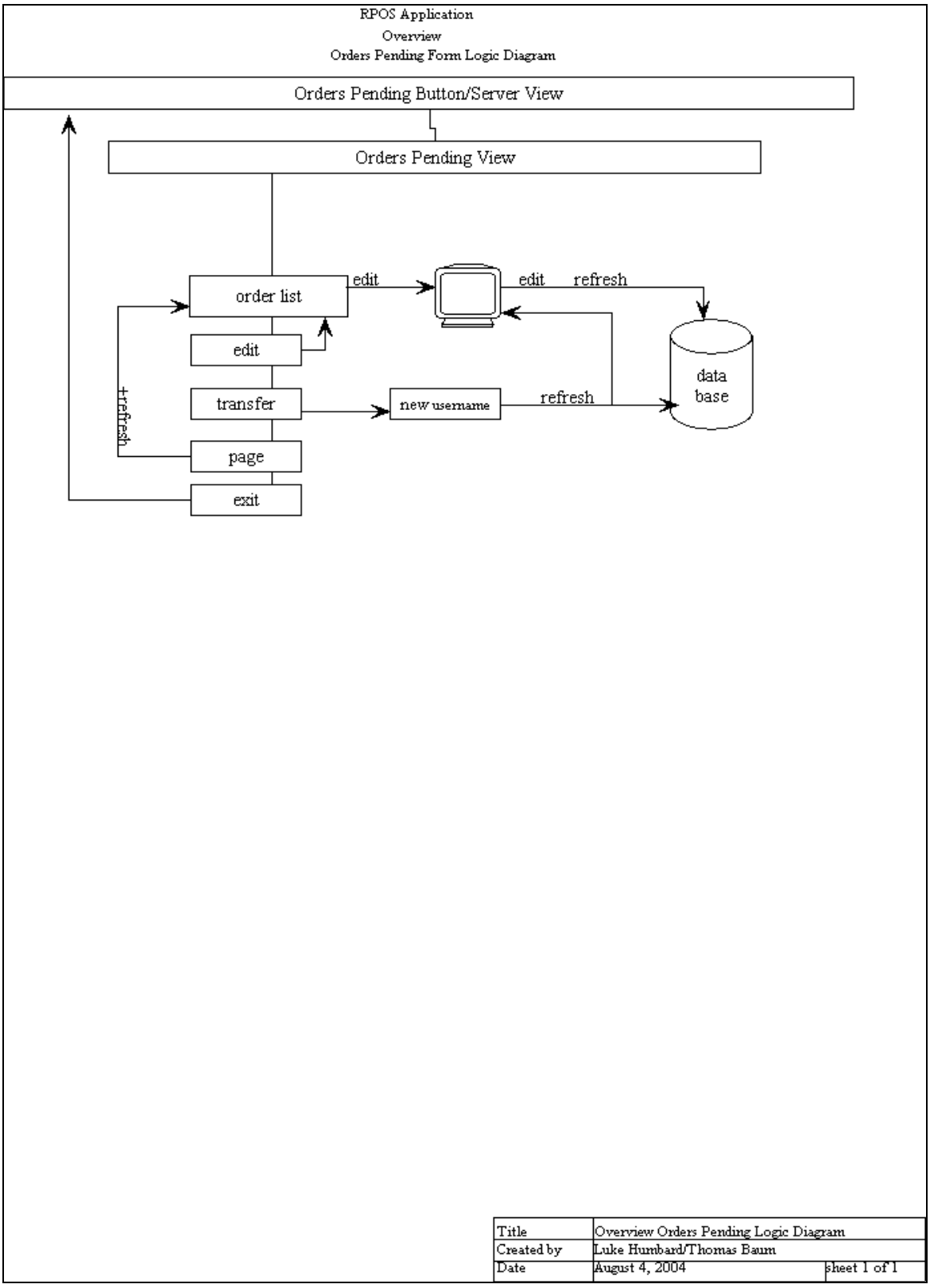


Figure A 5. (Orders Pending Logic Diagram)

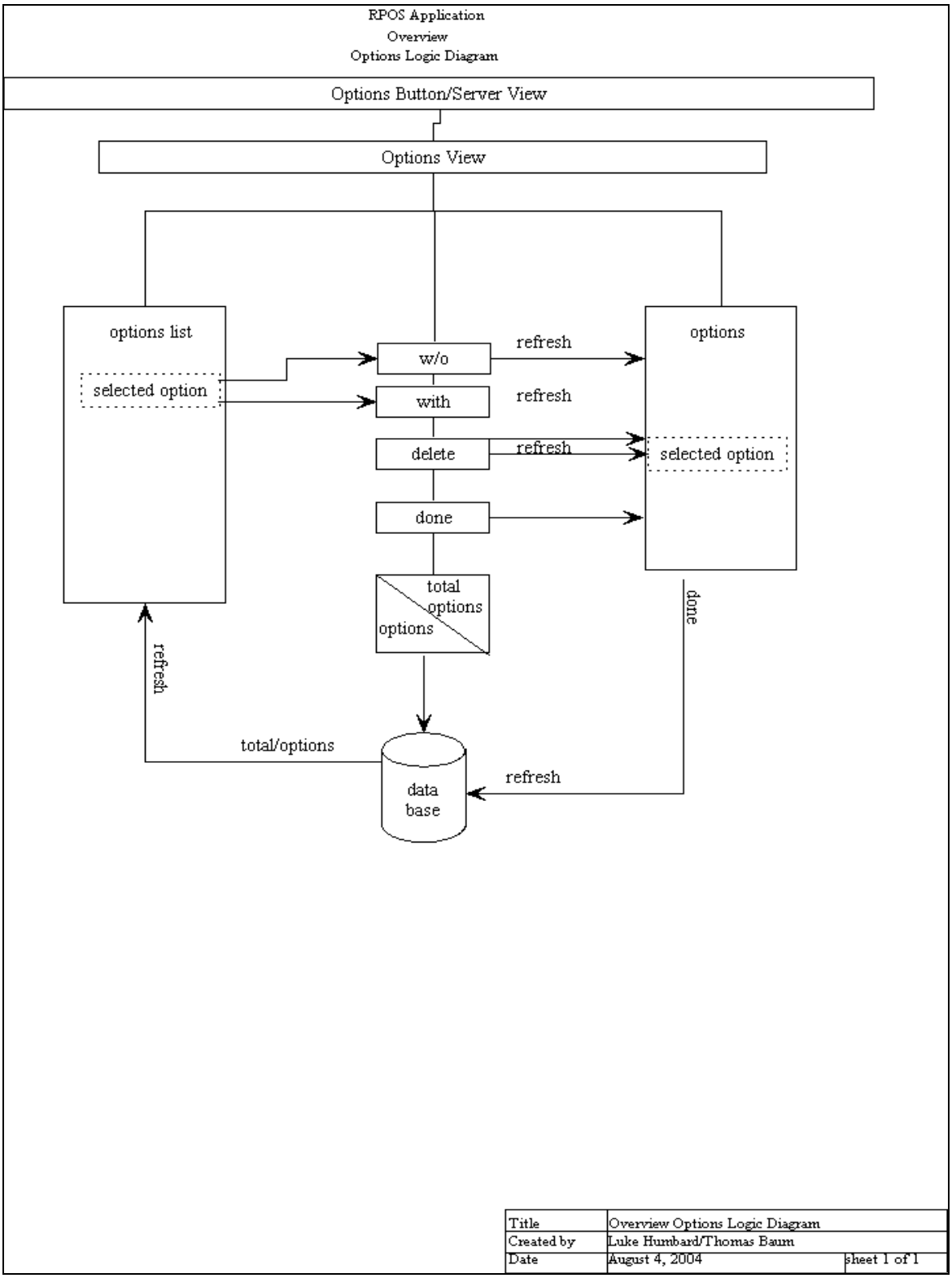


Figure A 6. (Options Logic View Diagram)

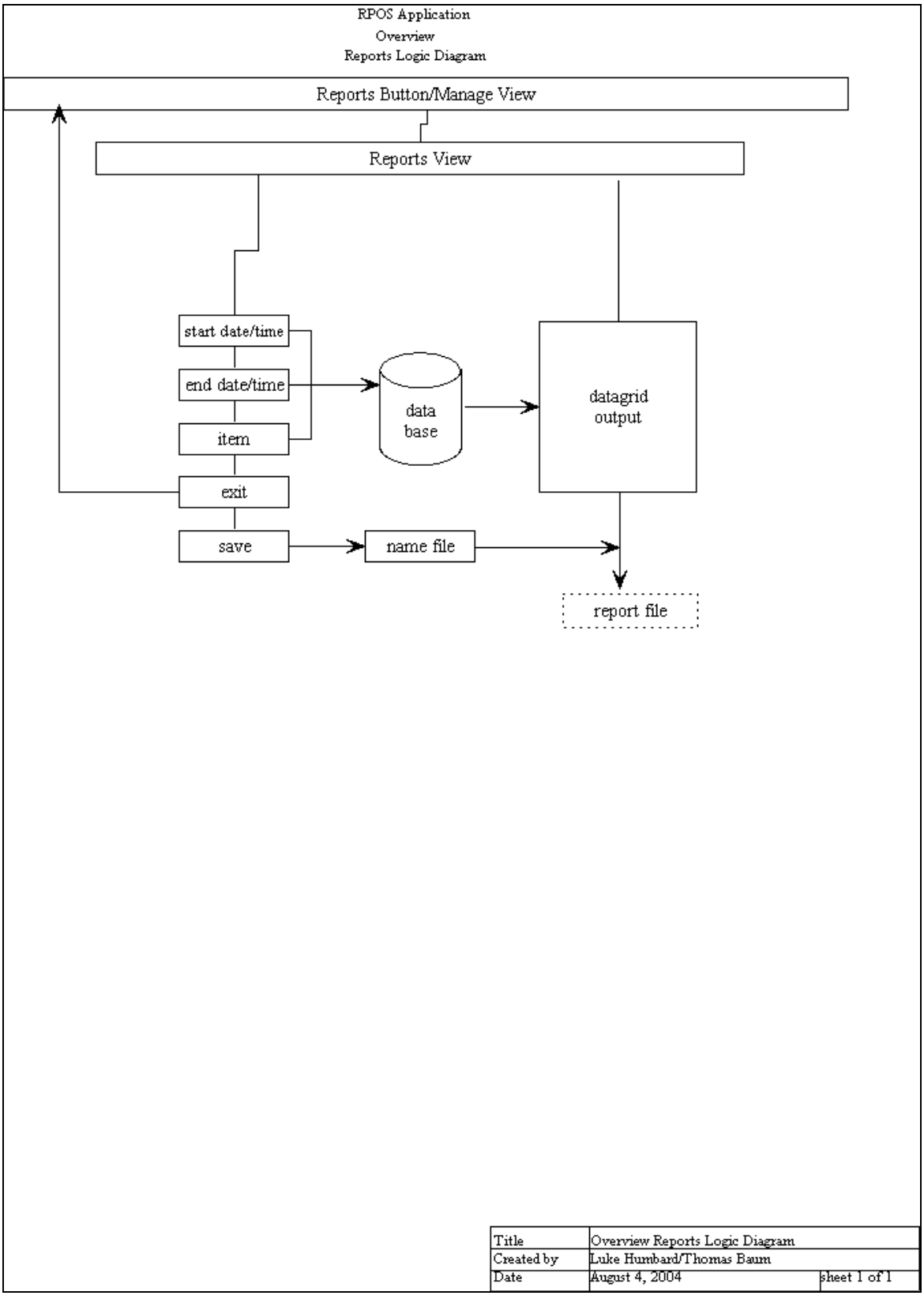


Figure A 7. (Reports Logic View Diagram