

---

# *Software Design Specification for Rub BBQ RMS*

---

Version 2.1  
1/21/2011



**Prepared for:**

Jared Leonard, Owner  
RUB BBQ  
2407 W Lunt Ave.  
Chicago, IL, 60645

**Prepared by:**

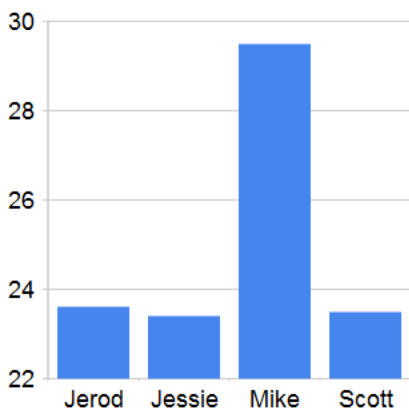
Jerod Hodgkin  
Mike Schenk  
Jessie Floyd  
Scott Leonard

Weber State University  
Ogden, UT 84408

### Responsibility Matrix

Levels	Team Member Name	Jerod	Jessie	Mike	Scott
	Project management (10 points)	75%		25%	
	Sec.3: Customer Statement of Requirements (6 points)	25%		25%	50%
	Sec.4: Glossary of Terms (4 points)		50%	50%	
	Sec.5: Functional Requirements Specification (37 points)	15%	40%	25%	25%
	Sec.6: Nonfunctional Requirements (6 points)		25%	25%	50%
	Sec.7: Domain Analysis (25points)	30%	10%	30%	30%
	Sec.8: User Interface Design (8 points)	50%		50%	
	Sec.9: Plan of Work (3 points)	25%	25%	25%	25%
	Sec.10: References (1 point)	50%		50%	

### Responsibility Allocation Chart



## Summary of Changes

Change Number	Date Completed	Location of Change	A M D	Description	Approved by (initials)	Date Approved
001	11/22/2010		A	Started the document.	JTH	11/22/2010
002	11/22/2010	Use Cases	M	Added or Updated Use Cases	JLF, JTH, SWL, MS	11/22/2010
003	11/29/2010	Use Cases	A	Added and updated Use Cases	JLF, SWL,MS	11/22/2010
004	11/30/2010	Entire Doc	M	Formatted the document for printing	JTH	11/30/2010
005	1/14/2011	User Stories	M	Added some user stories and mathched them all up with Jira.	JTH	1/19/2011
006	1/19/2011	History of work and Gantt Chart	M	Updated the project plan and gantt chart	JTH	1/21/2011

\*A - ADDED M - MODIFIED D - DELETED

## Table of Contents

---

Responsibility Matrix.....	2
Responsibility Allocation Chart.....	2
Summary of Changes.....	3
Introduction (Customer Statement of Requirements).....	7
Purpose.....	7
Scope .....	7
References .....	7
Audience.....	8
System Overview .....	9
System functionality .....	9
System Components.....	9
Basic Design Approach .....	10
User Interface Prototypes .....	11
Login Screen .....	11
Order Entry Screen .....	12
Cook's Screen .....	13
Inventory Management Touch-Screen.....	14
Inventory Management.....	15
Menu Management.....	16
Employee Management .....	17
Report Selection .....	18
Running Reports .....	19
Design Considerations.....	20
Assumptions and Dependencies .....	20
General Constraints.....	20
Architectural Strategies.....	20
System Architecture .....	22
Use Cases.....	22
Menu Management.....	22
Create/Update menu – BBQRMS-56 .....	23
Create menu item – BBQRMS-31 .....	23
Inventory Management.....	25
Add Supplier – BBQRMS-17 .....	25
Receive an Inventory Order – BBQRMS-18 .....	25

Reconcile Inventory – BBQRMS-19 .....	26
Place an Inventory Order – BBQRMS-57 .....	27
Cooks interface.....	28
Complete a customer order – BBQRMS-32.....	28
Customer Order Entry Subsystem .....	29
Place and save customer order without taking payment – BBQRMS-33.....	29
Take a customer order and accept payment – BBQRMS-35 .....	29
Recall and modify saved customer order – BBQRMS-37 .....	30
Cashiering Subsystem.....	32
Split a check – BBQRMS-34 .....	32
Accept payment for saved customer order – BBQRMS-36 .....	32
Take multiple payments for a customer order – BBQRMS-38 .....	33
Admin Functions.....	34
Managers Dashboard – BBQRMS-50.....	34
Add/Edit Employees – BBQRMS-39.....	34
Security.....	36
Log In – BBQRMS-22.....	36
Log Out – BBQRMS-23.....	36
Clock in and out – BBQRMS-30.....	37
Change own PIN – BBQRMS-25 .....	38
Program features are limited by assigned user roles – BBQRMS-24 .....	38
Reporting.....	40
Select and Run Report – BBQRMS-43.....	40
Developer Adds New Report – BBQRMS-43.....	40
Inventory Turns Report – BBQRMS-29 .....	41
Daily Sales Report – BBQRMS-28 .....	42
Weekly Prime Cost Report – BBQRMS-27 .....	42
Daily Shopping List Report – BBQRMS-2 .....	43
Logical View .....	45
Class Diagrams.....	46
Development View .....	51
Algorithms and Data structures .....	52
Process View.....	52
Order Placement and Completion.....	52
Run Report.....	53

## *Software Design Specification*

Menu Management.....	54
Inventory Management.....	55
Employee Management Activity Diagram.....	56
Physical View .....	57
Policies and Tactics .....	58
Detailed System Design .....	58
Database.....	58
Orders and menus .....	58
Inventory .....	59
Employees .....	59
Glossary .....	60
Bibliography.....	61
History of Work .....	62
Gantt Chart.....	63

## Introduction (Customer Statement of Requirements)

---

Jared Leonard owns a barbecue restaurant in Chicago, IL called Rub BBQ. He has requested an application for tracking his sales. He needs to track the number of menu items that were sold, what the method of payment was, whether it was an eat-in or take-out order, how it was ordered (phone-in, walk-in) and whether it was a order for catering or a single customer. He will eventually need to use this information to generate reports. In addition, the system to be built will include the ability to manage inventory, manage employee scheduling, and support accepting payment.

## Purpose

---

The system under development is proposed to assist in inventory management, product ordering, sales tracking, and profit/revenue projections. Jared Leonard, Ted Cowan and the development staff (Jerod Hodgkin, Mike Schenk, Jessie Floyd, and Scott Leonard) have input to this document. This document is intended to provide a basis for expectations and deliverables. It is managed by the development staff with input from the principals (Jared Leonard, Ted Cowan).

## Scope

---

Primary development efforts shall be focused on the following areas:

- Inventory management
- Taking customer orders
- Integration with a 3rd party payment processing system
- Managing menus to support taking orders
- Integrating a flexible reporting package
- Creating an initial set of reports
- Employee schedule management

This system will not provide:

- A payment processing system
- Support for 3rd party application integration
- Detailed accounting

This project is a new design and as such has not yet created a history of previous efforts. Operations or “sustainment” efforts are not scheduled into the project. All additional software modifications will be contracted on an “as-needed” basis and managed via the project charter.

## References

---

The development staff will maintain revisions of additional documentation via GSC-SVN. Changes will be checked in and maintained there. The following documents will be available for review upon request:

- RMS Project Charter
- WBS
- Project Proposal



## Audience

---

This document is intended for both the software developers and the customer (Jared Leonard) and is intended to be a “living” document to be updated as the project requirements become more definite and aspects of the project mature.



## System Overview

---

### System functionality

---

The RMS system will allow the manager to create one or more menus consisting of numbered menu items with their corresponding prices. These menus and menu items will be used to dynamically create an order-entry display that employees will use to take and record customer orders. The order entry screen will be built to be used with a touch-screen device to support quick data entry. In the absence of a touch screen, the menu item numbers will allow orders to be entered quickly by keyboard primarily using the numeric keypad. The order entry screen will also allow users to enter discounts or manually adjust prices. Once the order is complete, users will be able to accept payment from the customer and record in the system what the method of payment was. If it is a credit or debit card payment, the user will key-in the card information or swipe it through an attached card reader, after which the system will send a charge request to a third-party payment processing web service. The card information will not be permanently stored.

Once an order is entered, it will appear on a separate touch-screen terminal display in the kitchen where the cook(s) will begin preparing the customer's order. Once an order is prepared, the cook will press a button on the touch-screen to mark the order complete and dismiss it from the screen.

The RMS system will allow the owner to track restaurant inventory. Inventory tracking will occur in two parts. First, as the cooks pull ingredients from the pantry or refrigerator/freezer to be used, they will enter the amount they're using on another touch-screen terminal. The system will track the rate of usage of each inventory item. Second, the manager will use an inventory management screen to record the purchase of additional inventory. In this way, the RMS will be able to track and report inventory levels of those items being tracked. Periodically, a physical inventory count will need to be taken and the RMS inventory levels will need to be adjusted to account for inaccuracies in earlier data-entry and also due to waste and spoilage.

Employee management allows managers to control access to the RMS system for each individual employee. It will also support recording the information necessary to compute payroll by tracking employees names, pay type, wage or salary, and time worked.

Managers will be able to access screens for modifying the menu that regular employees will not be able to access. There, they will be able to add and remove items from menus, and change their prices.

Managers will also be able to access screens where they can choose from several reports to run. These reports have yet to be defined, but will include things like daily, weekly and monthly revenue, popular menu items, comparison of walk-in versus phone-in orders, comparison of catering business versus in-store business, employee hours worked, inventory costs, etc. Reports can be printed, saved as documents, or exported to a spreadsheet file.

### System Components

---

The RMS will be divided into the following separate components:

RMS database

RMS server

RMS order entry client

RMS cook's display

RMS menu management



- RMS inventory management
- RMS employee management
- RMS report interface
- RMS manager's dashboard

These components will be divided and deployed as three separate, communicating systems. All user interfaces will be included in a single executable program which allows users to log in and then limits their access to those interfaces appropriate for them. The RMS server will run in its own executable program, and the database will be deployed in a Microsoft SQL Server instance on the same machine as the RMS server.

### Basic Design Approach

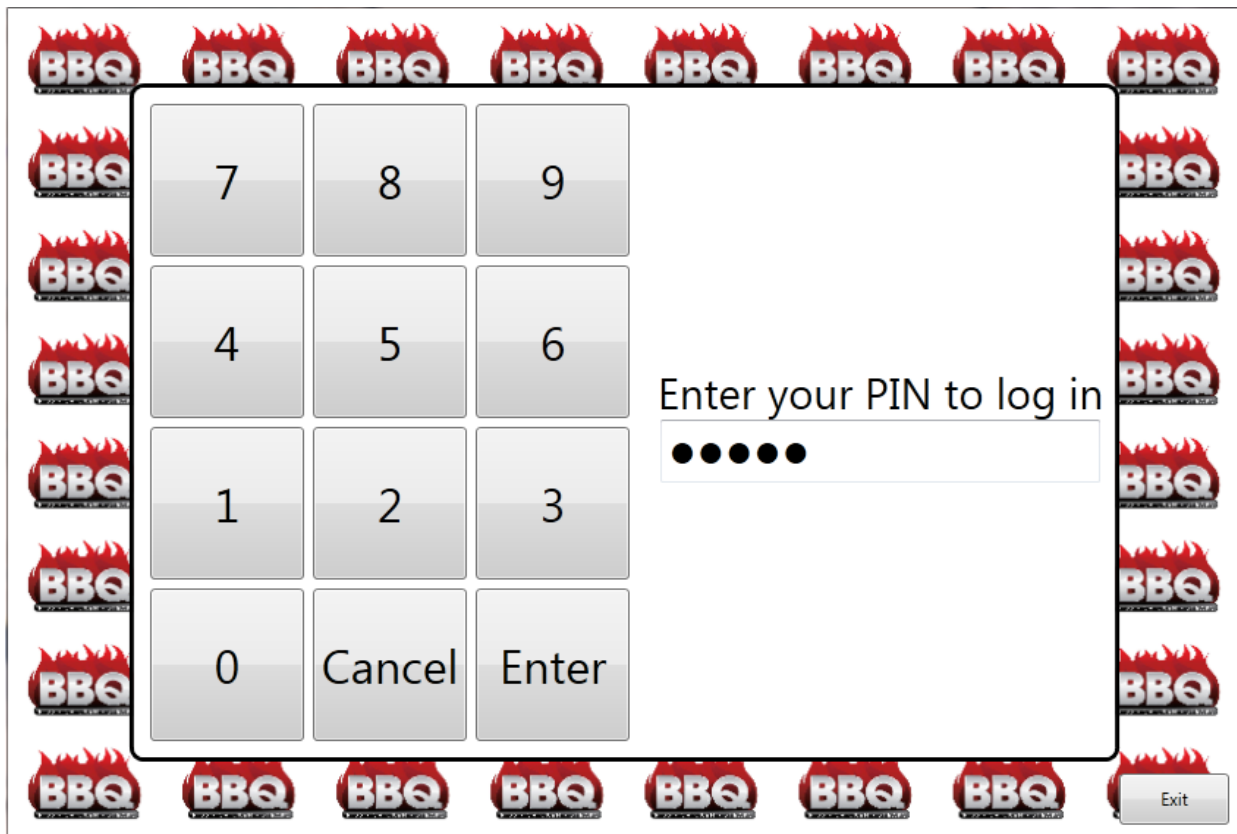
---

The RMS will be built as a client-server, Windows®-based application. The client and server may be deployed on a single machine, or may be deployed to separate machines connected with a standard TCP/IP network. User interfaces will be built using WPF with consideration for touch-screen constraints for the order-entry and cooks display. The management functions are expected to be used by a user sitting down with a keyboard and mouse, so normal input controls with regular sizes will be used in those areas.

The server will be a fairly simple set of WCF services that provide access to a central database of order, inventory, menu, and employee information.

## User Interface Prototypes

### Login Screen



The login screen features a central white rectangular area with a black border. Inside this area, on the left, is a numeric keypad with buttons for digits 0-9, a 'Cancel' button, and an 'Enter' button. To the right of the keypad, the text 'Enter your PIN to log in' is displayed above a PIN input field containing five black dots. The entire central area is surrounded by a decorative border of 'BBQ' logos, each featuring a flame icon. An 'Exit' button is located at the bottom right corner of the screen.

7	8	9
4	5	6
1	2	3
0	Cancel	Enter

Enter your PIN to log in

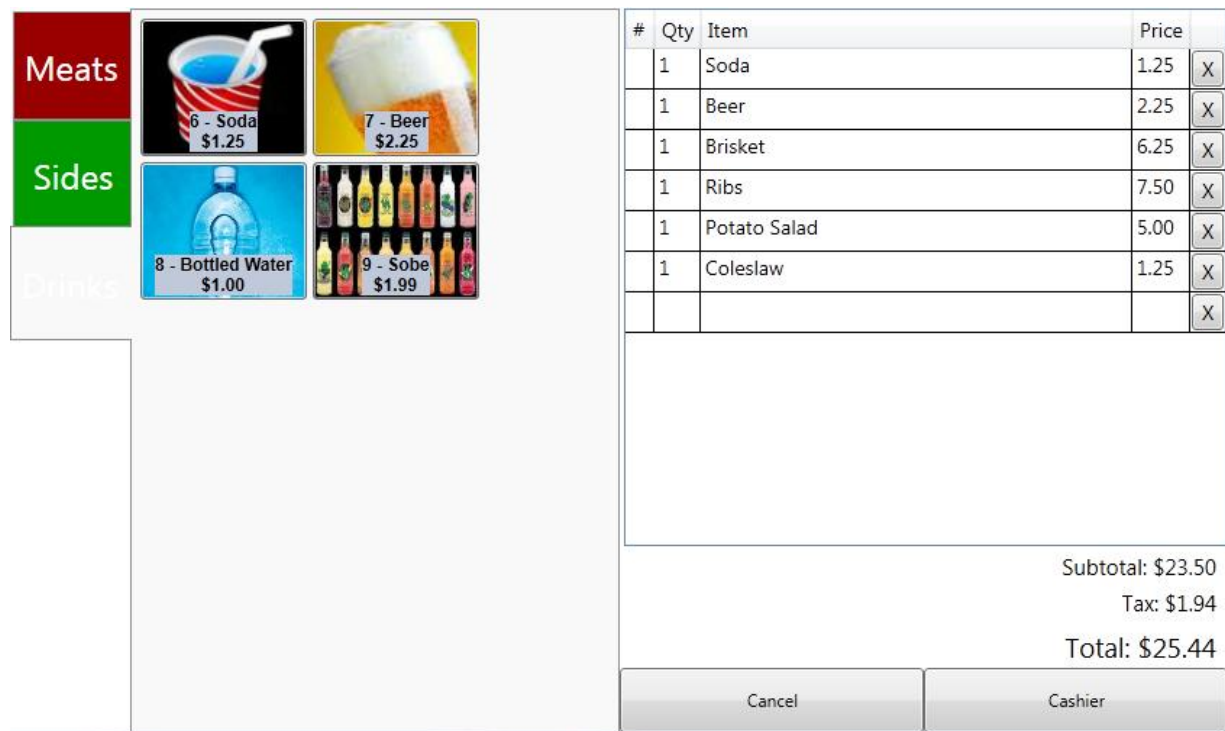
● ● ● ● ●

Exit

Figure 1. Login screen

To log in, users will be assigned a PIN number which they can enter through the touch-screen or keyboard.

## Order Entry Screen



The Order Entry Screen is divided into three main sections: a menu on the left, an order table on the right, and a summary/footer at the bottom.

**Menu Section (Left):**

- Meats:** A red tab.
- Sides:** A green tab.
- Drinks:** A light blue tab.

**Menu Items (Center):**

- 6 - Soda:** \$1.25 (Image of a soda cup)
- 7 - Beer:** \$2.25 (Image of a beer mug)
- 8 - Bottled Water:** \$1.00 (Image of a water bottle)
- 9 - Sobe:** \$1.99 (Image of several Sobe bottles)

**Order Table (Right):**

#	Qty	Item	Price	
	1	Soda	1.25	X
	1	Beer	2.25	X
	1	Brisket	6.25	X
	1	Ribs	7.50	X
	1	Potato Salad	5.00	X
	1	Coleslaw	1.25	X
				X

**Summary (Bottom Right):**

- Subtotal: \$23.50
- Tax: \$1.94
- Total: \$25.44

**Buttons (Bottom):**

- Cancel
- Cashier

Figure 2. Order entry screen

Orders may be entered through the touch-screen by pressing the menu item buttons, or through the keyboard, by entering menu item numbers and quantities. The screen opens with the keyboard focus in the “#” column. The user can enter the item number, then quantity. Using the touch screen, each touch of a menu item adds one more to the quantity of that item in the order. Pressing the “X” button on an order item will remove the item from the order.

Since this is intended to be used with a touch-screen interface, the buttons are large and easy to press.

Figure 2 shows an order in-progress. The user has just entered “11” to add a “BBQ Burger” to the order.

Each tab on the left represents a separate “menu.” Pressing a menu button will switch the screen to show that menu’s items in the central area.

The order entry prototype screen has changed since the version in the requirements document. It has been enhanced to add keyboard data-entry support.

## Cook's Screen



Figure 3. Cook's screen

The cook's screen shows the pending orders that need to be prepared. The oldest order is displayed on the left with a timer showing how long it has been since the order was placed.

## Inventory Management Touch-Screen

Pantry	Beans (#10 can)	Molasses (16 oz. bottle)	Catsup (bottle)
Fridge	<div>- 1 +</div>	<div>- 1 +</div>	<div>- 1 +</div>
Freezer	<div>Remove</div>	<div>Remove</div>	<div>Remove</div>

Figure 4. Inventory management touch screen

The kitchen staff uses this screen to quickly record inventory that is being used at the time they remove it. Since this is intended to be used with a touch-screen interface, the buttons are large and easy to press.

## Inventory Management

Inventory Management

Item	Qty on hand	Avg Daily Use	Reorder Threshold
Beef Ribs	8	5	10
Baby Back Ribs	8	5	10
Roast Beef	8	5	10
Brisket	8	5	10
Malassis	8	5	10
Brown Sugar	8	5	10
Katsup	8	5	10
Tender Loin	8	5	10
Pork Tender Loin	8	5	10
Chicken Breast	8	5	10
Smoke Sause	8	5	10
Elephant Thigh	8	5	10

Chicken Breast

Qty on hand: 8

Avg Daily Use: 5

Reorder Threshold: 10

Cost Per Unit: \$10.85

Last Ordered Date: 10/1/2010

Notes:

This is a hot selling item. Make sure to buy enough.

Distributor: Sysco Foods Co.

Unit Type: Case

Menu Items:

Family Meal Deal

Chicken Breast Meal

Chicken Soup

Oven Roasted Chicken Breast

Figure 5. Inventory management

Managers use the inventory management screen to:

Add new items to the inventory for tracking.

Reconcile the physical inventory count with the computed count.

Record spoilage and waste.

Stop tracking items.

## Menu Management

Item	Cost	Sale Price	Mark up %
St. Louis Style Spare Ribs	\$6.56	\$17.99	200%
Pulled Pork	8	\$6.99	10
Pulled Chicken	8	\$6.99	10
Brisket	8	\$7.99	10
Pulled Pork Sandwich	8	\$6.99	10
Pulled Pork Sliders (3)	8	\$9.99	10
Brisket Sandwich	8	\$7.99	10
Pulled Chicken Sandwich	8	\$6.99	10
Signature RUB Panini	8	\$7.99	10
RUB BBQ Burger	8	\$7.99	10
The Spiral Staircase of Cheese	8	\$6.99	10
Meaty Beans	8	\$6.99	0%

### Signature RUB Panini

<b>Name:</b>	Signature RUB Panini	<b>Cost:</b>	\$2.45	<b>Notes:</b> This is a hot selling item. Make sure to buy enough. Charts may appear here.
<b>Description:</b>	Pressed with either pulled chicken or pork and topped with sharp cheddar and house chutnev	<b>Sale Price:</b>	\$7.99	
<b>Category:</b>	SANDWICHES & BURGERS	<b>Mark up %:</b>	275%	
<b>Ingredients:</b>	Panini Roll Pulled Pork Signature BBQ Sauce Butter		<b>Avg Sold (Daily):</b> 26	

Figure 6. Menu management

Managers use the menu management screen to:

Add a menu to or remove a menu from the ordering screen.

Add items to or remove items from a menu, which changes the options on the ordering screen.

Change the name or price of menu items.

Change the inventory ingredients associated with each menu item.



## Employee Management

**Employee Management**

**Employees**

- John Doe
- Jane Doe
- Jane Doe
- Jane Doe
- Jane Doe
- Jane Doe
- Jane Doe

**John Doe**

**First Name:** 8

**Last Name:** 8

**Home Phone:** (801) 548-8467

**Cell Phone:** (801) 654-8945

**Other Phone:** (801) 894-2654

**Address:** 12345 Sumrandom Street

**Address Line 2:** Appt. #567

**City, State Zip:** West Valley City, UT 84095

**Role:** Cook

**User Privileges:** Edit Inventory, Manage Menu **Add Privileges**

**Comp. Type:** Hourly

**Comp. Rate:** \$4.25

**Last Clocked In:** 8:05am Oct. 25th 2010 **Clock on for:** 3:45:23

**Roles**

- Admin
- Cashier
- Cook
- Server

Figure 7. Employee Management

Managers will select an employee and edit the employee details.

## Report Selection

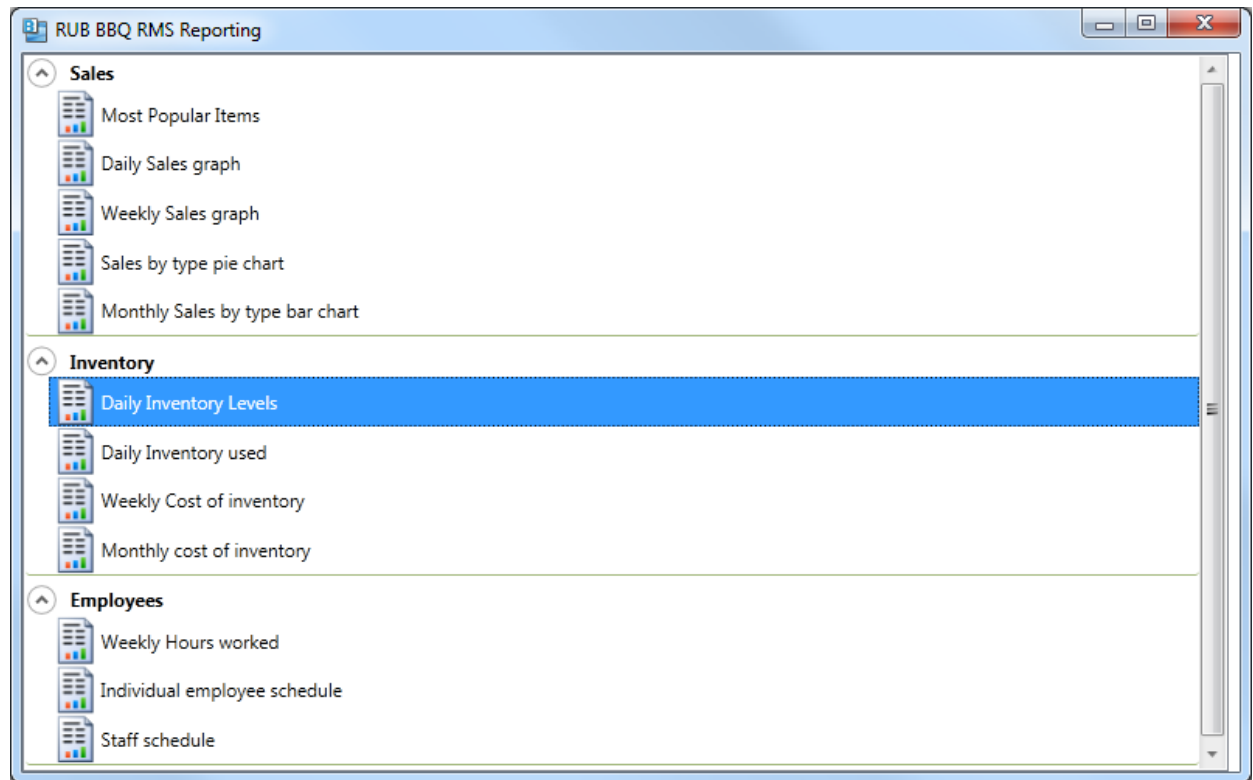


Figure 8. Report selection

Managers will select a report to run from this screen showing all the available reports organized into functional groups..

## Running Reports

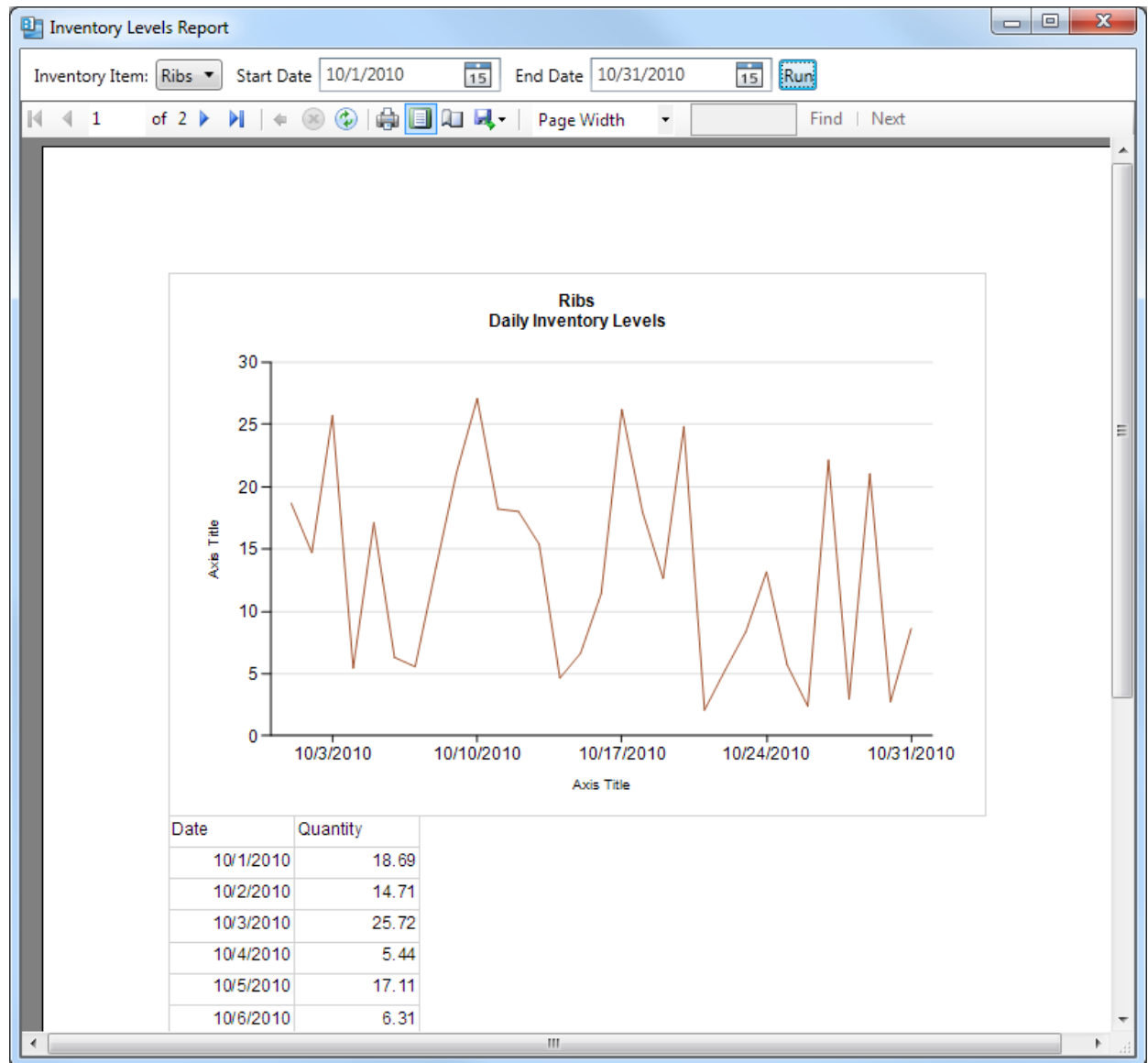


Figure 9. Running reports

Once a report has been selected, the report view screen will open and allow the user to enter the reports parameters at the top of the screen. Then after clicking "Run," the report will be generated and displayed below. The toolbar includes options for printing and exporting the report to various formats.

## Design Considerations

---

### Assumptions and Dependencies

---

The software application will be deployed on a computer running a Windows® operating system with the .NET 4.0 framework and having network access. Additional software can be installed by the user as required.

Specific system functions will depend on the availability of a credit card scanner, card payment processing system, a printer, and network access.

### General Constraints

---

All components will deploy and operate on Windows® based operating environments on commodity hardware.

Users are not expected to be computer-savvy and still should be able to operate the order entry and cooks screen after only minimal training.

The order entry and cook's terminals are expected to be in a noisy environment probably with poor lighting conditions. Screens and keyboards are expected to get dirty. Therefore, onscreen text, icons and controls should likely be quite large with good contrast.

Orders need to be taken and recorded even while the cashier's terminal is unable to communicate with the central server. For example, if the restaurant goes to a fair to sell food.

### Architectural Strategies

---

Selected a client-server, layered architecture (*answer 3a from report 2 instructions*)

- *The system should be able to run in a single user environment with one workstation acting as both client and server. As more users are added, the database should be moved to a server machine to allow access to clients.*

Selected Microsoft .Net 4 for the development platform.

- *Microsoft platforms give us the broadest and deepest options for data, user interface and development solutions. All development platforms used in this project will be Microsoft products. Microsoft .NET 4.0 is the latest Microsoft development platform.*

Selected WPF for the user interface platform

- *WPF is Microsoft's solution to provided rich, advanced user interfaces.*

Selected WCF (Data services) for the communication platform

Selected Microsoft SQL Server for the data storage

- *Microsoft SQL Server gives us the best database solution for the .NET platform.*

Selected Windows 7 for the target client operating system

- *This is the latest version of Microsoft Windows, and is the OS in use by the client on their current system.*

Intend to support touch-screen interaction

- *The client wants to maximize his customer interaction. A touch panel interface will allow him to maintain contact with the customer as orders are placed.*

Intend to support keyboard/mouse interaction

- *The keyboard and mouse will need to be used in areas where touch panels are*

*not possible, or not cost effective.*

Intend to allow users to use or not use the inventory module

- *The system will incorporate a modular design which will allow clients to use any or all parts of the system as needed.*

Intend to allow users to use or not use an integrated cashiering module

- *The system will incorporate a modular design which will allow clients to use any or all parts of the system as needed.*

Intend to allow orders to be taken while offline (i.e. off-site at a festival or fair)

- *BBQ restaurants make a lot of money at remote sites. This option is critical to allow the clients to use the same system in the restaurant or on the road.*

Intend to integrate with an online payment card processor already in use

- *The client needs to be able to take credit card payment using his existing payment processor.*

Intend to support a flexible reporting package

- *The true value of the system is in the ability to show the client how his business is performing and show him where he can improve.*

## System Architecture

### Use Cases



Figure 10. RMS use case diagram

Managers, Cooks and Order Takers are all users of the system so they all must log in. The other use cases are specific to the individual user's role. It is expected that some employees will have both the Cook and Order Taker roles, and Managers will most often also have Order Taker and Cook roles.

The customer is a secondary actor who interacts with the Order Taker during the Place Order and Record Payment cases.

### Menu Management

### Create/Update menu – BBQRMS-56

<b>Related use cases:</b> Inventory management Place order Complete order Create Menu Item	<b>Actors:</b> Managers
--	----------------------------

#### Summary

The menus will contain a list of items sold to the customers. Menus are constructed of menu items. The owners should be able to create new menus from existing menu items and change details of the current menu including, menu items, and description.

#### Preconditions

None.

#### Input/Output sequence for this feature

The Managers may edit or create a menu by clicking on the Manage Menu button. A list of existing menus will appear on the left with an add button on the bottom of the list. The Manager may edit an existing menu by selecting it from the list. The Manager may create a new menu by clicking the add menu button. Once the menu is selected the manager will then be able to alter the menu's information.

#### Postconditions

There exists a menu that can be displayed to the employees.

#### Design constraints of this feature

Menu management shall read the inventory to construct menu items. Menu items shall be available to be added to an order.

#### Performance requirements of this feature

No performance requirements have been identified.

### Create menu item – BBQRMS-31

<b>Related use cases:</b> Inventory management Place order Complete order	<b>Actors:</b> Managers
--	----------------------------

#### Summary

The menu items will be the items sold to the the customers. Menu items are constructed of inventory items. The owners should be able to create new menu items from existing inventory items and change details of the current menu items including, recipe, description and sale price.

#### Preconditions

There exist items in the inventory database with which to construct menu items. There exist menus in the database to add menu items to.

**Input/Output sequence for this feature**

The inventory items shall be read from the database and be combined to create a recipe that is associated with a menu item. The manager shall select a menu to which this item is to be added.

**Postconditions**

There exists a menu item that can be ordered by a customer.

**Design constraints of this feature**

Menu management shall read the inventory to construct menu items. Menu items shall be available to be added to an order.

**Performance requirements of this feature**

No performance requirements have been identified.



## Inventory Management

---

### Add Supplier – BBQRMS-17

---

<b>Related use cases</b> Log in Inventory Management	<b>Actors</b> Managers
--	---------------------------

**Summary**

Provide a user interface to add new suppliers. These suppliers are the companies or people that sell and deliver the products that make up the inventory.

**Preconditions**

None

**Input/Output sequence for this feature**

The user shall input the supplier information. This information may include (but is not limited to) address, contact phone number, contact name, discounts. After the supplier is entered it shall be available on the inventory menu when adding a supplier to an inventory item.

**Postconditions**

There exists a supplier that can be associated with an inventory item.

**Design constraints of this feature**

This screen does not need to be designed as a touch interface, but ease of use for potential touch interface should be considered. Supplier names shall be used on other screens such as inventory management. These screens may require new constraints be added to this user story.

**Performance requirements of this feature**

None.

### Receive an Inventory Order – BBQRMS-18

---

<b>Related use cases:</b> log in Place an Order	<b>Actors:</b> Managers
---	----------------------------

**Summary**

The manager should be able to receive an order from a supplier marking the order complete in the BBQ RMS removing the item from a pending receipt table.

**Preconditions**

Knowledge of a shipment and the supplier and item(s)

**Input/Output sequence for this feature**

Identify the supplier, locate pending order, mark complete



**Postconditions**

Inventory tables are updated to reflect current status and quantity.

**Design constraints of this feature**

Items to be received into inventory will be known by the system prior to receiving an order. This is accomplished by the placed an order scenario. Once an order is known, completing an inventory should be quickly accomplished using a mobile device similar in design to the touch-sensitive tablet computer used for customer order entry.

**Performance requirements of this feature**

Able to complete an entire order as well as a partial shipment, meaning only some of the ordered items are shipped and received into inventory leaving some items 'outstanding' or pending shipment.

[Reconcile Inventory – BBQRMS-19](#)

<b>Related use cases:</b> Inventory reporting Complete order	<b>Actors:</b> Managers
--	----------------------------

**Summary:**

The owners should be able to track the use of inventory through the day and make adjustments to inventory when a physical inventory count is taken

**Preconditions:**

We have not identified the preconditions.

**Steps**

We have not yet created user stories for the use cases. We need information from our client

**Input/Output sequence for this feature**

The client shall enter an initial inventory count when they go live on the system. The client shall reduce the inventory by an appropriate quantity as inventory items are used to make menu items. The client shall take a regular physical inventory to reconcile the difference between the system inventory count and the actual inventory count.

**Postconditions**

The client will have an accurate count of their inventory.

**Design constraints of this feature**

The inventory shall be stored in the database. The database shall exist such that the inventory items and quantities shall be available in other areas of the system. Reporting, ordering, and the menu shall access the inventory as needed.

**Performance requirements of this feature**

Inventory management shall be able to store up to 10,000 inventory items. The inventory screen shall take less than 10 seconds to load when all inventory items are listed.

## Place an Inventory Order – BBQRMS-57

---

<b>Related use cases:</b> Log on Add Item	<b>Actors:</b> Managers
---	----------------------------

### **Summary**

Reorder items from inventory when stock supplies drops below allowable threshold.

### **Preconditions**

Suppliers are known and inventory items loaded into database tables.

### **Steps**

Locate the item in the inventory management menu, identify quantity desired and click “re-order” which prints a order form to be called in to the supplier.

### **Input/Output sequence for this feature**

### **Postconditions**

Database tables and inventory management system records pending order and presents the manager an option to close the order marking it complete.

### **Design constraints of this feature**

Identified Suppliers, quantity, shipment time, and cost are known to the manager prior to order.

### **Performance requirements of this feature**

Large number of suppliers shall not inhibit the system function

## Cooks interface

---

### Complete a customer order – BBQRMS-32

---

<b>Related use cases:</b> Menu management Place order Receive payment	<b>Actors:</b> Cooks
--	-------------------------

**Summary:**

After an order is placed, it will show up on the cook's screen. An employee in the kitchen will prepare the items on the order and mark the order completed. Complete order shall allow the employees in the kitchen to receive orders that have been placed and mark the order complete once the items have been prepared.

**Preconditions:**

There exist orders that have not been completed or paid for.

**Input/Output sequence for this feature**

Cooks shall be presented with incomplete orders on a screen. The time since each order was placed shall be displayed. Once an order has been prepared, the cook shall press an on-screen button to indicate it is complete and it will disappear from the screen. The customer order will be marked complete in the database with a timestamp so it can be reported on later.

**Postconditions**

An order will be stored in the database in a completed status with a completed date and time.

**Design constraints of this feature**

More than one employee shall be able to view the active orders at once. A certain minimum number of orders needs to be visible on screen at once.

**Performance requirements of this feature**

This feature shall be capable of showing new orders in near real time. A large number of orders shall not impact the performance of this feature.

## Customer Order Entry Subsystem

---

### Place and save customer order without taking payment – BBQRMS-33

---

<b>Related use cases:</b> Menu management Complete customer order Accept payment for saved customer order Recall and modify saved customer order	<b>Actors:</b> Customers Order Takers
--	---

#### Summary

Employees shall be able to add menu items to a customer order. Employees shall be able to save a customer order for later recall. Employees shall be able to send customer orders to the kitchen for customer order completion. Employees shall have the option to accept payment as the order is placed.

#### Preconditions

There exist menu items that can be sold on the order.

#### Input/Output sequence for this feature

Menu items shall be added to an order. For a catering order, additional customer information shall be recorded. The order can be passed to the cook so the food can be prepared. The order can be saved for later recall.

#### Postconditions

An order shall exist in the system that can be completed, cashiered or saved for recall.

#### Design constraints of this feature

Place order shall read menu items in the database so an order can be created. Place order shall save orders in the database. Place order shall do some simple calculations to determine order price based on menu item price and tax information. Several people shall be able to place an order simultaneously.

#### Performance requirements of this feature

This feature shall be in continuous use when the restaurant is open. This feature shall run without slowing down over long periods of time. As more orders are placed the performance of this feature shall not be impacted. As menu items are added to an order the performance of this feature shall not be impacted.

### Take a customer order and accept payment – BBQRMS-35

---

<b>Related use cases:</b> Menu management Place order Complete order Sales reporting	<b>Actors:</b> Order Takers Customers
--	---

#### Summary:

After an order is placed, it will show up on the cook screen. An employee in the kitchen will prepare

the items on the order and mark the order completed. These orders will show up on the cashier screen so the employee can ring them up and receive payment for the order.

**Preconditions:**

There exist orders that have been completed but not paid for.

**Input/Output sequence for this feature**

Completed orders shall be viewed in receive payment. Orders shall be paid and go to history. Orders for which payment is not received shall be marked unpaid at the end of the day, and shall still be a part of the history.

**Postconditions**

A completed paid order will show up in the order history.

**Design constraints of this feature**

Receive payment shall read completed orders from the database and see the price of menu items so the invoice can be totalled. Receive payment shall be able to add tax and tip to the total.

**Performance requirements of this feature**

This feature shall be capable of showing completed orders in near real time. A large number of orders shall not impact the performance of this feature.

Recall and modify saved customer order – BBQRMS-37

<b>Related use cases:</b> Take multiple payments for a customer order Accept payment for saved customer order	<b>Actors:</b> Cashier Customer Cook
---	---

**Summary:**

The cashier shall retrieve a saved customer order. The cashier shall make adjustments to the order and resend it to the kitchen, and either cashier the order or save it again.

**Preconditions:**

At least one uncompleted order has been saved. The cook has not marked the order completed.

**Input/Output sequence for this feature:**

The cashier shall recall a saved order that has not been marked completed by the cook. The cashier shall make changes to the order which could include adding or removing items to the order, applying discounts or payments to the order. The cashier shall save the order. If the order exists on the cook screen, the order shall be marked to show there was a change.

**Postconditions:**

The order that was changed will now show with the changes.

**Design constraints of this feature**

The cook should be notified of changes to any orders on the cook screen.

**Performance requirements of this feature**

None

## Cashiering Subsystem

### Split a check – BBQRMS-34

<b>Related use cases:</b> Take a customer order and accept payment Complete order	<b>Actors:</b> Order Takers Customers
---	---

#### Summary

Order Takers shall be able to modify an existing customer order in order to divide the check to produce separate receipts and totals to be paid separately.

#### Preconditions

An order is saved in the database.

#### Input/Output sequence for this feature

An Order Taker shall retrieve an existing, un-paid customer order. The order taker shall select items from the existing order to place on a new check. Each check can then be printed separately.

#### Postconditions

An order will be split so that it can be paid separately.

#### Design constraints of this feature

If the order is split before it has been completed by the cook, splitting the check should not impact the cook's screen and the "Complete a customer order" use case.

#### Performance requirements of this feature

No performance requirements have been set.

### Accept payment for saved customer order – BBQRMS-36

<b>Related use cases:</b> Take multiple forms of payment	<b>Actors:</b> Cashier Customer
---	---------------------------------------

#### Summary:

The cashier shall retrieve a saved customer order. The cashier shall identify the type of payment and record the amount of the payment received.

#### Preconditions:

At least one unpaid order has been saved.

#### Input/Output sequence for this feature:

The cashier enters or selects a saved order. The cashier shall select a payment type then enter an amount paid. The cashier can take multiple forms of payment by following the respective use case. The



system shall print a receipt. When accepting credit or debit card payments, the cashier shall swipe or enter the card number before entering the payment amount. The system shall communicate with a third party payment processor to process the card payment.

**Postconditions:**

The selected order will have payment information recorded and saved with it. A receipt will have been printed for the customer.

**Design constraints of this feature**

The third party payment processing system needs to be accessible from the terminal where the cashier is taking payment.

**Performance requirements of this feature**

This feature shall be in continuous use when the restaurant is open. This feature shall run without slowing down over long periods of time. As more payments are accepted the performance of this feature shall not be impacted.

Take multiple payments for a customer order – BBQRMS-38

<b>Related use cases:</b> Place a customer order	<b>Actors:</b> Cashier
---	---------------------------

**Summary**

Provide the cashier the ability to receive multiple types of payments for a single order. Allowing the customer to provide multiple payment options. For example; some amount of cash and the difference of the balance due provided by a debit card or credit card.

**Preconditions**

An order is placed and received for items on the menu.

**Input/Output sequence for this feature**

An order is communicated by the customer to the cashier. The order is understood and items are available.

**Postconditions**

An order exists that has payments made on it that can be resaved or cashiered.

**Design constraints for this feature**

Upon completion the bill satisfied and the order is submitted to the cooks.

**Performance requirements of this feature**

multiple cashier actions shall be able to complete simultaneously.

## Admin Functions

---

### Managers Dashboard – BBQRMS-50

---

<b>Related use cases:</b> Log in	<b>Actors:</b> Managers
-------------------------------------	----------------------------

**Summary**

A manager needs to be able to see a real-time updated, status of the performance of the business.

**Preconditions**

A manager is logged in.

**Input/Output sequence for this feature**

A manager logs in, then shall select an option to display the dashboard. The system shall periodically update the display to show certain key performance metrics.

**Postconditions**

A managers dashboard screen is displayed.

**Design constraints for this feature**

The display needs to represent several measurements in an easily understandable display.

**Performance requirements of this feature**

The information displayed needs to be up-to-date to within 2 seconds.

### Add/Edit Employees – BBQRMS-39

---

<b>Related use cases:</b> Log in	<b>Actors:</b> Managers
-------------------------------------	----------------------------

**Summary**

The managers should be able to create user accounts and set or change their access roles. The managers should be able to track the employees working hours and their wage or salary. Track employees work hours and control their access to the RMS.

**Preconditions**

A manager is logged in.

**Input/Output sequence for this feature**

The user chooses an existing employee to go to an editing screen with that employees data. The user enters or changes the employee's information, then issues a command to save their changes. The changed employee record is saved in the central database.

**Postconditions**

The appropriate users will be able to log in to the system. Managers will be able to export necessary information to an external payroll system.

**Introduction/Purpose of this feature**

The employee management feature shall allow the owners to add, remove and update records of employees. Those employees also may be users of the system so managers can use this feature to control their access.

**Design constraints of this feature**

The employee data shall be stored in the database. Employee data includes private, personal information which needs to be managed securely.

**Performance requirements of this feature**

Not yet defined.

## Security

---

### Log In – BBQRMS-22

---

<b>Related use cases:</b> Log Out Clock in and out Program features are limited by assigned user roles	<b>Actors:</b> All users
---	-----------------------------

**Summary:**

All users identify themselves to the system and gain access to its features by entering a unique numeric PIN.

**Preconditions:**

Employees have been configured as users who can log in.

Or, the system has been installed with an initial administrative user.

**Input/Output sequence for this feature**

User taps the numeric keys on the touchscreen to enter their PIN and then taps “Enter”.

A successful login causes the primary post-login screen for that user to be displayed.

A failed login causes a notification to appear and allows the user to try again.

A successful login may also start the employee’s time clock.

**Postconditions**

The system grants the user access to the appropriate screens and features.

**Design constraints of this feature**

The log in occurs via a touchscreen interface.

Users will be logging in and out very frequently.

A single PIN is more efficient to enter, but does not have the advantage of a username/password combination that allows the password to be changed while still identifying the user by their username. The PIN must be constructed so that it can be changed when necessary but still identifies a user.

**Performance requirements of this feature**

A successful login must take no longer than 2 seconds on average including the user entering their PIN.

### Log Out – BBQRMS-23

---

<b>Related use cases:</b> Log In Clock in and out	<b>Actors:</b> All users
---	-----------------------------

**Summary:**

Logged in users can quickly log out to prevent unauthorized use of the system when they step away from the terminal.

**Preconditions:**

None.

**Input/Output sequence for this feature**

User taps an on-screen logout button.

The screen is changed to the login screen and the user is logged out.

**Postconditions**

The system's features are inaccessible until the next login.

**Design constraints of this feature**

This occurs via a touch screen interface.

Users will be logging in and out very frequently.

When the system is in a state with incomplected work, logout should not be possible until that is cancelled.

**Performance requirements of this feature**

A logout must take no longer than 1 second on average.

Clock in and out – BBQRMS-30

<b>Related use cases:</b> Employee management Log In Log Out	<b>Actors:</b> All users
---	-----------------------------

**Summary:**

The system shall track hours worked by hourly wage employees. Employees will 'clock in' when they begin their shift by simply logging in to the system. They will 'clock out' during breaks and at the end of their shift by specifically doing so.

**Preconditions:**

Hourly-wage employees have been configured as users who can log in.

**Input/Output sequence for this feature**

The first time an hourly-wage user logs in, the system will record that they have 'clocked in.'

When they log out normally, they will remain 'on the clock.'

To clock out, the user taps an on-screen 'clock out' button which also logs them out.

**Postconditions**

The user is no longer logged in.

The system is no longer counting the users wages for reports and dashboards.

The time the user was on the clock has been recorded for future reporting purposes.

**Design constraints of this feature**

This occurs via a touch screen interface.

Users will often remain on the clock even while not logged in to the system.

**Performance requirements of this feature**

Clocking out must take no longer than 1 second on average.

[Change own PIN – BBQRMS-25](#)

<b>Related use cases:</b> Log In	<b>Actors:</b> All users
-------------------------------------	-----------------------------

**Summary:**

Users should be able to change their own PIN used for logging in.

**Preconditions:**

User is logged in.

**Input/Output sequence for this feature**

User selects a change PIN command.

The systems prompts the user to enter their current PIN and the ending part of their new PIN twice. The beginning part of their PIN is system-assigned to identify the user and cannot be changed.

If the new PIN entry is successful, the user's PIN is changed and the prompting window is closed.

If the two new PIN entries don't match. A notification is displayed and the user is allowed to try again or cancel.

**Postconditions**

The next time the user logs in, only their new PIN will succeed.

**Design constraints of this feature**

This occurs via a touch screen interface.

A single PIN is more efficient to enter, but does not have the advantage of a username/password combination that allows the password to be changed while still identifying the user by their username. The PIN must be constructed so that it can be changed when necessary but still identifies a user.

**Performance requirements of this feature**

None.

[Program features are limited by assigned user roles – BBQRMS-24](#)

<b>Related use cases:</b> Log In	<b>Actors:</b> All users
-------------------------------------	-----------------------------

**Summary:**

The features of the system shall be accessible only to users with the proper permissions.

**Preconditions:**

Employees have been configured as users who can log in.

**Input/Output sequence for this feature**

User logs in.

The navigation and other commands the system displays are limited to those the currently-logged in

user has access to.

**Postconditions**

Users cannot use features they do not have permissions for.

**Design constraints of this feature**

If a user is currently logged in to the system at the time a manager changes their permissions. Those changes will not apply until the user logs out and logs in again.

**Performance requirements of this feature**

None.

## Reporting

---

### Select and Run Report – BBQRMS-43

---

<b>Related use cases:</b>	<b>Actors:</b> Managers
---------------------------	----------------------------

**Summary:**

Periodically, or on an ad-hoc basis, owners and managers will need to view reports to determine the performance of the restaurant or to help make decisions about the restaurant. The system shall provide a mechanism that allows users with manager permissions to select and run reports for print output or for output to a file. Owners and managers want to view or print reports customized by parameters such as the applicable time period, the employee, a menu item, etc.

**Preconditions:**

Some data has been generated through use of the system.

**Input/Output sequence for this feature**

A manager logs in to the system.

The user opens the reporting screen.

The user selects the desired report from a categorized list.

The user is prompted to enter parameters the specific report needs.

The user executes a command to run the report and views it on-screen.

The user has options to print the report or save it in various formats.

**Postconditions**

The manager has a printed report or an electronic file containing the desired information.

**Design constraints of this feature**

Reports including both textual (numeric) data as well as charts and graphs shall be supported.

Required output formats are print, PDF, and spreadsheet.

**Performance requirements of this feature**

Built in reports shall take no longer than 120 seconds to run.

### Developer Adds New Report – BBQRMS-43

---

<b>Related use cases:</b> Select report	<b>Actors:</b> Software developer Managers
--	--

**Summary:**

The system shall be open for developers to create and integrate custom reports. When an owner or manager discovers the need for a customized report, the effort required to make the new report and include it in the product must be reasonable.

**Preconditions:**

The software is built and deployed with data available in the database.



**Input/Output sequence for this feature**

Owner or manager identifies report needs

Software developer creates a Visual Studio project that outputs a .DLL. This project references the BBQRMS.ReportingInterface.DLL.

Software developer creates a class which implements a provided interface, collects parameter values and retrieves data.

Software developer uses Visual Studio report authoring tools to create the report based on the product's database schema.

Software developer, using manager privileges, "installs" the report file (.dll file) by selecting the "install" command on the report screen and selecting the .DLL file.

**Postconditions**

The manager has access to the newly defined report which is able to prompt for parameter values and produce the correct output.

**Design constraints of this feature**

It shall not be necessary to recompile or re-deploy the BBQRMS application to add a custom report.

**Performance requirements of this feature**

None.

Inventory Turns Report – BBQRMS-29

<b>Related use cases:</b> Select report Reconcile Inventory Receive an Inventory Order	<b>Actors:</b> Managers
---	----------------------------

**Summary:**

"Inventory turns" is the number of times the entire value of the inventory has been consumed during a time period. A larger number means inventory is not spending too much time waiting to be used. Owners want to know the efficiency of their inventory ordering process.

**Preconditions:**

A full inventory has been taken and a dollar value assigned in order to start the use of the inventory tracking module.

Inventory has been consumed and that has been recorded through the 'quick inventory' screen or by reconciling inventory at a later date.

**Input/Output sequence for this feature**

A manager logs in to the system.

The user selects the Inventory Turns report.

The user enters the desired time period.

The report computes the value of the inventory at the start of the period and at the end. It also sums the value of the inventory added and consumed during the period.

The report calculates and displays the "Inventory Turns" number.

**Postconditions**

The manager knows how many inventory turns occurred during the period.

**Design constraints of this feature**

None

**Performance requirements of this feature**

Report shall display results in under 30 seconds from the time the user has entered the parameters and executed the report.

Daily Sales Report – BBQRMS-28

<b>Related use cases:</b> Select report	<b>Actors:</b> Managers
--	----------------------------

**Summary:**

Managers and owners can view gross sales per day over a specified time period. The report shall also break sales down by menus, menu items and order type. Owners want to look for sales patterns to try to find ways to increase sales and profit.

**Preconditions:**

Sales have been recorded.

**Input/Output sequence for this feature**

A manager logs in to the system.

The user selects the Daily Sales report.

The user enters the desired time period.

The user runs the report.

The report summarizes and displays sales data stored in the system.

**Postconditions**

The manager sees daily sales trends.

**Design constraints of this feature**

None

**Performance requirements of this feature**

Report shall display results in under 30 seconds from the time the user has entered the parameters and executed the report.

Weekly Prime Cost Report – BBQRMS-27

<b>Related use cases:</b> Select report	<b>Actors:</b> Managers
--	----------------------------

**Summary:**

Managers and owners can view the weekly cost of the major costs of running the restaurant. This is compared to the weekly total sales to determine performance, but not to calculate profit. Owners want to monitor their prime costs and keep them below a certain threshold in order to ensure profitability.

**Preconditions:**

Sales have been recorded.

Employee hours worked have been recorded.

Employee gross wages (including payroll taxes and other overhead) have been recorded.

Inventory has been reconciled or tracked during the week.

**Input/Output sequence for this feature**

A manager logs in to the system.

The user selects the Weekly Prime Cost report.

The user selects the desired week.

The user runs the report.

The report totals food and beverage costs with labor costs.

The report totals sales for the week and displays both along with a percentage.

**Postconditions**

The manager sees prime costs as a dollar number and as a percentage of total sales weekly.

**Design constraints of this feature**

None

**Performance requirements of this feature**

Report shall display results in under 30 seconds from the time the user has entered the parameters and executed the report.

Daily Shopping List Report – BBQRMS-2

<b>Related use cases:</b> Select report Reconcile Inventory Receive an Inventory Order	<b>Actors:</b> Managers
---	----------------------------

**Summary:**

Managers and owners can print a shopping list generated by the system based on inventory usage and anticipated menu item sales including pending catering orders. Owners and managers want the system to generate a list of inventory items have fallen below restocking levels.

**Preconditions:**

Inventory information is up-to-date in the system.

Catering orders are entered in the system with the necessary date.

**Input/Output sequence for this feature**

A manager logs in to the system.

The user selects the Daily Shopping List report.

The user runs the report.



The user prints the report.

**Postconditions**

The manager carries a shopping list out to buy inventory from his purveyors.

**Design constraints of this feature**

None

**Performance requirements of this feature**

Report shall display results in under 30 seconds from the time the user has entered the parameters and executed the report.

## Logical View

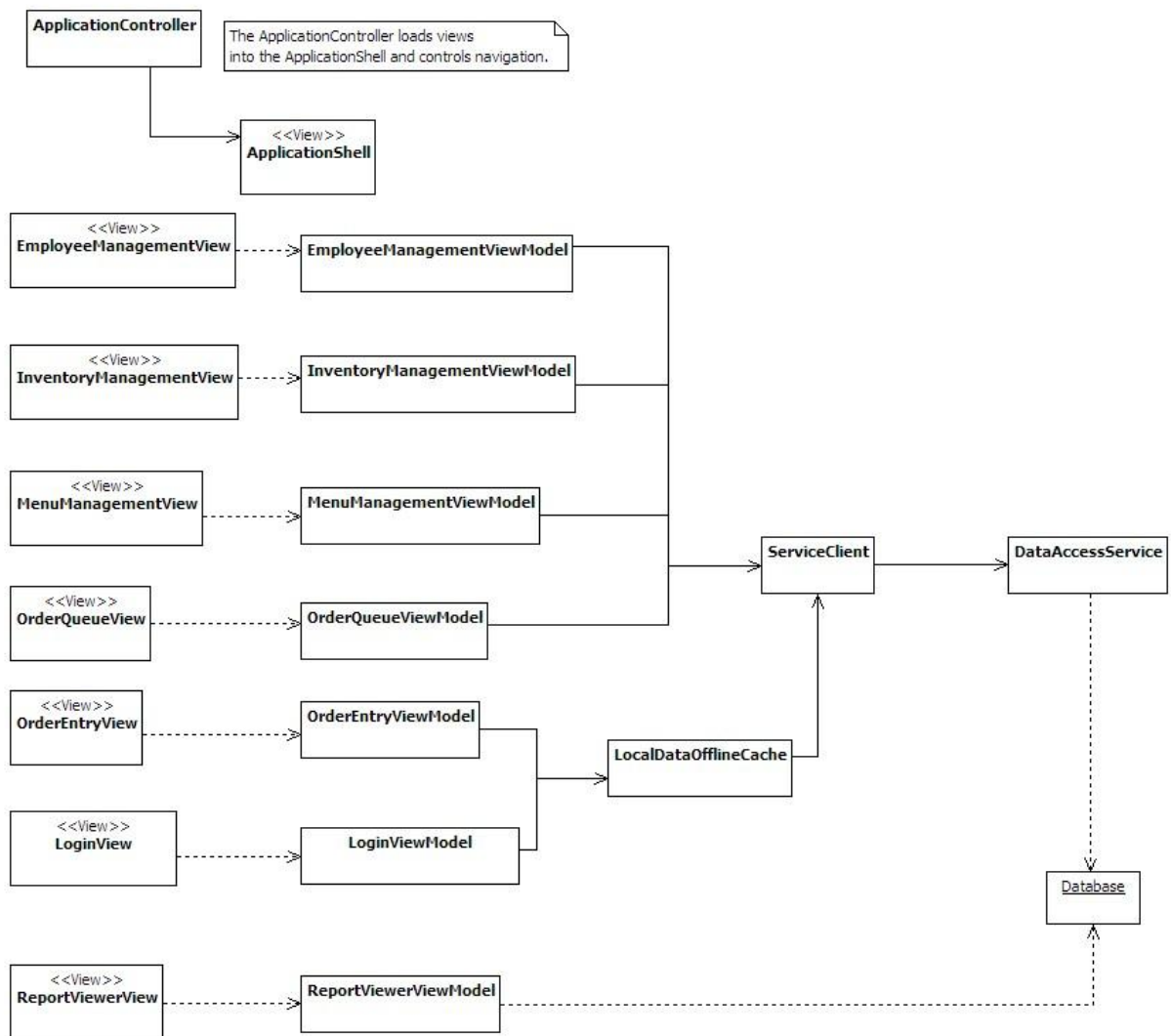


Figure 11a. Logical View

Figure 11a depicts the classes that will implement the behavior of the system. The classes follow a Model-View-ViewModel design pattern. The Model classes are not shown on this diagram, but their structure will mirror the entities in the entity relationship diagrams later in this document. They will be mapped from the database using Entity Framework.

## Class Diagrams

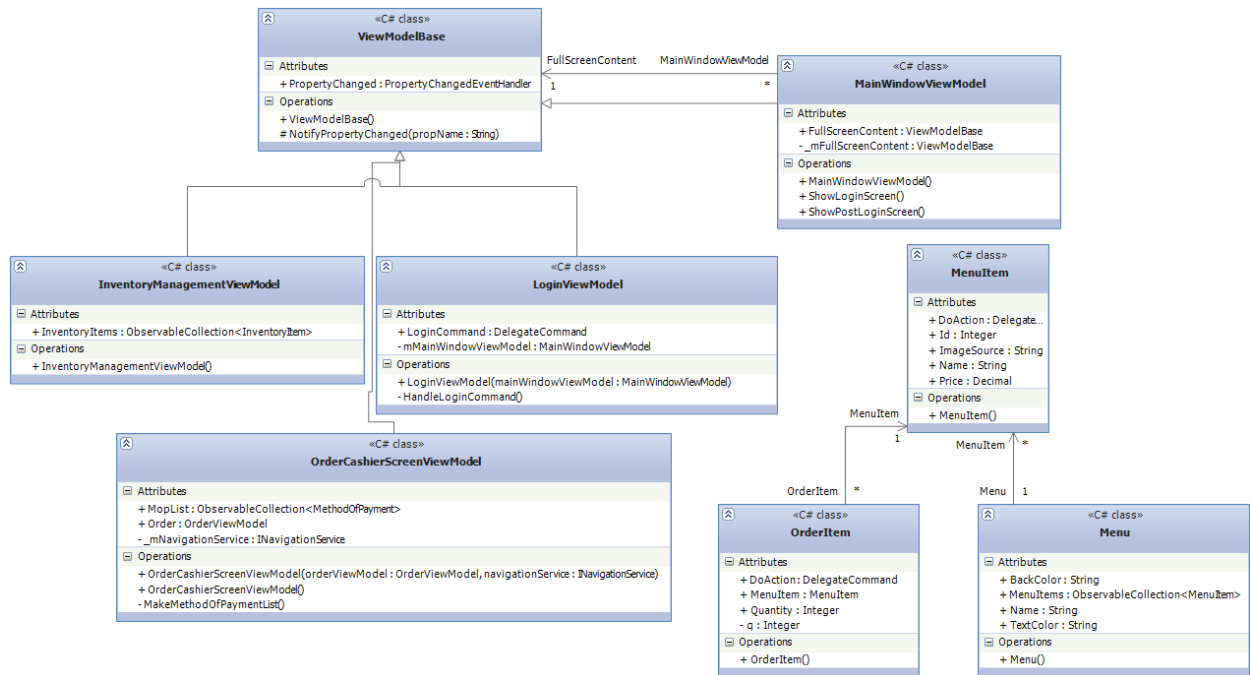


Figure 11b- Class diagrams with method signatures.

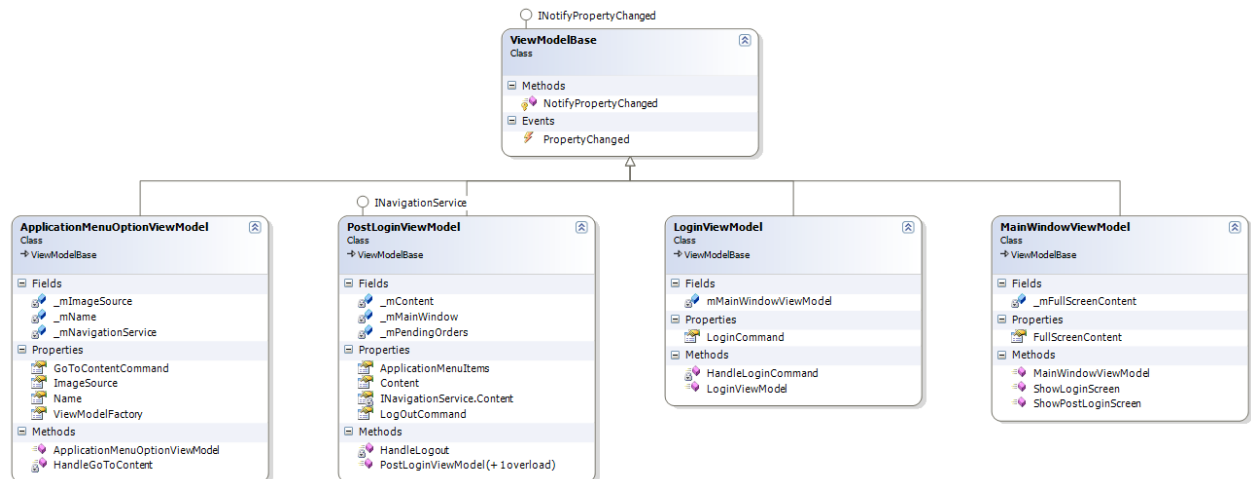


Figure 11b- View Models Related to Logging in and the Main Application

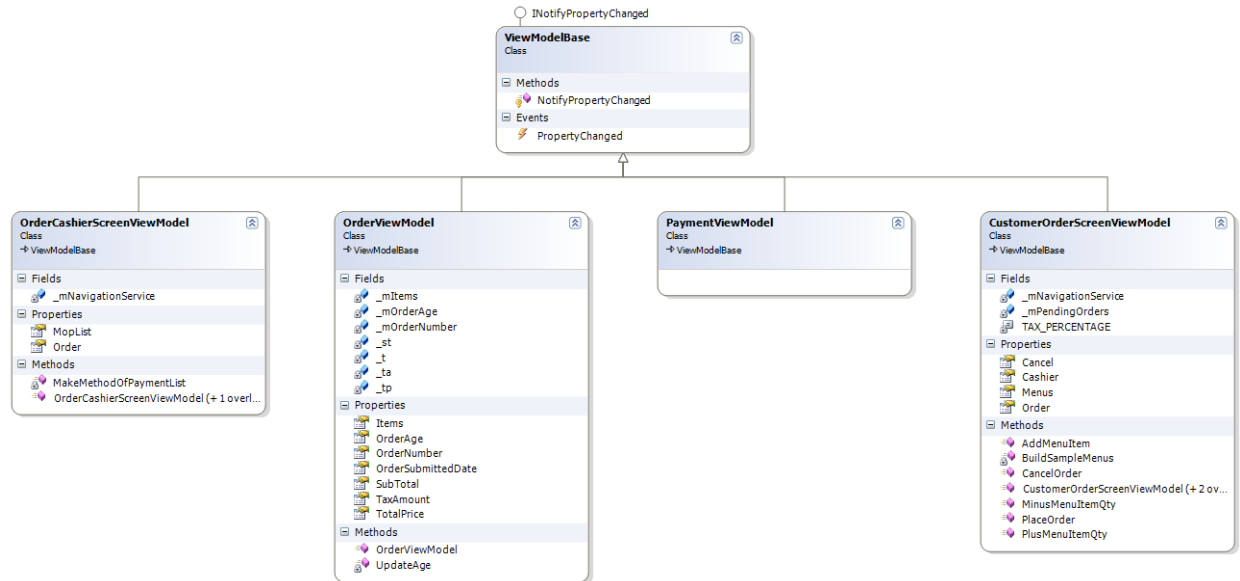


Figure 11c- View Models Related to Placing a Customer Order

Figure 11d- View Models Related to Reporting

11e

**Figure 11e- View Models Related to Inventory Management**

11f

**Figure 11f- View Models Related to Cook Screen**



a

**Figure 11g- Base Model and Descendants**

BBQ

**Figure 11h- Screens for User Controls (Code Behind)**



BBQ

**Figure 11i-Models and Helper Classes**

## Development View

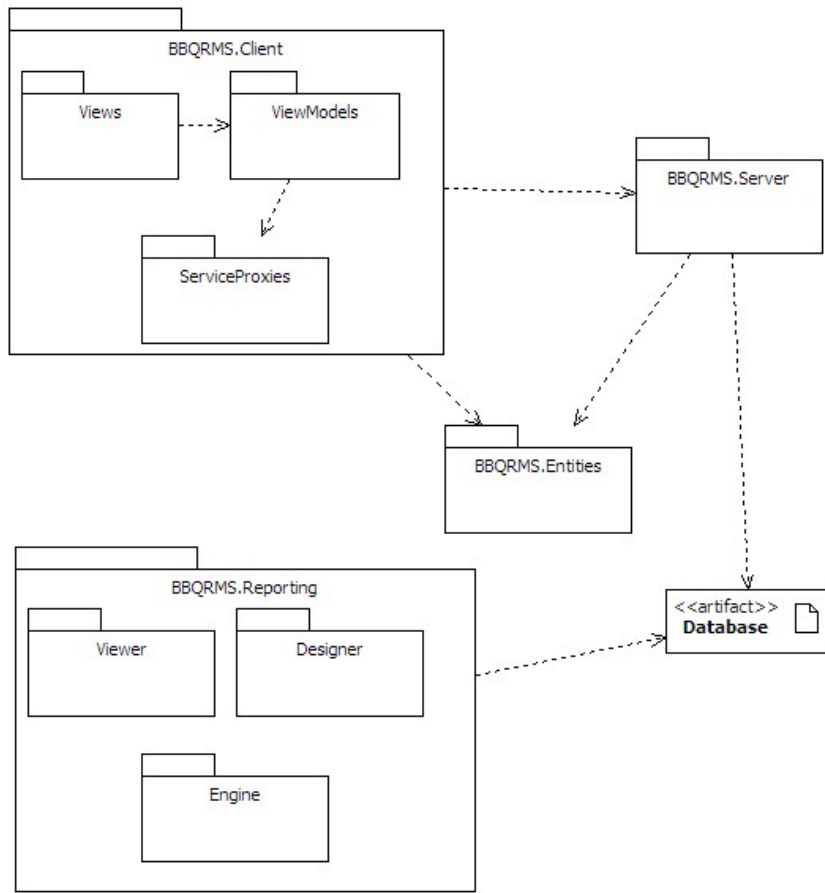


Figure 12. Package diagram

The RMS source code will be organized into packages as illustrated in Figure 12.

The BBQRMS.Client package will include all of the code running the user interfaces.

BBQRMS.Server will include WCF services exposed on the network providing access to the data stored in the central database. It will include Entity Framework classes, (the Object-Relational-Mapping system), to convert objects to database rows and vice versa.

BBQRMS.Entities will hold the data classes for the business entities that will be persisted in the central database. Instances of these entities will be transferred in serialized form from the server to the client and back.

BBQRMS.Reporting will hold the utility classes to support the integration of a pre-existing reporting package including its viewer component, rendering engine and report designer. Pre-defined report definitions will also be in the BBQRMS.Reporting package. The reporting subsystem will connect directly to the database since that is how most reporting packages expect to be used. It will not go through the same data access services and ORM layer as the other user interface components.

## Algorithms and Data structures

Algorithms used in the system will be straight forward summations of order items, order item prices, inventory orders, reporting, etc. This application will include few, if any, complex algorithms due to the nature of the application.

The system will utilize the data structures from the underlying architecture provided in the .Net Framework, primarily the Collection classes.

## Process View

## Order Placement and Completion

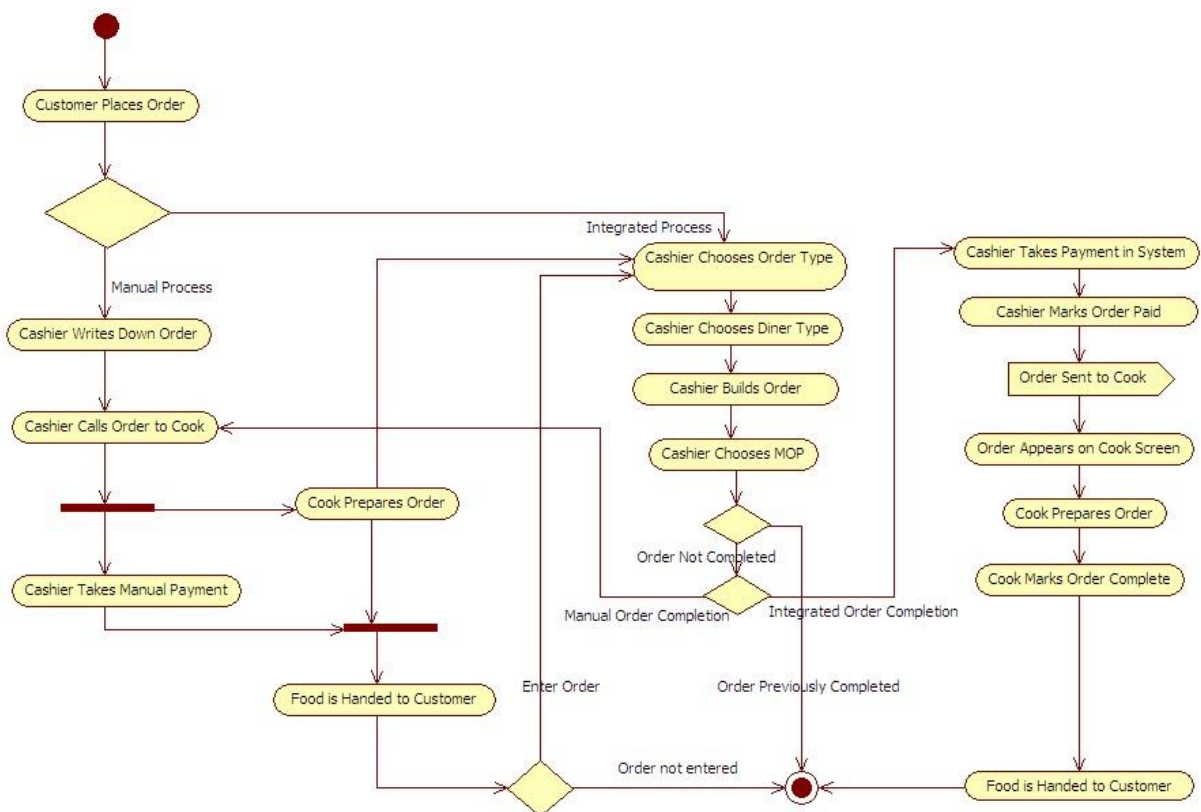


Figure 13. Order Placement and Completion Activity Diagram

## Run Report

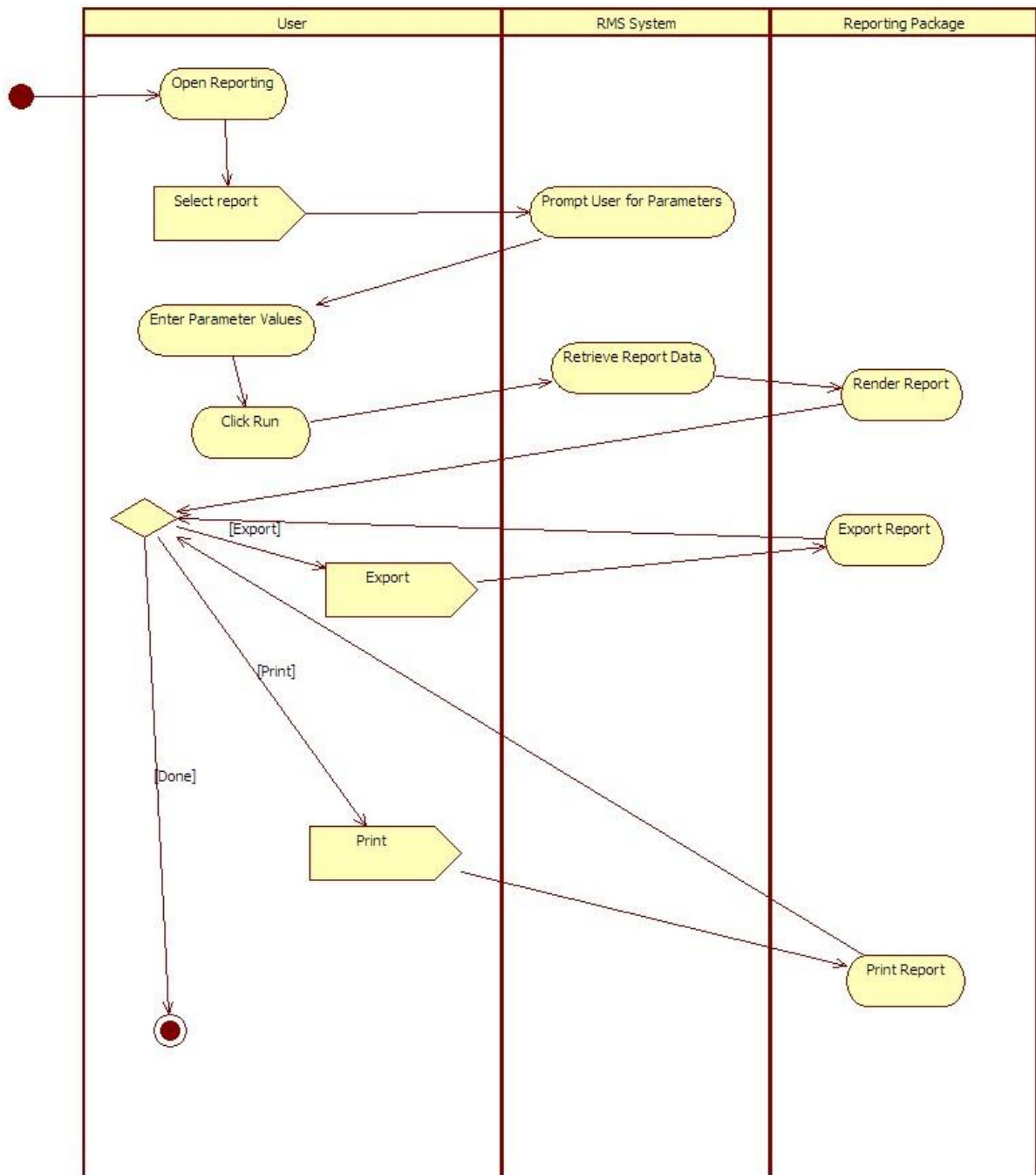


Figure 14. Run Report Activity Diagram

## Menu Management

---

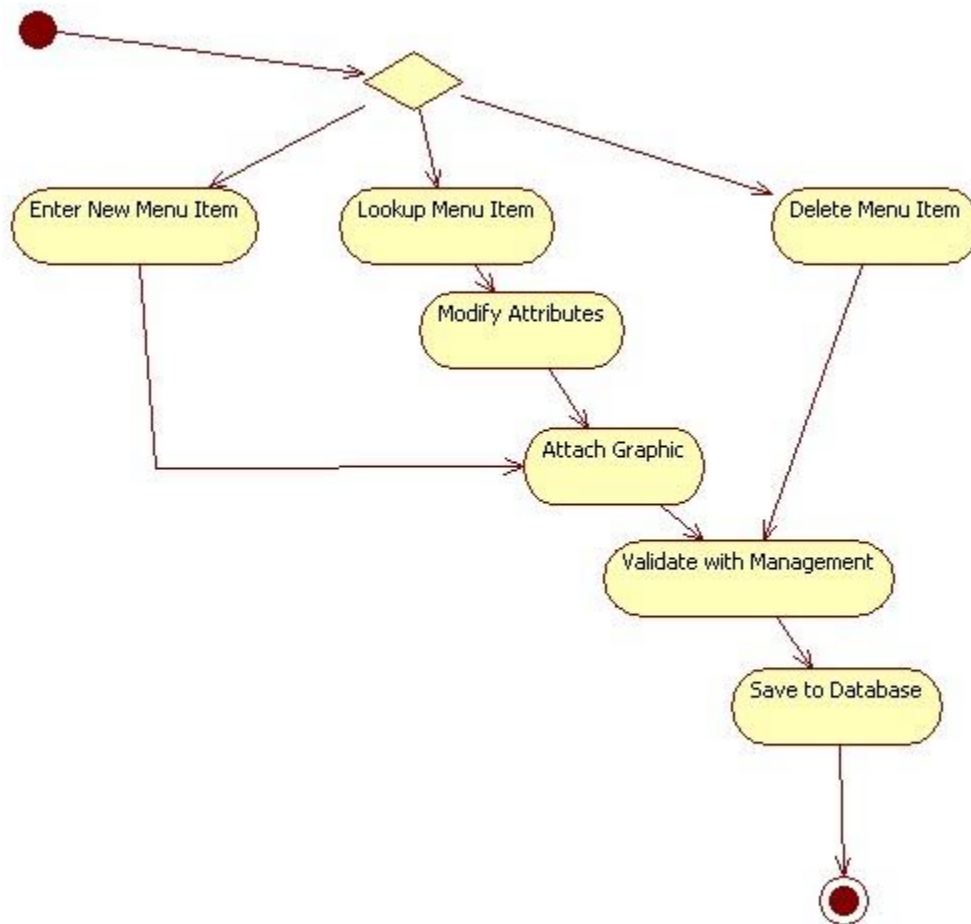


Figure 15. Menu Management Activity Diagram

## Inventory Management

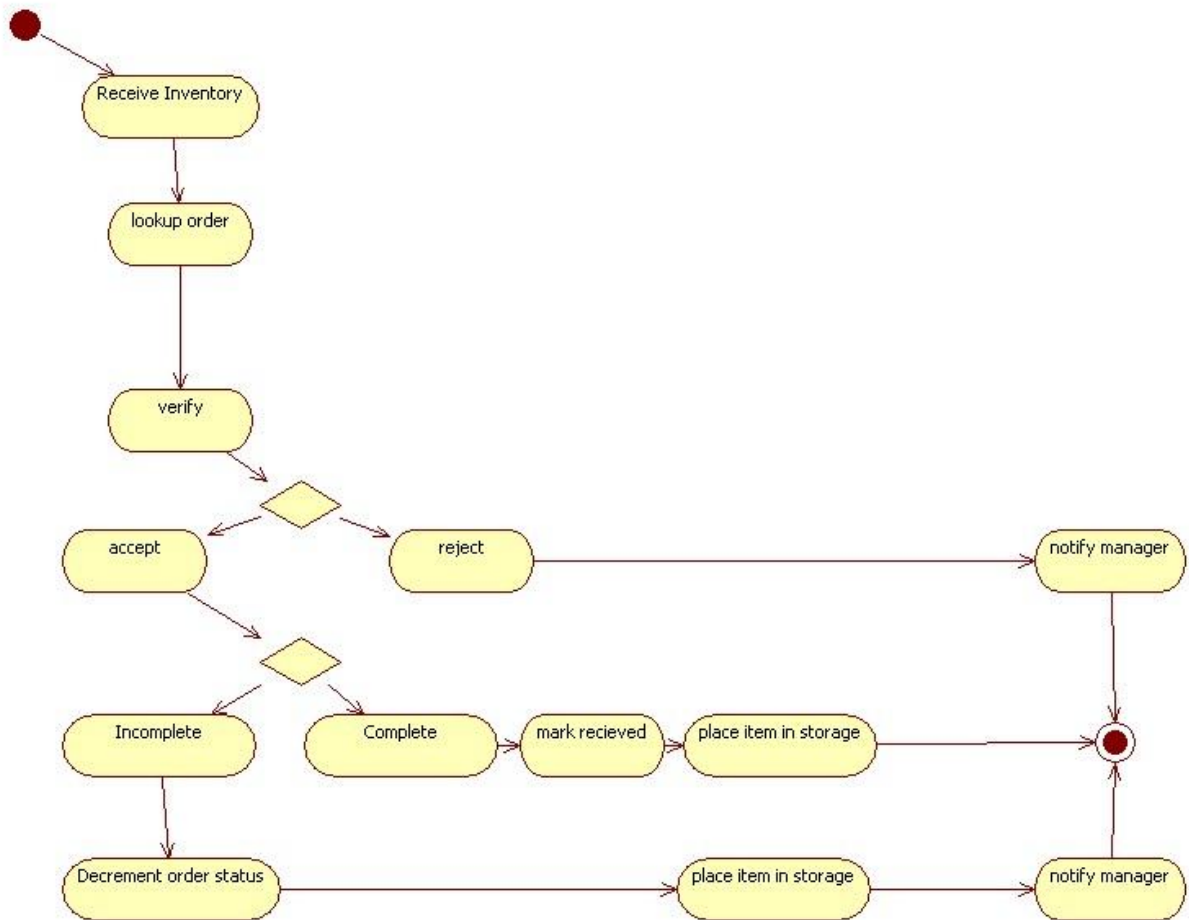


Figure 16. Inventory Management Activity Diagram

## Employee Management Activity Diagram

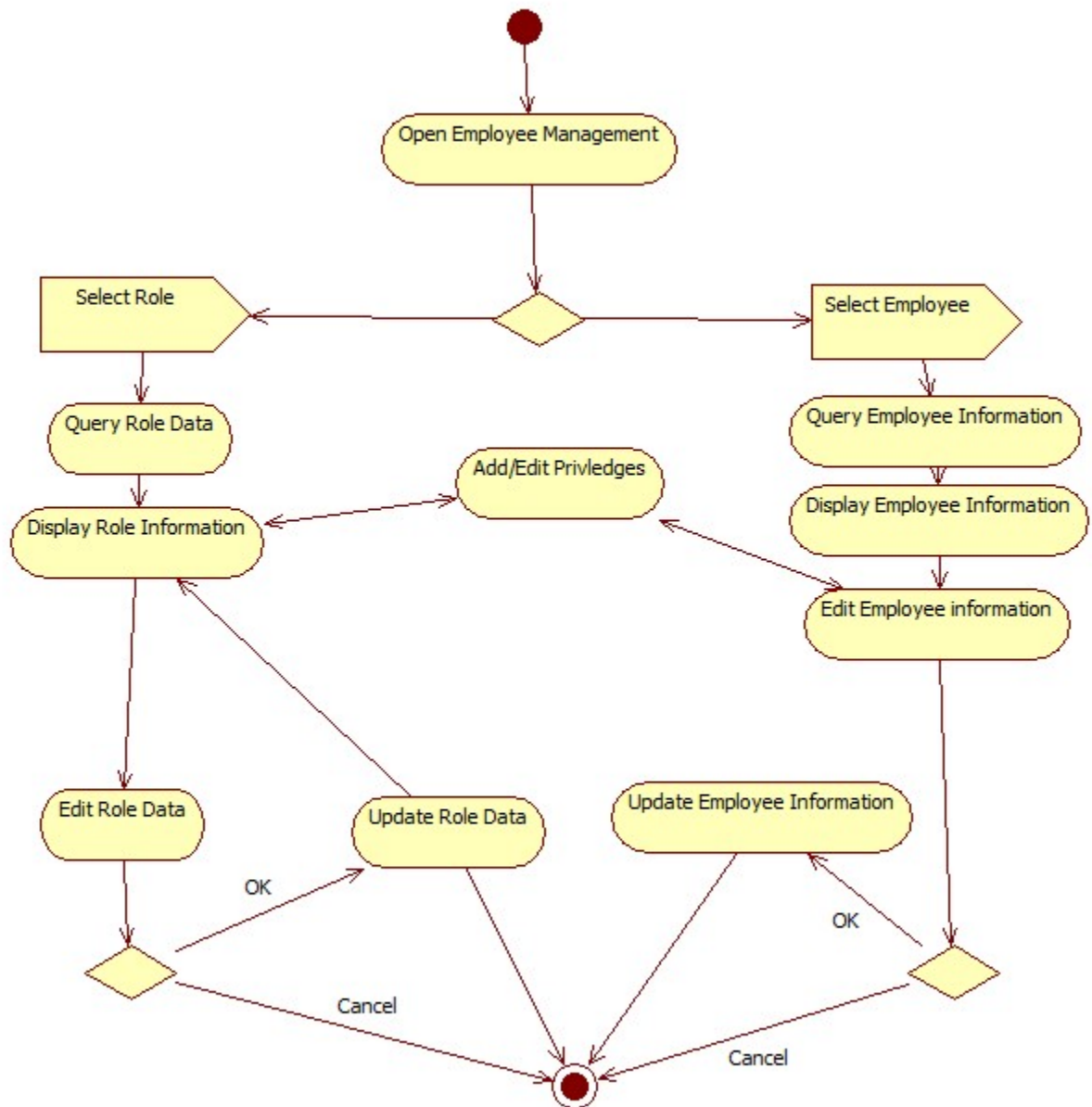


Figure 17. Employee Management Activity Diagram



## Physical View

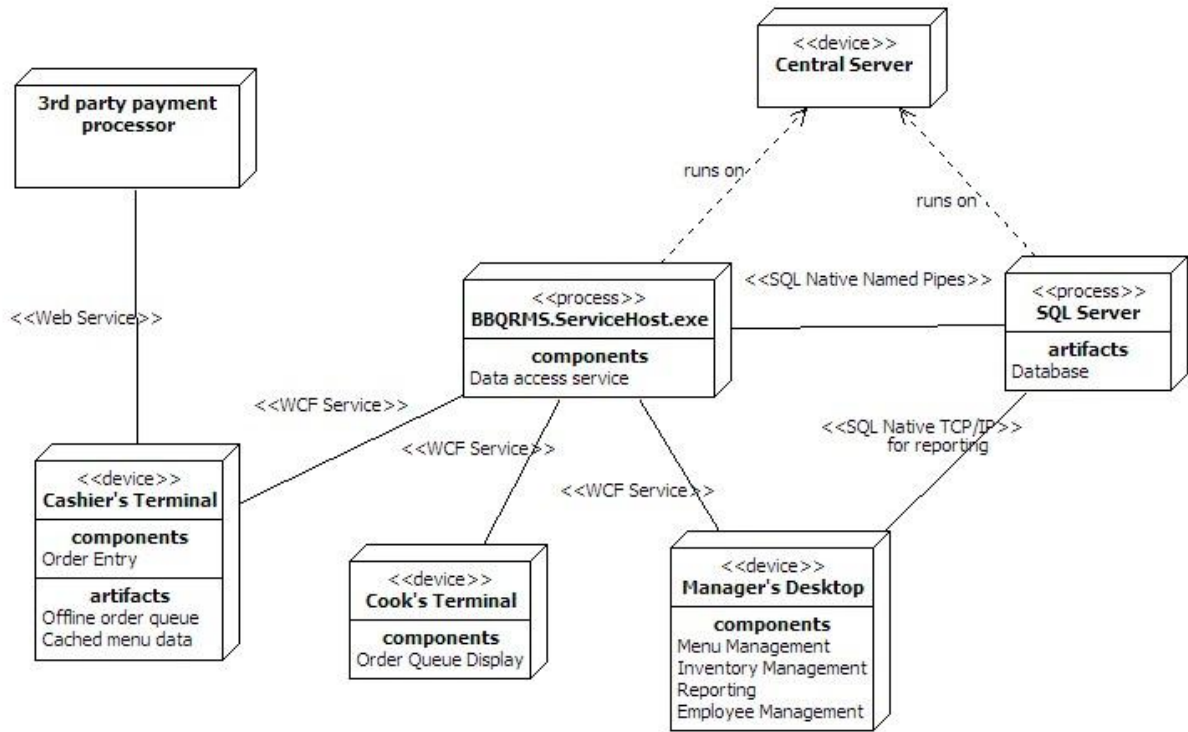


Figure 18. Deployment diagram

A full deployment of the RMS will include:

- one central server machine running the SQL Server database service and the service hosting the BBQRMS data access services

- one or more cashier's terminals where the BBQRMS client program can be run to use the Order Entry component.

- one or more cook's terminals where the BBQRMS client program is run to use the order queue display component.

- one or more manager's terminals where the BBQRMS client program is run to use the management and administrative components.

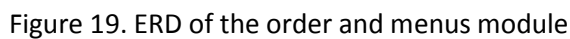
The cashier's terminal will also include local data storage to be used when a connection to the central server is unavailable.

Communication between processes, (except the connection to the database), will utilize WCF channels. This decision allows us to configure the details of the bindings later without impacting the system architecture.

A minimal deployment will also be supported where all the components and all three processes can run on a single machine.

## Using WCF Data Services for the data access layer

### Using continuous integration



## Inventory

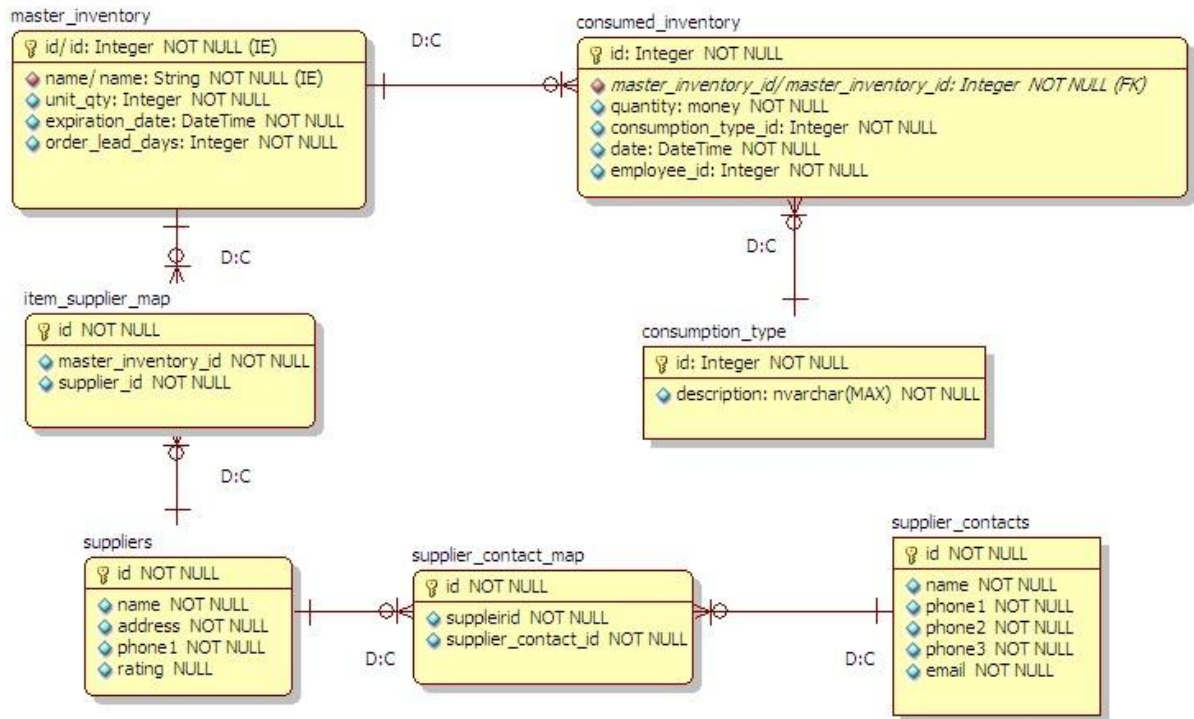


Figure 20. ERD of the inventory module

## Employees

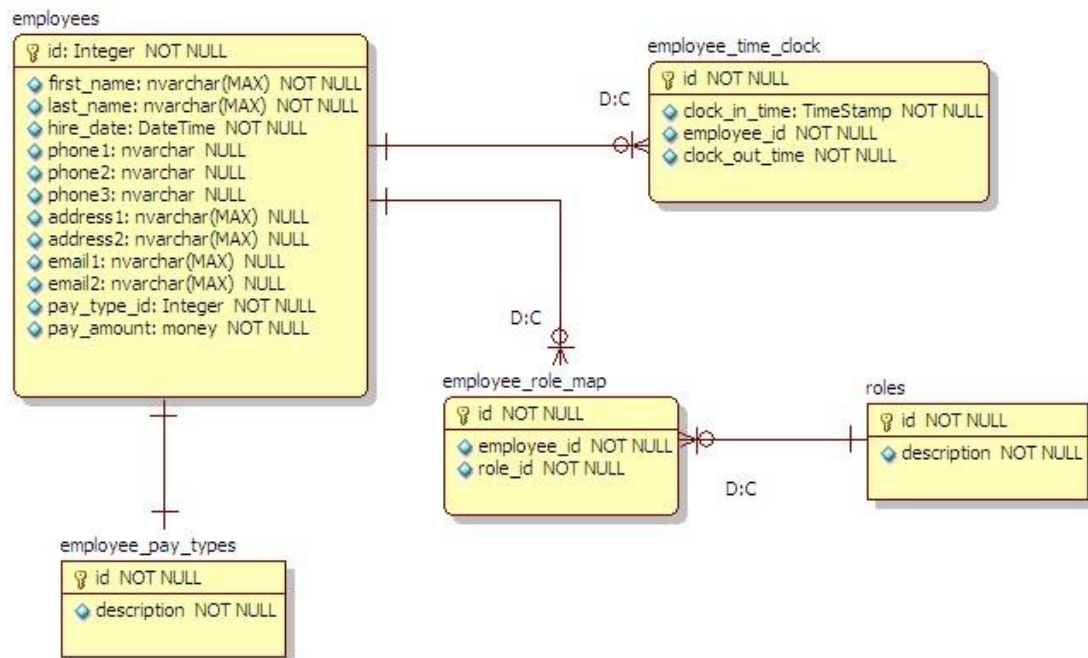


Figure 21. ERD of the employee management module

## Glossary

Term or Acronym	Definition
Alpha test	Limited release(s) to selected, outside testers
Beta test	Limited release(s) to cooperating customers wanting early access to developing systems
Final test	aka, Acceptance test, release of full functionality to customer for approval
DFD	Data Flow Diagram
SDD	Software Design Document, aka SDS, Software Design Specification
SRS	Software Requirements Specification
SSRS	System and Software Requirements Specification
RMS	Restaurant Management System
POS	Point of Sale
ERD	Entity Relationship Diagram
WPF	Windows Presentation Foundation
WCF	Windows Communication Foundation
RDBMS	Relational Database Management System
GSC-SVN	Google Source Control - subversion
WBS	Work Break-down Structure (MS project file)
Use case	A diagram and description of a certain function of the system.
GUI	Graphical user interface
MOP	Method of Payment

## **Bibliography**

---

Foundations of Programming Building Better Software By Karl Seguin [www.codebetter.com](http://www.codebetter.com)

Model-View-ViewModel Pattern By Chistian Moser <http://www.wpftutorial.net/MVVM.html>

UML Design Tool: Star UML <http://staruml.sourceforge.net/en/>

fyiReporting RDL Project <http://www.fyireporting.com>

SQL Server <http://www.microsoft.com/sqlserver/2008/en/us/>

Window Communication Foundation (WCF) <http://msdn.microsoft.com/en-us/netframework/aa663324.aspx>

Entity Framework <http://msdn.microsoft.com/en-us/library/aa697427%28VS.80%29.aspx>

Windows Presentation Foundation (WPF) <http://msdn.microsoft.com/en-us/library/ms754130.aspx>

## History of Work

	Task Name	Status	Duration	Start	Finish	Pre	Resource Names
1	Iteration 1 (1/3-1/19)	Late	13 days	Mon 1/3/11	Wed 1/19/11		
2	Revise Project Plan	Complete	6 days	Mon 1/3/11	Mon 1/10/11		Jerod, Scott, Mike, Jessie
3	Revise SSRS	Complete	2 days	Tue 1/11/11	Wed 1/12/11	2	Jerod, Scott, Mike, Jessie
4	Revised SSRS Due	Late	0 days	Wed 1/19/11	Wed 1/19/11	2,3	
5	BBQRMS-47 Build Initial Database structure	Complete	5 days	Thu 1/13/11	Wed 1/19/11		Jerod
6	BBQRMS-42 Shell - Main application navigation	Late	2 days	Thu 1/13/11	Fri 1/14/11		Mike
7	BBQRMS-44 Shell - Setup database server and communication	Complete	1 day	Mon 1/17/11	Mon 1/17/11		Mike
8	BBQRMS-22 Security - Users can log in	Late	0.5 days	Tue 1/18/11	Tue 1/18/11	6,7	Mike
9	BBQRMS-23 Security - Users can log out	Late	0.5 days	Tue 1/18/11	Tue 1/18/11	8	Mike
10	BBQRMS-18 Inventory Management - Receive an inventory order	Late	5 days	Thu 1/13/11	Wed 1/19/11		Jessie
11	BBQRMS-45 Order Entry - Design User Interface for Placing Custo	Complete	5 days	Thu 1/13/11	Wed 1/19/11		Scott
12	Graded Checkpoint 1	Late	0 days	Wed 1/19/11	Wed 1/19/11		
13	Iteration 2 (1/20-1/31)	Late	8 days	Thu 1/20/11	Mon 1/31/11		
14	BBQRMS-24 Security - Program features are limited by assigned u	Future Task	2 days	Fri 1/28/11	Mon 1/31/11		Mike
15	BBQRMS-43 Reports - Integrated reporting tool	On Schedule	6 days	Thu 1/20/11	Thu 1/27/11		Mike
16	BBQRMS-33 Order Entry - Place and save customer order without	On Schedule	4 days	Thu 1/20/11	Tue 1/25/11	11	Scott
17	BBQRMS-37 Order Entry - Recall and modify saved customer orde	Future Task	4 days	Wed 1/26/11	Mon 1/31/11	11	Scott
18	BBQRMS-17 Inventory Management - Add Supplier	On Schedule	4 days	Thu 1/20/11	Tue 1/25/11		Jessie
19	BBQRMS-57 Inventory Management - Place an Inventory Order	Future Task	4 days	Wed 1/26/11	Mon 1/31/11	18	Jessie
20	BBQRMS-39 admin - Add/Modify Employees	Late	3 days	Thu 1/20/11	Mon 1/24/11		Jerod
21	BBQRMS-40 admin - Add/Modify Roles and Privs	Future Task	5 days	Tue 1/25/11	Mon 1/31/11	20	Jerod
22	Graded Progress Check 2	Future Task	0 days	Mon 1/31/11	Mon 1/31/11		
23	Iteration 3 (2/1-2/14)	Future Task	10 days	Tue 2/1/11	Mon 2/14/11		
24	BBQRMS-36 Cashiering - Accept payment for saved customer ord	Future Task	5 days	Tue 2/1/11	Mon 2/7/11	11	Scott
25	BBQRMS-35 Order Entry - Take a customer order and accept payr	Future Task	5 days	Tue 2/8/11	Mon 2/14/11	24	Scott
26	BBQRMS-19 Inventory Management - Reconcile Inventory	Future Task	10 days	Tue 2/1/11	Mon 2/14/11	10	Jessie
27	BBQRMS-30 Security - Users can clock in and clock out	Future Task	4 days	Tue 2/1/11	Fri 2/4/11	8,9	Mike
28	BBQRMS-25 Security - Users can change their PIN	Future Task	2 days	Mon 2/7/11	Tue 2/8/11	27	Mike
29	BBQRMS-28 Reports - Daily Sales Report	Future Task	2 days	Wed 2/9/11	Thu 2/10/11	15,	Mike
30	BBQRMS-56 Menu Management - Create/Edit/Delete Menu	Future Task	4 days	Tue 2/1/11	Fri 2/4/11		Jerod
31	BBQRMS-31 Menu Management - Create/Edit/Delete Menu Item	Future Task	5 days	Mon 2/7/11	Fri 2/11/11	30	Jerod
32	Iteration 4 (2/15-2/28)	Future Task	11 days?	Tue 2/15/11	Tue 3/1/11		
33	BBQRMS-48 Cashiering - Print Receipt	Future Task	5 days	Tue 2/15/11	Mon 2/21/11		Mike
34	BBQRMS-2 Reports - Daily Shopping List	Future Task	3 days	Tue 2/22/11	Thu 2/24/11	33,	Mike
35	BBQRMS-32 Cook Screen - Complete a Customer Order	Future Task	5 days	Tue 2/15/11	Mon 2/21/11		Scott
36	BBQRMS-34 Cashiering - Split a Check	Future Task	5 days	Tue 2/22/11	Mon 2/28/11	24,	Scott
37	BBQRMS-16 Inventory Management - Touch-screen	Future Task	1 day?	Tue 2/15/11	Tue 2/15/11	26	Jessie
38	BBQRMS-50 Dashboard - Display managers dashboard	Future Task	1 day	Mon 2/28/11	Mon 2/28/11		Jerod
39	BBQRMS-51 Dashboard - Display whose on the clock	Future Task	1 day	Tue 3/1/11	Tue 3/1/11	38	Jerod
40	Test Plan	Future Task	1 day	Wed 2/16/11	Wed 2/16/11		Jerod, Jessie
41	System testing	Future Task	5 days	Thu 2/17/11	Wed 2/23/11	40	Jerod, Jessie
42	Testing Results	Future Task	0 days	Wed 2/23/11	Wed 2/23/11	41	
43	Graded Checkpoint 3	Future Task	0 days	Mon 2/28/11	Mon 2/28/11		
44	Iteration 5 (3/1-3/14)	Future Task	10 days	Tue 3/1/11	Mon 3/14/11		
45	BBQRMS-52 Cashiering - Open Cash drawer	Future Task	5 days	Tue 3/1/11	Mon 3/7/11		Mike
46	BBQRMS-29 Reports - Inventory turns	Future Task	2 days	Tue 3/8/11	Wed 3/9/11	15,	Mike
47	BBQRMS-27 Reports - Weekly prime cost report	Future Task	1 day	Thu 3/10/11	Thu 3/10/11	15,	Mike
48	BBQRMS-38 Cashiering - Take multiple methods of payment for a c	Future Task	5 days	Tue 3/1/11	Mon 3/7/11		Scott
49	BBQRMS-53 Cashiering - Process Credit Card	Future Task	5 days	Tue 3/8/11	Mon 3/14/11	48	Scott
50	BBQRMS-5 Update UI for user management to be touch friendly	Future Task	4 days	Tue 3/1/11	Fri 3/4/11		Jerod
51	BBQRMS-54 Dashboard - Display estimated daily profit.	Future Task	3 days	Mon 3/7/11	Wed 3/9/11		Jerod
52	BBQRMS-55 Dashboard - Display estimated cost of goods	Future Task	3 days	Thu 3/10/11	Mon 3/14/11		Jerod
53	System testing	Future Task	10 days	Tue 3/1/11	Mon 3/14/11		Jessie
54	Iteration 6 (3/15-4/4)	Future Task	15 days	Tue 3/15/11	Mon 4/4/11		
55	Beta testing	Future Task	7 days	Tue 3/15/11	Wed 3/23/11		Jerod, Scott, Mike, Jessie
56	Beta Test Results	Future Task	0 days	Wed 3/23/11	Wed 3/23/11	55	
57	Bug Fixing	Future Task	7 days	Thu 3/24/11	Fri 4/1/11		Scott, Mike
58	System Testing	Future Task	7 days	Thu 3/24/11	Fri 4/1/11		Jerod, Jessie
59	Graded Checkpoint 4	Future Task	0 days	Mon 4/4/11	Mon 4/4/11		
60	Iteration 7 (4/5-4/18)	Future Task	10 days	Tue 4/5/11	Mon 4/18/11		
61	Burn in	Future Task	10 days	Tue 4/5/11	Mon 4/18/11		Jerod, Scott, Mike, Jessie
62	Final Delivery to Instructor	Future Task	0 days	Wed 4/13/11	Wed 4/13/11		
63	Project Go Live	Future Task	0 days	Mon 4/18/11	Mon 4/18/11		



## Gantt Chart

