DATA MART FOR CSUS HORNET BOOKSTORE

Pooja Ramesh
B.E., Visveswaraiah Technological University, Karnataka, India, 2008

PROJECT

Submitted in partial satisfaction of
the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

at

CALIFORNIA STATE UNIVERSITY, SACRAMENTO

SPRING
2011

DATA MART FOR CSUS HORNET BOOKSTORE

A Project

by

Pooja Ramesh

Approved by:

_____, Committee Chair
Dr. Meiliu Lu

_____, Second Reader
Dr. Mary Jane Lee

_____
Date

Student: Pooja Ramesh

I certify that this student has met the requirements for format contained in the University format manual, and that this project is suitable for shelving in the Library and credit is to be awarded for the Project.

_____, Graduate Coordinator _____
Dr. Nikrouz Faroughi Date

Department of Computer Science

Abstract

of

DATA MART FOR CSUS HORNET BOOKSTORE

by

Pooja Ramesh

Data mart is the subset of an organizational data warehouse which is designed to focus on a single subject such as Sales, Finance or Marketing. Data marts draw data only from few sources based on the single subject focus. Data marts are smaller and less complex when compared to Data warehouse. Hence, it is easier to build and maintain a data mart.

The advantage of using data mart is that the users gain faster access to common data by having little or no knowledge about data mart and obtains data as required. It improves performance and end users response time.

The main objective of this project is to develop a data mart prototype for the CSUS Hornet bookstore. The purpose of creating this prototype is to help the Hornet bookstore to retrieve the data in an efficient manner based on inventory and sales of the bookstore. The top two priority searches are:

- To find the warranty of computers purchased by the customers in order to find out whether the computers can be repaired at no cost or at partial expense.

- To find sales rate of different brand laptops for each semester and then decide on ordering more number of particular brand laptops for next semester based on the laptops which was sold out in more number in the previous semester.

In this project, only the top two search priorities are considered. Currently the bookstore keeps track of all the items in an excel sheet where in a user has to manually search for the correct data by scrolling through different excel sheet and find the needed data. Therefore, this data mart will help the Hornet bookstore computer department to retrieve the required data in a simpler manner by providing the data based on the query they are running to obtain information about specific items and as well as sales on certain dates.

This project not only helps the hornet bookstore but in turn helps the students who get enrolled in CSC177 coursework because it provides a practical example while learning data warehousing and data mart concepts.

To achieve this objective we will be creating a star schema data mart design, data preprocessing and use MY SQL database in creating a data mart

_____, Committee Chair
Dr. Meiliu Lu

_____
Date

ACKNOWLEDGMENTS

I owe my sincere and heartfelt thanks to my project advisor, Dr. Meiliu Lu for giving me an opportunity to work on my Master's project under her guidance and providing her invaluable support throughout the project. Her suggestions and ideas were of great help in getting this project into reality.

I would like to express my deepest gratitude to the CSUS Hornet Bookstore, Computer department for their enormous support; encouragement and guidance in making this project a real success. I would like to thank them for their patience and time in attending several demos which was carried out throughout this project.

My sincere thanks to Professor Mary Jane Lee for being my second reader and providing detailed remarks in reviewing my project report. I also thank Dr. Nikrouz Faroughi for his ideas and support during the course of the project.

My warm and sincere thanks to my friend Shashi Kumar Nanjaiah for being a great mentor in completion of my Master's project. He was my guide, supporter, critic and teacher throughout the journey of my project. You have guided me from silly to the most complicated tasks; I have learnt a lot from you and shall always have my deepest respect to you Catty.

My special thanks to my friends Mayur Anand, Pramukh Jadhav and Adithya Shreyas for providing their valuable inputs in reviewing my project report, support and correcting my work throughout my Master's program. I would also like to thank all my other friends who have always been throughout my Master's program.

Last but not the least; I owe my deepest and heartfelt thanks to my Family for their unconditional love and support throughout my life. The sole reason for what I am today is my family. I am grateful to my Father Mandya Boraiah Ramesh and Mother N.Vijayalakshmi for providing me financial and mental support to come over seas and get my Master's degree. I am happy in fulfilling mine and my parent's dream of accomplishing my Master's degree. I am very happy to have my Brother R. Pavan Kumar for his irreplaceable love and support from the day I have born. He is always besides me for any help and day. Having such a lovely and caring family I have always projected me in a wrong way. I would therefore like to tell them that I am blessed to have such a family and I LOVE YOU Appa, Amma and Anna.

I would also like to thank our family friend Mahesha and others who are all responsible for providing financial support and encouragement during the journey of my Master's program.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

Chapter 1

INTRODUCTION

1.1 Data Mart

A Data Warehouse is storage of raw or formatted data which is used for analysis, archival and security purposes. It plays a very important role in a Decision Support System. [1] Data Warehouse is now becoming an important part of any business enterprise. It helps an enterprise analyze its current situation and predict the future. Organizations that do Financial Forecasting, Decision support, Logistics and Inventory Management applications uses data warehouse to learn various topics including product profitability, customer demographics, and external data to provide analysis and so on. The primary objective of a data warehouse is to maintain and analyze historical data.

Data Mart is a repository of data which is formed by collecting data from operational data or any other sources that satisfy the demands of a particular group or community. Data marts are the subsets of an organizational data warehouse which draws data only from few sources based on the single subject focus such as Sales, Finance or Marketing. Majority of business related analytical activities takes place in a data mart. Each data mart is tailored for a particular function or activity. The organization of data in a data mart follows a simple schema structure called Star schema. The data which is being analyzed resides at the star's center, and around the center, resides other data that describe the dimensions. For example, facts constitute the sales of a product in an organization and dimensions holds the analysis of product, customer, and date of

purchase and so on. . Data marts are smaller and less complex when compared to Data warehouse. For example, a data warehouse of an IT organization contains all the data related to their business and the data is collected in such a way that it can be later used. Whereas, designing a Data Mart is based on the analysis of the needs of the user. Therefore the IT organization possessing a data warehouse can also have a data mart based on their needs which includes but not limited to Finance, marketing, sales and so on [2].

The advantage of using data mart is that the users gain faster access to common data by having little or no knowledge about data mart and obtains data as required. It improves performance and end users response time.

The steps involved in implementing a data mart are as follows

- Design – This is the first step in the data mart process which involves in gathering business and technical requirements, identifying data sources, selecting appropriate subset of data and finally designing logical and physical structure of the data mart.
- Construction – This step involves in creating physical database and storage structures which is associated with the data mart. Designing schema objects such as tables and indexes and finally determining the best way to set up the tables and access structures.
- Populating the data - In this step we perform all the task in getting the data from the source which involves mapping data sources to target data structures,

cleansing and transforming the data, extracting data, loading data into data mart creating and storing metadata.

- Accessing – This step involves putting the data in use i.e. querying the data, creating reports, charts, and graphs and then publishing these results.
- Managing – This step involves in managing the data mart over the lifetime by providing secure access to the data, managing the growth of the data and optimizing the data for better performance.

In this project we will be developing an application that will be used by CSUS Hornet Bookstore to retrieve data on inventory and sales of the bookstore. To achieve this, we have developed an expressive web application that can be displayed consistently on all major web browsers. PHP with Zend framework is used in the middle tier and MYSQL in the data tier. To ease the development of cross-platform and JavaScript/Ajax-based application JQuery (JavaScript library) is used.

1.2 Motivation

There are three main reasons behind developing a data mart prototype system for the CSUS Hornet bookstore, Computer department. They are:

1. It helps Hornet bookstore computer department to leave behind their manual approach in gathering sales and customer information and retrieve the required data in a simpler manner through the data mart.

2. This project will be set as a practical example for the students who are learning data mart concepts.

3. Lastly and most importantly, this project served as a learning experience during my Master's project as I was newly introduced to the concept of Data mart, Zend framework tool and PHP programming language.

1.3 Research Road

A good design always begins with a good design concept that tries to solve a problem. The concept will lead the way to give a direction for your design decisions. But this concept was introduced to me for the first time so the biggest question was how do you form a concept? What questions need to be asked in order to develop one? How does your concept become the roadmap for your design?

Not knowing anything about Data mart, it took me a while to understand the concept and get the basics right. Initially, I found it difficult to gather information about the data mart because I had not taken any course related to data mart during my under graduate or in my Master's program. During my research I noticed that not many people had worked on this topic, as a result gathering information and understanding the concept was not an easy task.

It was my first time that I was working on Zend framework. The main challenge that I faced was configuration of the tool. On the other hand the tool is user friendly, easy to understand and can be used by any novice. Another roadblock related to Zend framework tool is the licensing issue as it has only thirty day free trial.

The third roadblock that I faced was using PHP as the programming language in designing my application. I had no experience in using this language. As a result it took

me some time in understanding the concepts of PHP and to get started in building the application.

1.4 Project Report Organization

This section provides a brief guideline and description of topics in each chapter of the project report. In chapter 2, we will have a detailed explanation of Data mart and tools used which includes description on Zend framework, its functionalities and features; description on programming language PHP and MYSQL. Chapter 3 provides us detailed user requirements analysis. The Design and implementation of data mart is covered under chapter 4 and chapter 5 respectively. Finally, in chapter 5 we summarize the report by providing the test results of the application.

Chapter 2

BACKGROUND AND TECHNOLOGY

2.1 Introduction

The data mart application features reusable components across the enterprise using an n-tier model. This provides the ability, in the future, to separate components onto different physical machines, thereby, enabling scalability and reliability when enhancements are made to the application. Centralized business rules support low-risk development and simple ongoing maintenance. Centralized data access allows database independency. Necessary data is collected to support operational, strategic decision-making, and reporting needs for the effective administration of the Data mart application.

2.2 PHP

PHP is a server side scripting language. It is a powerful tool for making dynamic and interactive Web pages. The goal of PHP is to enable a quicker web development process. PHP allows you to use the existing HTML content while you add the PHP code to it. PHP runs on almost any web server, on any operating system. One of the other strong points about PHP is its support for a wide variety of databases.

PHP was created by Rasmus Lerdof in 1995 for tracking accesses to his online resume [3]. It was originally developed in Perl scripts and was initially known as “Personal Home Page” tools. It was later rewritten in C with database functionalities. Few amendments followed in 1998, 2000 and 2004 which eventually added a bundle with

Zend Engine with object oriented programming, robust XML, SOAP extension for interoperability with web services and SQLite packages. PHP now stands for PHP Hypertext Preprocessor.

Few other advantages of PHP are listed below:

- Faster developments of large websites
- Customized user experience would be provided to the users/visitors based on the information provided by the users
- Makes creation of shopping carts for e-commerce websites easy

Since the syntax of PHP is very similar to C, JAVA or PERL with little PHP specific additions, the pre-requisites to using PHP are basic programming knowledge and knowledge of HTML syntax and forms.

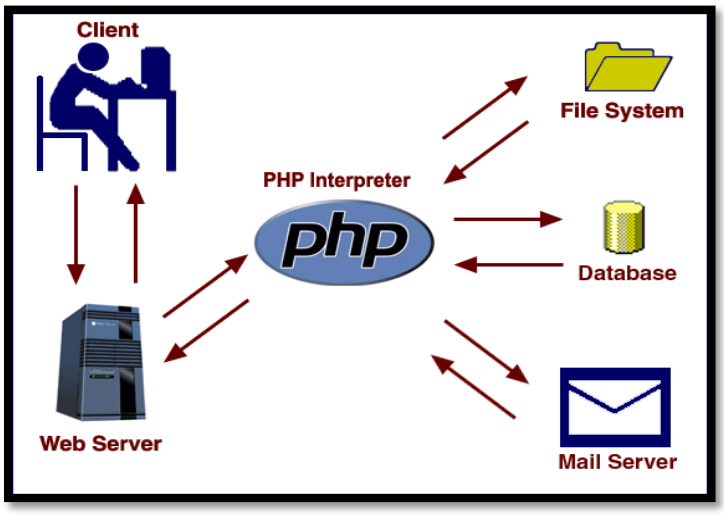

Figure 1: Working of PHP

Figure 1[3] shows the working of PHP. When a user requests for a web service from a PHP page from a web browser, a request is sent to the web server and the server in turn initiates the PHP interpreter to take the control. Then the PHP interpreter communicates to the file systems, databases and email servers and does the required action and then delivers the webpage to the web server, to return to the browser. The interpreter uses a configuration file to determine the setting that needs to be used which is usually defined in a “php.ini” file [3].

PHP along with MySQL provides a very powerful solution to collect the data from the users, create specific content on the fly and perform other operations like printing data, making comparison, performing numeric calculations which is not possible with HTML alone.

2.3 MySQL

MySQL is an open source, enterprise-level, multi-threaded, relational database management system that stores and retrieves data using the Structured Query Language [SQL]. The MySQL is currently owned by Oracle Corporation, but the source code is made available for free under the terms of GNU General Public License [GPL] with proprietary agreements.

Relational Database is one that stores data in the form of relational tables unlike the flat files. A Structured Query Language [SQL] is a standardized query language for getting information from a relational database. A Database Management System [DBMS] is a system that manages relational databases. It can also be referred to a collection of

programs that enables storage, modification and extraction of information from a database.

The advantages of MySQL are high performance, high reliability and ease of use. Although MySQL is accessible to many programming languages, it is very popular with PHP as they couple together very easily. PHP can collect the data from the user and MySQL can store this information. PHP creates the dynamic calculations and the required variables are provided by MySQL. Similar numerous possibilities could be added to an application when PHP and MySQL are used together.

2.4 ZEND Framework

At the heart of the design is the Zend Framework (ZF). ZF is an open source framework for developing web applications and services with PHP. ZF is implemented using 100% object-oriented code. The component structure of ZF is somewhat unique; each component is designed with few dependencies on other components. This loosely coupled architecture allows our development team to use components individually. It is also consistence with one of our key architectural principles.

ZF components in the standard library form a powerful and extensible web application framework when combined. ZF offers a robust and performance oriented Mode; View Controller (MVC) implementation, business logic, and database abstraction that is simple to use, and a forms component that implements HTML form rendering, validation, and filtering so that our developers can consolidate all of these operations using one easy-to-use, object oriented interface. Other components, such as Zend_Auth (Security

Authentication) and Zend_Acl (Security Authorization), provide user authentication and authorization against all common credential stores.

This framework is suitable for a major project like this. One of the easy-to-use scripting languages currently is PHP. A unique framework to compliment PHP is ZEND framework. ZEND Framework is an open source framework used to develop applications using PHP v5.0 and above. ZEND Framework's implementation uses 100% object-oriented code.

Few advantages of ZEND Framework are:

- Robust
- High performance MVC implementation
- Provides database abstraction which is simple to use
- Provides forms component that implements HTML form rendering, validation, and filtering

The Zend Framework makes writing web applications easier, simpler, and faster. It provides developers with a body of source code that has already been developed by dozens of developers which are unit tested. This allows developers to skip developing similar applications and focus more on the other functionalities of their application. One way to choose a framework for web development is looking at whether the framework makes web development easier, faster and simpler. The other factors that must be considered are maintainability, adaptability, ease of testing, technical and community

support, quality of documentation, regularity of upgrades and fixes, quality control, base performance, the learning curve, features and innovation, hosting availability, support for current best practices, and community sources literature [4]. Like other frameworks Zend framework has also adopted the Model-View-Controller [MVC] architecture.

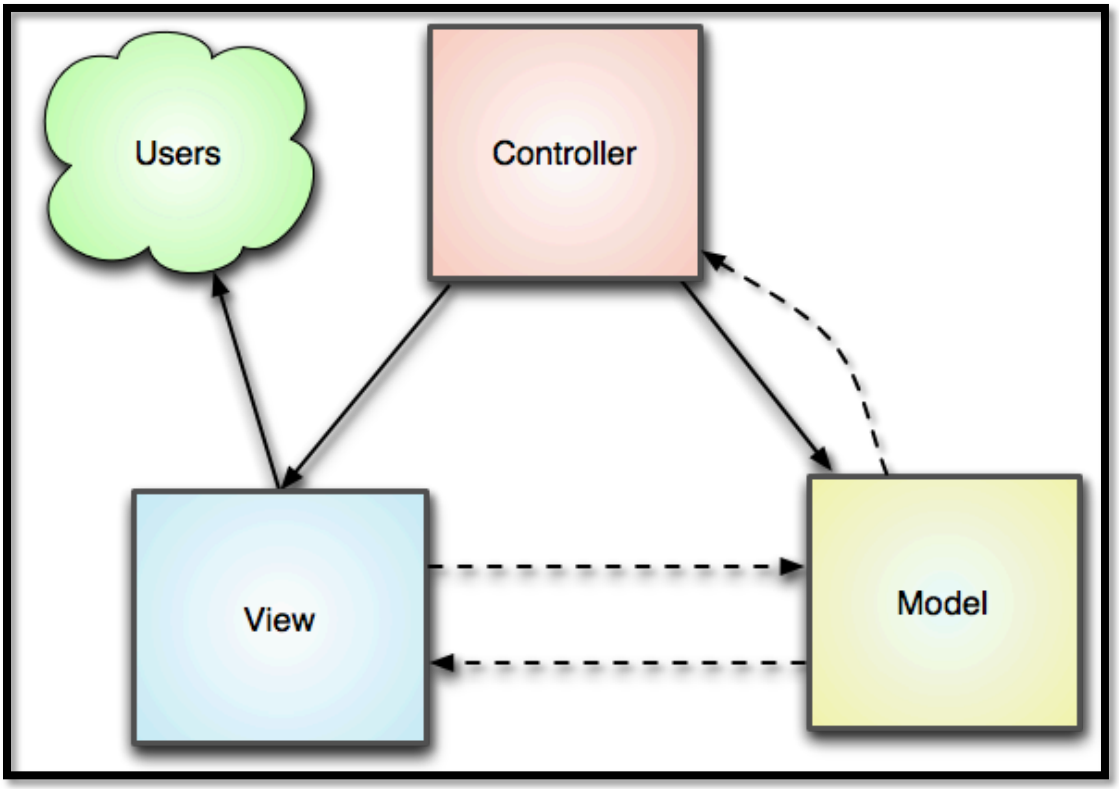

Figure 2: MVC Architecture

The Model-View-Controller [See Figure 2 [5]] as the name indicates has three component separations which are:

Model: A Model is responsible for maintaining the state between each HTTP request. Model incorporates all the rules, restraints and behavior governing and utilizing this

information. An example for this could be in an online banking application, any transactions/purchases above $2000 could be defined to be illegal and the transaction is forced to abort. In essence, a Model centers on the data and its underlying behavior.

View: The View is responsible for generating a user interface for your application. In PHP it is usually defined as the presentational HTML, RSS, and XML, JSON or anything else that is sent to the client application through a browser or requesting web service. The View is organized into template files which can also be echoed from, or manipulated by the Controller prior to its output. The view also contains the PHP code to organize, filter and manipulate the format based on the data retrieved from one or more Models.

Controller: The primary function of the controller is to control and delegate. In MVC architecture for a web application, the controller takes the input from the user, translates it into actions on one or more Models and returns the output generated by the View to the user.

2.4.1 ZEND Configuration

```
Listen 10081
NameVirtualHost *:10081
<VirtualHost *:10081>
    Alias /ZendServer "C:\Program Files (x86)\Zend\ZendServer\GUI\html"
    DocumentRoot "C:\Program Files (x86)\Zend\ZendServer\GUI\html"
    RewriteEngine On
    RewriteRule ^/$ /ZendServer/ [R]
    RewriteRule ^/Login$ /ZendServer/Login [R]
    <Directory "C:\Program Files (x86)\Zend\ZendServer\GUI\html">
```

AllowOverride All

Allow from all

</Directory>

</VirtualHost>

To enclose a group of directives which includes the named directory, sub – directories of that directory and all the files present within these directories, `<Directory>` and `</Directory>` tags are used. In a directory context if any directive is allowed, then those directives can be used. The *Directory-path will either be* a full path to a directory or a wild card string that is written using Unix shell-style matching. In a wild-card string, '?' is used to match any single character, `*` to match any sequences of characters and `[]is used for` character ranges. But none of the wild cards match with the '/' backslash character. Therefore, `<Directory /*/public_html>` will not match /home/user/public_html, but matches <Directory /home/*/public_html>. [6]

Server needs to know the directives declared in a file when it finds an .httpd file that is specified by AccessFileName in order to know whether that file can override directives that were configured earlier. [6]

AllowOverride is specified without any regular expressions. It is valid only in <Directory> sections and not in <Location>, <DirectoryMatch> or <Files> sections.

The .httpd files are completely ignored when this directive is set to None. As a result, the server will not attempt to read .httpd files present in the filesystem.

Whereas, when the same directive is set to All, then any directive which contains the .httpd Context is allowed in .httpd files.

The RewriteRule is simple and can be analyzed as “for any url, redirect to index.php” [6]. Now that we have introduced the tools and technology required in designing the Data Mart application for Hornet Bookstore of CSUS, in the next chapter, we will go into the detailed User requirements and analysis for our application.

Chapter 3

USER REQUIREMENT AND ANALYSIS

3.1 Introduction

The purpose of User Requirements Analysis phase is detailed understanding of the business need and break down into discrete requirements, which are then clearly defined, reviewed and agreed upon with the Subject Matter Experts/Business partners/Decision-Makers. Requirements Analysis phase is such an important phase because, the framework for the application is developed, providing the foundation for all future design and development efforts.

Data mart application project consisted of project start-up activities where we worked closely with the staff at the Computer Department of the Hornet Bookstore, at the California State University Sacramento, to set up the project infrastructure, tools, Project Management processes and techniques. Additionally, a work plan was developed that included a detailed work breakdown structure; schedule of activities; milestones; tasks; task dependencies; and review/acceptance criteria.

Interviews were conducted with Hornet Bookstore staff to analyze the existing business and technical environment and then listed the business needs. The following groups were interviewed to assess organizational imperatives, business imperatives and directions, case management and fiscal needs, and technical standards and vision:

- Hornet Bookstore, Computer Department staff

- Engineering and Computer Science IT staff

The Computer Department at the Hornet bookstore, maintain all the specifications of laptops and information on customer purchase in a database. This database is created and maintained by the staff at the bookstore.

During our interview we noticed that the bookstore currently keeps record of each transaction that is processed in an excel spreadsheet document. As a result, the bookstore has to manually scroll through different excel spread sheets and then find the required data. For this reason, we came up with the idea of designing a data mart for the Hornet bookstore. This data mart will make life easier for a bookstore administrator to retrieve data in an efficient manner based on inventory and sales of the bookstore.

3.2 User Requirements

Functional requirement is a requirement which defines the functions of a system based on its input, behavior and outputs which is being developed. It is defined during the planning phase of the project [7].

The functional requirements in this project are listed based on the top priority searches performed by Hornet bookstore, they are:

1. To find the warranty of computers purchased by the customers in order to find out whether the computers can be repaired at no cost or at some expense.
    a. Designated library admin should be able to search warranty for particular customer purchase.

b. The system should display the full first and last names of each customer along with contact information such as Email and warranty information.

c. The system should provide options to select Brand name and Product name of customer laptop.

d. The system should provide options to select the year and term they purchased.

2. To find sales rate of different brand laptops for each semester and then decide on ordering more number of particular brand laptops for next semester based on the laptops which was sold out in more number in the previous semester.

    a. Designated library admin should be able to find Quantity sold for particular product.

    b. The system should provide the list of all Brand names and related Product names.

    c. The system should provide the option to select term and year to find quantity sold for the selected Brand name and Product name.

Apart from the top priority searches, the Hornet Bookstore suggested us to provide them a link which provides us the list of all the updated laptops by making necessary selection.

3.3 System Requirements

A non functional requirement is a requirement that specifies certain standards to evaluate the functions of the system. The implementation of non-functional requirement is detailed in system architecture which defines how a system is supposed to be [8].

System requirements of Data mart application is as follows:

- Data mart application must be built using open source tools and technology in order to be compatible with California State University, Sacramento server.
- Hornet Bookstore, Computer Science department of California State University, Sacramento must be able to access Data mart application through web.
- Hornet Bookstore must access Data mart application using standard web browsers and used across multiple Operating system platforms.
- Maintenance of the built Data mart application must be easy to handle by the Bookstore without much training, which includes knowledge transfer and reference manuals.
- The Data mart application must have a good response time.

3.4 Scope of the Project

After the requirement gathering and analysis phase the scope of this project was determined to design a data mart which helps the bookstore to retrieve data based on

inventory and sales information which in turn supports the needs of the bookstore. To achieve this task, we were forced to use open source tools and technology in order to be compatible on the CSUS server.

The following technologies were used in this project:

1. Zend framework: This is an open source web application framework which is user friendly, simple and corporate friendly licensing [4].

2. Zend Studio: This is an open source PHP Integrated Development Environment. It is integrated with Zend Framework enabling the application to build quickly [5].

3. PHP and JavaScript: Coding languages used.

4. MySQL: It is the most popular open source database and runs on more than 20 platforms including Linux, Windows, Mac OS, Solaris, HP-UX, IBM AIX, giving us the kind of flexibility you want to control.

Additional features added to this application are listed below:

- Link to find product information of all different brand laptops.
- Added Table sorter which is a JavaScript library for sorting the contents in the table.
- Set of instructions on How to perform tasks using the application.
- Matched the tables and columns of the data mart with respect to the database which is been currently used at the Hornet Bookstore, Computer department.

- Pagination added to display product information into discrete pages.

Chapter 4

SYSTEM DESIGN

This chapter provides a detailed design and summarizes the technical approach used to develop and implement the Data mart application. It also discusses the various tiers, their interactions and how they synchronize to form a complete system.

4.1 Conceptual View

The Data mart application provides a robust system meeting the Hornet Bookstore requirement. The key architectural principles include:

- An n-tier (Here n=3) compliant solution designed to CSUS hornet book store requirements;
- A web-based platform using a MySQL backend; and

The Data mart application is based on an n-tier model. The solution is designed using ZF, which provides the core underpinning of n-tier and object oriented design principles. We used processes and procedures to enforce compliance with enterprise standards throughout the entire Software Development Lifecycle. The following standards help maintain the n-tiered model during the life of the project:

- A front-end User Interface (UI) using Web pages
- Implementation of business rule component as a separate dedicated component

- A data access layer component as a separate Class Library project. This data layer uses classes to encapsulate access. Typed datasets are used; they provide the flexibility of the dataset class and strong typing for each column in the tables.

4.2 Logical View

The three distinct logical tiers:

- Presentation
- Business
- Data Access

These defined tiers help support maximum re-use capability and facilitate rapid deployment supporting future functional or technical platforms changes. Our proposed solution employs a modular, scalable, business oriented technical architecture that meets the requirements of the Hornet Bookstore, Computer Department of CSUS.

Our architecture is based on ZF, which implements a MVC (Model-View-Controller) design pattern that allows us to create well defined components with reduced interdependencies, enabling change. Figure 3 below provides a logical view of the solution's n-tier architecture, and shows that our architecture supports heterogeneity and renders itself to reuse of the existing hardware.

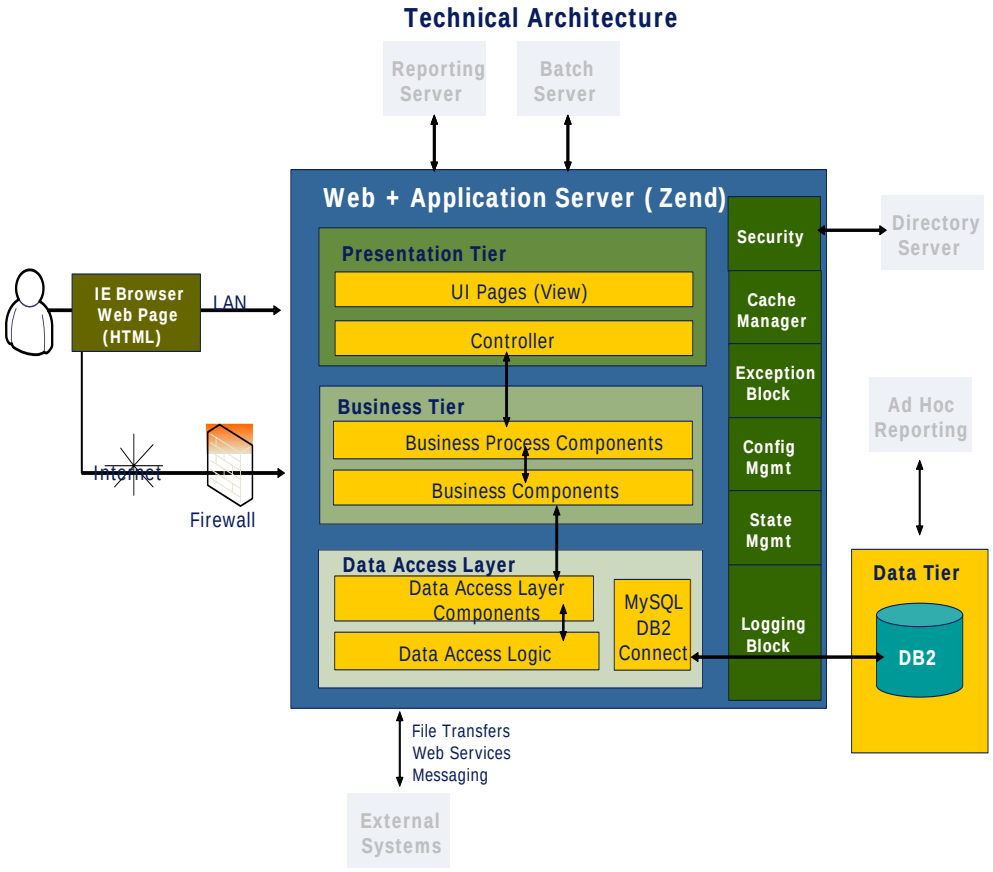

Figure 3: Technical Architecture

4.2.1 Presentation Layer

The Presentation Tier is responsible for rendering information stored in the database in a form that is easily understood by the end user. For the Data mart application, the Presentation Tier will consist of PHP and HTML pages and associated objects, which will enable proper generation of web pages on the user's browser. These objects will facilitate interaction with the Business Tier and the Data Access Tier.

4.2.2 Business Layer

The Business Tier encapsulates the core functionality of an application. As the presentation tier captures user interaction and passes it through, a disciplined approach is required to adequately process the incoming data. Based on Object Oriented best practices and principles, the solution's business tier provides an extensible and robust solution for all application needs.

Functionality will be divided between three types of objects 1) process components that can each be mapped to a single business functional requirement; 2) business components that handle the breakdown of this functionality into smaller, manageable tasks; and 3) Data Transfer Objects that provide the data transports mechanism for moving data between the architect layers.

- Business Process Component (BPC): The business process component creates a single entity to encompass complex business functionality and manages the execution of each part of the task through multiple business components. Thus, a single BPC is used for the execution of the high-level task, while the Create, Read, Update, and Delete (CRUD) behaviors of that task are executed by business components.
- Business Component (BC): The business component is used to perform a single work item and directly deal with the database stored procedures. When designed properly, one business component may be used by multiple business

process components if the simple task being performed is repetitive among multiple BPCs.

- Data Transfer Objects (DTO): In order to facilitate a wider range of flexibility and looser component coupling, our solution employs the use of Data Transfer Objects. DTO's are lightweight and serializable components, which primarily serve as the transport vehicle, which moves data between all layers of the solution. The link to the DTO is passed between the tiers of an application, without copying and recopying data. This saves resources and provides for a scalable design.

4.2.3 Data Access Tier

This tier isolates Data Access code and provides a highly optimized mechanism for communication with MySQL data stores. It makes sense to abstract this necessary logic into a separate layer, since doing so centralizes data access functionality and makes it easier to configure, maintain and re-use. This is in line with the methodology of providing a decoupled, yet robust, scalable and maintainable architecture.

The Data Access Tier is also architected to be database agnostic for all the known database platforms, yet our target implementation will be for MySQL. Since the execution code for interaction with all data stores is contained within the Data Access Tier, the rest of the application is coded without coupling to a specific database. The Data Access Tier can thus be specified to point to any of the major database packages, and

proper use of Object Oriented Design helps ensure that minimal overhead is incurred for this functionality. In addition, enhancements such as parameter caching are in place to help ensure a high-level of performance.

Summarization of the above tier is shown in the below table:

Table 1: Data Mart application Architecture

Data mart application Architecture (Zend Framework)		
Component	Description	Advantage
Business Tier		
Business Process Components	Manages the execution of business process. Uses one to many business components.	Creates a single entity to encompass complex business functionality
Business Components.	Encapsulates business logic on to discreet reusable business components. Deals directly with data access layer components.	Each business component may be reused by multiple business processes.
Data Access Layer		
Data Access Layer Components	Passes data components to necessary stored procedures	Contains optimized, pre-compiled code specific to application and data access requirements
Data Access Logic	Provides highly optimized mechanism for communication with data stores	Centralizes data access functionality and makes it easier to configure, maintain, and re-use
MySQL	Connects Zend (PHP) Data Access Components to the database.	
Interfaces	Provides an open interface architecture with multiple interface methods	Open architecture allows easy interfacing with other systems

Our solution is packaged with several reusable services developed to provide an application with pre-built, tested and highly scalable components that provide basic functionality. In addition to transaction management and multiple database integration, our Data Access layer has the capability for future use to output XML files that can used to share information with other systems.

4.2.3.1 Dimensional Model

Figure 4 describes the dimensional model design for our Data mart application. The ProductDimension, CustomerDimension and DateDimension constitute the dimension tables, and QuantityFact table is our Fact Table.

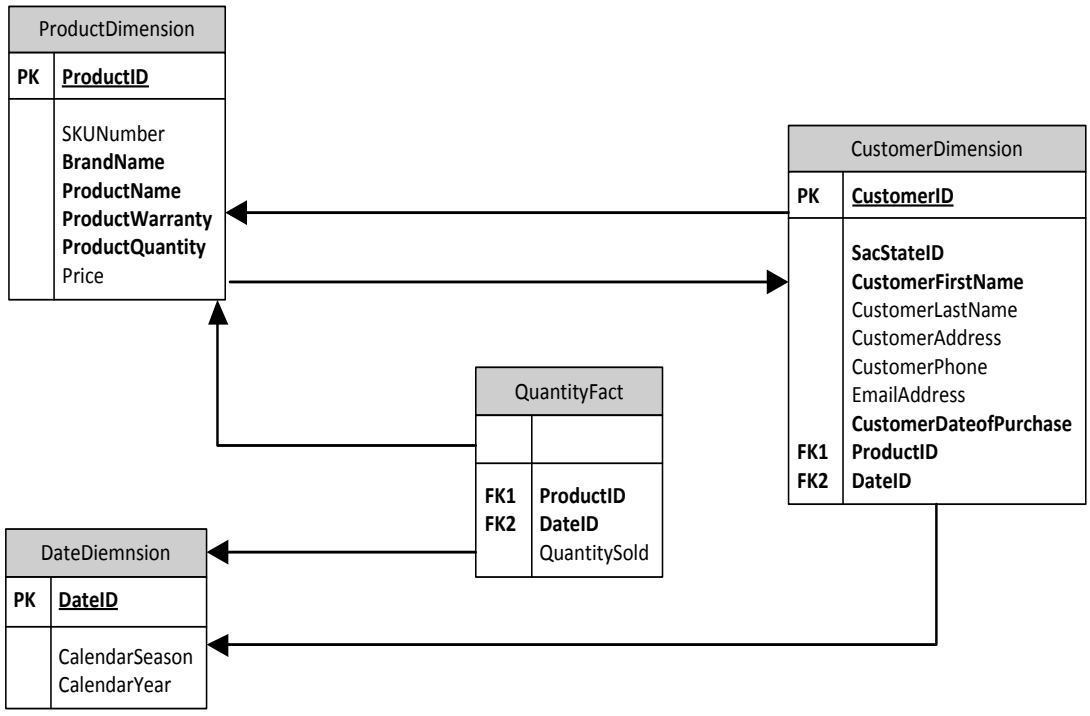

Figure 4: Dimensional Model

Dimension table is a table which is used to categorize the information into different hierarchical levels of attributes that provides the complete summarization to the fact table. Here the dimension tables namely, ProductDimension, CustomerDimension and DateDimension contains detailed information of products, customer and Date on different hierarchical levels respectively.

Fact table is a table in a data mart which stores measures or facts of a business process. “The measures are generally numeric and correspond to the *how much* or *how many* aspects of a question. Examples of measures are price, product sales, product inventory, revenue, and so forth. A measure can be based on a column in a table or it can be calculated” [10]. In our application, the QuantityFact table stores information regarding number of different brand’s product laptops based on the term and year.

4.3 User Interface Specifications

This section of the Software Design contains details for the screens that will be included in the system. These specifications have the details to develop the system.

4.3.1 Web Interface Design

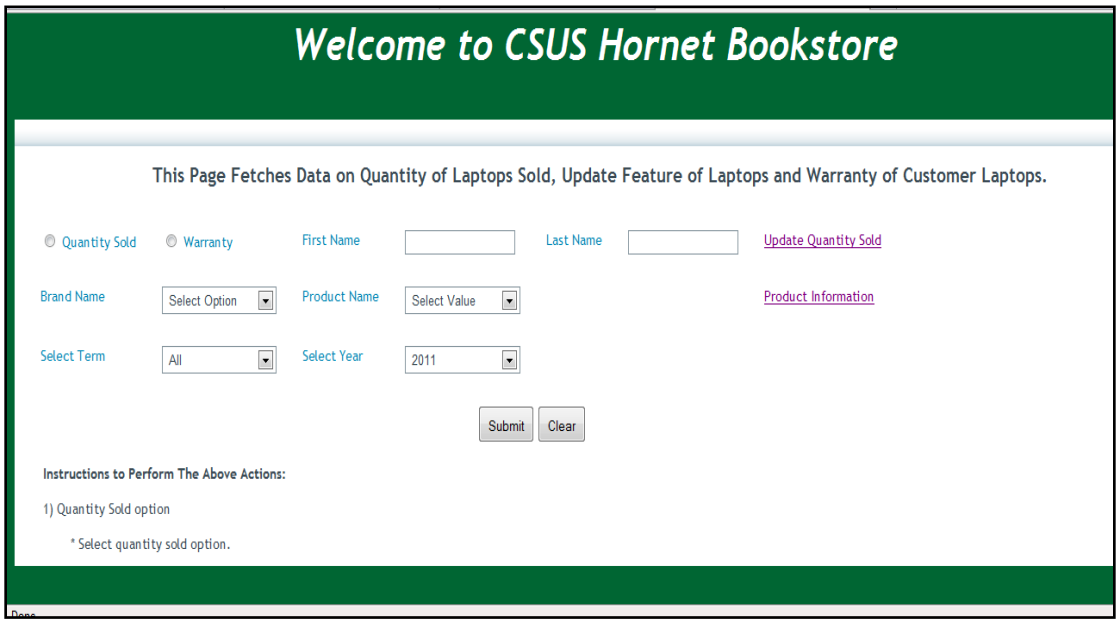

Figure 5: Web Interface Design

The web interface design covers three distinct aspects which are Usability – refers to the flow and navigation of pages, Functionality – meeting needs and supporting the tasks and Visualization – creating interest in accessing the application and avoiding unnecessary distractions [9].

The above figure is the front end design of the Data mart application for the Hornet bookstore, Computer science department. This design provides the bookstore the ability to trace the number of laptops sold based on the selection criteria, finding the warranty for a customer's laptops, view detailed information of products at the bookstore and update information of number of laptops sold for a particular brand name, product name, term and/or year.

Additional to these functionalities, the web interface also provides the set of instructions on how to perform the above mentioned tasks.

4.3.2 Use Case Diagram

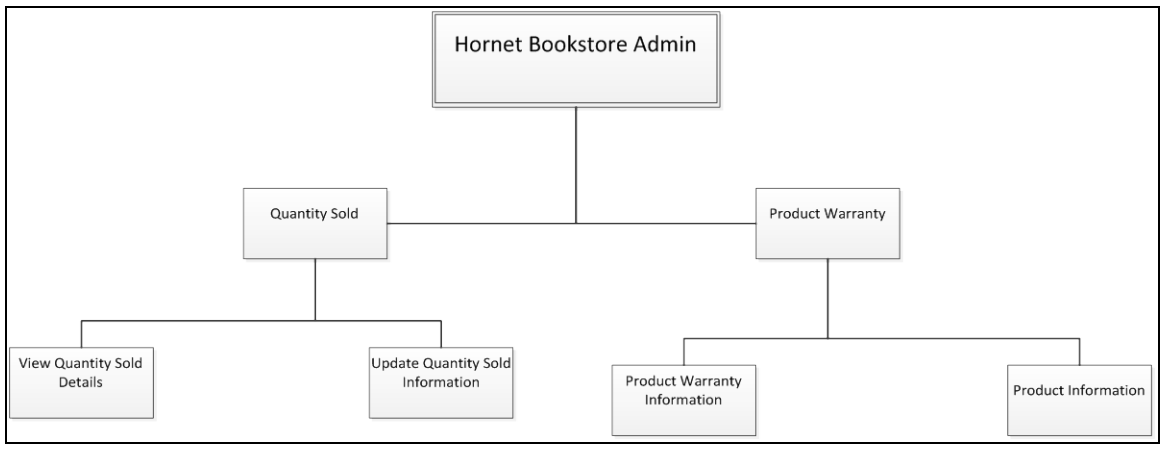

Figure 6: Use Case Diagram

The above figure gives an overview of the data flow in the application from user's perspective. Hornet Bookstore administrator has the authority to perform the below tasks using this application, they are:

- To find out number of laptops sold based on brand name, product name, term and/or year.
- To retrieve warranty information of customer laptops.
- To retrieve complete details of laptops which includes Product ID, SKU number, Brand name, Product name, warranty period, quantity and price of the laptops.
- To update Quantity sold information for each brand laptops.

Use Cases List

The Hornet Bookstore Data mart application use cases developed are listed in the table below.

Table 2: Use Case List

Use Case #	Use Case Name	Description	Actor
DM001	Quantity Sold	User views the number of laptops sold based on the selection criteria.	Hornet Bookstore Admin
DM002	Product Warranty	User views warranty of customer laptops.	Hornet Bookstore Admin
DM003	Update Quantity sold	Number of laptops sold is updated	Hornet Bookstore Admin
DM004	Product Information	User views complete details of the selected product.	Hornet Bookstore Admin

Chapter 5

IMPLEMENTATION

5.1 Getting Started with MVC

To maintain this type of an application in the long run is not simple. This is because the client keeps requesting for modifications as per their requirements or needs. As a result, changes are made in various places in the code [11].

resources.db.params.host = localhost
resources.db.params.username = root
resources.db.params.password = password
resources.db.params.dbname = DataMart [11]

Therefore, the maintainability of an application can be improved by separating the code into three manifest parts with separate files. They are:

Model: The Model part of the application holds specific data that is to be displayed**.** It is responsible for maintaining the state between each HTTP request. Model incorporates all the rules, restraints and behavior governing and utilizing this information. An example for this could be in an online banking application, any transactions/purchases above $2000 could be defined to be illegal and the transaction is forced to abort. In essence, a Model centers on the data and its underlying behavior.

View: View is the part of the application which is concerned with the display to the user. It is responsible for generating a user interface for your application. In PHP it is usually defined as the presentational HTML, RSS, and XML, JSON or anything else that is sent to the client application through a browser or requesting web service. The View is

organized into template files which can also be echoed from, or manipulated by the Controller prior to its output. The view also contains the PHP code to organize, filter and manipulate the format based on the data retrieved from one or more Models.

Controller: The primary function of the controller is to control and delegate. It is responsible to put together the specifics of the model and the view to ensure that the right data is displayed to the user. In MVC architecture for a web application, the controller takes the input from the user, translates it into actions on one or more Models and returns the output generated by the View to the user [11].

5.2 Configuring ZEND Framework and Library Application

Listen 10090

*NameVirtualHost *:10090*

*<VirtualHost *:10090>*

DocumentRoot "C:\Users\\Zend\workspaces\DefaultWorkspace7\Datamart\public”
RewriteEngine On
<Directory "C:\Users\ R\Zend\workspaces\DefaultWorkspace7\Datamart\public">
AllowOverride All
Allow from all
</Directory>
</VirtualHost>

AllowEncodedSlashes On

To enclose a group of directives which includes the named directory, sub – directories of that directory and all the files present within these directories, `<Directory>` and `</Directory>` tags are used. In a directory context if any directive is allowed, then those directives can be used.The *Directory-path will either be* a full path to a directory or a wild card string that is written using Unix shell-style matching. In a wild-card string, `?` is used to match any single character, `*` to match any sequences of characters and `[]is used for` character ranges. But none of the wild cards match with `/' character. Therefore, `<Directory /*/public_html>` will not match `/home/user/public_html`, but matches `<Directory /home/*/public_html>` [5].

Server needs to know the directives declared in a file when it finds an .httpd file that is specified by AccessFileName in order to know whether that file can override directives that were configured earlier [4].

AllowOverride is specified without any regular expressions. It is valid only in <Directory> sections and not in <Location>, <DirectoryMatch> or <Files> sections.

The .httpd files are completely ignored when this directive is set to None. As a result, the server will not attempt to read .httpd files present in the filesystem.

Whereas, when the same directive is set to All, then any directive which contains the .httpd Context is allowed in .httpd files.

The *directive-type* can be one of the following groupings of directive [5].

The RewriteRule is simple and can be analyzed as “for any url, redirect to index.php” [5].

We should make sure that a couple of php.ini must be set correctly for security purposes.

Additionally, we must make sure that if we use mod_php, only then the php_flag settings in .httpd must work. If we use CGI/FastCGI, then care must be taken with respect to php.ini settings.

Images, JavaScript files and CSS files should be placed within the public subdirectory and any requests to these files must not be directed to Bootstrapper. This way it helps in easy configuration of Apache and serves all these files with other httpd file in Datamart/public directly.

Datamart/public/.httpd

RewriteEngine off

Whilst not strictly necessary with out current rewrite rules, we can add a couple more .httpd files to ensure that our application and library directories are protected:

Datamart/application/.htppd

deny from all

Datamart/library/.htppd

deny from all

Note that for .htaccess files be used by Apache, the configuration directive AllowOverride

must be set to All within your httpd.conf file [5].

5.3 Directory Structure

The data mart's application looks similar to the below mentioned structure. This structure shows that we have complete control over our Apache configuration, however a little modification is made to the same structure. The modification is as follows:

Create a directory in the web server's root directory called Data mart i.e. the URL to go to the application is http://localhost/Datamart [11].

Application's files are located by creating the following subdirectories:

Datamart/
/application
/controllers
/layouts
/models
/views
/library
/public
/images
/scripts
/styles

From the above subdirectories created we noticed that there are separate directories for the model, view and controller files in our application. Supporting images, JavaScript files and CSS files are placed within the public subdirectory. The Zend Framework files which are downloaded will be saved in the library folder and if any additional libraries used they can also be placed in the same folder [11].

5.4 Bootstrapping

The Zend_Controller of Zend framework supports all the websites with clear or valid URL's. This is achieved by sending all requests to pass through a single index.php file called Bootstrapper. Bootstrapper acts as a central place for all pages in an application and guarantees that the environment setup is perfect for running the application. This is accomplished by using an .httpd file in the Datamart root directory [11].

5.4.1 The Bootstrap File: index.php

The bootstrap file in our application is Datamart/index.php. We start with the following code:

Datamart/index.php

```
<?php
set_include_path(implode(PATH_SEPARATOR, array(
    realpath(dirname(__FILE__) . '/../library'),
    get_include_path(),
)));

/* Define path to application directory*/
defined('APPLICATION_PATH')
    || define('APPLICATION_PATH', realpath(dirname(__FILE__) . '/../application'));
```

```
/* Define application environment */
defined('APPLICATION_ENV')
    || define('APPLICATION_ENV', (getenv('APPLICATION_ENV') ?
getenv('APPLICATION_ENV') : 'development'));

/** Zend_Application */
require_once 'Zend/Application.php';

/* Create application, bootstrap, and run */
$application = new Zend_Application(
    APPLICATION_ENV,
    APPLICATION_PATH . '/configs/application.ini'
);
$application->bootstrap();
$application->run();
?>
```

In the Bootstrap file make sure that the Zend Framework is in the include_path by setting the initial include_path. Also move this file to the php.ini or web server configuration for performance reasons. [11]

```
set_include_path(implode(PATH_SEPARATOR, array(
    realpath(dirname(__FILE__) . '/../library'),
    get_include_path(),
```

)));

The design of the Zend Framework is such that the files should be on the include path. In order to make the work easier in loading the models classes, place the models directory on the include path.

Zend.php files are to be included to have access to the Zend class because it has the required static functions which help us to load any other Zend Framework class.

Zend::loadClass('Zend_Controller_Front');

Here the Zend::loadClass loads the named class. This is accomplished by changing the underscores in the class name to path separators and adding .php at the end. Now the class Zend_Controller_Front is loaded from the file Zend/Controller/Front.php. Following the same naming conventions for other library classes will help us load the files by utilizing the Zend::loadClass(). To achieve this, the front controller and router class needs to be loaded.

The Router class is used by the first controller to map the desired URL to the right PHP function to display the pages. The router operates to know which part of the URL is the path to the index.php and then it looks at the URL element beyond that point. In regulation, the request object which is an instance of Zend_Controller_Request_Http must be able to detect the correct URL. The other altenative option in setting the base URL is byusing the setBaseUrl() function which is available within the http request class and within the front controller.

$_SERVER['PHP_SELF'] is used to set the base URL to the right URI in index.php for the router. This works for most server configurations. If this setting fails, then the designation is changed to baseUrl to the correct URL to index.php for the system.

To know in which directories the controllers are present, front controller configuration is required [11].

$frontController->setControllerDirectory('./application/controllers');

The front controller is implemented in such a way that it should throw exceptions for all the occurrences. The front controller catches these exceptions and stores them in the created _exceptions property of the “Response” object. The main purpose of the “Response” object is that it stores all the response to the requested URL. The front controller will then send the headers and displays the page content.

Now, we run the configured application:

// run!

$frontController->dispatch();

Now go to http://localhost/Datamart/ to test, fatal error similar to the below line should be displayed:

Fatal error: Uncaught exception 'Zend_Exception' with message 'File

"./application/ [11].

5.5 Results

This section provides a detailed explanation of the results obtained for the tasks performed using this application. They are:

1. To find sales rate of different brand laptops for each semester and then decide on ordering more number of particular brand laptops for next semester based on the laptops which was sold out in more number in the previous semester.

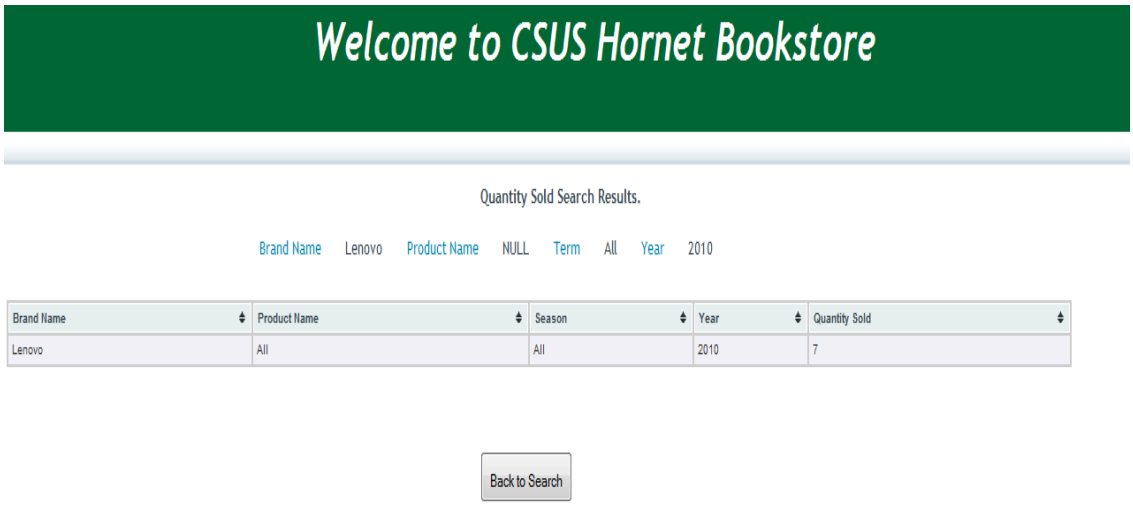

Figure 4: Quantity Sold Search Results Page

The above figure is the result for warranty of Lenovo laptops for all the product names, both fall and spring term and for the year 2010. This screen provides the number of Lenovo laptops sold for the above mentioned selections. All the selections made to view this result are displayed above the table of the result.

2. To find the warranty of computers purchased by the customers in order to find out whether the computers can be repaired at no cost or at some expense.

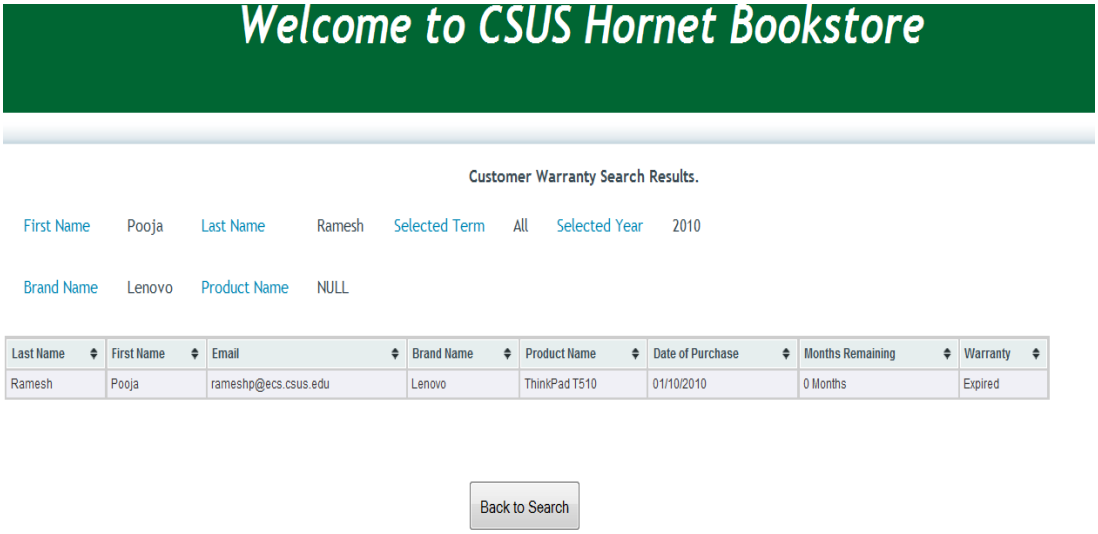

Figure 5: Customer Warranty Search Results Page

The above figure is the result of the task to view warranty of laptops purchased by the customers to find out whether the laptops can be repaired at no cost or at some expense. The result is obtained by selecting warranty option on the main screen and providing customer name and customer laptop details to view warranty of customer laptop purchased.

The result screen displayed shows the selections made along with the remaining warranty period, warranty status and date of purchase of the customer laptop.

3. Product information of all different brand laptops

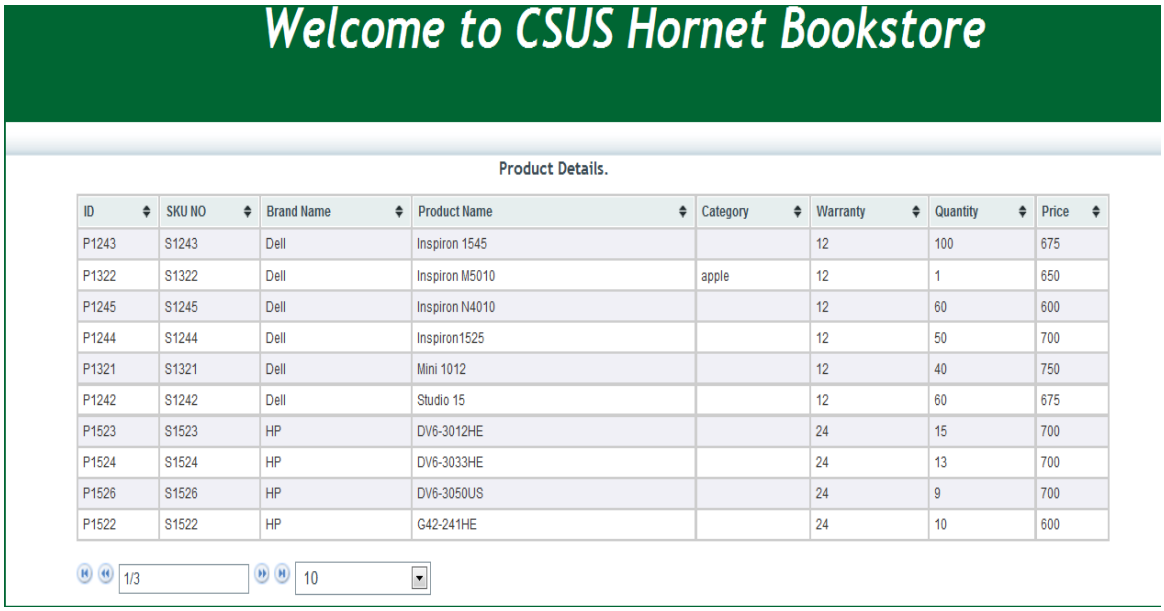

Figure 6: Product Details

The above figure provides the entire list of all the brand laptops sold at the Bookstore. This page provides all the details of laptops which includes product names, warranty period, number of laptops of sold of each product and the price of these laptops.

This page is displayed by clicking Product Information link in the main screen of the Data mart application.

4. Displaying Update Product information.

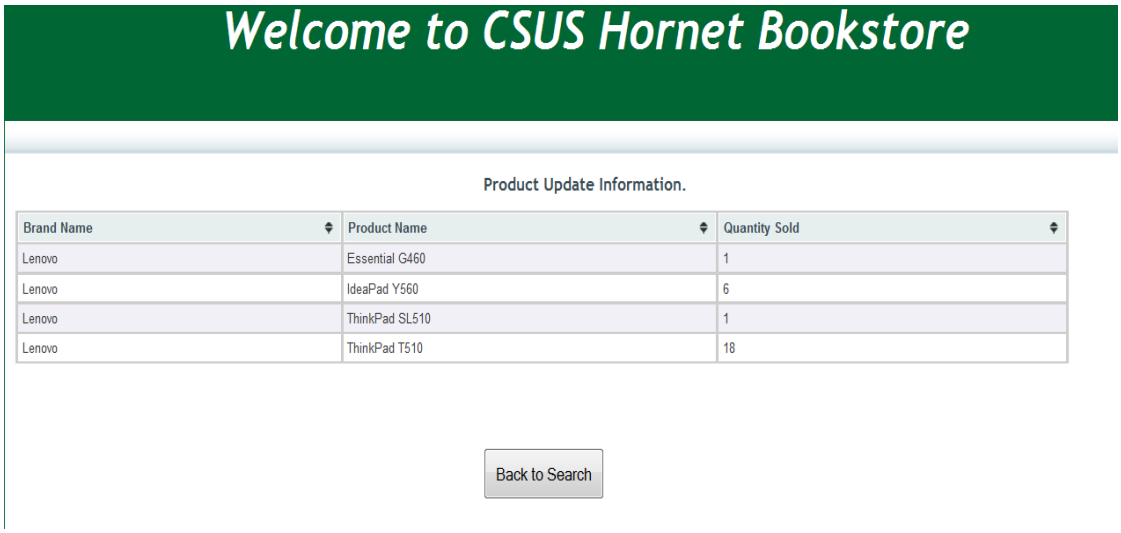

Figure 7: Product Update Information

The above figure shows the product update information when selected Lenovo as the brand name. This action is performed by clicking on the Update Quantity Sold link which is present on the main screen of the Data mart application.

5.5 Performance Analysis

Currently CSUS Hornet Bookstore is performing this task manually. In the sense, the laptop information and the customer information are stored in Excel files and any information has to be searched manually through those excel files. The two main business objective of this project is to automate the whole process and improve the efficiency of quantity fact and warranty searches. The normal database would have been served the purpose, but in order to provide the better performance Data mart comes into picture. The performance of the searches is improved by using the Fact tables namely, Quantity Sold

and Warranty and the Dimension tables. Once the related query is run, the fact table retrieves data from0 dimension tables and stores the data. This helps in faster retrieval of data when compared to the database which runs queries every time to view the same result and thereby taking more time in gathering data when compared to a Data Mart. Therefore, the designed Data Mart provides efficient data access, improves performance and Hornet Bookstore users response time.

The CSUS Hornet Bookstore Data Mart application built has a proxy layer between Front end and application layer with a layer of memory storage. This layer helps in increasing performance of the application by not overloading the application server when user requests are made. The proxy server stores returned data from the application, and when the same requests are made from Front end, the proxy server will return the data stored in the memory. This way it removes any extra calls to the application layer. [12]

Based on the feedback from CSUS Hornet Bookstore changes were made to the application. Firstly, a link was provided to list all the updated and entire product information of laptops by making necessary selection. This way, it makes the work of the Bookstore staff easier and retrieving of the result is faster when compared to retrieval of data from a database by running a query. Secondly, added Table sorter which is a JavaScript library for sorting the contents in the table. Also with the table sorter we can restrict the amount of data flowing to the web application. For example if the user performs a search which results in 500 rows of data only first 10 results can be shown when the page is initially loaded. The user can then scroll down the table for the next 10 results. This makes the application faster to show the minimum data in the initial go

rather than waiting for the application to load all the 500 rows for the first time. This is one of the functionality the bookstore was impressed, also a functionality to change the number of rows to be displayed on the page. Later, to make the better use of this application a set of instructions on how to use this application was provided, which served as the purpose of user manual. Overall the bookstore can completely eradicate the use of Excel files and desired searches can be done in a button click.

The advantages in using this application are: [12]

- Delivers content quickly.
- Improves efficiency and response time.
- Provides better usability to users as data is retrieved quickly.

Tools which helped in better design of the application are as below:

HTML5: HTML 5 makes the structure pages much easier by adding whole set of new elements. A variety of common structures are included in HTML 4 pages, they are headers, footers and columns. These are marked up as div elements by giving a descriptive id or class for each of them. Many of the sophisticated features of sites relay on connections which are created between different web technologies i.e. HTML, JavaScript and Cascading Style Sheets.

A standard way to run JavaScript in the background by making multiple treads runs at the same time which helps to run multiple applications at the same time. These threads in the background can perform complex mathematical calculations, network requests or to access local storage while the main web page users actions such as scrolling, typing or

clicking. Separate threads in the background perform different actions without affecting the performance of the webpage. This way of multiple thread functionality is very useful for web applications to perform functions without affecting other actions at the same time on the applications [13].

AJAX: The main drawback in using traditional HTML based websites is the waiting time for pages to get reload and refresh when the user clicks on an option or a Hypertext link. Over several years of attempts to improve dynamism of web pages by introducing individual techniques such as JavaScript, DHTML, CSS and Microsoft's MLHttpRequest ActiveX tool. Web2.0 applications and services leads to the approval of using one specific group of technologies called Asynchronous Javascript + XML (AJAX). In AJAX, once the page is loaded only small amounts of data pass from and through the server. This helps only a portion of the web page to be reloaded in real time and provides a more responsive interface with the application. Hence, it is with the introduction of AJAX that has helped to overcome the drawback of traditional HTML [14].

Chapter 6

CONCLUSION

This project is to develop a Data mart application for the Computer department of Hornet Bookstore at California State University, Sacramento. The Data mart application built facilitates the Hornet Bookstore to have an online prototype instead of maintaining inventory and sales information on an excel spread sheet. This application provides the book store to retrieve the needed data in a simpler manner thereby improving performance and saving the search time.

The Data mart application focuses on the top two priority searches which is frequently performed by the Hornet book store and also the detailed laptop information. They are: (1) Finding sales rate of different brand laptops for the mentioned year. (2) Finding warranty of customer laptops. (3) Update warranty information; and (4) Detail Laptop information.

Finally, we corroborate the success of our project because the built Data mart application met the requirements of the book store thereby satisfying the administrator demands. The application is hence handed to the book store as they plan to integrate it along with their current website for their future use. This application is made available only to the Hornet Bookstore Administrator and will be integrating within their website by setting access permission only to the book store administrator.

There is always scope for further developments, for this project in a given time the future work or enhancements could be done in the following areas:

- Implementation of Data Preprocessing, Data Mining for example: It is used to compare the sales rate of different brand laptops based on their selection criteria.
- Extract, Transform and Load process (ETL). ETL is highly recommended, because MYSQL has the capability of exporting data into XML format and Zend Framework has a XML parser which makes implementation easy to transfer the legacy data which is in the Excel files to the new Data mart.
- The data mart application can further be extended by implementing other priority searches performed by the Hornet book store.

APPENDIX A

Project Source Code: Index

```
<?php
// Set the initial include_path. You may need to change this to ensure
that
// Zend Framework is in the include_path; additionally, for performance
// reasons, it's best to move this to your web server configuration or
php.ini
// for production.
set_include_path(implode(PATH_SEPARATOR, array(
    realpath(dirname(__FILE__) . '/../library'),
    get_include_path(),
)));

// Define path to application directory
defined('APPLICATION_PATH')
    || define('APPLICATION_PATH', realpath(dirname(__FILE__) .
'/../application'));

// Define application environment
defined('APPLICATION_ENV')
    || define('APPLICATION_ENV', (getenv('APPLICATION_ENV') ?
getenv('APPLICATION_ENV') : 'development'));

/** Zend_Application */
require_once 'Zend/Application.php';

// Create application, bootstrap, and run
$application = new Zend_Application(
    APPLICATION_ENV,
    APPLICATION_PATH . '/configs/application.ini'
);

$application->bootstrap();

$application->run();
```

APPENDIX B

Project Source Code: Bootstrap

```php
<?php

/**
 * Application bootstrap
 *
 * @uses    Zend_Application_Bootstrap_Bootstrap
 * @package QuickStart
 */
class Bootstrap extends Zend_Application_Bootstrap_Bootstrap
{
    /**
     * Bootstrap autoloader for application resources
     *
     * @return Zend_Application_Module_Autoloader
     */
    protected function _initAutoload()
    {
      // This is required for the Default namespace to work correctly
        $autoloader = new Zend_Application_Module_Autoloader(array(
            'namespace' => 'Default',
            'basePath'  => dirname(__FILE__),
        ));

            // Add custom namespaces for autoloading
            $autoloader = Zend_Loader_Autoloader::getInstance();
            $autoloader->registerNamespace('Ebilling_');

        return $autoloader;
    }

    protected function _initSession()
    {
      // Read and initialize any config values
      $session = $this->getPluginResource('session');
      if ($session)
      {
            $session->init();
      }

        // Start the session (Zend recommended best practice)
        Zend_Session::start();

    }

    /**
     * Bootstrap the view doctype
```

```
     *
     * @return void
     */
    protected function _initDoctype()
    {
        $this->bootstrap('view');
        $view = $this->getResource('view');
        $view->doctype('XHTML1_STRICT');
            $view->addHelperPath('Zend/Dojo/View/Helper/',
'Zend_Dojo_View_Helper');

    }
}
```

APPENDIX C

Project Source Code: Application

```
[production]
phpSettings.display_startup_errors = 0
phpSettings.display_errors = 0
includePaths.library = APPLICATION_PATH "/../library"
bootstrap.path = APPLICATION_PATH "/Bootstrap.php"
bootstrap.class = "Bootstrap"

; Bootstrap resources:
; - Front Controller
; - Layout
; - Database
resources.frontController.controllerDirectory = APPLICATION_PATH
"/controllers"
resources.layout.layoutPath = APPLICATION_PATH "/layouts/scripts"
resources.view[] =
resources.view.helperPath.Zend_Dojo_View_Helper = APPLICATION_PATH
"/../library/Zend/Dojo/View/Helper/"
resources.db.adapter = "PDO_MYSQL"

[staging : production]

[testing : production]
phpSettings.display_startup_errors = 1
phpSettings.display_errors = 1

[development : production]
; For development, we want to display errors and use a different
database
phpSettings.display_startup_errors = 1
phpSettings.display_errors = 1
resources.db.params.host = localhost
resources.db.params.username = root
resources.db.params.password = password
resources.db.params.dbname = DataMart
```

APPENDIX D

Project Source Code: Layouts

```
<?php echo $this->doctype();

$controller = Zend_Controller_Front::getInstance()->getRequest()-
>getControllerName();
$action     = Zend_Controller_Front::getInstance()->getRequest()-
>getActionName();

$viewlist = array(
                        "home" => array(
                                                "search"
=>array("search", "index")

      ,"searchresult"=>array("searchresult", "index")

      ,"warrantyresult"=>array("searchresult", "index")
                                            )
                  );
?>

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>Library Application</title>
<?php
      echo $this->headLink()->appendStylesheet( $this-
>baseurl('/css/general.css'));
      echo $this->headLink()->appendStylesheet( $this-
>baseurl('/css/test.css'));
      echo $this->headLink()->appendStylesheet( $this-
>baseurl('/css/navigation.css'));
?>
</head>
<body onLoad="init()">
      <div id="wrapper">
      <div id="navigation">
                  Welcome to CSUS Hornet Bookstore
      </div>

            <div id="sub-navigation">
                  <div class="layout">
                        <table cellpadding="0" cellspacing="0"
class="tabs t100">
                              <tr>
                                    <td class="layout-left"></td>
                                          <td class="layout-center">
```

```
                                                <div id="sidebar">
                                                      <div
class="sidebar-inner">
                                                      <table
class="sub-tabs">

                                                      </table>
                                                      </div>
                                                </div>
                                          </td>
                                    <td class="layout-right"></td>
                              </tr>
                        </table>
                  </div>
            </div>

            <div id="main" style="width: 100%">
                  <div class="layout">
                        <div class="layout-left"></div>
                              <div class="layout-center"
      style="overflow: auto">

                                    <?php echo $this->layout()->content
?>
                              </div>
                        <div class="layout-right"></div>
                  </div>
            </div>

            <div id="footer">
            <div class="layout">
            <table cellpadding="0" cellspacing="0" class="footer t100">
                  <tr>
                        <td class="layout-left"></td>
                              <td class="layout-center"></td>
                              <td class="layout-right"></td>
                  </tr>
            </table></div></div></div>
</body>
</html>
```

APPENDIX E

Project Source Code: Controller (Error Handler)

```php
<?php

class ErrorController extends Zend_Controller_Action
{
    public function errorAction()
    {
        $errors = $this->_getParam('error_handler');
        switch ($errors->type) {
            //case
Zend_Controller_Plugin_ErrorHandler::EXCEPTION_NO_ROUTE:
            case
Zend_Controller_Plugin_ErrorHandler::EXCEPTION_NO_CONTROLLER:
            case
Zend_Controller_Plugin_ErrorHandler::EXCEPTION_NO_ACTION:
                // 404 error -- controller or action not found
                $this->getResponse()->setHttpResponseCode(404);
                $this->view->message = 'Page not found';
                break;
            default:
                // application error
                $this->getResponse()->setHttpResponseCode(500);
                $this->view->message = 'Application error';
                break;
        }

        $this->view->exception = $errors->exception;
        $this->view->request   = $errors->request;
    }
}
```

APPENDIX F

Project Source Code: Home Controller

```
<?php

class HomeController extends Zend_Controller_Action
{
	public function searchAction()
	{
		$ifSelected = $this->getRequest()->getParam( 'brandName' );
		//echo $ifSelected;
		$brandInstance = new Default_Model_Search();
		$brandName = $brandInstance->fetchinfo();
		if ($ifSelected != "")
		{
			$params['qsold']		= $this->getRequest ()-
>getParam ("quantitySold");
			$params['warranty']		= $this->getRequest ()-
>getParam ("isInWarranty");
			$params['firstName']	= $this->getRequest ()-
>getParam ("firstName");
			$params['lastName']	= $this->getRequest ()-
>getParam ("lastName");
			$productName			= $brandInstance-
>fetchproductinfo($ifSelected);
			//print_r($params); //exit;
			$this->view->initdata	= $params;
			$this->view->brandName	= $brandName;
			$this->view->setSelected = $ifSelected;
			$this->view->productName = $productName;
		}
		else
		{
			$this->view->setSelected	= 'null';
			$this->view->brandName		= $brandName;
		}
	}

	public function productnameAction()
	{
		$redirectlink	= '/home/search';
		$this->_redirect ($redirectlink);

	}

	public function searchresultAction()
	{
```

```
            $params['qsold']        = $this->getRequest ()->getParam
("quantitySold");
            $params['warranty']           = $this->getRequest ()-
>getParam ("isInWarranty");
            $params['firstName']    = $this->getRequest ()->getParam
("firstName");
            $params['lastName']     = $this->getRequest ()->getParam
("lastName");
            $params['brandName']    = $this->getRequest ()->getParam
("brandName");
            $params['productName']  = $this->getRequest ()->getParam
("productName");
            $params['term']               = $this->getRequest ()-
>getParam ("term");
            $params['year']               = $this->getRequest ()-
>getParam ("year");

            $resultInstance               = new Default_Model_Search();
            $results                      = $resultInstance-
>fetchproductresults($params);
            //print_r($results); exit;
            $this->view->results    = $results;
            $this->view->params           = $params;
      }

      public function warrantyresultAction()
      {
            $params['qsold']        = $this->getRequest ()->getParam
("quantitySold");
            $params['warranty']           = $this->getRequest ()-
>getParam ("isInWarranty");
            $params['firstName']    = $this->getRequest ()->getParam
("firstName");
            $params['lastName']     = $this->getRequest ()->getParam
("lastName");
            $params['brandName']    = $this->getRequest ()->getParam
("brandName");
            $params['productName']  = $this->getRequest ()->getParam
("productName");
            $params['term']               = $this->getRequest ()-
>getParam ("term");
            $params['year']               = $this->getRequest ()-
>getParam ("year");
            $resultInstance               = new Default_Model_Search();
            $results                      = $resultInstance-
>fetchproductresults($params);
            $this->view->params           = $params;
            $this->view->results    = $results;

      }

      public function updateAction()
```

```
	{
		$ifSelected	= 'null';
		$ifSelected	= $this->getRequest()->getParam(
'brandName' );
		$success	= $this->_getParam('success');
		$message	= $this->_getParam('message');
		$brandInstance  = new Default_Model_Search();
		$brandName	= $brandInstance->fetchinfo();

		if ($ifSelected != "")
		{
			$productName			= $brandInstance-
>fetchproductinfo($ifSelected);
			$this->view->productName = $productName;
		}

		$this->view->brandName   = $brandName;
		$this->view->setSelected = $ifSelected;
		$this->view->message     = $message;
	}

	public function updatequantityAction()
	{
		$params['brandName']    = $this->getRequest ()->getParam
("brandName");
		$params['productName']  = $this->getRequest ()->getParam
("productName");
		$params['term']		= $this->getRequest ()-
>getParam ("term");
		$params['year']		= $this->getRequest ()-
>getParam ("year");
		$updateInstance		= new Default_Model_Search();

		try
		{
			$updatedRecords   = $updateInstance-
>updatequantitysold($params);

			$arr['success']   = 1;
			$arr['msg']       = "Updated $updatedRecords Records
to Quantity Sold Successfuly";
		}
		catch (Exception $e)
		{
			$arr['msg'] = $e->getMessage();
		}

		if($updatedRecords)
		{
			$updatedRecords = $updateInstance-
>fetchupdated($params['brandName']);
		}
```

```
            $this->view->updatedRecords = $updatedRecords;
      }

      public function productinfoAction()
      {
            $productInstance        = new Default_Model_Search();
            $productResults               = $productInstance-
>fetchproducts();
            $this->view->results    = $productResults;
      }
}

?>
```

APPENDIX G

Project Source Code: Model (Search Page)

```
<?php
class Default_Model_Search
{
	public function fetchinfo()
	{
		$db  = Zend_Db_Table::getDefaultAdapter ();
		$sql = $db->query('SELECT DISTINCT BRANDNAME FROM
BRANDNAMEDROPDOWN');
		$brandname = $sql->fetchall();
		return $brandname;
	}

	public function fetchproductinfo($brandName)
	{
		$db          = Zend_Db_Table::getDefaultAdapter ();
		$dbquery     = "SELECT DISTINCT PRODUCTNAME FROM
BRANDNAMEDROPDOWN
                                        WHERE BRANDNAME =
'$brandName'";
		$sql         = $db->query($dbquery);
		$productName = $sql->fetchall();
		return $productName;
	}

	public function fetchproductresults($params)
	{
		$db  = Zend_Db_Table::getDefaultAdapter ();

		if($params['warranty'] == "true")
		{
			$dbquery  = "SELECT TIMESTAMPDIFF(MONTH,
CUSTOMERDATEOFPURCHASE, NOW()) < PRODUCTWARRANTY AS DIFFMONTH,
                                    CUSTOMERLASTNAME,
CUSTOMERFIRSTNAME, BRANDNAME, PRODUCTNAME,

	DATE_FORMAT(CUSTOMERDATEOFPURCHASE,'%m/%d/%Y') AS
CUSTOMERDATEOFPURCHASE, EMAILADDRESS
                                    FROM CUSTOMERDIMENSION,
PRODUCTDIMENSION, DATEDIMENSION
                                    WHERE PRODUCTDIMENSION.PRODUCTID =
CUSTOMERDIMENSION.PRODUCTID
                                    AND CUSTOMERDIMENSION.DATEID =
DATEDIMENSION.DATEID";
```

```
                  $dbquery .= " AND CUSTOMERFIRSTNAME = ".
"'".$params['firstName']."'".
                                    " AND CUSTOMERLASTNAME = ".
"'".$params['lastName']."'".
                                    " AND BRANDNAME =  ".
"'".$params['brandName']."'";
            }

            else
            {
                  if($params['productName'] == "NULL")
                  {
                        $dbquery = "SELECT SUM(QUANTITYSOLD) AS
QUANTITYSOLD, BRANDNAME,
                                    CALENDARSEASON, CALENDARYEAR,
0 AS PRODUCTNAME FROM
                                    QUANTITYFACT,
PRODUCTDIMENSION, DATEDIMENSION
                                    WHERE BRANDNAME = ".
"'".$params['brandName']."'"."
                                    AND
PRODUCTDIMENSION.PRODUCTID = QUANTITYFACT.PRODUCTID
                                    AND DATEDIMENSION.DATEID =
QUANTITYFACT.DATEID";
                  }
                  else
                  {
                        $dbquery = "SELECT SUM(QUANTITYSOLD) AS
QUANTITYSOLD, BRANDNAME,
                                    CALENDARSEASON, CALENDARYEAR,
PRODUCTNAME FROM
                                    QUANTITYFACT, PRODUCTDIMENSION,
DATEDIMENSION
                                    WHERE BRANDNAME = ".
"'".$params['brandName']."'"."
                                    AND PRODUCTDIMENSION.PRODUCTID =
QUANTITYFACT.PRODUCTID
                                    AND DATEDIMENSION.DATEID =
QUANTITYFACT.DATEID";

                  }
            }
            if ($params['year'] != "NULL")
            {
                   $dbquery .= " AND CALENDARYEAR = ".
"'".$params['year']."'";
            }

            if($params['productName'] != "NULL")
            {
                  $dbquery .= " AND PRODUCTNAME = ".
"'".$params['productName']."'";
```

```
            }

            if($params['term'] != "")
            {
                  switch ($params['term'])
                  {
                      case "All":
                          $dbquery .= " AND
(DATEDIMENSION.CALENDARSEASON = 'Spring' OR

      DATEDIMENSION.CALENDARSEASON = 'Fall')";
                          break;
                      case "Spring":
                          $dbquery .= " AND
DATEDIMENSION.CALENDARSEASON = 'Spring'";
                          break;
                      case "Fall":
                          $dbquery .= " AND
DATEDIMENSION.CALENDARSEASON = 'Fall'";
                          break;
                  }
            }
            $dbquery .= " GROUP BY BRANDNAME";
            //echo $dbquery; exit;
            $sql = $db->query($dbquery);
            $results = $sql->fetchall();
            return $results;
      }

      public function updatequantitysold($params)
      {
            $db  = Zend_Db_Table::getDefaultAdapter ();
            $dbquery = "SELECT P.BRANDNAME, P.PRODUCTNAME,
D.CALENDARYEAR, D.CALENDARSEASON,
                              COUNT(C.PRODUCTID) AS QUANTITYSOLD,
C.PRODUCTID, C.DATEID
                              FROM CUSTOMERDIMENSION C,
PRODUCTDIMENSION P, DATEDIMENSION D
                              WHERE C.PRODUCTID = P.PRODUCTID
                              AND C.DATEID = D.DATEID
                              AND P.BRANDNAME = ".
"'".$params['brandName']."'";

            if($params['productName'] != "NULL")
            {
                  $dbquery .= " AND P.PRODUCTNAME = ".
"'".$params['productName']."'";
            }

            if($params['year'] != "NULL")
            {
```

```
                  $dbquery .= " AND D.CALENDARYEAR = ".
"'".$params['year']."'";
            }

            if($params['term'] != "")
            {
                  switch ($params['term'])
                  {
                      case "All":
                          $dbquery .= " AND (D.CALENDARSEASON =
'Spring' OR
                                                  D.CALENDARSEASON =
'Fall')";
                          break;
                      case "Spring":
                          $dbquery .= " AND D.CALENDARSEASON =
'Spring'";
                          break;
                      case "Fall":
                          $dbquery .= " AND D.CALENDARSEASON = 'Fall'";
                          break;
                  }
            }
            $dbquery .= " GROUP BY P.PRODUCTNAME, D.CALENDARYEAR,
D.CALENDARSEASON,
                              P.BRANDNAME, C.PRODUCTID, C.DATEID";
            //echo $dbquery; //exit;
            $sql = $db->query($dbquery);
            $results = $sql->fetchall();

            $counter=0;
            try {
                  $db->beginTransaction ();
                  // Insert into EBILLINC
                  foreach($results as $row)
                  {
                        $dbquery = 'UPDATE QUANTITYFACT SET
QUANTITYSOLD = "'.$row['QUANTITYSOLD'].'"
                                          WHERE PRODUCTID =
"'.$row['PRODUCTID'].'" AND
                                          DATEID =
"'.$row['DATEID'].'"';
                        $sql = $db->query ($dbquery);
                        $counter++;
                        if($counter > 100){
                              $db->commit();
                              $counter=0;
                              $db->beginTransaction();
                        }
                  }
                  $db->commit();
            }           catch ( Exception $e )
```

```
            {
                  // An exception has been thrown We must rollback the
transaction
                  $db->rollBack ();
                  echo "here";
                  //throw new Exception($e->getMessage());
                  exit ();
            }
            return $counter;
      }

      public function fetchupdated($productName)
      {
            $db  = Zend_Db_Table::getDefaultAdapter ();
            $dbquery = 'SELECT SUM(QUANTITYSOLD) AS QUANTITYSOLD,
BRANDNAME, PRODUCTNAME
                        FROM QUANTITYFACT, PRODUCTDIMENSION WHERE
                        BRANDNAME = "'.$productName.'"
                        AND PRODUCTDIMENSION.PRODUCTID =
QUANTITYFACT.PRODUCTID
                        GROUP BY BRANDNAME, PRODUCTNAME';
            $sql = $db->query($dbquery);
            $results = $sql->fetchall();
            return $results;
      }

      public function fetchproducts()
      {
            $db  = Zend_Db_Table::getDefaultAdapter ();
            $dbquery = 'SELECT * FROM PRODUCTDIMENSION ORDER BY
BRANDNAME, PRODUCTNAME';
            $sql = $db->query($dbquery);
            $results = $sql->fetchall();
            return $results;
      }
}
```

APPENDIX H

Project Source Code: View

Error.phtml

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>Zend Framework Default Application</title>
</head>
<body>
  <h1>An error occurred</h1>
  <h2><?php echo $this->message ?></h2>

  <?php if (isset($this->exception)): ?>

  <h3>Exception information:</h3>
  <p>
      <b>Message:</b> <?php echo $this->exception->getMessage() ?>
  </p>

  <h3>Stack trace:</h3>
  <pre><?php echo $this->exception->getTraceAsString() ?>
  </pre>

  <h3>Request Parameters:</h3>
  <pre><?php echo var_export($this->request->getParams(), true) ?>
  </pre>
  <?php endif ?>

</body>
</html>
```

APPENDIX I

Project Source Code: View (Home Page)

a. ProductInfo

```
<?php
$result = $this->results;
?>

<script type="text/javascript" src="/js/jquery/pack.js"></script>
<script type="text/javascript"
src="/js/jquery/tablesorter.pager.js"></script>
<script type="text/javascript"
src="/js/jquery/tablesorter.js"></script>

<script type="text/javascript">
function init()
{
      // do nothing
}

$(document).ready(function() {
    $("#tableid")
    .tablesorter({widthFixed: true, widgets: ['zebra']})
    .tablesorterPager({container: $("#pager")});
});

function SubmitForm()
{
      document.productinfoform.action = "/home/search";
      document.productinfoform.submit();
}
</script>

<style>
/* tables */
 @import '/css/tablesorter.css';
</style>

<form name="productinfoform" >
<div style="padding-left: 40%; padding-top: 10px;">
<b>Product Details.</b>
</div>
<div style="padding-left: 5%;">
<?php
```

```
$o = '<table class="tablesorter" id="tableid">
            <thead>
                  <tr>
                        <th>ID</th>
                        <th>SKU NO</th>
                        <th>Brand Name</th>
                        <th>Product Name</th>
                        <th>Category</th>
                        <th>Warranty</th>
                        <th>Quantity</th>
                        <th>Price</th>
                  </tr>
            </thead>
            <tbody>';
            foreach($result as $rows)
            {
            $o .= '<tr><td>'.$rows['ProductID'].'</td>
                        <td>'.$rows['SKUNumber'].'</td>
                        <td>'.$rows['BrandName'].'</td>
                        <td>'.$rows['ProductName'].'</td>
                        <td>'.$rows['ProductCategory'].'</td>
                        <td>'.$rows['ProductWarranty'].'</td>
                        <td>'.$rows['ProductQuantity'].'</td>
                        <td>'.$rows['Price'].'</td>
                        </tr>';
            }

            $o .= '</tbody></table>';
echo $o;

?>
<div id="pager" class="pager">
            <img src="/images/tablesorter/first.png" class="first"/>
            <img src="/images/tablesorter/prev.png" class="prev"/>
            <input type="text" class="pagedisplay"/>
            <img src="/images/tablesorter/next.png" class="next"/>
            <img src="/images/tablesorter/last.png" class="last"/>
            <select class="pagesize">
                  <option selected="selected"  value="10">10</option>
                  <option value="20">20</option>
                  <option value="30">30</option>
                  <option  value="40">40</option>
            </select>
</div>
</div>

<div style='padding-left:40%; padding-top: 50px;'>
      <span>
            <button style="height: 40px;" id="Submit" type="button"
onclick=SubmitForm()>Back to Search </button>
      </span>
</div>
```

```
</form>
```

b. Search.phtml

```
<script type="text/javascript">
function init()
{
	if(document.form1.warranty.checked)
	{
		document.form1.firstName.disabled = false;
		document.form1.lastName.disabled = false;
	}
	populateYearSelect();
}

function dropdown(sel)
{
   if(sel.options.selectedIndex == 0){
      alert('Please choose an option');
      return false;
   }
   else
   {
        document.form1.isInWarranty.value =
document.form1.warranty.checked;
        document.form1.quantitySold.value =
document.form1.qsold.checked;
      document.form1.action = "/home/search";
      sel.form.submit();
   }
}

var currentEnabled = null;
function enableElement(elem1, elem2)
{
	if (currentEnabled)
	{
		currentEnabled.disabled = true;
	}
	elem1.disabled = false;
	elem2.disabled = false;
	currentEnabled = elem1;
}

function disableElement(elem1, elem2)
{
	document.form1.firstName.value = "";
	document.form1.lastName.value = "";
	elem1.disabled = true;
	elem2.disabled = true;
	currentEnabled = elem1;
```

```
}

function SubmitForm()
{
	document.form1.isInWarranty.value =
document.form1.warranty.checked;
	document.form1.quantitySold.value = document.form1.qsold.checked;
	if(document.form1.warranty.checked == false &&
			document.form1.qsold.checked == false)
	{
		alert("Please Select Quantity Sold or Warranty Before
Search");
		return false;
	}
	if(document.form1.brandName.options.selectedIndex == 0)
	{
		alert("Please Select a Brand Name Before Search");
		return false;
	}
	if(document.form1.isInWarranty.value == "true")
	{
		if(document.form1.firstName.value == "" ||
document.form1.lastName.value == "")
		{
			alert("Please Enter Customer First Name and Last
Name");
			return false;
		}
		document.form1.action= "/home/warrantyresult";
	}
	else
	{
		document.form1.action = "/home/searchresult";
	}
	document.form1.submit();
}

function ClearForm(oForm)
{
	var elements = oForm.elements;
    oForm.reset();
	for(i=0; i<elements.length; i++)
	{
	    field_type = elements[i].type.toLowerCase();
		switch(field_type)
		{
			case "text":
			case "password":
			case "textarea":
		    case "hidden":
				elements[i].value = "";
				break;
```

```
                case "radio":
                    case "checkbox":
                          if (elements[i].checked)
                          {
                                elements[i].checked = false;
                          }
                          break;

                    case "select-one":
                    case "select-multi":
                                elements[i].selectedIndex = -1;
                          break;

                    default:
                          break;
              }
        }
}

function populateYearSelect()
{
   d = new Date();
   curr_year = d.getFullYear();
   for(i = 0; i < 10; i++)
   {
       document.getElementById('year').options[i] = new
Option(curr_year-i,curr_year-i);
   }
}

</script>

<?php
$brandName = $this->brandName;
$setSelected = $this->setSelected;
$params = $this->initdata;
//print_r($params); exit;
?>

<form name="form1" action="home/searchresult" >
<div style="padding-left: 20px;
      padding-bottom: 20px;
      padding-top:10px;
      font-size: 14pt;
      text-align: center;" >
<b>This Page Fetches Data on Quantity of Laptops Sold,
Update Feature of Laptops and Warranty of Customer Laptops.</b>
</div>
<div class='section'>
<table cellspacing="10px" cellpadding="10px">
      <tr>
            <td class="label">
```

```
            <input type="radio"  name="group" id="qsold"
                  <?php if ($params['qsold'] == 'true') echo
'checked="checked"; else echo ""';?>"

      onclick="disableElement(this.form.elements['firstName'],

      this.form.elements['lastName']);" />
                  Quantity Sold
            </td>

            <td class="label">
            <input type="radio" name="group" id="warranty"
            <?php if ($params['warranty'] == 'true') echo
'checked="checked";'?>"

      onclick="enableElement(this.form.elements['firstName'],

this.form.elements['lastName']);"/>
                  Warranty
            </td>

            <td class="label">First Name</td>
            <td><input id="firstName" type="text" name="firstName"
disabled="disabled"
                  value= "<?php if ($params['firstName'] != '') echo
$params['firstName']; else echo '';?>"
                   />
            </td>

            <td class="label">Last Name</td>
            <td><input id="lastName" type="text" name="lastName"
disabled="disabled"
                  value= "<?php if ($params['lastName'] != '') echo
$params['lastName']; else echo '';?>"/>
            </td>
            <td>
                  <a href="/home/update"  >Update Quantity Sold</a>
            </td>
      </tr>

      <tr>
            <td class="label">Brand Name </td>
            <td><select name="brandName" id = "brandName"
onchange="return dropdown(this)">
                        <option value=NULL> Select Option</option>
                        <?php
                        foreach($brandName as $name)
                              {
                              ?>
                              <option <?php if($setSelected ==
$name['BRANDNAME']) echo "selected"; else echo ""; ?> id="<?php echo
$name['BRANDNAME']?>">
```

```
                              <?php echo $name['BRANDNAME'] ?>
</option>
                        <?php }?>
                  </select>
            </td>

            <td class="label">Product Name</td>
            <td><select id="productName" name="productName">
                        <option value=NULL>Select Value</option>
                        <?php
                              $productName = $this->productName;
                              foreach ($productName as $name)
                              {
                        ?>    <option value="<?php echo
$name['PRODUCTNAME']?>">
                              <?php echo $name['PRODUCTNAME']?>

</option>
                              <?php
                              } ?>
                  </select>
            </td>
            <td></td>
            <td></td>
            <td>
                  <a href="/home/productinfo"  >Product Information</a>
            </td>
      </tr>

      <tr>
            <td class="label"> Select Term</td>
            <td>
                  <select name="term" id = "term">
                        <option>All</option>
                        <option>Fall</option>
                        <option>Spring</option>
                  </select>
            </td>
            <td class="label"> Select Year</td>
            <td>
                  <select name="year" id="year"></select>
            </td>
      </tr>
</table>
</div>

<div style='padding-left: 40%; padding-top: 10px;'>
      <span>
            <button style="height: 35px;" id="Submit" type="button"
onclick=SubmitForm()>Submit </button>
      </span>
      <span>
```

```
            <button style="height: 35px;" id="clear" type="button"
onclick=ClearForm(this.form)>Clear</button>
      </span>
</div>

<div style="padding-left: 1%; padding-top: 10px; text-align: left;">
<table cellspacing="5px" cellpadding="5px">
      <tr>
            <td><b>Instructions to Perform The Above Actions:</b></td>
      </tr>
      <tr>
            <td>1) Quantity Sold option</td>
      </tr>
      <tr>
            <td>       
                  * Select quantity sold option.</td>
      </tr>
      <tr>
            <td>       
                  * Select Brand name and Product name of your choice,
then select Term - FALL,
                  Spring or ALL ("ALL" option displays values for both
FALL and Spring) and Year.</td>
      </tr>
      <tr>
            <td>       
                  * Displays result for the choice made.</td>
      </tr>
      <tr>
            <td>2) Warranty option</td>
      </tr>
      <tr>
            <td>       
                  * Select Warranty option.</td>
      </tr>
      <tr>
            <td>       
                  * Type in the Customer First Name and Last
                  Name whose laptop warranty is being searched.</td>
      </tr>
      <tr>
            <td>       
                  * Select the Band name, Product name and Year of
                  Purchase of Customer's laptop.</td>
      </tr>
      <tr>
            <td>       
                  * Displays Customer's information which includes
                  Date of Purchase, Warranty status and Warranty
remaining.</td>
      </tr>
      <tr>
```

```
            <td>3) Update Quantity Sold option</td>
      </tr>
   <tr>
            <td>       
                  * Select Update Quantity Sold link, which navigates
to the next page.</td></tr>
</table>

</div>
<input type="hidden" id="isInWarranty" name="isInWarranty" value=""/>
<input type="hidden" id="quantitySold" name="quantitySold" value=""/>
</form>
```

c. SearchResults.phtml

```
<?php
$params = $this->params;
?>

<script type="text/javascript" src="/js/jquery/pack.js"></script>
<script type="text/javascript"
src="/js/jquery/tablesorter.js"></script>
<script>
function init()
{
      // do nothing
}
$(document).ready(function() {
        $("#myTable").tablesorter({widgets: ['zebra']});
});
</script>

<script type="text/javascript">
function SubmitForm()
{
      document.searchresultsform.action = "/home/search";
      document.searchresultsform.submit();
}
</script>

<style>

/* tables */
 @import '/css/tablesorter.css';

</style>

<form name="searchresultsform" >
<div style="padding-left: 40%; padding-top: 10px;">
<b>Quantity Sold Search Results.</b>
</div>
```

```
<div style="padding-left: 20%">
<table cellspacing="10px" cellpadding="10px">
      <tr>
            <td class="label">Brand Name </td>
            <td><?php echo $params['brandName'];?></td>

            <td class="label">Product Name</td>
            <td><?php echo $params['productName']?></td>

            <td class="label">Term</td>
            <td><?php echo $params['term']?></td>

            <td class="label">Year</td>
            <td><?php echo $params['year']?></td>
      </tr>
</table>
</div>



<?php
$result = $this->results;
$o = '<table class="tablesorter" id="myTable">
            <thead>
                  <tr>
                        <th>Brand Name</th>
                        <th>Product Name</th>
                        <th>Season</th>
                        <th>Year</th>
                        <th>Quantity Sold</th>
                  </tr>
            </thead>
            <tbody>';
            foreach($result as $rows)
            {
            $o .= '<tr><td>'.$rows['BRANDNAME'].'</td><td>'
                        .$rows['PRODUCTNAME'] = (($rows['PRODUCTNAME']
== "0") ? "All" : $rows['PRODUCTNAME']).'</td>
                        <td>'.$rows['CALENDARSEASON']=
(($params['term'] == "All") ? "All" : $rows['CALENDARSEASON']).'</td>
                        <td>'.$rows['CALENDARYEAR'].'</td>
                        <td>'.$rows['QUANTITYSOLD'].'</td></tr>';
            }

            $o .= '</tbody></table>';

echo $o;

?>

<div style='padding-left:40%; padding-top: 50px;'>
```

```
      <span>
            <button style="height: 40px;" id="Submit" type="button"
onclick=SubmitForm()>Back to Search </button>
      </span>
</div>
</form>
```

d. Update.phtml

```
<?php
$brandName        = $this->brandName;
$setSelected      = $this->setSelected;
$productName      = $this->productName;
$message          = $this->message;
?>

 <script type="text/javascript">
function init()
{
      populateYearSelect();
}

function populateYearSelect()
{
   d = new Date();
   curr_year = d.getFullYear();
   for(i = 0; i < 10; i++)
   {
       document.getElementById('year').options[i] = new
Option(curr_year-i,curr_year-i);
   }
}

function dropdown(sel)
{
   if(sel.options.selectedIndex == 0){
      alert('Please choose an option');
      return false;
   }
   else
   {
      document.quantitySold.action = "/home/update";
      sel.form.submit();
   }
}

function Back()
{
      document.quantitySold.reset();
      document.quantitySold.action = "/home/search";
      document.quantitySold.submit();
}
```

```
function SubmitForm()
{
	document.quantitySold.action = "/home/updatequantity";
	document.quantitySold.submit();
}
</script>

<form name="quantitySold" action="/home/update" >
<div style="padding-left: 20px;
	padding-bottom: 20px;
	padding-top:10px;
	font-size: 14pt;
	text-align: center;" >
<b>This page updates the quantity of laptops sold for a particular
brand and product</b>
</div>

<div style='padding-left:10%; padding-top: 10px;'>
<table cellspacing="10px" cellpadding="10px">
	<tr>
		<td class="label">Brand Name </td>
		<td><select name="brandName" id = "brandName"
onchange="return dropdown(this)">
			<option value=NULL> Select Option</option>
			<?php
			foreach($brandName as $name)
				{
				?>
				<option <?php if($setSelected ==
$name['BRANDNAME']) echo "selected"; else echo ""; ?> id="<?php echo
$name['BRANDNAME']?>">
				<?php echo $name['BRANDNAME'] ?>
</option>
			<?php }?>
		</select>
		</td>

		<td class="label">Product Name</td>
		<td><select id="productName" name="productName">
			<option value=NULL>Select Value</option>
			<?php
				$productName = $this->productName;
				foreach ($productName as $name)
				{
			?>	<option value="<?php echo
$name['PRODUCTNAME']?>">
				<?php echo $name['PRODUCTNAME']?>
</option>
				<?php
				} ?>
		</select>
```

```
            </td>
      </tr>

      <tr>
            <td class="label"> Select Term</td>
            <td>
                  <select name="term" id = "term">
                        <option>All</option>
                        <option>Fall</option>
                        <option>Spring</option>
                  </select>
            </td>
            <td class="label"> Select Year</td>
            <td>
                  <select name="year" id="year"></select>
            </td>
      </tr>
</table>
</div>

<div style='padding-left:40%; padding-top: 50px;'>
      <span>
            <button style="height: 40px;" id="Submit" type="button"
onclick=SubmitForm()>Update</button>
      </span>
      <span>
            <button style="height: 40px;" id="Submit" type="button"
onclick=Back()>Back to Search</button>
      </span>
</div>

<div style="padding-left: 1%; padding-top: 10px; text-align: left;">
<table cellspacing="5px" cellpadding="5px">
      <tr>
            <td><b>Instructions to Perform Update Action:</b></td>
      </tr>
      <tr>
            <td>* Select Brand name, Product name, Term and Year in
order to fetch the updated
                        quantity sold result for the above selection
made.</td>
      </tr>
      <tr>
            <td>* Displays the recent updates of laptops sold along
with the details of laptops
                        for which the selection is made.</td>
      </tr>
</table>
</div>
</form>
```

e. UpdateQuantity.phtml

```
<style>

/* tables */
 @import '/css/tablesorter.css';

</style>

<script type="text/javascript" src="/js/jquery/pack.js"></script>
<script type="text/javascript"
src="/js/jquery/tablesorter.js"></script>

<script>
function init()
{
      // do nothing
}
$(document).ready(function() {
        $("#myTable").tablesorter({widgets: ['zebra']});
});

function SubmitForm()
{
      document.updatequantityform.action = "/home/search";
      document.updatequantityform.submit();
}
</script>

<?php
$updatedRecords = $this->updatedRecords;
//print_r($updatedRecords); exit;
?>

<form name="updatequantityform" >
<div style="padding-left: 40%; padding-top: 10px;">
<b>Product Update Information.</b>
</div>
<?php
$o = '<table class="tablesorter" id="myTable">
            <thead>
                  <tr>
                        <th>Brand Name</th>
                        <th>Product Name</th>
                        <th>Quantity Sold</th>
                  </tr>
            </thead>
            <tbody>';
            foreach($updatedRecords as $rows)
```

```
            {
            $o .= '<tr><td>'.$rows['BRANDNAME'].'</td><td>'
                        .$rows['PRODUCTNAME'].'</td>
                        <td>'.$rows['QUANTITYSOLD'].'</td></tr>';
            }

            $o .= '</tbody></table>';

echo $o;
?>

<div style='padding-left:40%; padding-top: 50px;'>
      <span>
            <button style="height: 40px;" id="Submit" type="button"
onclick=SubmitForm()>Back to Search </button>
      </span>
</div>
</form>
```

f. WarrantyResults.phtml

```
<?php
$params = $this->params;
?>
<script type="text/javascript" src="/js/jquery/pack.js"></script>
<script type="text/javascript"
src="/js/jquery/tablesorter.js"></script>
<script>
function init()
{
      //do nothing
}
$(document).ready(function() {
        $("#myTable").tablesorter({widgets: ['zebra']});
});
</script>

<script type="text/javascript">
function SubmitForm()
{
      document.warrantyresultsform.action = "/home/search";
      document.warrantyresultsform.submit();
}
</script>

<style>

/* tables */
 @import '/css/tablesorter.css';
```

```
</style>

<form name="warrantyresultsform" >
<div style="padding-left: 40%; padding-top: 10px;">
<b>Customer Warranty Search Results.</b>
</div>
<div>
<table cellspacing="10px" cellpadding="10px">
	<tr>
		<td class="label">First Name</td>
		<td><?php echo $params['firstName'];?></td>

		<td class="label">Last Name</td>
		<td><?php echo $params['lastName'];?></td>

		<td class="label"> Selected Term</td>
		<td><?php echo $params['term']?></td>

		<td class="label"> Selected Year</td>
		<td><?php echo $params['year']?></td>
	</tr>

	<tr>
		<td class="label">Brand Name </td>
		<td><?php echo $params['brandName'];?></td>

		<td class="label">Product Name</td>
		<td><?php echo $params['productName']?></td>
	</tr>
</table>
</div>

<?php

$result = $this->results;

$o = '<table class="tablesorter" id="myTable">
		<thead>
			<tr>
				<th>Last Name</th>
				<th>First Name</th>
				<th>Email</th>
				<th>Brand Name</th>
				<th>Product Name</th>
				<th>Date of Purchase</th>
				<th>Months Remaining</th>
				<th>Warranty</th>
			</tr>
		</thead>
		<tbody>';
		foreach($result as $rows)
		{
```

```
            $o .= '<tr>
                                <td>'.$rows['CUSTOMERLASTNAME'].'</td>
                                <td>'.$rows['CUSTOMERFIRSTNAME'].'</td>
                                <td>'.$rows['EMAILADDRESS'].'</td>

      <td>'.$rows['BRANDNAME'].'</td><td>'.$rows['PRODUCTNAME'].'</td>

      <td>'.$rows['CUSTOMERDATEOFPURCHASE'].'</td>
                                <td>'.$rows['DIFFMONTH'].' Months</td>
                                <td>'.(($rows['DIFFMONTH'] > 0)? "In
Warranty": "Expired").'</td>
                           </tr>';
            }

            $o .= '</tbody></table>';

echo $o;

?>

<div style='padding-left:40%; padding-top: 50px;'>
      <span>
            <button style="height: 40px;" id="Submit" type="button"
onclick=SubmitForm()>Back to Search </button>
      </span>
</div>
</form>
```

BIBLIOGRAPHY

1. Data Warehouse [Online]

Available: http://www.tech-faq.com/data-warehouse.html

2. Data mart does not equal Data warehouse [Online]

Available:http://www.information-management.com/infodirect/19991120/1675-1.html

3. Working of PHP image. Available online: http://www.learnphp-tutorial.com/PHPBasics.cfm

4. "Zend Framework: Survive The Deep End" a free book available online:

Available Online: http://www.survivethedeepend.com/

5. "Programmer's Reference Guide". [Online]

Available:http://framework.zend.com/manual/en/learning.quickstart.intro.html

6. Apache HTTP Server Version 2.2 [Online]

Available: http://httpd.apache.org/docs/current/mod/core.html#allowoverride

7. Sommerville, Ian (2006). *Software Engineering* (8th ed.). ISBN 978-0321313799. http://www.cs.st-andrews.ac.uk/~ifs/Books/SE8/index.html

8. Stellman, Andrew, Greene, Jennifer (2005). *Applied Software Project Management* O'Reilly Media. p. 113. ISBN 978-0-596-00948-9. http://www.stellman-greene.com/aspm/

9. Web Interface Design [Online]

Available: http://www.edtech.vt.edu/edtech/id/interface/index.html

10. IBM Informix Database Design and Implementation Guide [Online]

Available:http://publib.boulder.ibm.com/infocenter/idshelp/v10/index.jsp?topic=/com.ibm.ddi.doc/ddi223.htm

11. Rob Allen, “Getting Started with Zend Framework”. [Online]

Available: http://www.tanit.hu/files/getting-started-with-zend-framework-152.pdf

12. Data Mart [Online]

Available: http://www.shinstudio.com/blog/backend-tech/use-cache-technologies-to-improve-performance-of-your-site/

13. Web Development Tools and articles [Online]

Available:http://www.xibl.com/web-development/html5-features-tips-and-techniques-you-must-know/

14. Chrisina Draganova “Asynchronous JavaScript Technology and XML (AJAX)”. [Online]

Available:

http://www.londonmet.ac.uk/fms/MRSite/acad/foc/research/conferences/jicc11.pdf