2007

Web-based inventory management system for the West Virginia State Police

Robert M. Horwatt
West Virginia University

Follow this and additional works at: <https://researchrepository.wvu.edu/etd>

Recommended Citation

Horwatt, Robert M., "Web-based inventory management system for the West Virginia State Police" (2007).
Graduate Theses, Dissertations, and Problem Reports. 1797.
<https://researchrepository.wvu.edu/etd/1797>

This Thesis is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/ or on the work itself. This Thesis has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.

Web-Based Inventory Management System for the
West Virginia State Police

Robert M. Horwatt

Thesis submitted to the College of Engineering and Mineral Resources at West
Virginia University in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science

Roy S. Nutter, Ph.D., Chair
Bojan Cukic, Ph.D.
John Moody, Ph.D.

Lane Department of Computer Science
and Electrical Engineering

Morgantown, West Virginia
2007

Keywords: IEEE std. 830, software engineering, software requirements specifications,
database, web-based database, normalization, VPN, data security

ABSTRACT

Web-Based Inventory Management System for the West Virginia State Police

Robert M. Horwatt

Abstract: This thesis describes the conversion of a DOS-based database system to meet very specific web-based database requirements of the West Virginia State Police. The need for conversion was proposed by the West Virginia State Police in January of 2007. The conversion process given here: allows for normalization of all data; to make the process of inventory management easier for employees of the West Virginia State Police (WVSP); enumerates inventory management problems currently being faced by the WVSP; present a brief explanation of databases and the database model being used; and provides the initial requirements, specifications for this system. These requirements for the new database system will minimize current problems faced by the WVSP. A first prototype was developed and is presented as well as part of this thesis.

DEDICATED TO
my family and friends
for all their love and support
throughout the years
and especially to
Rick, for all your guidance,
love, support, and patience.

ACKNOWLEDGEMENTS

I would like to thank the following people for their assistance in furthering my education and proving advice in writing this thesis.

Dr. Roy Nutter, for his patience, guidance, and knowledge whenever it was needed.

Dr. Bojan Cukic, for his knowledge and guidance he provided without being asked.

Dr. John Moody, for his knowledge and consideration in preparing this thesis.

Dr. Afzel Noore and Brian Powell, for the opportunity to instill my knowledge to a new generation of students.

Dr. John Atkins, for the opportunity to continue my education.

To the many friends I have made during my time at W.V.U.

Thank you all.

TABLE OF CONTENTS

1.0 Introduction	1
1.1 Brief Justification	1
1.2 Statement of Problem	2
1.3 Database and Software Engineering Background	3
1.3.1 Database Background	3
1.3.2 Software Engineering Background	4
1.3.3 Rapid-Prototyping Life-Cycle Model Defined	4
2.0 Related Work	6
3.0 Methodology	8
3.1 Approach to Solve the Problem	8
3.1.1 Detailed User Level Description	15
3.1.2 Process of Adding a New Item to Inventory	16
3.1.3 Database Design	18
3.1.3.1 Database Table Example	18
4.0 Results	21
4.1 Security Achieved	21
4.2 Prototype Results	22
4.3 System Concerns	23
5.0 Conclusion	24
5.1 Bibliography	25
6.0 Future Issues	26
6.1 Presentation Notes	27
Appendix A – Requirements Document	29
A.1 Application	30
A.1.1 Application Purpose	30
A.1.2 Application Context/Overview	31
A.1.3 Major Constraints	32
A.1.4 User Characteristics	34
A.1.5 References	35
A.2 Architectural and Component – Level Design	36
A.2.1 Description of Architecture	36
A.2.1.1 Architectural Diagram	38
A.2.2 Functions	39
A.2.2.1 Login Functions	39
A.2.2.1.1 Sign_In	39
A.2.2.1.2 New_User	40

A.2.2.1.3 Forget Password	41
A.2.2.2 User-Level Zero Functions	42
A.2.2.2.1 Add_New_Item	43
A.2.2.2.2 Delete_Item	44
A.2.2.2.3 Backup_Database	45
A.2.2.3 User-Level One Functions	46
A.2.2.3.1 Current_Inventory	46
A.2.2.3.2 Update_Logistic_Inventory	47
A.2.2.3.3 Logistic_Repair	48
A.2.2.3.4 Logistic_Create_Repair	48
A.2.2.4 User-Level Two Functions	49
A.2.2.4.1 Assign	50
A.2.2.4.2 Assigned_Items	51
A.2.2.4.2 Available_Items	52
A.2.2.4.3 Personnel_Inventory	53
A.2.2.4.4 Vehicle_Inventory	54
A.2.2.4.2 Change_Location	55
A.2.2.4.3 Change_User_Level	56
A.2.2.4.4 Change_User_Level_Back	57
A.2.2.5 User-Level Three Functions	58
A.2.2.5.1 Update	58
A.2.2.5.2 Log_Repair	59
A.2.2.5.4 Mileage_Update	60
A.2.2.5.4 Check_Oil	61
A.2.2.6 User-Level Four Functions	62
A.2.2.6.2 Inventory	62
A.2.2.6.2 Find	63
A.2.2.6.3 Printer_Friendly	64
2.2.6.3 Exit	65
A.3 Table Descriptions and Overviews	66
A.3.2 Table Overview	66
A.3.2.1 User's Table	66
A.3.2.2 Personnel Table	66
A.3.2.3 Personnel Items Table	66
A.3.2.4 Vehicle Table	66
A.3.2.5 Vehicle Item Table	67
A.3.2.6 Vehicle Repair Table	67
A.3.2.7 Detachment Table	67
A.3.2.8 Gun Table	67
A.3.2.9 Badge Table	67
A.3.3.10 General Inventory Table	68
A.3.3 Table Definitions	68
3.3.1 User's Table	68
A.3.3.2 Personnel Table	68
A.3.3.3 Personnel Items Table	70
A.3.3.4 Vehicle Table	71
A.3.3.5 Vehicle Item Table	72
A.3.3.6 Vehicle Repair Table	73
A.3.3.7 Detachment Table	74
A.3.3.8 Gun Table	75
A.3.3.9 Badges	76
A.3.3.10 General Inventory	77
4 Description of User Interfaces	78

A.4.1 Level Zero User	78
A.4.2 Level One User	78
A.4.3 Level Two User	78
A.4.4 Level Three User	78
A.4.5 Level Four User	79
<i>A.5 Conclusion</i>	79
<i>Appendix B – Prototype Code</i>	80
B.1 Added	81
B.2 Add New Item	81
B.3 Add New Item – PHP	81
B.4 Badge Enter	82
B.5 Badge Enter – PHP	83
B.6 DellItem - PHP	85
B.7 DellItemSelection	85
B.8 Enter – PHP	86
B.9 EnterGI	87
B.10 EnterGun	90
B.11 EnterNewUser – PHP	91
B.12 Exit	92
B.13 Exit-PHP	93
B.14 Frames – PHP	93
B.15 Functions – PHP	93
B.16 GIEnter - PHP	95
B.17 GunEnter	97
B.18 ItemEnter - PHP	99
B.19 Level0Links	100
B.20 Level1Links	103
B.21 Level2Links	105
B.22 Level3Links	106
B.23 Level4Links	107
B.24 LogInventDisplay - PHP	108
B.25 LogisticsInvent - PHP	110
B.26 Main	110
B.27 MainFrame - PHP	111

B.28 NewItemSelected	111
B.29 NewUser	112
B.30 NewUserSetup – PHP	113
B.31 PerEnter	114
B.32 PerEnter – PHP	117
B.33 PerInvDisplay – PHP	119
B.34 PerInvFrames - PHP	122
B.35 PerInvSelect – PHP	123
B.36 PerItemEnter	124
B.37 PerItemEnter – PHP	126
B.38 TitleFrame	128
B.38 tools	129
B.39 UpdateCount – PHP	129
B.40 UpdateCountMath	129
B.41 UpdateCountSelect – PHP	132
B.42 UpdateCountSelectFrames – PHP	133
B.43 VehEnter	134
B.44 VehicleEnter	136
B.45 VehicleItemEnter – PHP	138
B.46 VehicleItemItem - PHP	140
B.47 Welcome – PHP	142

Table of Figures

Figure 1: Database Described	4
Figure 2: Rapid Prototype life cycle model [5]	5
Figure 3: Detachment layout in reference to the main server	10
Figure 4: User Login to the Application	13
Figure 5: Entering a new inventory item into the IMS	17
Figure 6: General Inventory Table Record Example	19
Figure 7: The Relationships between Tables in the Database	20

1.0 Introduction

1.1 *Brief Justification*

This thesis shall illustrate one technique to convert a DOS-based database system to meet web-based database requirements. The West Virginia State Police has a critical need to update its DOS-based inventory system. A proper conversion will also allow normalization of all data in the database and will make the process of inventory management easier for employees of the West Virginia State Police (WVSP).

The current inventory system being used by the West Virginia State Police is an old commercial DOS-based database titled “Team Up”. This application was developed in the early 1980s [1]. Consequently the exact history and information regarding this application is no longer available. The West Virginia State Police database contains important information regarding many aspects of their personnel and inventory. The database includes detailed information of personnel, vehicles, objects assigned to personnel, such as guns and badges, and objects assigned to vehicles, such as laptop computers and light bars. In addition, included in this database is each detachment’s general inventory, such as desktop computers, desks, and forensic equipment. Other information items that are to be added in this database include indications of how these items were obtained, whether by a grant or a budget line item. Although this database is working for the West Virginia State Police, it is outdated and has many limitations. These limitations include but are not limited to the following:

- The application is only available to the main headquarters near Charleston, WV.

- Detachments cannot view their own inventory; a printed list from Charleston must be sent to each detachment.
- Multiple copies of the database are being used by three database administrators, causing the normalization of data to be non-existent.

When designing the tables and relationships for a database, certain logical inconsistencies commonly occur. A process called normalization helps you to ensure that these inconsistencies do not occur.

Normalization is the process of refining tables, keys, columns, and relationships to create a consistent database design[3].

1.2 Statement of Problem

This thesis shall describe a process for updating the West Virginia State Police's current Inventory Management System by developing the requirements specification and a prototype system. This prototype Inventory Management System should assist the next design team in building a complete system. The inventory system, once completed by the next design team, shall be accessible by all West Virginia State Police detachments through the use of a virtual private network (VPN). The main server will be located near Charleston, WV at WVSP headquarters. As requested by the WVSP this proposed application will incorporate a barcode scanner to make the task of entering and managing inventory much easier.

1.3 Database and Software Engineering Background

1.3.1 Database Background

A “database” can be described as “a collection of information organized in such a way that a computer program can quickly select desired pieces of data. You can think of a database as an electronic filing system [2].” A query is a request to the database system for information that satisfies user-defined criteria. The results or answers returned from these queries are data or information that can be used to obtain a conclusion or decision on the part of the user. The essential building block of a database is an organized collection of data that creates a “record”. A collection of records creates a “table”, and a collection of tables create the “database”. The structure, or organization, of these tables is known as a “schema”. The most common schema is a “relational model”, although there are other models. This relational model schema uses one or more columns or fields to relate to a field in a different table. This allows the use of multiple tables that contain data that describe a single entity, including the relationships of that entity to other data tables and records.

Figure 1: Database Described

1.3.2 Software Engineering Background

Software engineering can be defined as “The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software [4].” There are many different models that can be used for software engineering. However, this thesis shall describe the rapid-prototyping life-cycle model which will be utilized for the prototype selected for the West Virginia State Police Inventory System.

1.3.3 Rapid-Prototyping Life-Cycle Model Defined

A rapid-prototype is a working model that is functionally similar to that of the finished product. All functionalities of the finished product must be incorporated in the rapid-prototype model. This allows the user the ability to test the functionalities of the prototype as if it actually was the finished product. In addition, the design of

the application can be tested as well. This model, described in Figure 1, has the same construction as that of the finished products life-cycle; all phases of the life cycle must be completed.

The purpose of the rapid-prototype is to allow the client and the future users to interact and experiment with a model that is similar to the final product.

Figure 2: Rapid Prototype life cycle model [5]

2.0 Related Work

The application software needed by each police department to track inventory is unique to each department. As with business inventory management tracking software, there does not exist a single piece of software that can be configured to meet the inventory tracking needs of all businesses. This quandary seems to exist with police departments throughout the world. Similar to businesses, the inventory tracking system application for each and every police department seems to be unique. Each police department has different required information that needs to be stored. In the next section, a few related software applications shall be discussed with an explanation of why this type of inventory software does not already exist.

One major issue with commercial inventory applications that can be purchased from various vendors is that they do not normally include how an item was actually purchased. Since there is no standard for police inventory, the acquisition information of an item may not be available in commercial software.

Another concern deals with the issue of security - being able to define who can access various types of information. Current inventory management systems do not provide hierarchy of levels that is needed by all police departments. For example, the ability to add a new item to inventory should not be available to just any employee. Allowing this type of function at any user level can affect the integrity of the data within the inventory management database.

Commercial software such as IntelliTrack [6] can be integrated with existing databases. This software can manage and track inventory or can manage warehouse inventory. Both of these features are required by the WVSP. Although it offers many useful features, it does not include features such as how an item was acquired. Another feature that is not offered is tracking personnel information. This is an important part of a police inventory system. Items that are assigned to a police officer, such as a gun or badge, are not available with this commercial application. Most commercial inventory management applications are designed for a point of sale type of business. These types of systems can not be modified for the requirements of a police department.

Although there are a few software packages designed for inventory tracking for police departments, they do not offer all the functionalities needed by the West Virginia State Police. The needs of the WVSP are extremely specific and are not achieved by any commercial software that is available.

3.0 Methodology

The proposed solution to the West Virginia State Police Inventory Management System will be described in detail in this section.

The proposed system shall create a database application that will improve the current system that the West Virginia State Police (the client) is using for inventory management. By implementing this system, the client should improve the accuracy of their inventory and make the process of inventory management considerably easier. This system will make use of one and only one database that will be shared by all users. This will ensure that all data that resides in the database is normalized. By doing this, all data will be current and up to date. The users of the system shall include database administrators, who have full “read” and “write” capabilities, a variety of other users that have varying permission levels of “read” and “write” capabilities, and also users that have read-only capabilities.

The approach taken for this application shall be described in detail in the following section.

3.1 Approach to Solve the Problem

Once the general system concept was agreed upon between the parties involved, meetings were conducted. From the beginning, it was decided that the entire application was too large for one person to complete. It was decided that this thesis and resulting prototype would be used as a reference and guide for a future design

team to create the final product. Due to the size of the final product, assumptions were made in creating this prototype. The prototype will include several, but not all, of the features the client requested.

Initially requirements information was gathered with Dr. Roy Nutter, West Virginia University Professor, and Sgt. Christopher Casto, West Virginia State Police, to discuss the database currently being used and its shortcomings. In addition to the current system's deficiencies, many ideas of how to improve the current system were discussed. Furthermore, the hierarchical structure of the West Virginia State Police was discussed. This hierarchy consists of the main office, or headquarters, located near Charleston, West Virginia with sixty-one satellites, or detachment locations, positioned throughout West Virginia. These sixty-one detachments are organized into districts consisting of multiple detachments. The primary objective for this application is to make certain that all detachments are capable of accessing the inventory database to view and alter their own inventory, thus insuring that they are working with a current up-to-date inventory list. This would make the process of monthly inventory management effortless. The current system being used requires the main headquarters to run a query for a detachment, print the results and mail these results to the detachment. It was decided to create a new database using SQL Server 2005. The design required that all table information could be transferred from the old database to the new database using a comma separated value (CSV) file for each table that exists in the current database system. Access to the new database system shall be established by a web application developed in ASP.NET. The new system shall only

be accessible through a virtual private network (VPN). By using a VPN, security measures can guarantee that only West Virginia State Police employees can enter the inventory management system. Finally, incorporating a barcode scanner into the inventory management process was requested.

Figure 3: Detachment layout in reference to the main server

Shortly after gathering some requirements information, a small rapid prototype was developed and presented to Dr. Roy Nutter and Sgt. Christopher Casto. Both parties

agreed that this prototype was on the correct track and was the type of application that was needed to improve the current system.

Finally design information was gathered with other employees of the West Virginia State Police; three of the eight employees attending this meeting were the database administrators (DBAs). The following people were in attendance;

- Dave Saffel Database Administrator
- Shirley Schneider Database Administrator
- Rick Nissel Database Administrator
- Lieutenant G. E. McCabe Communications Officer
- Corporal Jerry Dornburg Communications Officer
- Sergeant Christopher Casto Digital Forensics Officer
- Dr. Roy Nutter West Virginia University Professor
- Robert Horwatt West Virginia University Graduate Student

Many design aspects of the current system were further explained in detail. Design features that were required and/or sought were delineated. A major feature was the normalization of the data that resides in the database. Currently, the three DBAs are working with multiple copies of the same database. Therefore information may not be accurate throughout all copies. It was decided that this application was too large for one person to handle alone in the allotted time. It was decided by Dr. Nutter that the system designer should make assumptions regarding the application and database that would meet as many of the requirements stated by the client as possible. This

design shall be used by a future design team to aid in the creation of an application that the client will install and utilize.

Once all the information gathered during the three meetings was separated and organized, the design process of the application was started. First and foremost, the programming languages and database application needed to create the rapid prototype had to be selected. The language chosen for the front-end, or client-side, of the application was hypertext markup language (HTML). This language was chosen because of the ease of use and the knowledge the designer has with this programming language. The back-end, or server-side, of the application will be MySQL. This database was chosen for its ease of use, previous experience the designer had with MySQL and the current unavailability of SQL Server 2005. PHP (PHP: Hypertext Preprocessor) was selected as the scripting language. This scripting language, that will be used to tie the client-side to the server-side, was preferred for the ease of use, and the designer's knowledge of this scripting language. This rapid prototype was created on a UNIX server running Apache. The space on the UNIX server was provided by Mr. Jeff Sumey, a California University of Pennsylvania professor, a former professor of the designer.

The system shall incorporate five distinct levels of users. These levels are numbered zero through four. A brief description of the five users follows:

Level 0: This user-level shall include all of the database administrators. These users shall have the ability to read and write freely to the database.

Level 1: This user-level shall include the logistics officers. These users shall have limited write capabilities and restricted read capabilities.

Level 2: This user-level shall include personnel that oversee a district. These users shall have further restricted read and write capabilities.

Level 3: This user-level shall include employees that are in charge of a detachment. These users shall have even more restricted read and write capabilities.

Level 4: This user-level shall include all other personnel at a detachment location. These users shall have read-only capabilities.

Figure 4: User Login to the Application

By organizing the users of the system into this type of hierarchical structure, different levels of information security can be provided. This is accomplished by only allowing each level of user to view information that is applicable to that level of employment with the WVSP. Further, information security management is attained by allowing each user level to write only what is essential to their particular user-level.

3.1.1 Detailed User Level Description

When a user logs into the application, the user level field will be used to determine the level of read/write permissions for that user. When a new user registers with the application, they will default to user level 4. The database administrator will change the new user's level if needed. The user levels shall be described below in more detail:

Level 0 – This user shall be the database administrators that are employed by the West Virginia State Police. This is the highest user level. They shall have permission to read any data that exists in the database and have the capability to freely change any information within the database. A few of these permissions shall be to add a record to any table in the database. They shall also have the ability to delete any record in the database. Also, they shall have the ability to back-up the database at any time.

Level 1 – This user shall be the logistics officer. This user is responsible for maintaining items that need to be repaired. Also, the user may track inventory items, such as a case of individually wrapped batteries for flashlights. For example, if a logistics officer opens a new case of batteries, he/she will be able to alter the number of unopened cases in inventory. This user shall have somewhat restricted read and write capabilities to the database.

Level 2 – This user shall be an employee that oversees a district. This user shall oversee multiple detachments. One feature this user shall have is the ability to alter the status of an item. If an item needs to be repaired,

borrowed, or transferred to another detachment, this user shall have the ability to modify the location of this item. This user shall have even more restrictions on what can be read or written to the database.

Level 3 – This user shall consist of the head of a detachment. This user shall have extremely restricted read and write capabilities. A few functionalities this user will have are the ability to alter a user's permission level and change the location of an item within their detachment. This user will only be permitted to alter the permission level of a user that is employed at the same detachment as the detachment head. This type of altering will be useful when the detachment head goes on vacation, for instance. This shall allow another employee of the detachment the same permissions as his/her superior, while they are absent. This level also permits an item location change, for example, if a computer is relocated to a different room within the detachment.

Level 4 – This user shall consist of any other employee at a detachment. This user shall only have limited read capabilities. This user type may be doing the actual inventory management for their detachment.

All users that have a lower user level shall have the same capabilities as the user below them.

3.1.2 Process of Adding a New Item to Inventory

The West Virginia State Police have a system to add new items to inventory. This process must be thoroughly documented by utilizing the proposed system. Initially, the item is purchased. The database administrators, level 0, must generate a unique

inventory item number and a barcode associated with the item. Once the new inventory item obtains a unique number, all information (manufacturer, model, serial number, etc.) associated with this item must be added to the database. Furthermore, the item must be assigned to a detachment, and it must be related to a location within the detachment, where the item resides.

Figure 5: Entering a new inventory item into the IMS

Inventory items can be assigned to a person, a detachment or a vehicle. Managing this type of inventory requires multiple tables that need to be related. For example, guns and badges are assigned to a person. General inventory items, such as desktop computers, printers, etc., are considered detachment items. Items assigned to vehicles consist of laptop computers, radios, etc. Even a vehicle can be assigned to a person.

3.1.3 Database Design

The database design will be a relational database. By utilizing this type of database, the West Virginia State Police database administrators shall have the capability to create new tables without undue difficulty, and to track any type of inventory items, from a vehicle or a person to an item assigned to a vehicle or person. An example of a table, with corresponding fields and records is described below.

3.1.3.1 Database Table Example

The General Inventory Table will consist of the following fields.

• DPS Number	• Purchase Date
• Loc Code	• Cost
• Manufacturer	• Entered By
• Model	• Entered On
• Serial	• Count
• Description	

The **DPS Number** field shall contain the unique identification number for the new item being added to inventory and shall be the primary key for this table. The **Loc Code** shall contain the detachment ID for where the inventory item shall reside. This field shall also be used as a primary key. These two fields together shall create a composite key for this table. The next field, **Manufacturer**, shall store the manufacturer of the item being added. The **Model** field shall store the model number and/or name of the item. **Serial** shall hold the serial number of the item, if one exists. A brief description of the item shall be stored in the **Description** field. **Purchase Date** shall be used to store the date the item was purchased. The **Cost** field shall be used to store the price of the item. Who and when this item was entered into the inventory shall be stored in the **Entered By** and **Entered On** fields. Finally, **Count**

shall be used for inventory items utilized by the Logistics Office. This field shall be used for non-serialized inventory items where the count currently in inventory needs to be tracked.

DPS Number	Loc Code	Manufacturer	Model	Serial	Description	Purchase Date	Cost	Entered By	Entered On	Count
11111	000	Dell	Dimensi on E510	EZ506 2	512Mb Ram, 200Gb HD	01-23-2002	\$1395.00	Bob Horwatt	01-25-2002	
22222	111	Lon	4DRFP		4 drawer fire proof filing cabinet	05-15-1999	\$2549.99	Joan Smith	05-25-1999	
33333	222	Duracell	9 volt		Case of 9-volt batteries, 72 count	12-16-2003	\$119.99	Pete Jones	12-16-2003	53
4444	222	Duracell	Size D		Case of D cell batteries, 144 2pk count	08-30-2005	\$89.95	Bob Horwatt	08-31-2005	130

Figure 6: General Inventory Table Record Example

This prototype is assuming all items are purchased by the WVSP. In reality, items may be purchased, donated or acquired by a grant. This level of detailed information was never thoroughly discussed in any meeting. Therefore the final disposition of these types of data will not be addressed in this thesis. These items will need to be explored in further detail with the West Virginia State Police to ensure that the finished product is delivered to meet the client's needs.

Figure 7: The Relationships between Tables in the Database

4.0 Results

The West Virginia State Police's application request was an application that is very specific in nature. This type of specific application does not exist as a consumer piece of software that can be purchased. There are many generic consumer products that can be purchased that have the capabilities to achieve some of the features requested. However, there is no single commercially available software application that has enough configuration capabilities to meet the requirements needed by the West Virginia State Police. Therefore, this application needed to be specific in nature and called for the use of the software engineering development process. The prototype, which is a product of this thesis, is an application that meets many, but not all, of the required specifications. Additional meetings with the WVSP will be needed along with alterations of this prototype.

4.1 Security Achieved

The application makes use of only one database that assures data normalization. The application also assures the security of the data, via the hierarchical system of user levels. The user level hierarchy allows users to only view and/or change certain aspects of the database. Furthermore, additional security is obtained by the use of a VPN. This guarantees only West Virginia State Police employees are logging into the inventory management system.

4.2 Prototype Results

The prototype is an application that meets many, but not all, requirements that are needed by the WVSP. Some features were not included, either because certain current system knowledge was not available from WVSP, or because of the time constraints for the project and the need for a larger design team.

The application recognizes the user's level of permission and grants them access to the application. The user will only have the ability to read and/or write to the database determined by their user level. The system permits users the ability to convert any query result into a format that is printer friendly, for the use of inventory management. Furthermore the prototype includes the functionality to incorporate a barcode scanner. Since a barcode scanner was not provided during this stage of development, it could not be integrated at this point in the system design. In addition, the analysis of the prototype application showed that it is possible to meet the requirements that are needed to convert the WVSP DOS- based database to a web-based database. Finally, the small data set used to test this prototype produced results that were accurate, dependable and up-to-date.

4.3 System Concerns

One item that was discussed with the WVSP was the idea of building a modular system. This would allow the ability to add other software modules for different features in the future. One feature mentioned was the ability to have employees log work hours and add a payroll function to the inventory system. One major concern with this idea is security. Do the employees of WVSP want all of their personal information, such as social security numbers in cyberspace?

Another concern is having a totally centralized database. Having this type of database is totally dependant on the network. If the network should go down for any reason, the entire system would become inoperable. The idea of a decentralized database, so that each detachment can have its own database, with resynchronization done periodically, would prove much more beneficial in the long term.

5.0 Conclusion

A commercial off-the-shelf system for converting a DOS-based inventory management database of the WVSP to meet the requirements of a new web inventory management application is not readily available. Many assumptions were made to meet some of the requirements needed by the client. As a result, a prototype was developed that met many of the requirements. One attribute of the prototype, which is not available in the WVSP system, is security. When a web-based system was proposed, security needed to be incorporated into the new inventory management system. First, by using a VPN, entrance into the application is attained and the validity of the user is verified. Subsequently by assigning user levels, information security is accomplished. This hierarchy of users restricts the viewing and writing of information to the database depending on the user's level of permissions. Finally, creating one centralized database to be used by all users maintains the integrity of the information in the database. An initial requirements and specifications document has been developed and is presented here-in.

5.1 Bibliography

- [1] David W. Saffel, Chief Engineer, West Virginia State Police. Meeting held on January 18, 2007
- [2] Webopedia, "Online Encyclopedia of Computer Terms", June, 2003.
<http://www.webopedia.com/TERM/d/database.html>
- [3] Microsoft. "MicrosoftTechNet", January, 2007.
<http://www.microsoft.com/technet/prodtechnol/sql/70/books/c0318260.msp>
- [4] Roger S. Pressman. "Process: A Generic View," in *Software Engineering: A Practitioner's Approach*. 6th ed. New York: McGraw-Hill, 2005, p 21. Chapter 2.
- [5] Stephen R. Schach. "Software Life-Cycle Models," in *Object-Oriented & Classical Software Engineering*, 6th ed. New York: McGraw-Hill, 2005, p 51. Chapter 2.
- [6] IntelliTrack Inc., "Intellitrack Software", January, 2002.
http://www.intellitrack.net/barcode_software.asp
- [7] IEEE Std. 830 – 1998 – IEEE Recommended Practice for Requirements Specifications. 1998
- [8] IEEE Std. 1233 – 1998 – IEEE Guide for Developing System Requirements Specifications. 1998
- [9] Christian, Wenz, *PHP Phrasebook*. Indianapolis: Sams Publishing. 2005.
- [10] Rasmus, Lerdorf, *PHP:Pocket Reference*, 2nd ed. Sebastopol, CA: O'Reilly. 2002.
- [11] Luke Welling and Laura Thomas, *PHP and MySQL Web Development*, 3rd ed. Indianapolis: Sams Publishing. 2004.

6.0 Future Issues

Some future issues with the West Virginia State Police inventory management system that the next design team will need to address will be described in this section.

Policies concerning usernames and passwords will need to be developed by the WVSP. These policies should include how many letters and/or numbers a username will need to be. Passwords should also be addressed in the same manner as the username. Along with the above mentioned, rules regarding passwords need to be defined by the WVSP, such as how often passwords need to be changed and whether older passwords can or cannot be used again.

Another concern is the number of user levels needed by the WVSP for the inventory management system. The current prototype only includes five levels of information security. A complete description of all information security levels will need to be obtained from the West Virginia State Police. The questions of who will be allowed to see what information, and who will be allowed to make alterations to what information, must be carefully defined. In addition, the encryption of sensitive data should be considered. This data would include: social security numbers, addresses, etc., but the WVSP may wish to encrypt all transmitted data to insure higher levels of security.

Finally, a more in-depth understanding is needed of how the current inventory system works. Specifically current data must be redefined and organized into proper database

tables and field descriptions. These definitions must also include how inventory items are acquired, categorized, labeled, and into which table(s) these data items should be stored.

By spending more time with the client, the next design team will be able to obtain the necessary information and database knowledge needed to create an inventory management system that will meet all the requirements needed by the West Virginia State Police.

6.1 Presentation Notes

A presentation of this thesis along with a demonstration of the prototype was presented to employees of the West Virginia State Police. These employees included;

- Dave Saffel Database Administrator
- Shirley Schneider Database Administrator
- Rick Nissel Database Administrator
- Lieutenant G. E. McCabe Communications Officer
- Corporal Jerry Dornburg Communications Officer
- Sergeant Christopher Casto Digital Forensics Officer
- Dr. Roy Nutter West Virginia University Professor
- Robert Horwatt West Virginia University Graduate Student

The following list represents comments following the presentation.

- The idea of redundant data being ported to another database.
- They would like the application front end to be as simple as possible.

- ASP.Net and SQL 2005 is a must for this application.
- The possibility of integrating this application with the existing WVCrime application.
- Migrate existing data from database to the new database using a .csv (comma separated value) file. This will make the conversion very simple.
- History reports on all inventory
- Plate number and vehicle color needs to be added to the vehicle tables.
- Transportation reports.
- Commodity codes need to be added to the new database.
- Warning windows that ask the user “Do you want to do this?”
- The currently have approximately 800 tables that need to be incorporated in the new application.
- Education information of each employee that states what classes they have taken.

Appendix A – Requirements Document

A.1 Application

A.1.1 Application Purpose

The purpose of the West Virginia State Police Inventory Tracking System is to update the current inventory system currently being utilized. The current system is a DOS-based application that can only be used by the West Virginia State Police headquarters in Charleston. The system that is being proposed shall be a web-based application that shall allow all detachments to access their own inventory for monthly checking and updating. This application shall be only available through a Virtual Private Network (VPN) into the West Virginia State Police Intranet and utilize an Imagetech 4600/4800 hand-held barcode scanner.

A.1.2 Application Context/Overview

This document specifies the requirements for the West Virginia State Police Inventory Tracking System (WVSP-ITS). This document is not a design, but is a formal listing of what the system should do once it is completed. It details the functions that the system shall perform. Due to time constraints in developing this project, this document shall serve as a starting point for future design teams. This specification document also serves as the basis for all later deliverables.

The WVSP-ITS application is designed to make the procedure of inventory management easier for all employees involved in this process. By implementing one common database to be utilized by all users, the need to update and maintain duplicate databases shall be non-existent.

A.1.3 Major Constraints

The WVSP-ITS shall adhere to the following constraints:

- The application shall integrate an Imagetech 4600/4800 hand-held barcode scanner.
- Development of the application shall be free of cost.
- The application shall be compatible with the existing operating systems and hardware in all WVSP facilities.
- The application shall use passwords to protect the database.
- The application shall only be available via the WVSP Intranet.
- The database shall run on Windows XP, Windows 2000, and future operating systems Windows Vista and Windows Longhorn.
- The end user interface data shall be accessible from all web-browsers running on Windows and MAC operating systems.
- The future development team shall provide user manuals for all permission levels of the application.
- The future development team shall provide a programmer's manual to assist with maintenance of the WVSP-ITS.
- The future development team shall provide training for operating the application.
- The administrator shall have full capabilities to alter database tables and fields along with user interface screens.

- Sixty-four WVSP detachments and eight WVSP headquarters shall have access to WVSP-ITS via a VPN connection.
- The database shall be implemented with Microsoft SQL Server 2005.

A.1.4 User Characteristics

- The application shall have distinct user classes ranging from level zero to level four. These distinct levels shall have varying degrees of user permissions.
- Level Zero, Administrator, shall have the ability to read and write freely to the database.
- Level One shall have read capabilities, along with the following added write capabilities: The ability to add a new item that has been approved.
- Level Two shall have read capabilities, along with the following added write capabilities: the ability to alter the field that displays whether an item is in for repair, borrowed, transferred to a different detachment, or returned to the original detachment, or the decommission of an item.
- Level Three shall have read capabilities, along with the ability to change a level zero's level to a level one status. Another write capability shall be the ability to alter the location of an item within their detachment, for example, moving a light bar from one patrol car to another patrol car.
- Level Four shall have read-only capabilities. These users shall only be able to view inventory from their detachment.

A.1.5 References

IEEE Std. 830 – 1998 – IEEE Recommended Practice for Requirements Specifications.

IEEE Std. 1233 – 1998 – IEEE Guide for Developing System Requirements Specifications.

A.2 Architectural and Component – Level Design

A.2.1 Description of Architecture

WVSP-ITS shall have a data-centric client/server architecture. The central database management system (DBMS) shall reside on the server.

All users shall use a secure Virtual Private Network (VPN) to enter the WVSP Intranet. This shall ensure that no security breaches shall occur.

The server shall have five clients: Level Zero, Level One, Level Two, Level Three, and Level Four. Level Zero, the administrator, shall have full read and write capabilities to the DBMS. Level One, Logistics Officer, shall have the ability to add a new item to the DBMS, and track smaller inventory such as batteries. Level Two, District Manager, shall have the ability to change an item to the status of repair, borrowed, transferred or returned to detachment, along with an indication of when an item needs to be sent for checking. Level Three, Sergeant of a detachment, shall have the ability to change a user's level that is below him/her. In addition, this user level shall have the capability to change the location of an item within their detachment. Level Four shall have Read-Only capabilities.

Note that all users shall have the same capabilities as the users below them. So,

Level Zero shall have the capabilities of Level One through Level Four in addition to their own capabilities. Also, Level One shall have the capabilities of Level Two through Level Four and so on.

A.2.1.1 Architectural Diagram

WVSP – ITS Architecture

A.2.2 Functions

A.2.2.1 Login Functions

A.2.2.1.1 Sign_In

This function shall allow a user to enter the program. Once the user has entered his/her username and password, a query shall be sent to the DBMS for verification. While checking for validity, the query shall access and return the user's real name, the detachment to which they belong, and the current date and time.

Input

Username and password

Source of Input

The user utilizing a computer keyboard

Output

Entrance into the application

Destination of output

The user's user-level welcome page

Processing

This function shall extract the username and password information from the login screen. It shall then query the database and check for validity. If the data entered is valid, it shall retrieve the user's real name, user level and the detachment name from the database. The user-level data shall be used to determine which of the five screens to present to the user and welcome the user with their real name, detachment name and the current date and time.

Requirements

Correct username and password

Precondition

Successful VPN into the West Virginia State Police Intranet

Post Condition

Welcome screen displayed

Side effects

None

Responses to abnormal behavior

Present the user with a screen that states the username does not match the password.

A.2.2.1.2 New_User

This function shall add a new user to the application. Once the required fields are submitted, the application shall query the data to see if the username has already been used. If so, the application shall inform the user to select a different username.

Input

Username, Password, First Name, Last Name, Middle Initial, Address1, Address2, Address3, City, State, Zip, Phone, SSN, Rank, Detachment Code, Employee Number

Source of Input

The user utilizing a computer keyboard

Output

Either entrance into the application, or an indication that the username has already been taken.

Destination of output

The user's user-level welcome page or username already taken page

Processing

This function shall extract all required information from the new users screen. It shall then query the data and check for duplicate usernames and correct format. If the data entered is valid, the program shall welcome the user to the application

with the user's real name and the detachment name from the database. The user-level data shall be used to determine which of the five screens to present the user. If the username has already been taken, the form shall be displayed again with all information filled in except for the username and password. The user must then try a different username; if the username is valid, the user shall be admitted to the application. If the new username is taken, the process shall repeat until a unique username is entered.

Requirements

All fields must be filled and validated

Precondition

Successful VPN into the West Virginia State Police Intranet

Post Condition

Welcome screen displayed

Side effects

None

Responses to abnormal behavior

Present the user with a screen that states the username does not match the password

A.2.2.1.3 Forget Password

This function shall require the user to supply his/her social security number as validation before releasing the password to the user. The user shall be given the option to change the password or leave it as is.

Input

Username, Social Security Number

Source of Input

The user utilizing a computer keyboard

Output

Displays password

Destination of output

Password Page

Processing

This function shall extract the username and social security number from the page and query the data base for validity. If the information supplied is correct, the password for the user shall be displayed. The user shall be given the option to change the password if desired. If the user decides to change the password, the new password shall be written to the database.

Requirements

All fields must be filled and validated

Precondition

Successful VPN into the West Virginia State Police Intranet

Post Condition

Password displayed

Side effects

None

Responses to abnormal behavior

Present the user with a screen that states the username does not match the password

A.2.2.2 User-Level Zero Functions

This user level shall have complete read and write capabilities to the database. The Add_New_Item and Delete_Item shall be available for the Personnel, Personnel Item, Vehicle, Vehicle Item, Logistic Inventory, General Inventory options presented to the DBA. This user shall have all the capabilities and functionalities as the users beneath him/her.

A.2.2.2.1 Add_New_Item

This function shall allow the Database Administrators (DBA) to enter a new item to the database

Input

The new item's information

Source of Input

The DBA

Output

A message stating that a new item has been added to the database, along with the items information.

Destination of output

Add_New_Item page

Processing

The information entered by the DBA shall be used to add a new record to the database

Requirements

Successful login, a new item to be added to the database

Precondition

None

Post Condition

Database altered

Side effects

None

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.2.2 Delete_Item

This function shall allow the DBA to delete an item from the database

Input

Item information

Source of Input

The DBA

Output

A message stating that an item has been deleted from the database, along with the item's information.

Destination of output

Delete_Item page

Processing

The information shall be extracted from the page and used to query the database. Once the correct record has been found it shall be deleted from the database.

Requirements

The information entered by the DBA shall be used to delete a record from the database

Precondition

Successful login

Post Condition

Database altered

Side effects

If the item does not exist in the database, a message stating so shall be displayed

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.2.3 Backup_Database

This function shall allow the DBA to back-up the database periodically.

Input

The date of the back-up

Source of Input

DBA

Output

A message stating that the database has successfully been backed up, the name of the file, and the date it occurred.

Destination of output

Backup_Database page

Processing

The information entered by the DBA shall be used to title the database back-up image, and a copy of the current database shall be stored on the server

Requirements

Successful login

Precondition

None

Post Condition

A copy of the database shall be created with the date it occurred as its file name.

Side effects

If the server's hard drive should be full, a message shall be displayed stating an error occurred while trying to back-up the database

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.3 User-Level One Functions

This user shall have complete read capabilities and limited write capabilities.

This user shall also have the same functionalities as level two, level three and level four users. This user shall have the following unique functions:

A.2.2.3.1 Current_Inventory

This function shall display the current inventory of the logistics officer.

Input

None

Source of Input

None

Output

Displays current inventory for the logistics officer

Destination of output

Logistics Inventory Page

Processing

A query of the Logistics Inventory Table shall return all inventories and the count of records.

Requirements

Successful login

Precondition

None

Post Condition

Data presented to the user

Side effects

None

Responses to abnormal behavior

An error shall be displayed stating the information for that barcode is not in the database

A.2.2.3.2 Update_Logistic_Inventory

This function shall allow the Logistics Officer to update inventory.

Input

Barcode or item number

Source of Input

Imageteam 4600/4800 hand-held barcode scanner or keyboard

Output

Displays information for item(s) scanned

Destination of output

The Scan List screen

Processing

The information from the barcode that is scanned or input from the keyboard shall be used to query the database. The count field of the Logistics Inventory table shall be reduced by one.

Requirements

Successful login

Precondition

None

Post Condition

Database altered

Side effects

None

Responses to abnormal behavior

An error shall be displayed stating the information for that barcode is not in the database

A.2.2.3.3 Logistic_Repair

This function shall allow the Logistics officer to log a repair to an item in inventory.

Input

Item ID

Source of Input

Keyboard

Output

Display item

Destination of output

Repair Page

Processing

This function shall query the database and display the repair history of all items that have been repaired.

Requirements

Successful login, correct item ID

Precondition

Successful login

Post Condition

None

Side effects

None

Responses to abnormal behavior

An error message shall be displayed if the command could not be executed

A.2.2.3.4 Logistic_Create_Repair

This function shall allow the Logistics officer to enter a repair order for an item that can be repaired.

Input

Item ID, Repair, Entered By, Entered On, Cost, Repair By

Source of Input

Keyboard

Output

Display Item information

Destination of output

Repair Enter Page

Processing

This function shall extract the information entered by the user and create a new record in the Logistic Repair History Table.

Requirements

Successful login

Precondition

Successful login

Post Condition

The database is altered

Side effects

None

Responses to abnormal behavior

An error message shall be displayed if the command could not be executed

A.2.2.4 User-Level Two Functions

This user shall have limited write capabilities and full read capabilities. This user shall also have all functionalities of users below him/her. The user shall be presented with five unique options. They are Personnel, Personnel Items, Vehicle, Vehicle Items, and User Level Change.

A.2.2.4.1 Assign

This function is available for the Personnel Items, Vehicle, and Vehicle Items listed above. This function shall assign an item to a vehicle or person. If the Vehicle option is chosen, the vehicle shall be associated linked to a person.

Input

Personnel or Vehicle information and Item to be assigned

Source of Input

The district head by using the keyboard

Output

The information for the Person or Vehicle plus the information for the Item to be assigned

Destination of output

Item assign page

Processing

A query containing both the item and the person or vehicle that shall be associated with the item shall be displayed to the user. Once the user has confirmed the item and person or vehicle, the user shall click “OK” and the item that has been assigned shall be updated. This update shall relate the vehicle or person to that item.

Requirements

Successful Login

Precondition

None

Post Condition

Database update

Side effects

None

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.4.2 Assigned_Items

This function is available for the Personnel Items, Vehicle and Vehicle Items choices mentioned in the level description. This function shall display all items associated with one of the options. If Personnel Items is chosen, this function shall display a list of all items that have been assigned and to whom they are assigned.

Input

None

Source of Input

None

Output

The item information

Destination of output

Assigned page

Processing

The database shall be queried based on the Issued To field of the table associated with the options mentioned above. If the field contains data, the record shall be displayed.

Requirements

Successful Login

Precondition

None

Post Condition

None

Side effects

None

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.4.2 Available_Items

This function is available for the Personnel Items, Vehicle and Vehicle Items choices mentioned in the level description. This function shall display all items that are available for assignment from the option chosen. For example, if Personnel Items is chosen, this function shall display a list of all items that are available for assignment.

Input

None

Source of Input

None

Output

The item information

Destination of output

Unassigned page

Processing

The database shall be queried based on the Issued To field of the table associated with the options mentioned above. If the field contains no data, the record shall be displayed.

Requirements

Successful Login

Precondition

None

Post Condition

None

Side effects

None

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.4.3 Personnel_Inventory

This function shall display all items associated with Personnel. This function is only available when the user chooses Personnel from the options listed in the level description.

Input

Personnel ID number

Source of Input

Keyboard

Output

The items associated to the person entered

Destination of output

Personnel Inventory List page

Processing

A query with the personnel's ID number shall return all items associated with him/her.

Requirements

Successful Login

Precondition

None

Post Condition

None

Side effects

None

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.4.4 Vehicle_Inventory

This function shall display all items associated with a vehicle. This function is only available when the user chooses Vehicle from the options listed in the level description.

Input

VIN of vehicle

Source of Input

Keyboard

Output

The items associated with that vehicle

Destination of output

Vehicle inventory Page

Processing

A query with the VIN of the vehicle shall return all items associated with that vehicle.

Requirements

Successful Login

Precondition

None

Post Condition

None

Side effects

None

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.4.2 Change_Location

This function shall allow the detachment Sergeant to change the location of an item within their detachment.

Input

The user from level four who shall have the capabilities of the user level three and the to and from date

Source of Input

Data entered from the keyboard on Change_Location page by the detachment head

Output

A list of items whose location has changed within the detachment

Destination of output

The Change_Location page

Processing

A list shall be presented to the detachment head. They shall have the ability to only alter the location of the item(s)

Requirements

Successful login and item(s) that need to have their locations altered within the detachment

Precondition

Successful login

Post Condition

The location of the item(s) being altered within the database

Side effects

None

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.4.3 Change_User_Level

This function shall allow the detachment head to change a User-Level Four to a User-Level Three. This function shall also ask for the date range that this user's level shall be altered.

Input

Person selected from a drop-down list and the dates his/her level shall be altered

Source of Input

The detachment head

Output

A message stating that the selected user's level has been altered

Destination of output

The Change_User_Level_Result page

Processing

The information shall be extracted from the Change_User_Level page. This information shall be used to query the database and this record shall be altered for the time span inputted by the detachment head.

Requirements

Successful login

Precondition

None

Post Condition

Database altered

Side effects

If the detachment head has selected a user whose level has already been changed to a higher level, a message shall be displayed

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.4.4 Change_User_Level_Back

This function shall allow the detachment head to change the user's level back to User-Level Four. This function shall display a list of users whose level has been changed. The user shall select a name to change his/her level back to its original level.

Input

Person selected from a drop-down list and the dates his/her level shall be altered

Source of Input

The detachment head

Output

A message stating that the selected user's level has been altered

Destination of output

The Change_User_Level_Result page

Processing

The information shall be extracted from the Change_User_Level page. This information shall be used to query the database and this record shall be altered for the time span input by the detachment head.

Requirements

Successful login

Precondition

None

Post Condition

Database altered

Side effects

If the detachment head has selected a user whose level has already been changed to a higher level a message shall be displayed

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

A.2.2.5 User-Level Three Functions

This user level shall have read capabilities along with limited write permissions.

This user's functionalities are listed below, and shall have all the functionalities of user-level four. As this level of user enters the application, they shall be presented with six options. These options are Personnel, Personnel Items, Vehicle, Vehicle Items, General Inventory and Detachment.

A.2.2.5.1 Update

This function shall be available for all six options listed above. It shall allow the user to make updates to his/her inventory. This update only includes the location of an item within the detachment or vehicle, a change of address, phone, etc., for personnel working in their detachment, and it includes detachment information such as a phone number change, change of address, etc.

Input

Item that needs to be changed

Source of Input

Keyboard

Output

Updated record

Destination of output

Updated record page

Processing

The user shall be presented with a list of records for the option chosen and the user shall click on the item that needs to be altered. He/She shall be presented with the record and shall only have permission to change the items location within the detachment.

Requirements

Successful login

Precondition

Successful login and selected item to be changed

Post Condition

The database is altered

Side effects

None

Responses to abnormal behavior

An error message shall be displayed if the command could not be executed.

A.2.2.5.2 Log_Repair

This function is only available with the Vehicle option mentioned above. This function shall allow the user to enter a repair that is needed and approved for a vehicle that is part of the detachment.

Input

VIN Number, Mileage, Repair, Repair Date, Repair Done, Cost, Entered By, Entered On

Source of Input

Keyboard

Output

Display new record

Destination of output

Updated record page

Processing

This function shall extract information entered by the user and add a new record to the Vehicle repair table.

Requirements

Successful login

Precondition

Successful login and VIN of vehicle

Post Condition

The database is altered

Side effects

None

Responses to abnormal behavior

An error message shall be displayed if the command could not be executed.

A.2.2.5.4 Mileage_Update

This function shall allow the user to update the mileage of the detachments vehicles and is only available for the Vehicles option.

Input

VIN

Source of Input

Keyboard

Output

Display vehicle information

Destination of output

Vehicle information page

Processing

This function shall query the Vehicle table and display its information. The user shall only have permission to alter the mileage field. This information shall be written to the database.

Requirements

Successful login

Precondition

Successful login and current mileage entered

Post Condition

Database altered

Side effects

None

Responses to abnormal behavior

An error message shall be displayed if the command could not be executed.

A.2.2.5.4 Check_Oil

This function shall allow the user to check when an oil change is due for the fleet of vehicles of the detachment. This function is only available for with the Vehicle option.

Input

None

Source of Input

None

Output

Display all vehicles within the detachment and the mileage before next oil change.

Destination of output

Display oil change page

Processing

This function shall query the Vehicle Repair table and subtract mileage since last oil change from the current mileage and display that mileage for each vehicle.

Requirements

Successful login

Precondition

Successful login and current mileage entered

Post Condition

None

Side effects

None

Responses to abnormal behavior

An error message shall be displayed if the command could not be executed.

A.2.2.6 User-Level Four Functions

The user-level four users upon successful login shall be presented with a screen with five options. The user can choose from the five following options: Personnel, Personnel Items, Vehicle, Vehicle Items, and General Inventory. After selecting one of the options, the user shall be presented with the following three choices. This user level shall only allow the user to view items and nothing more.

A.2.2.6.2 Inventory

This function shall display all items for the detachment of the user. The inventory list shall depend on the option the user chose from the five options listed above.

Input

None

Source of Input

None

Output

Displays information for item for the detachment in the category chosen.

Destination of output

The inventory screen

Processing

The application shall query the database and display information from the chosen list mentioned above.

Requirements

Successful login

Precondition

None

Post Condition

Data presented to the user

Side effects

None

Responses to abnormal behavior

An error shall be displayed if the database cannot return all of the information for the chosen inventory

A.2.2.6.2 Find

This function shall allow the level four users to view other detachment inventory.

This shall be used in the event that an item needs to be borrowed.

Input

Detachment code

Source of Input

Keyboard

Output

Displays information for items from detachment chosen.

Destination of output

The detachment inventory screen

Processing

The application shall query the database and display information from the chosen detachment and from the options list mentioned above.

Requirements

Successful login

Precondition

None

Post Condition

Data presented to the user

Side effects

None

Responses to abnormal behavior

An error shall be displayed if the database cannot return all of the information for the chosen inventory

A.2.2.6.3 Printer_Friendly

This function shall allow the user to convert the Hyper-Text Markup Language (HTML) into a format that allows the user to print out the report in a format that is easier to read. The data presented on the webpage is a result of a query to the database based on the detachment code.

Input

Inventory List

Source of Input

Query to the database based on detachment code.

Output

Inventory List

Destination of output

Printer friendly page

Processing

This function shall take the information from the successful login page textbox and present it in a printer friendly format.

Requirements

Successful login

Precondition

Successful login and successful detachment query

Post Condition

Data presented in a printer friendly format.

Side effects

None

Responses to abnormal behavior

An error message shall be displayed to the user if the system cannot process the request.

2.2.6.3 Exit

This function shall allow the user to exit the application, returning them to the login screen.

Input

Clicking on the exit link

Source of Input

Mouse click

Output

Returns user to the login screen

Destination of output

The monitor

Processing

Once the Exit link is pressed the program shall stop and return the user to the login screen.

Requirements

Successful login and clicking the Exit link

Precondition

None

Post Condition

The program exits

Side effects

None

Responses to abnormal behavior

None

A.3 Table Descriptions and Overviews

A.3.2 Table Overview

A.3.2.1 User's Table

This table shall hold the user's username and password along with the user's social security number. The social security number shall be used as verification in the event the user forgets his/her password.

A.3.2.2 Personnel Table

This table shall contain the information of all employees that are employed with the West Virginia State Police.

A.3.2.3 Personnel Items Table

This table shall include all items that can be assigned to an employee. For example, a trooper has many items that are assigned to him/her. They include but are not limited to: handcuffs, baton, flashlight, cell phone, and belt.

A.3.2.4 Vehicle Table

This table shall include all information relating to a West Virginia State Police vehicle.

A.3.2.5 Vehicle Item Table

Just as items can be assigned to an employee, items can be assigned to a vehicle.

This table shall contain information related to items that can be assigned to a vehicle. These items include but are not limited to: a light bar, CB radio and laptop computer.

A.3.2.6 Vehicle Repair Table

This table shall store information regarding a vehicles repair history. Also, this table shall track when oil changes are due.

A.3.2.7 Detachment Table

This table shall store information about each detachment that make up the West Virginia State Police Force.

A.3.2.8 Gun Table

This table shall contain information regarding guns. Information includes but is not limited to: general gun information, which gun was issued to whom, who issued the gun, etc.

A.3.2.9 Badge Table

This table shall store information concerning badges and which badge was issued to whom.

A.3.3.10 General Inventory Table

This table shall store all other types of inventory for the West Virginia State Police. The inventory stored in this table shall be mostly Detachment office inventory

A.3.3 Table Definitions

3.3.1 User's Table

This table shall contain the following fields:

UID Code
Username
Password
SSN

UID Code: Uniquely identifies each record

Username: The username of the user

Password: The password of the user

SSN: The social security number of the user. Used for verification methods in the event the user forgets his/her password

Table Constraints

Primary Keys – UID Code, Username

Every field must not be null

A.3.3.2 Personnel Table

This table shall contain the following fields:

ID Code
F Name

L Name
M Initial
Address1
Address2
Address3
City
State
Zip
Phone:
SSN
Rank
Detach Code
Level

ID Code: Uniquely identifies each record

F Name: The first name of the employee

L Name: The last name of the employee

M Initial: The middle initial of the employee

Address1: Street address of employee

Address2: Street address of employee

Address3: Street address of employee

City: City of employee

State: State of employee

Zip: Zip code of employee

Phone: Phone number of employee

SSN: The social security number of the employee

Rank: The rank of the employee

Detach Code: The detachment code of the employee

Level: Denotes the user's level of permissions in the application

Table Constraints

Primary Keys – ID Code

Every field must not be null

A.3.3.3 Personnel Items Table

This table shall include the following fields:

Record ID
Item
Manufacturer
Model
Serial Number
Description
Cost
Date Purchased
Issued To
Issued Date
Issued By

Record ID: Uniquely identifies each record

Item: Item to be issued

Manufacturer: The items manufacturer

Model: Model number/name of the item

Serial Num: Serial number or the item

Description: Description of the item

Cost: Cost of the item

Date Purchased: Date the item was purchased

Issued To: Who the item was issued to

Issue Date: When it was issued

Issued By: Who issued the item

Table Constraints

Primary Key – Record ID

Every field must not be null

A.3.3.4 Vehicle Table

This table contains the following fields:

VIN Number
Manufacturer
Model
Description
Cost
Date Purchased
Entered By
Entered On
Notes

VIN Number: Uniquely identifies each record and is the Vehicle Identification number

Manufacturer: The manufacturer of the vehicle

Model: Model of the vehicle

Description: A brief description of the vehicle

Cost: Purchase price of the vehicle

Date Purchased: The date the vehicle was purchased

Entered By: Who entered the record

Entered On: The date the record was entered

Notes: Added notes concerning the vehicle

Table Constraints

Primary Key – VIN Number

Every field must not be null

A.3.3.5 Vehicle Item Table

This table shall contain the following fields:

Record ID
Item
Manufacturer
Model
Serial Number
Description
Cost
Date Purchased
VIN Number
Entered By
Entered On

Record ID:	Uniquely identifies each record
Item:	Item to be issued
Manufacture:	Manufacturer of the item
Model:	Model name/number of the item
Serial Number:	Serial Number of the item
Description:	Brief description of the item
Cost:	Purchase price of the item
Date Purchased:	Date the item was purchased
VIN Number:	The VIN number of the vehicle where the item resides
Entered By:	Who entered the record
Entered On:	The date the record was entered

Table Constraints

Primary Key – Record ID

Every field must not be null

A.3.3.6 Vehicle Repair Table

This table shall contain the following fields:

Record ID
VIN Number
Mileage
Mileage Since
Oil Date
Repair
Repair Date
Repair Done
Cost
Entered By
Entered On

Record ID:	Uniquely identifies each record
VIN Number:	VIN Number of the vehicle
Mileage:	Current mileage
Mileage Since:	Mileage since the last oil change
Oil Date:	Date of the last oil change
Repair:	The repair that was done to the vehicle
Repair Date:	Date the repair was done
Repair Done:	Repair performed by
Cost:	Cost of the repair
Entered By:	Who entered the record
Entered On:	The date the record was entered

Table Constraints

Primary Key – Record ID

Every field must not be null

A.3.3.7 Detachment Table

This table shall contain the following fields:

Loc Code
Detachment
District
County
Address 1
Address 2
Address 3
City
State
Zip
Phone
Detach Head

Loc Code: The code associated with a detachment

Detachment: The name of the detachment

District: The district where the detachment resides

County: County name of the detachment

Address 1: First line of the address

Address 2: Second line of the address

Address 3: Third line of the address

City: City of the detachment

State: State of the detachment

Zip: Zip code of the detachment

Phone: Phone number of the detachment

Detach Head: The detachment head

Table constraints:

Primary Key – Loc Code

All fields must not be null, except Address 2 and Address 3

A.3.3.8 Gun Table

This table shall contain the following fields:

Gun ID
Gun Type
Gun Manufacturer
Gun Model
Serial Num
Issued To
Issue Date
Issued By
Return Date
Cost
Purchase Date
Entered By
Entered On

Gun ID:	Uniquely identifies each record
Gun Type:	The type of gun
Gun Manufacturer:	The manufacturer of the gun
Gun Model:	The model name of the gun
Serial Num:	The serial number of the gun
Issued To:	The person the gun was issued to
Issue Date:	The date the gun was issued
Issued By:	Who issued the gun
Return Date:	Date the gun was returned
Cost:	Purchase price of the gun
Purchase Date:	Date the gun was purchased
Entered By:	Who entered the record

Entered On: The date the record was entered

Table Constraints

Primary Keys – Gun ID – Serial Num

All fields must not be null

A.3.3.9 Badges

This table shall contain information on badges that were issued to personnel.

It contains the following fields:

ID Code

Serial

Issued To

Issued Date

Return Date

Issued By

ID Code: Uniquely identifies each record

Serial: The serial number of the badge issued

Issued To: Who received the badge

Issued Date: The date the badge was issued

Return Date: The date the badge was returned

Issued By: The person who issued the badge

Table Constraints

Primary Key – ID Code

Every field must not be null

A.3.3.10 General Inventory

This table shall contain information associated with general inventory

This table contains the following fields:

DPS Number
Loc Code
Manufacturer
Model
Serial
Description
Purchase Date
Cost
Entered By
Entered On
Count

DPS Number: The unique number associated with the item

Loc Code: The detachment location where the item shall reside

Manufacturer: The manufacturer of the item

Model: The model of the item

Serial: The serial number of the item

Description: A small description of the item

Cost: The cost of the item

Purchase Date: Date the item was purchased

Entered By: Who entered the record

Entered On: The date the record was entered

Count: This shall track the count of certain items

Table Constraints:

Primary Key – DSP Number

All fields must be not null

4 Description of User Interfaces

The five classes of users have a distinct user interface. These interfaces are broken up into the five levels of users. These interfaces are described in the following sections. The functions in each user class stated in section 2 – except internal functions – shall have a corresponding control within the graphical user interface (GUI) of that user class. Screen shots of a prototype shall soon be completed and presented when finished.

A.4.1 Level Zero User

The level zero users shall have complete access to all function of the application. They shall also have complete access to the database. The interface shall reflect these privileges.

A.4.2 Level One User

The level one user shall have limited access to functions of the application. The interface shall reflect these privileges.

A.4.3 Level Two User

The level two user shall have limited access to functions of the application. The interface shall reflect these privileges.

A.4.4 Level Three User

The level three user shall have limited access to functions of the application. The interface shall reflect these privileges.

A.4.5 Level Four User

The level four user shall have read-only privileges.

A.5 Conclusion

The proposed application shall contain only one database that shall be altered by users that obtain proper permission levels. Users of all levels shall have the ability to read specific data within the database. However, the amount of information that shall be presented to the user shall be limited depending on the user's permission level.

Following this Requirements Document shall be a Test Plan and a Specifications Document. The Test Plan shall describe what shall happen during each function when incorrect and correct data has been entered. The Specifications Document shall aid the future design team in completing this application.

Due to time constraints, this application shall be a prototype for the West Virginia State Police. They shall have the ability to add, delete and change whatever they deem necessary to obtain an application that shall meet their needs fully.

This application shall also incorporate a barcode. At the time this document was completed, the barcode scanner has not been provided.

Appendix B – Prototype Code

B.1 Added

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FFCC00">
<div align="center"><font size="+2">You have been added
  </font>
</div>
</body>
</html>
```

B.2 Add New Item

```
<html>

<Select>
<option></option>
<option>General Inventory</option>
<option>Personnel</option>
<option>Personnel Item</option>
<option>Vehicle</option>
<option>Vehicle Item</option>
</select>
</html>
```

B.3 Add New Item – PHP

```
<?php
  session_start();
  require_once('Functions.php');
  $page1 = "Level0Links.htm";
  $page2 = "NewItemSelection.htm";
  buildframes($page1,$page2);
?>
```

B.4 Badge Enter

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post" action="BadgeEnter.php"
target="_self">

  <p align="center"><strong><font size="+2">Enter the new
Badge Information. All fields are
required.</font></strong></p>
  <table width="45%" border="1" align="center">
    <tr>
      <td><div align="right"><strong>Serial
Number:</strong></div></td>
      <td><input type="text" name="SerNum"></td>
    </tr>
    <tr>
      <td><div align="right"><strong>Issued
To:</strong></div></td>
      <td><input type="text" name="IssTo"></td>
    </tr>
    <tr>
      <td><div align="right"><strong>Issued
By:</strong></div></td>
      <td><input type="text" name="IssBy"></td>
    </tr>
    <tr>
      <td><div align="right"><strong>Issued
Date:</strong></div></td>
      <td><input type="text" name="IssDate">
        <strong>yyyy-mm-dd</strong></td>
    </tr>
    <tr>
      <td><div align="right"><strong>Return
Date:</strong></div></td>
      <td><input type="text" name="RetDate">
        <strong>yyyy-mm-dd</strong></td>
    </tr>
  </table>
```



```

$sql = "insert into
badges (Serial, IssuedTo, IssuedBy, IssuedDate, ReturnDate,
        EnteredBy, EnteredOn) values
('$SerNum', '$IssTo', '$IssBy', '$IssDate', '$RetDate', '$EntB
y', '$EntOn')";

$result = mysql_query($sql, $database) or
die(mysql_error());
        mysql_close($database);
?>
<body bgcolor="#000099" text="#FF9900">
<div align="center">
    <p><strong><font size="+2">The following information
has been </font></strong></p>
    <p><strong><font size="+2">entered to the
        Badges Inventory.</font></strong></p>
    <table width="20%" border="1" cellpadding="3">
        <tr>
            <td align="right">Serial Number:</td>
            <td><?php echo $SerNum?>&nbsp;</td>

        </tr>
        <tr>
            <td align="right">Issued To:</td>
            <td><?php echo $IssTo?>&nbsp;</td>

        </tr>
        <tr>
            <td align="right">Issued By:</td>
            <td><?php echo $IssBy?>&nbsp;</td>

        </tr>
        <tr>
            <td align="right">Issued Date:</td>
            <td><?php echo $IssDate?>&nbsp;</td>

        </tr>
        <tr>
            <td align="right">Return Date:</td>
            <td><?php echo $RetDate?>&nbsp;</td>
        </tr>
        <tr>
            <td align="right">Entered By:</td>
            <td><?php echo $EntBy?>&nbsp;</td>

        </tr>
        <tr>

```

```

        <td align="right">Entered On:</td>
        <td><?php echo $EntOn?>&nbsp;</td>

    </tr>

</table>
<p>&nbsp;</p>
</div>

```

B.6 DelItem - PHP

```

<?php
    session_start();
    require_once('Functions.php');
    $page1 = "Level0Links.htm";
    $page2 = "DelItemSelection.htm";
    buildframes($page1,$page2);
?>

```

B.7 DelItemSelection

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Delete Item Select</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FF9900">
<p></p>
<p></p>

<form name="form1" method="post"
action="DelItemEnter.php" target="_top">
<p align="center">
<Select name="Select">
<option></option>
<option>generalinventory</option>
<option>personnel</option>
<option>personnelitem</option>
<option>vehicle</option>
<option>vehicleitem</option>

```

```

<option>gun</option>
<option>badges</option>
</select>
<p align="center">
<input type="submit" name="Submit" value="Select">
</p></p></form>
<p align="center">&nbsp;</p>
</body>
</html>

```

B.8 Enter – PHP

```

<?php
session_start();
require_once('Functions.php');
$Username = $_POST['Username'];
$Password = $_POST['Password'];

$_SESSION['USER'] = $Username;

$databse = mysql_connect("localhost","bamb","senior1");
mysql_select_db("WVSPITS");
if(!$databse)
{
    echo mysql_error();
    exit;
}

$sql = "SELECT * FROM users where username = '$Username'
and password = '$Password'";

$results = mysql_query($sql, $databse);
$newArray = mysql_fetch_array($results);
$level = $newArray['userlevel'];
$fname = $newArray['fname'];
$number_of_rows = mysql_num_rows($results);
$_SESSION['LEVEL'] = $level;
$_SESSION['FNAME'] = $fname;
if($number_of_rows > 0)
{
    if($level == 0)
    {
        $page1 = "Level0Links.htm";
        $page2 = "Welcome.php";
        buildframes($page1,$page2);
    }
}

```



```

    }
    if($level == 1)
    {
        $page1 = "Level1Links.htm";
        $page2 = "Welcome.php";
        buildframes($page1,$page2);
    }
    if($level == 2)
    {
        $page1 = "Level2Links.htm";
        $page2 = "Welcome.php";
        buildframes($page1,$page2);
    }
    if($level == 3)
    {
        $page1 = "Level3Links.htm";
        $page2 = "Welcome.php";
        buildframes($page1,$page2);
    }
    if($level == 4)
    {
        $page1 = "Level4Links.htm";
        $page2 = "Welcome.php";
        buildframes($page1,$page2);
    }
}
else
{
    echo mysql_error();
    exit;
}
mysql_free_result($results);
mysql_close($database);
?>

```

B.9 EnterGI

```

<?php
session_start();
require_once('Functions.php');
$DPS          = $_POST['DPS'];
$LocCode      = $_POST['LocCode'];
$Manufacturer  = $_POST['Manufacturer'];
$Model        = $_POST['Model'];
$SerNum       = $_POST['SerNum'];

```

```

$Desc      = $_POST['Desc'];
$Cost      = $_POST['Cost'];
$PurDate   = $_POST['PurDate'];
$EntBy     = $_POST['EntBy'];
$EntOn     = $_POST['EntOn'];
$Count     = $_POST['Count'];

$database = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");
if(!$database)
{
    echo mysql_error();
    exit;
}
$sql = "insert into
generalinventory(DPS,LocCode,Manufacturer,Model,Serial,
Description,PurchaseDate,Cost,EnteredBy,EnteredOn,Count)
values
('$DPS','$LocCode','$Manufacturer','$Model','$SerNum','$D
esc','$PurDate','$Cost','$EntBy','$EntOn','$Count')";
$result = mysql_query($sql, $database) or
die(mysql_error());
mysql_close($database);
?>
<body bgcolor="#000099" text="#FF9900">
<div align="center">
    <p><strong><font size="+2">The following information
has been </font></strong></p>
    <p><strong><font size="+2">entered to the
        general inventory.</font></strong></p>
    <table width="15%" border="1" cellpadding="3">
        <tr>
            <td>DPS:</td>
            <td><?php echo $DPS ?>&nbsp;</td>

        </tr>
        <tr>
            <td>Location Code:</td>
            <td><?php echo $LocCode ?>&nbsp;</td>

        </tr>
        <tr>
            <td>Manufacturer:</td>
            <td><?php echo $Manufacturer ?>&nbsp;</td>

```

```

</tr>
<tr>
  <td>Model:</td>
  <td><?php echo $Model?>&nbsp;</td>

</tr>
<tr>
  <td>Serial Number:</td>
  <td><?php echo $SerNum?>&nbsp;</td>

</tr>
<tr>
  <td>Description:</td>
  <td><?php echo $Desc?>&nbsp;</td>

</tr>
<tr>
  <td>Purchase Date:</td>
  <td><?php echo $PurDate?>&nbsp;</td>

</tr>
  <tr>
    <td>Cost:</td>
    <td><?php echo $Cost?>&nbsp;</td>

</tr>

<tr>
  <td>Entered By:</td>
  <td><?php echo $EntBy?>&nbsp;</td>

</tr>
<tr>
  <td>Entered On:</td>
  <td><?php echo $EntOn?>&nbsp;</td>

</tr>
<tr>
  <td>Count:</td>
  <td><?php echo $Count?>&nbsp;</td>

</tr>

</table>
<p>&nbsp;</p>
</div>

```

B.10 EnterGun

```
<?php
session_start();
require_once('Functions.php');
$GunType      = $_POST['Type'];
$Manufacturer  = $_POST['Manufacturer'];
$Model        = $_POST['Model'];
$SerNum       = $_POST['SerNum'];
$IssTo        = $_POST['IssTo'];
$IssBy        = $_POST['IssBy'];
$IssDate      = $_POST['IssDate'];
$RetDate      = $_POST['RetDate'];
$Cost         = $_POST['Cost'];
$PurDate      = $_POST['PurDate'];
$EntBy        = $_POST['EntBy'];
$EntOn        = $_POST['EntOn'];

$database = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");

if(!$database)
{
    echo mysql_error();
    exit;
}

$sql = "insert into gun
(GunType,Manufacturer,Model,SerNum,IssuedTo,IssuedBy,
IssuedDate,Cost,PurchaseDate,EnteredBy,EnteredOn)
values
('$GunType','$Manufacturer','$Model','$SerNum','$IssTo','
$IssBy','$IssDate','$Cost','$PurDate','$EntBy','$EntOn') "
;

$result = mysql_query($sql, $database) or
die(mysql_error());
mysql_close($database);
?>
<body bgcolor="#000099" text="#FF9900">
<div align="center">
    <p><strong><font size="+2">The following information
has been entered into the database.</font></strong></p>
    <table width="15%" border="1" cellpadding="3">
        <tr>
            <td><?php echo $GunType ?>&nbsp;  </td>
        </tr>
    </table>

```



```

$_SESSION['User']      = $Username;
$_SESSION['FName']     = $FName;
$_SESSION['SSN']       = $SSN;
$_SESSION['PASS']      = $Pass;
$_SESSION['PASS1']     = $Pass1;

if($Pass == $Pass1)
{
    $database = mysql_connect("localhost","bamb",
    "senior1");

    mysql_select_db("WVSPITS");
    if(!$database)
    {
        echo mysql_error();
        exit;
    }

    $sql = "insert into users (username,password,fname,
    ssn, userlevel)
    values
    ('$Username','$Pass','$FName','$SSN', 6)";

    $result = mysql_query($sql, $database) or
    die(mysql_error());
    if($result)
    {
        $page1 = "Level4Links.htm";
        $page2 = "Added.htm";
        buildframes($page1, $page2);
    }
}

mysql_close($database);
?>

```

B.12 Exit

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">

```

```

</head>

<body bgcolor="#000099" text="#FF9900">
<div align="center">
  <p>&nbsp;</p>
  <p>&nbsp;</p>
  <p>&nbsp;</p>
  <p><strong><font size="+3">It is safe to disconnect
from the VPN.
  </font> </strong>
  </p>
</div>
</body>
</html>

```

B.13 Exit-PHP

```

<?php
session_start();
require_once('Functions.php');
$page1 = "tools.htm";
$page2 = "Exit.htm";
buildframes($page1,$page2);
?>

```

B.14 Frames – PHP

```

<?php
require_once('Functions.php');
$page1 = '';
$page2 = '';
buildframes($page1, $page2);
?>

```

B.15 Functions – PHP

```

<?php
session_start();
function buildframes($page1, $page2)
{
  if($page1 == '' && $page2 == '')

```

```

    {
?>
<frameset rows="97,*" cols="*" framespacing="0"
frameborder="yes" border="0" bordercolor="#666666">

    <frame src="TitleFrame.htm" name="topFrame"
    scrolling="NO" noresize >

    <frameset rows="*" cols="123,*" framespacing="0"
    frameborder="yes" border="0" bordercolor="#666666">

    <frame src= tools.htm name="leftFrame" scrolling="NO"
    noresize>

    <frame src= Main.htm name="mainFrame">
    </frameset>
</frameset><noframes></noframes>
<?php
    }
    else
    {
?>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">

</head>
<frameset rows="97,*" cols="*" framespacing="0"
frameborder="yes" border="2" bordercolor="#666666">

<frame src="TitleFrame.htm" name="topFrame"
scrolling="NO" noresize >

<frameset rows="*" cols="150,*" framespacing="0"
frameborder="yes" border="2" bordercolor="#666666">

<?php    echo "<frame src= $page1 name= 'leftFrame'
scrolling= 'YES' noresize>

<frame src= $page2 name='mainFrame'>
</frameset>
</frameset>"; ?>
<noframes><body>
</body></noframes>
</html>

```



```

<?php
    }
}
?>

```

B.16 GIEnter - PHP

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>General Inventory Enter</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FF9900">
<div align="center">
  <p><font size="+2"><strong>Enter the General Inventory
Item Information</strong></font></p>
  <form name="form1" method="post" action="EnterGI.php">
  <table width="56%" border="0">
    <tr>
      <td width="34%"><div align="right">DPS:</div></td>
      <td width="44%"><div align="center">
        <input type="text" name="DPS">
      </div></td>
      <td width="22%">&nbsp;</td>
    </tr>
    <tr>
      <td><div align="right">Location Code:</div></td>
      <td><div align="center">
        <input type="text" name="LocCode">
      </div></td>
      <td>&nbsp;</td>
    </tr>
    <tr>
      <td><div align="right">Manufacturer:</div></td>
      <td><div align="center">
        <input type="text" name="Manufacturer">
      </div></td>
      <td>&nbsp;</td>
    </tr>
    <tr>
      <td><div align="right">Model:</div></td>

```

```

        <td><div align="center">
            <input type="text" name="Model">
        </div></td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td><div align="right">Serial:</div></td>
        <td><div align="center">
            <input type="text" name="SerNum">
        </div></td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td><div align="right">Description:</div></td>
        <td><div align="center">
            <textarea name="Desc"></textarea>
        </div></td>
        <td>256 characters</td>
    </tr>
    <tr>
        <td><div align="right">Purchase Date:</div></td>
        <td><div align="center">
            <input type="text" name="PurDate">
        </div></td>
        <td>yyyy-mm-dd</td>
    </tr>
    <tr>
        <td><div align="right">Cost:</div></td>
        <td><div align="center">
            <input type="text" name="Cost">
        </div></td>
        <td>111.00</td>
    </tr>
    <tr>
        <td><div align="right">Entered By:</div></td>
        <td><div align="center">
            <input type="text" name="EntBy">
        </div></td>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td><div align="right">Entered On:</div></td>
        <td><div align="center">
            <input type="text" name="EntOn">
        </div></td>
        <td>yyyy-mm-dd</td>
    </tr>

```

```
|  |  |
| --- | --- |
| Count: |  |
|  |  |

```

B.17 GunEnter

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Gun Enter</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post" action="EnterGun.php"
target="_self">
<p align="center"><strong><font size="+2">Enter the new
gun information. All fields are
required.</font></strong></p>
<table width="45%" border="1" align="center">
|  |  |
| --- | --- |
| Gun Type: |  |

```

```
|  |
| --- |
| Manufacturer: |
| Model: |
| Serial Number: |
| Issued To: |
| Issued By: |
| Issued Date:   yyyy-mm-dd |
| Cost:   ex. 111.00 |
| Purchase Date:   yyyy-mm-dd |
| Entered By: |
| Entered On:   yyyy-mm-dd |
|  |

```

```

        <td><input type="submit" name="Submit"
value="Submit">&nbsp;&nbsp; <input type="reset"
name="Submit2" value="Reset"></td>
    </tr>
</table>
</form>
</body>
</html>

```

B.18 ItemEnter - PHP

```

<?php
session_start();
require_once('Functions.php');
$Selection = $_POST[Select];
if($Selection == "Gun")
{
    $page1 = "Level0Links.htm";
    $page2 = "GunEnter.htm";
    buildframes($page1,$page2);
}
if($Selection == "Badges")
{
    $page1 = "Level0Links.htm";
    $page2 = "BadgeEnter.htm";
    buildframes($page1,$page2);
}
if($Selection == "General Inventory")
{
    $page1 = "Level0Links.htm";
    $page2 = "GIEnter.htm";
    buildframes($page1,$page2);
}
if($Selection == "Personnel")
{
    $page1 = "Level0Links.htm";
    $page2 = "PerEnter.htm";
    buildframes($page1,$page2);
}
if($Selection == "Personnel Item")
{
    $page1 = "Level0Links.htm";
    $page2 = "PerItemEnter.htm";
    buildframes($page1,$page2);
}

```

```

if($Selection == "Vehicle")
{
    $page1 = "Level0Links.htm";
    $page2 = "VehEnter.htm";
    buildframes($page1,$page2);
}
if($Selection == "Vehicle Item")
{
    $page1 = "Level0Links.htm";
    $page2 = "VehItemEnter.htm";
    buildframes($page1,$page2);
}
?>

```

B.19 Level0Links

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FF9900" >
<form name="form1" method="post" action="AddNewItem.php"
target="_top">
    <p>

        <input name="Add" type="submit" id="Add" value="Add
New Item">
    </p></form>
    <form name="form2" method="post" action="DelItem.php"
target="_top"> <p>
        <input name="Delete" type="submit" id="Delete"
value="Delete Item">
    </p></form>
    <p>    <input name="Backup" type="submit" id="Backup"
value="Backup DB">
    </p>
    <p>
        <form name="form3" method="post" action =
"LogisticsInvent.php" target="_top">

```

```

        <input name="Current" type="submit" id="Current"
value="Current Inventory">
    </form>
</p>
<p>
    <form name="form4" method="post" action =
"UpdateCountSelectFrames.php" target="_top">
        <input name="UpdateInven" type="submit"
id="UpdateInven" value="Update Inventory">
    </form>
</p>
<p>
    <input name="RepairReport" type="submit"
id="RepairReport" value="Repair Report">
</p>
<p>
    <input name="LogRepair" type="submit" id="LogRepair"
value="Log Repair">
</p>
<p>
    <input name="Assign" type="submit" id="Assign"
value="Assign">
</p>
<p>
    <input name="Assigned" type="submit" id="Assigned"
value="Assigned Items">
</p>
<p>    <input name="Available" type="submit"
id="Available" value="Available Items">
</p>
<p>
    <form name="form4" method="post" action =
"PerInvFrames.php" target="_top">
        <input name="Personnel" type="submit" id="Personnel"
value="Personnel Inventory">
    </form>
</p>
<p>
    <input name="Vehicle" type="submit" id="Vehicle"
value="Vehicle Inventory">
</p>
<p>
    <input name="ChangeLoc" type="submit" id="ChangeLoc"
value="Change Location">
</p>
<p>    <input name="Back" type="submit" id="Back"
value="Change Back">

```

```

    </p>
    <p>
        <input name="Update" type="submit" id="Update"
value="Update Loc">
    </p>
    <p>
        <input name="Log Repair" type="submit" id="Log
Repair" value="Log Repair">
    </p>
    <p>
        <input name="Update Mileage" type="submit" id="Update
Mileage" value="Update Mileage">
    </p>
    <p>
        <input name="Check Oil" type="submit" id="Check Oil"
value="Check Oil">
    </p>
    <p>
        <input name="Inventory" type="submit" id="Inventory"
value="Inventory">
    </p>
    <p>
        <input name="Find" type="submit" id="Find"
value="Find">
    </p>
    <p>
        <input name="Printer" type="submit" id="Printer"
value="Printer">
    </p>
    <p>
        <form name="Exit" method="post" action="Exit.php"
target="_top">
            <input name="Exit" type="submit" id="Exit"
value="Exit">
        </form>
    </p>

    <p>&nbsp;</p>

</body>
</html>

```


B.20 Level1Links

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post" action="">
  <p>
    <input name="Current" type="submit" id="Current"
value="Current Inventory">
  </p>
  <p>
    <input name="UpdateInven" type="submit"
id="UpdateInven" value="Update Inventory">
  </p>
  <p>
    <input name="RepairReport" type="submit"
id="RepairReport" value="Repair Report">
  </p>
  <p>
    <input name="LogRepair" type="submit" id="LogRepair"
value="Log Repair">
  </p>
  <p>
    <input name="Assign" type="submit" id="Assign"
value="Assign">
  </p>
  <p>
    <input name="Assigned" type="submit" id="Assigned"
value="Assigned Items">
  </p>
  <p>
    <input name="Available" type="submit"
id="Available" value="Available Items">
  </p>
  <p>
    <input name="Personell" type="submit" id="Personell"
value="Personnel Inventory">
  </p>
  <p>
    <input name="Vehicle" type="submit" id="Vehicle"
value="Vehicle Inventory">
  </p>
</form>
</body>
</html>
```

```

    </p>
    <p>
        <input name="ChangeLoc" type="submit" id="ChangeLoc"
value="Change Location">
    </p>
    <p>
        <input name="Back" type="submit" id="Back"
value="Change Back">
    </p>
    <p>
        <input name="Update" type="submit" id="Update"
value="Update Loc">
    </p>
    <p>
        <input name="Log Repair" type="submit" id="Log
Repair" value="Log Repair">
    </p>
    <p>
        <input name="Update Mileage" type="submit" id="Update
Mileage" value="Update Mileage">
    </p>
    <p>
        <input name="Check Oil" type="submit" id="Check Oil"
value="Check Oil">
    </p>
    <p>
        <input name="Inventory" type="submit" id="Inventory"
value="Inventory">
    </p>
    <p>
        <input name="Find" type="submit" id="Find"
value="Find">
    </p>
    <p>
        <input name="Printer" type="submit" id="Printer"
value="Printer">
    </p>
    <p>
        <input name="Exit" type="submit" id="Exit"
value="Exit">
    </p>
</form>
</body>
</html>

```

B.21 Level2Links

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post" action="">
  <p>
    <input name="Assign" type="submit" id="Assign"
value="Assign">
  </p>
  <p>
    <input name="Assigned" type="submit" id="Assigned"
value="Assigned Items">
  </p>
  <p>    <input name="Available" type="submit"
id="Available" value="Available Items">
  </p>
  <p>
    <input name="Personell" type="submit" id="Personell"
value="Personnel Inventory">
  </p>
  <p>
    <input name="Vehicle" type="submit" id="Vehicle"
value="Vehicle Inventory">
  </p>
  <p>
    <input name="ChangeLoc" type="submit" id="ChangeLoc"
value="Change Location">
  </p>
  <p>    <input name="Back" type="submit" id="Back"
value="Change Back">
  </p>
  <p>
    <input name="Update" type="submit" id="Update"
value="Update Loc">
  </p>
  <p>
    <input name="Log Repair" type="submit" id="Log
Repair" value="Log Repair">
  </p>
```

```

    <p>
      <input name="Update Mileage" type="submit" id="Update
Mileage" value="Update Mileage">
    </p>
    <p>
      <input name="Check Oil" type="submit" id="Check Oil"
value="Check Oil">
    </p>
    <p>
      <input name="Inventory" type="submit" id="Inventory"
value="Inventory">
    </p>
    <p>
      <input name="Find" type="submit" id="Find"
value="Find">
    </p>
    <p>
      <input name="Printer" type="submit" id="Printer"
value="Printer">
    </p>
    <p>      <input name="Exit" type="submit" id="Exit"
value="Exit">
    </p>
  </form>
</body>
</html>

```

B.22 Level3Links

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post" action="">
  <p>
    <input name="Update" type="submit" id="Update"
value="Update Loc">
  </p>
  <p>

```

```

        <input name="Log Repair" type="submit" id="Log
Repair" value="Log Repair">
    </p>
    <p>
        <input name="Update Mileage" type="submit" id="Update
Mileage" value="Update Mileage">
    </p>
    <p>
        <input name="Check Oil" type="submit" id="Check Oil"
value="Check Oil">
    </p>
    <p>
        <input name="Inventory" type="submit" id="Inventory"
value="Inventory">
    </p>
    <p>
        <input name="Find" type="submit" id="Find"
value="Find">
    </p>
    <p>
        <input name="Printer" type="submit" id="Printer"
value="Printer">
    </p>
    <p>        <input name="Exit" type="submit" id="Exit"
value="Exit">
    </p>
</form>
</body>
</html>

```

B.23 Level4Links

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>

<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post" action="">
    <p>

```

```

        <input name="Inventory" type="submit" id="Inventory"
value="Inventory">
</p>
    <p>
        <input name="Find" type="submit" id="Find"
value="Find">
</p>
    <p>
        <input name="Printer" type="submit" id="Printer"
value="Printer">
</p>
    <p>        <input name="Exit" type="submit" id="Exit"
value="Exit">
</p>
</form>
</body>
</html>

```

B.24 LogInventDisplay - PHP

```

<?php
$database = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");
if(!$database)
{
    echo mysql_error();
    exit;
}
?>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<p>&nbsp;</p>
<p align="center"><strong><font size="+2">Current
Inventory for the Logistics Office</font></strong></p>
<table width="100%" border="1">
    <tr align="center" bgcolor="#333333">
        <td>&nbsp;<DPS</td>

```

```

        <td>&nbsp;Loc Code</td>
        <td>&nbsp;Manufacturer</td>
        <td>&nbsp;Model</td>
        <td>&nbsp;Serial Num</td>
        <td>&nbsp;Description</td>
        <td>&nbsp;Purchase Date</td>
        <td>&nbsp;Cost</td>
        <td>&nbsp;Entered By</td>
        <td>&nbsp;Entered On</td>
        <td>&nbsp;Count</td>
    </tr>
<?php
$sql = "SELECT * FROM generalinventory where LocCode =
666";

$result = mysql_query($sql, $database) or
die(mysql_error());
while ($newArray = mysql_fetch_array($result))
{
    echo "<tr align = 'center'>";echo "<td>";echo
    $newArray['DPS'];echo "</td>";

    echo "<td>";echo $newArray['LocCode'];echo "</td>";
    echo "<td>";echo $newArray['Manufacturer'];
    echo "</td>";
    echo "<td>";echo $newArray['Model'];echo "</td>";
    echo "<td>";echo $newArray['Serial'];echo "</td>";
    echo "<td>";echo $newArray['Description'];
    echo "</td>";
    echo "<td>";echo $newArray['PurchaseDate'];
    echo "</td>";
    echo "<td>";echo $newArray['Cost'];echo "</td>";
    echo "<td>";echo $newArray['EnteredBy'];
    echo "</td>";echo "<td>";
    echo $newArray['EnteredOn'];echo "</td>";
    echo "<td>";echo $newArray['Count'];
    echo "</td></tr>";
}

mysql_close($database);
?>
</table>
</body>
</html>

```

B.25 LogisticsInvent - PHP

```
<?php
session_start();
require_once('Functions.php');
$page1 = "Level0Links.htm";
$page2 = "LogInventDisplay.php";
buildframes($page1,$page2);
?>
```

B.26 Main

```
<title>Login</title><body bgcolor="#000099"
text="#FF9900">
<p align="center"><font size="+3"><strong>Welcome to the
West Virginia State Police</strong></font></p>
<p align="center"><strong><font size="+3">Inventory
Tracking System </font></strong></p>
<p align="center"><strong><font size="+3">WVSP-
ITS</font></strong></p>
<form name="form1" method="post" action="Enter.php"
target="_top">

<table border="0" align="center">
<tr>
  <td><div align="left"><strong>Username:
  </strong></div></td>

  <td><div align="center"><input type="text"
  name="Username"></div></td>
</tr>
<tr>
  <td><div align="right"><strong>Password:
  </strong></div></td>
  <td><div align="center"></div><input type="Password"
  name="Password"></div></td>
</tr>
</table>
<p align="center">
<input type="submit" name="Submit" value="Enter">
<input name="Cancel" type="reset" id="Cancel"
value="Cancel">
</p>
</form>
```



```

<form name="form1" method="post"
action="NewUserSetUp.php" target="_top">
<p align="center"><strong><font size="+3">
<input name="NewUser" type="submit" id="NewUser"
value="New User">
</font></strong></p>
</form>

```

B.27 MainFrame - PHP

```

<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<frameset rows="97,*" cols="*" framespacing="0"
frameborder="yes" border="0" bordercolor="#666666">
<frame src="TitleFrame.htm" name="topFrame"
scrolling="NO" noresize >
<frameset rows="*" cols="123,*" framespacing="0"
frameborder="yes" border="0" bordercolor="#666666">
<frame src= tools.htm name="leftFrame" scrolling="NO"
noresize>
<frame src= Main.htm name="mainFrame">
</frameset>
</frameset><noframes></noframes>
<noframes><body>

```

B.28 NewItemSelection

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<p></p>
<p align="center"><strong><font size="+2">Select what
type of item you would like to enter.</font></strong></p>

```

```

<form name="form1" method="post" action="ItemEnter.php"
target="_top">
<p align="center">
<Select name="Select">
<option></option>
<option>General Inventory</option>
<option>Personnel</option>
<option>Personnel Item</option>
<option>Vehicle</option>
<option>Vehicle Item</option>
<option>Gun</option>
<option>Badges</option>
</select>
<p align="center">
<input type="submit" name="Submit" value="Select">
</p></p></form>
<p align="center">&nbsp;</p>
</body>
</html>

```

B.29 NewUser

```

!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>New User</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<p align="center"><font size="+2">Please enter the
following data</font></p>
  <form name="form1" method="post"
action="EnterNewUser.php">
    <table border="0" align="center">
      <tr>
        <td><div align="Right"> First Name:</div></td>
        <td><div align="center"><input name="FName"
type="text" size="30">
</div></td>
        <td><div align="left"> (upto 20 char)</div></td>
      </tr>
      <tr>

```



```

$page1 = "tools.htm";
$page2 = "NewUser.php";
buildframes($page1,$page2);
?>

```

B.31 PerEnter

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post" action="PerEnter.php"
target="_self">
<p align="center"><strong><font size="+2">Enter the new
Employee's Information.</font></strong></p>
  <table width="70%" border="1" align="center">
    <tr>
      <td><div align="right"><strong>*First
      Name:</strong></div></td>
      <td><em><strong>
        <input type="text" name="FName">
        (John)</strong></em></td>
    </tr>
    <tr>
      <td><div align="right"><strong>*Last
      Name:</strong></div></td>
      <td><em><strong>
        <input type="text" name="LName">
        (Smith)</strong></em></td>
    </tr>
    <tr>
      <td><div align="right"><strong>*M
      Initial:</strong></div></td>
      <td><em><strong>
        <input type="text" name="MInt">
        (M) </strong> </em></td>
    </tr>
    <tr>
      <td><div align="right"><strong>*Address
      1:</strong></div></td>

```

```

        <td><em><strong>
            <input type="text" name="Addy1">
                (123 Elm Street)</strong></em></td>
    </tr>
    <tr>
        <td><div align="right"><strong>Address
            2:</strong></div></td>
        <td><em><strong>
            <input type="text" name="Addy2">
                </strong></em></td>
    </tr>
    <tr>
        <td><div align="right"><strong>Address
            3:</strong></div></td>
        <td><em><strong>
            <input type="text" name="Addy3">
                </strong></em></td>
    </tr>
    <tr>
        <td><div
            align="right"><strong>*City:</strong></div></td>
        <td><em><strong>
            <input type="text" name="City">
                (Morgantown)</strong></em></td>
    </tr>
    <tr>
        <td>
            <div align="right"><strong>*State:</strong></div>
        </td>
        <td><em><strong>
            <input type="text" name="State">
                (WV)</strong></em></td>
    </tr>
    <tr>
        <td>
            <div align="right"><strong>*Zip:</strong></div>
        </td>
        <td><em><strong>
            <input type="text" name="Zip">
                (12345)</strong></em></td>
    </tr>
    <tr>
        <td>
            <div
                align="right"><strong>*Phone:</strong></div>
        </td>
        <td><em><strong>
            <input type="text" name="Phone">
                (3045551234)</strong></em></td>
    </tr>

```


B.32 PerEnter – PHP

```
<?php
session_start();
require_once('Functions.php');

$FName      = $_POST['FName'];
$LName      = $_POST['LName'];
$MInt       = $_POST['MInt'];
$Addy1      = $_POST['Addy1'];
$Addy2      = $_POST['Addy2'];
$Addy3      = $_POST['Addy3'];
$City       = $_POST['City'];
$State      = $_POST['State'];
$Zip        = $_POST['Zip'];
$Phone      = $_POST['Phone'];
$SSN        = $_POST['SSN'];
$Rank       = $_POST['Rank'];
$DetCode    = $_POST['DetCode'];
$Level      = $_POST['Level'];

$database = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");
if(!$database)
{
    echo mysql_error();
    exit;
}

$sql = "insert into
personnel(FName,LName,MInitial,Addy1,Addy2,Addy3,City
,State,Zip,Phone,SSN,Rank,DetachCode,Level)

values
('$FName','$LName','$MInt','$Addy1','$Addy2','$Addy3',
'$City','$State','$Zip',$Phone','$SSN','$Rank','$DetCode'
,'$Level')";

$result = mysql_query($sql, $database) or
die(mysql_error());
mysql_close($database);
?>
<body bgcolor="#000099" text="#FF9900">
<div align="center">
    <p><strong><font size="+2">The following Employee  has
    been </font></strong></p>
    <p><strong><font size="+2">entered to the
```

```

Personnel Inventory.</font></strong></p>
<table width="30%" border="1" cellpadding="3">
  <tr>
    <td align="right">First Name:</td>
    <td><?php echo $FName?>&nbsp;</td>

  </tr>
  <tr>
    <td align="right">Last Name:</td>
    <td><?php echo $LName?>&nbsp;</td>

  </tr>
  <tr>
    <td align="right">M Initial:</td>
    <td><?php echo $MInt?>&nbsp;</td>

  </tr>
  <tr>
    <td align="right">Address 1:</td>
    <td><?php echo $Addy1?>&nbsp;</td>

  </tr>
  <tr>
    <td align="right">Address 2:</td>
    <td><?php echo $Addy2?>&nbsp;</td>
  </tr>
  <tr>
    <td align="right">Address 3:</td>
    <td><?php echo $Addy3?>&nbsp;</td>

  </tr>
  <tr>
    <td align="right">City:</td>
    <td><?php echo $City?>&nbsp;</td>

  </tr>
  <tr>
    <td align="right">State:</td>
    <td><?php echo $State?>&nbsp;</td>

  </tr>
  <tr>
    <td align="right">Zip:</td>
    <td><?php echo $Zip?>&nbsp;</td>

  </tr>
</tr>

```



```

<td align="right">Phone:</td>
    <td><?php echo $Phone?>&nbsp;</td>
</tr>
<tr>
    <td align="right">SSN:</td>
    <td><?php echo $SSN?>&nbsp;</td>

</tr>
<tr>
    <td align="right">Rank:</td>
    <td><?php echo $Rank?>&nbsp;</td>

</tr>
<tr>
    <td align="right">Detachment Code:</td>
    <td><?php echo $DetCode?>&nbsp;</td>

</tr>
<tr>
    <td align="right">User Level:</td>
    <td><?php echo $Level?>&nbsp;</td>

</tr>

</table>
<p>&nbsp;</p>
</div>

```

B.33 PerlInvDisplay – PHP

```

<?php
session_start();
$Selection = $_POST[Select];
$databse = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");
if(!$databse)
{
    echo mysql_error();
    exit;
}
if($Selection != "All Employees")
{
    $pieces = explode(' ', $Selection);
    $Name = $pieces[1].' '.$pieces[0].' '.$pieces[2];
}

```

```
$sql = "SELECT * FROM personnelitems where IssuedTo =
'$Name'";
```

```
$result = mysql_query($sql, $database) or
die(mysql_error());
```

```
?>
```

```
<body bgcolor="#000099" text="#FF9900">
<p align="center"><strong><font size="+2">Personnel
Inventory Assigned to</font></strong></p>
<p align="center"><strong><font size="+3">
```

```
<?php echo $Selection ?>
```

```
&nbsp;  </font></strong></p>
```

```

    <table width="100%" border="1">
    <tr align="center" bgcolor="#333333">
    <td>&nbsp;  Item</td>
    <td>&nbsp;  Manufacturer</td>
    <td>&nbsp;  Model</td>
    <td>&nbsp;  Serial Num</td>
    <td>&nbsp;  Description</td>
    <td>&nbsp;  Cost</td>
    <td>&nbsp;  Purchase Date</td>
    <td>&nbsp;  Issued To</td>
    <td>&nbsp;  Issued Date</td>
    <td>&nbsp;  Issued By</td>
    </tr>
```

```
<?php
```

```

    while ($newArray = mysql_fetch_array($result))
    {
        echo "<tr align = 'center'>";
        echo "<td>";echo $newArray['Item'];
        echo "</td>";
        echo "<td>";echo $newArray['Manufacturer'];
        echo "</td>";
        echo "<td>";echo $newArray['Model'];
        echo "</td>";
        echo "<td>";echo $newArray['SerNum'];
        echo "</td>";
        echo "<td>";echo $newArray['Description'];
        echo "</td>";
        echo "<td>";echo $newArray['Cost'];
        echo "</td>";
        echo "<td>";echo $newArray['DatePur'];
        echo "</td>";
        echo "<td>";echo $newArray['IssuedTo'];
```

```

        echo "</td>";echo "<td>";
        echo $newArray['IssuedDate'];echo "</td>";
        echo "<td>";echo $newArray['IssuedBy'];
        echo "</td>";echo "</tr>";
    }
    ?>
</table>
<?php
mysql_close($database);
}
else
{
    $sql1 = "SELECT FName, LName, MInitial from
    personnel";
    $result1 = mysql_query($sql1, $database) or
    die(mysql_error());

    while($newArray1 = mysql_fetch_array($result1))
    {
        $Name = $newArray1['FName']." ".
        $newArray1['LName']." ".$newArray1['MInitial'];
        $sql = "SELECT * from personnelitems where
        IssuedTo = '$Name'";

        $result = mysql_query($sql, $database) or
        die(mysql_error());
        $row = mysql_num_rows($result);
        if($row > 0)
        {
            echo "In the If Statement",$row;
        }
    }
    ?>

    <body bgcolor="#000099" text="#FF9900">
    <p align="center"><strong><font
    size="+2">Personnel Inventory Assigned
    to</font></strong></p>
    <p align="center"><strong><font size="+3">
    <?php echo $Name ?>
    &nbsp;</font></strong></p>
    <table width='100%' border='1'>
        <tr align='center' bgcolor='#333333'>
            <td>&nbsp;<strong>Item</strong></td>
            <td>&nbsp;<strong>Manufacturer</strong></td>
            <td>&nbsp;<strong>Model</strong></td>
            <td>&nbsp;<strong>Serial Num</strong></td>
            <td>&nbsp;<strong>Description</strong></td>
            <td>&nbsp;<strong>Cost</strong></td>
            <td>&nbsp;<strong>Purchase Date</strong></td>

```

```

        <td>&nbsp;Issued Date</td>
        <td>&nbsp;Issued By</td>
    </tr>
<?php
    while ($newArray = mysql_fetch_array($result))
    {
        echo "Second While";
        echo "<tr align = 'center'>";
        echo "<td>";echo $newArray['Item'];
        echo "</td>";echo "<td>";
        echo $newArray['Manufacturer'];echo "</td>";
        echo "<td>";echo $newArray['Model'];
        echo "</td>";
        echo "<td>";echo $newArray['SerNum'];
        echo "</td>";
        echo "<td>";
        echo $newArray['Description'];echo "</td>";
        echo "<td>";echo $newArray['Cost'];
        echo "</td>";
        echo "<td>";echo $newArray['DatePur'];
        echo "</td>";echo "<td>";
        echo $newArray['IssuedDate'];echo "</td>";
        echo "<td>";echo $newArray['IssuedBy'];
        echo "</td>";echo "</tr>";
    }
}
}
mysql_close($database);
}
?>
</table>
</body>

```

B.34 PerInvFrames - PHP

```

<?php
session_start();
require_once('Functions.php');
$page1 = "Level0Links.htm";
$page2 = "PerInvSelect.php";
buildframes($page1,$page2);
?>

```

B.35 PerlInvSelect – PHP

```
<body bgcolor="#000099" text="#FF9900">
<?php
$database = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");
if(!$database)
{
    echo mysql_error();
    exit;
}
?>
<p align="center">
<p align="center"><strong><font size="+2">Please Select
an Employee to Display their Inventory</font>
</strong></p>
<P align="center">
<form method="post" action="PerInvDisplay.php"
target="_self">

<p align="center">
<select name="Select">
    <option>Please Select</option>
    <option>All Employees</option>
<?php
    $sql = "SELECT FName,LName,MInitial FROM personnel";
    $result = mysql_query($sql, $database) or
    die(mysql_error());

    while($newArray = mysql_fetch_array($result))
    {
        $Name = $newArray['LName'].'. '.$newArray['FName'];
        if($newArray['MInitial'] != '')
        {
            $Name = $Name .'. '.$newArray['MInitial'];
        }
        echo '<option>';
        echo $Name;
        echo '</option>';
    }
?>
</select>
</p>
<p align="center">
<input type="submit" name="Submit" value="Select">
</p>
</form>
```

</P>

```
<?php
mysql_close($database);
?>
```

B.36 PerlItemEnter

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post"
action="PerlItemEnter.php" target="_self">
<p align="center"><strong><font size="+2">Enter the new
Employee's Information.
</font></strong></p>
<table width="60%" border="1" align="center">
  <tr>
    <td>
      <div align="right"><strong>*Item:</strong></div></td>
    <td><em><strong>
      <input type="text" name="Item">
        (compter, Hand Cuffs, etc)
      </strong></em></td>
  </tr>
  <tr>
    <td>
      <div align="right"><strong>*Manufacture:</strong>
      </div></td>
    <td><em><strong>
      <input type="text" name="Man">
      </strong></em></td>
  </tr>
  <tr>
    <td>
      <div align="right"><strong>*Model:</strong>
      </div></td>
    <td><em><strong>
      <input type="text" name="Model">

```

```

        </strong> </em></td>
</tr>
<tr>
    <td><div align="right"><strong>
        Serial Number:</strong></div></td>
    <td><em><strong>
        <input type="text" name="SerNum">
        </strong></em></td>
</tr>
<tr>
    <td><div align="right"><strong>*Description:
        </strong></div></td>
    <td><em><strong>
        <textarea name="Desc"></textarea>
        </strong></em></td>
</tr>
<tr>
    <td><div align="right"><strong>*Cost:
        </strong></div></td>
    <td><em><strong>
        <input type="text" name="Cost">
        </strong></em></td>
</tr>
<tr>
    <td><div align="right"><strong>*Date Purchased:
        </strong></div></td>
    <td><em><strong>
        <input type="text" name="DatePur">
        (yyyy-mm-dd)</strong></em></td>
</tr>
<tr>
    <td><div align="right"><strong>Issued To:
        </strong></div></td>
    <td><em><strong>
        <input type="text" name="IssTo">
        </strong></em></td>
</tr>
<tr>
    <td><div align="right"><strong>Issued Date:</strong>
        </div></td>
    <td><em><strong><input type="text" name="IssDate">
        (yyyy-mm-dd)</strong></em></td>
</tr>
<tr>
    <td><div align="right"><strong>Issued By:</strong>
        </div></td>
    <td><em><strong>

```



```
DatePur,IssuedTo,IssuedDate,IssuedBy)
```

```
values
```

```
(' $Item', '$Man', '$Model', '$SerNum', '$Desc', '$Cost',  
 '$DatePur', '$IssTo', '$IssDate', '$IssBy')";
```

```
$result = mysql_query($sql, $database) or  
die(mysql_error());  
mysql_close($database);  
?>
```

```
<body bgcolor="#000099" text="#FF9900">  
<div align="center">  
  <p><strong><font size="+2">  
    The following Employee has been </font></strong></p>  
  <p><strong><font size="+2">  
    entered to the PersonnelInventory.  
    </font></strong></p>  
  <table width="30%" border="1" cellpadding="3">  
    <tr>  
      <td align="right">Item:</td>  
      <td><?php echo $Item?>&nbsp;</td>  
  
    </tr>  
    <tr>  
      <td align="right">Manufacturer:</td>  
      <td><?php echo $Man?>&nbsp;</td>  
  
    </tr>  
    <tr>  
      <td align="right">Model:</td>  
      <td><?php echo $Model?>&nbsp;</td>  
  
    </tr>  
    <tr>  
      <td align="right">Serial Number:</td>  
      <td><?php echo $SerNum?>&nbsp;</td>  
  
    </tr>  
    <tr>  
      <td align="right">Description:</td>  
      <td><?php echo $Desc?>&nbsp;</td>  
    </tr>  
    <tr>  
      <td align="right">Cost:</td>  
      <td><?php echo $Cost?>&nbsp;</td>
```

```

</tr>
<tr>
  <td align="right">Date Purchased:</td>
  <td><?php echo $DatePur?>&nbsp;</td>

</tr>
<tr>
  <td align="right">Issued To:</td>
  <td><?php echo $IssTo?>&nbsp;</td>

</tr>
<tr>
  <td align="right">Issued Date:</td>
  <td><?php echo $IssDate?>&nbsp;</td>
</tr>
<tr>
  <td align="right">Issued By:</td>
  <td><?php echo $IssBy?>&nbsp;</td>
</tr>
</table>
<p>&nbsp;</p>
</div>

```

B.38 TitleFrame

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>TitleFrame</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<p align="center"><font size="+2"><strong><font
size="+3">West Virginia State Police
</font></strong></font><font size="+3"><strong>Inventory
Tracking System </strong></font></p>
</body>
</html>

```

B.38 tools

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Tools</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<p>&nbsp;</p>
<p>&nbsp;</p>
</body>
</html>
```

B.39 UpdateCount – PHP

```
<?php
session_start();
require_once('Functions.php');
$page1 = "Level0Links.htm";
$page2 = "UpdateCountMath.php";
buildframes($page1,$page2);
?>
```

B.40 UpdateCountMath

```
<?php
session_start();
$Selection = $_POST[itemselect];
$databse = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");
if(!$databse)
{
    echo mysql_error();
    exit;
}
?>
```



```

while ($Flag == -1)
{
    $newArray = mysql_fetch_array($result);
    $S = $newArray['Manufacturer'] ." ".
    $newArray['Model'];

    if( $S == $Selection)
    {
        $Flag = 0;
    }
}

echo "<tr align = 'center'>";echo "<td>";
echo $newArray['DPS'];echo "</td>";
echo "<td>";echo $newArray['LocCode'];echo "</td>";
echo "<td>";echo $newArray['Manufacturer'];
echo "</td>";
echo "<td>";echo $newArray['Model'];echo "</td>";
echo "<td>";echo $newArray['Serial'];echo "</td>";
echo "<td>";echo $newArray['Description'];
echo "</td>";
echo "<td>";echo $newArray['PurchaseDate'];
echo "</td>";echo "<td>";
echo $newArray['Cost'];
echo "</td>";
echo "<td>";echo $newArray['EnteredBy'];
echo "</td>";
echo "<td>";echo $newArray['EnteredOn'];
echo "</td>";
echo "<td>";echo $newArray['Count'];
echo "</td></tr>";
mysql_close($database);
?>
</body>
</html>

```

B.41 UpdateCountSelect – PHP

```

<?php
$database = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");
if(!$database)
{
    echo mysql_error();
    exit;
}

```

```

    }
?>
<body bgcolor="#000099" text="#FF9900">
<p align="center">&nbsp;</p>
<p align="center"><strong><font size="+2">Select which
inventory item you want to decrease the inventory
quantity</font></strong></p>
<form name="form1" method="post"
action="UpdateCountMath.php" target="_self">
<p align="center">
<select name="itemselect" >
    <option>Please Select</option>
<?php
    $sql = "SELECT Manufacturer,Model FROM
    generalinventory where LocCode = 666";

    $result = mysql_query($sql, $database) or
    die(mysql_error());

    while ($newArray = mysql_fetch_array($result))
    {
        echo "<option>";
        echo $newArray['Manufacturer'];
        echo " ";
        echo $newArray['Model'];
        echo "</option>";
    }
?>
</select>
</p><p align="center">
    <input type="submit" name="Submit" value="Select">
</p>
</form>
<p>&nbsp;</p>
<?php
    mysql_free_result($result);
    mysql_close($database);
?>

```

B.42 UpdateCountSelectFrames – PHP

```

<?php
session_start();
require_once('Functions.php');
$page1 = "Level0Links.htm";

```

```

$page2 = "UpdateCountSelect.php";
buildframes($page1,$page2);
?>

```

B.43 VehEnter

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post"
action="VehicleEnter.php" target="_self">
<p align="center"><strong><font size="+2">Enter the new
Employee's Information.</font></strong></p>
<table width="60%" border="1" align="center">
  <tr>
    <td><div align="right"><strong>*VIN:
    </strong></div></td>
    <td><em><strong>
    <input type="text" name="VIN">
    </strong></em></td>
  </tr>
  <tr>
    <td><div align="right"><strong>*Manufacture:
    </strong></div></td>
    <td><em><strong>
    <input type="text" name="Man">
    </strong></em></td>
  </tr>
  <tr>
    <td><div align="right"><strong>*Model:</strong>
    </div></td>
    <td><em><strong>
    <input type="text" name="Model">
    </strong> </em></td>
  </tr>
  <tr>
    <td><div align="right"><strong>*Description:
    </strong></div></td>

```



```

        <p align="center"><strong><em>*required</em>
        </strong></p>
    </form>
</body>
</html>

```

B.44 VehicleEnter

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body bgcolor="#000099" text="#FF9900">
<form name="form1" method="post"
action="PerItemEnter.php" target="_self">
<p align="center"><strong><font size="+2">Enter the new
Employee's Information.
</font></strong></p>
<table width="60%" border="1" align="center">
<tr>
    <td><div align="right"><strong>*Item:</strong>
    </div></td>
    <td><em><strong>
    <input type="text" name="Item">
    (computer, Hand Cuffs, etc) </strong></em></td>
</tr>
<tr>
    <td><div align="right"><strong>*Manufacture:
    </strong></div></td>
    <td><em><strong>
    <input type="text" name="Man">
    </strong></em></td>
</tr>
<tr>
    <td><div align="right"><strong>*Model:</strong>
    </div></td>
    <td><em><strong>
    <input type="text" name="Model">
    </strong> </em></td>
</tr>

```

```
|  |  |
| --- | --- |
| Serial Number: |  |
| *Description: |  |
| *Cost: |  |
| *Date Purchased: | (yyyy-mm-dd) |
| Issued To: |  |
| Issued Date: | (yyyy-mm-dd) |
| Issued By: |  |

```

```

        <input type="text" name="IssBy">
    </strong></em></td>
</tr>
<tr>
    <td><div align="right"></div></td>
    <td><input type="submit" name="Submit"
        value="Submit">&nbsp;&nbsp; <input type="reset"
        name="Submit2" value="Reset"></td>
</tr>
</table>
<p align="center"><strong><em>*required</em></strong></p>
</form>
</body>
</html>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1">
</head>
<body>
</body>
</html>

```

B.45 VehicleItemEnter – PHP

```

<?php
session_start();
require_once('Functions.php');

$item      = $_POST['Item'];
$man       = $_POST['Man'];
$model     = $_POST['Model'];
$serNum    = $_POST['SerNum'];
$desc      = $_POST['Desc'];
$cost      = $_POST['Cost'];
$datePur   = $_POST['DatePur'];
$vin       = $_POST['VIN'];
$entBy     = $_POST['EntBy'];
$entOn     = $_POST['EntOn'];

$dbase = mysql_connect("localhost","bamb", "senior1");

```

```

mysql_select_db("WVSPITS");
if(!$database)
{
    echo mysql_error();
    exit;
}
$sql = "insert into vehicleitem
(Item,Manufacturer,Model,SerNum,Description,Cost,
DatePur,VIN,EnteredBy,EnteredOn)

values

('$Item','$Man','$Model','$SerNum','$Desc','$Cost',
'$DatePur','$VIN','$EntBy','$EntOn')";

$result = mysql_query($sql, $database) or
die(mysql_error());
mysql_close($database);
?>
<body bgcolor="#000099" text="#FF9900">
<div align="center">
<p><strong><font size="+2">The following Employee has
been </font></strong></p>
<p><strong><font size="+2">entered to the
Personnel Inventory.</font></strong></p>

<table width="30%" border="1" cellpadding="3">
<tr>
    <td align="right">Item:</td>
    <td><?php echo $Item?>&nbsp;</td>
</tr>
<tr>
    <td align="right">Manufacturer:</td>
    <td><?php echo $Man?>&nbsp;</td>
</tr>
<tr>
    <td align="right">Model:</td>
    <td><?php echo $Model?>&nbsp;</td>
</tr>
<tr>
    <td align="right">Serial Number:</td>
    <td><?php echo $SerNum?>&nbsp;</td>
</tr>
<tr>
    <td align="right">Description:</td>
    <td><?php echo $Desc?>&nbsp;</td>
</tr>

```

```

<tr>
  <td align="right">Cost:</td>
  <td><?php echo $Cost?>&nbsp;</td>
</tr>
<tr>
  <td align="right">Date Purchased:</td>
  <td><?php echo $DatePur?>&nbsp;</td>
</tr>
<tr>
  <td align="right">VIN:</td>
  <td><?php echo $VIN?>&nbsp;</td>
</tr>
<tr>
  <td align="right">Entered By:</td>
  <td><?php echo $EntBy?>&nbsp;</td>
</tr>
<tr>
  <td align="right">Entered On:</td>
  <td><?php echo $EntOn?>&nbsp;</td>
</tr>
</table>
<p>&nbsp;</p>
</div>

```

B.46 VehicleItemItem - PHP

```

<?php
session_start();
require_once('Functions.php');

$Item      = $_POST['Item'];
$Man       = $_POST['Man'];
$Model     = $_POST['Model'];
$SerNum    = $_POST['SerNum'];
$Desc      = $_POST['Desc'];
$Cost      = $_POST['Cost'];
$DatePur   = $_POST['DatePur'];
$VIN       = $_POST['VIN'];
$EntBy     = $_POST['EntBy'];
$EntOn     = $_POST['EntOn'];

$databse = mysql_connect("localhost","bamb", "senior1");
mysql_select_db("WVSPITS");

```

```

if(!$database)
{
    echo mysql_error();
    exit;
}

$sql = "insert into vehicleitem
(Item,Manufacturer,Model,SerNum,Description,Cost,
DatePur,VIN,EnteredBy,EnteredOn)

values

('$Item','$Man','$Model','$SerNum'$Desc','$Cost',
'$DatePur','$VIN','$EntBy','$EntOn')";

$result = mysql_query($sql, $database) or
die(mysql_error());
mysql_close($database);
?>
<body bgcolor="#000099" text="#FF9900">
<div align="center">
    <p><strong><font size="+2">The following Employee has
    been </font></strong></p>
    <p><strong><font size="+2">entered to the
    Personnel Inventory.</font></strong></p>
    <table width="30%" border="1" cellpadding="3">
    <tr>
        <td align="right">Item:</td>
        <td><?php echo $Item?>&nbsp;</td>
    </tr>
    <tr>
        <td align="right">Manufacturer:</td>
        <td><?php echo $Man?>&nbsp;</td>
    </tr>
    <tr>
        <td align="right">Model:</td>
        <td><?php echo $Model?>&nbsp;</td>
    </tr>
    <tr>
        <td align="right">Serial Number:</td>
        <td><?php echo $SerNum?>&nbsp;</td>
    </tr>
    <tr>
        <td align="right">Description:</td>
        <td><?php echo $Desc?>&nbsp;</td>
    </tr>
    <tr>

```

```

        <td align="right">Cost:</td>
        <td><?php echo $Cost?>&nbsp;</td>
    </tr>
    <tr>
        <td align="right">Date Purchased:</td>
        <td><?php echo $DatePur?>&nbsp;</td>
    </tr>
    <tr>
        <td align="right">VIN:</td>
        <td><?php echo $VIN?>&nbsp;</td>
    </tr>
    <tr>
        <td align="right">Entered By:</td>
        <td><?php echo $EntBy?>&nbsp;</td>
    </tr>
    <tr>
        <td align="right">Entered On:</td>
        <td><?php echo $EntOn?>&nbsp;</td>
    </tr>
</table>
<p>&nbsp;</p>
</div>

```

B.47 Welcome – PHP

```

<?php
session_start();
echo '<body bgcolor="#000099" text="#FF9900">';
echo '<p align= "center"><strong><font size="+3">Welcome
to the WVSP-ITS.';
echo '</font></strong></p>';
echo '<p align = "center"><strong><font size="+3">';
echo $_SESSION['FNAME'],' your user level is ',
$_SESSION['LEVEL'],'</font></strong></p>';
echo '<p align = "center"><strong><font size="+3">Current
Date and Time is: ';
echo date('jS F, H:i');
echo '</font></strong></p>';
?>

```