

Master's Thesis 2014

Laboratory Inventory System

Design and Implementation

Candidate:

Dona Nathasha Nakandalage

Telemark University College
Faculty of Technology
Kjølnes ring 56
3918 Porsgrunn

<http://www.hit.no>



Telemark University College

Faculty of Technology
M.Sc. Programme

MASTER'S THESIS, COURSE CODE FMH606

Student: Dona Nathasha Nakandalage

Thesis title: Design and Implementation of Laboratory Inventory System

Signature:

Number of pages: <#>

Keywords: Web Programming, ASP.NET, Database Systems, SQL Server, OOP, Equipment

Supervisor: Hans Petter Halvorsen sign.:

2nd Supervisor: <name> sign.:

Censor: <name> sign.:

External partner: <name> sign.:

Availability: Open

Archive approval (supervisor signature): sign.: **Date :**

Abstract:

Lab Inventory System is a resource management system for a laboratory for handling loans and reservations, managing user profiles and allowing search facilities for users to find out information of equipment in detail. It is a better solution with many flexible and convenient features, allowing lab administrators and users to maximize efficiency while reducing time wastage. It gives detailed information about users and equipment with vendor and salesman details and keeps track of available equipment and issued equipment.

Lab Inventory System contains a relational database, a website, a web service and two desktop applications ("LabelWriter" application and "Loan/Return Equipment" application). "Loan/Return Equipment" application is supposed to install and configure in each lab for loaning and returning equipment. "LabelWriter" application which is going to install in administrator's computer, is for printing out labels for registered equipment. A web service is for storing and retrieving data from the database when using desktop applications and "Lab Inventory System" web site is for handling loans and reservations, managing user profiles and registering equipment.

The thesis report describes software development life cycle of Lab Inventory System (Analysis, Design, Implementation, Testing and Deployment phases) with Use Case, class and database design diagrams. Further it details the system overview, detailed system description with its architecture.

Telemark University College accepts no responsibility for results and conclusions presented in this report.

Contents

Contents	3
Abbreviation	7
Part 1: Introduction	9
1 Introduction.....	10
1.1 Task Description and Task Background	10
1.2 Aim and Scope of the Work.....	10
1.2.1 Why Lab Inventory System.....	11
1.3 The structure of the report	12
Part 2: Theory and Background	14
2 Programming Concepts.....	15
2.1 Object Oriented Programming Concepts.....	15
2.1.1 Encapsulation	16
2.1.2 Inheritance	17
2.1.3 Polymorphism	17
2.1.4 Advantages of Object Oriented Programming	17
2.1.5 Disadvantages of Object Oriented Programming.....	18
2.2 Web Services	18
2.3 Visual Studio 2013.....	19
2.3.1 Windows Forms	20
2.3.2 ASP.NET	20
2.3.3 Team Foundation Server (TFS).....	21
3 Database Design Concepts	23
3.1 Relational Database Model.....	24
3.2 Database Diagram	25
3.2.1 ERWin.....	25
3.3 SQL Server 2012.....	26
3.3.1 Data Retrieval and Manipulation in SQL Server	26
3.3.2 Concurrency and Locking	26
3.3.3 SQL Server Management Studio.....	27

Part 3: Analysis and Design	28
4 Analysis	29
4.1 System Overview	29
4.1.1 Database	30
4.1.2 “Lab Inventory System” Website.....	30
4.1.3 “LabelWriter” Application.....	30
4.1.4 “Loan/Return Equipment” Application.....	31
4.1.5 Web Service	31
4.2 Requirements for “Lab Inventory System” Website	31
4.3 Requirements for “LabelWriter” Application	37
4.4 Requirements for “Loan/Return Equipment” Application	39
5 Design	42
5.1 Application Design	42
5.1.1 System Architecture	42
5.2 Database Design	44
Part 4: Implementation, Testing and Deployment	47
6 Implementation	48
6.1 Implementation of “Lab Inventory System” Website	48
6.2 Implementation of Web Service	53
6.3 Implementation of “LabelWriter” and “Loan/Return Equipment” Applications	54
6.4 Database Scripts, Stored Procedures and Views	56
6.5 Source Code Control	57
6.6 Best Coding Practices	58
6.6.1 Commenting	58
6.6.2 Exception Handling.....	59
7 Testing	60
7.1 Testing of Lab Inventory System	60
7.1.1 Bugs/Tasks Tracking with Team Foundation Server.....	62
8 Deployment	66
8.1 Deploy “Lab Inventory System” Website and Web Service	66
8.1.1 IIS Configuration.....	67
8.1.2 Deploy Database Scripts	69

8.2	Deploy “Loan/Return Equipment” Application and “LabelWriter” Application	70
9	Conclusion and Future Work	73
9.1	Suggestions for future work	74
	References	75
	Appendix 1: Task Description.....	77
	Appendix 2: Fully Dressed Use Case Documents for “Lab Inventory System” Website.....	78
	Appendix 3: User Interface Sketches for “Lab Inventory System” Website	80
	Appendix 4: Fully Dressed Use Case Documents for “LabelWriter” Application.....	82
	Appendix 5: Fully Dressed Use Case Documents for “Loan/Return Equipment” Application	83
	Appendix 6: SQL Server Database Diagrams	84
	Appendix 7: Class Diagram.....	86
	Appendix 8: Screen dumps of “Lab Inventory System” Website	90
	Appendix 9: Web Methods of “Lab Inventory System” Web Service.....	92

Preface

The thesis, “Design and Implementation of Laboratory Inventory System” has been carried out at Telemark University College under the supervision of Hans Peter Halverson. The thesis work has been a mandatory part of System and Control Engineering master program at Telemark University College during spring semester.

I would like to thank my supervisor Hans Petter Halverson for close supervision and good collaboration. He encouraged me by giving feedback for my work and providing necessary equipment on time for testing the system. Indeed, he devoted his precious time to discuss things each and every week throughout the whole semester. In addition I would like to thank college library for providing books, various journals and research papers.

Many thanks also go to my colleagues at Telemark University College for their support. Finally, my very special thanks go to my loving husband, Deshaka Kottage for being my emotional anchor and my parents & family members for their encouragement and support.

Dona Nathasha Nakandalage

The system is available in <http://128.39.35.248/LabSystem>

Abbreviation

API	Application Programming Interface
CLR	Common Language Runtime
CSS	Cascading Style Sheet
DB	Database
DBMS	Database Management System
ETL	Extract, Transform, Load
FTP	File Transfer Protocol
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IIS	Internet Information Services
OLAP	Online Analytical Processing
OOP	Object Oriented Programming
RFID	Radio Frequency Identification
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
TFS	Team Foundation Server
T-SQL	Transact-Structured Query Language

UDDI	Universal Description, Discovery and Integration
UI	User Interface
UML	Unified Modeling Language
VB	Visual Basic
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language
XSLT	EXtensible Stylesheet Language Transformations

Part 1: Introduction

Part 1 contains “Introduction” chapter which describes the task description, aim and scope of work and the structure of the report.

1 Introduction

This chapter explains the tasks given for the master thesis titled, “Design and Implementation of Laboratory Inventory System” in spring semester at Telemark University College 2014. It also includes the tasks description and background with aim and scope of the work. Furthermore, it includes an explanation of different chapters in the report.

1.1 Task Description and Task Background

See Appendix 1 for task description and background given by the supervisor.

1.2 Aim and Scope of the Work

This section explains the aim of the thesis with describing necessity of such a system and specifies the scope of each task.

Presently an enormous amount of lab equipment (some equipment are very expensive) have been used by students and staff from Faculty of Technology, Telemark University college during their day to day study/teaching activities like research and developments, lab assignments, projects, thesis and lectures. The records of lab equipment (borrower name, reason for borrowing, borrowed date, student number/staff number) are maintained in a written form by lab administrators or owners of the equipment. This takes lots of time of both lab administrators and borrowers as borrower has to go to the office of lab administrator and has to write the details in a form when borrowing or returning the equipment. In the meantime it is the responsibility of lab administrator to find the appropriate form when borrowing or returning equipment, to check deadlines of each record, to maintain the records (may be at the end of each semester to check whether all the equipment are available in the lab or not), to get corrective actions when missing/damaging equipment. Therefore it is obvious that it is a long, time consuming process and the existing system is not so efficient.

“Design and Implementation of Laboratory Inventory System” comes as a better solution with many flexible and convenient features, allowing lab administrators and users to maximize efficiency while reducing time wastage. It gives detailed information about users and equipment with vendor and salesman details and will keep track of available equipment and

issued equipment. It will definitely minimize errors and will be a very good replacement to existing repetitive manual process.

1.2.1 Why Lab Inventory System

Several benefits of Lab Inventory System over the manual process can be listed as follows:

- Reduce errors and eliminate long and repetitive manual processing
- Improved efficiency and effectiveness in administration and management of records
- More economical as it minimizes the time usage
- Easy to take corrective actions in a timelier manner
- Easy access to equipment details as it has search functionality
- Increased productivity

The whole thesis has been divided into five main tasks as follows:

Task 1

The very first task is to design a database for Lab Inventory System using SQL server and ERWin software applications. The database should include tables for storing information about different lab equipment and equipment categories, handling reservations and loans, storing user information, login details and access level information. Moreover, the database should be designed according to database concepts and rules to reduce data redundancy.

Task 2

Task two is to design and implement a “LabelWriter” application for printing out barcode labels that can be attached to different equipment and user identity card. This is a standalone application, so that lab administrators can install and configure it in their personal computers. “LabelWriter” application allows searching users or equipment through a web service, selecting an appropriate users or equipment and printing a label with a barcode and other specific data. This allows selecting a default label template or to upload and use their own label template. DYMO Label Writer should be connected and drivers should be installed in the computer for printing labels.

Task 3

In task three, a web application has been created for managing and tracking lab equipment data. This allows creating/updating/deleting user profiles, reserving lab equipment, borrowing or returning equipment, granting different access levels, registering/modifying equipment,

displaying equipment details, searching equipment with name or vendor name or filtered search with equipment categories. This is an online web site, so that each and every registered user can access it from everywhere.

Task 4

Task four is to design and implement a smartphone application for easy management of the lab equipment. But design and implementation of “Loan/Return Equipment” application is more important and useful to enhance the functionalities of whole Lab Inventory System than smartphone application, “Loan/Return Equipment” application has been implemented although it was not in the initial task description.

“Loan/Return Equipment” application is a standalone application that can be installed and configured in each and every lab with a barcode reader, so that the users can borrow or return equipment with few clicks. It allows scanning equipment and user barcodes and storing records in the database while ensuring minimum time usage.

Task 5

Task five is to deliver a written report (thesis report).

1.3 The structure of the report

The report is divided into four main parts with nine different chapters to simplify the structure of the report. Each part consists of one or more chapters. The nine chapters in the report contain the following:

Chapter 1: Introduction

The introduction chapter describes the objectives of the thesis, scope of each and every task and the structure of the report.

Chapter 2: Programming Concepts

This chapter describes OOP concepts, pros and cons of OOP and theory behind web services. It also details the study carried out to familiarize with Visual Studio and its tools with their features.

Chapter 3: Database Design Concepts

Chapter three describes the concepts behind database designing, database models and features and functionalities of database design tools.

Chapter 4: Analysis

This chapter details system overview, user categories and functional requirements of “Lab Inventory System” website, “Loan/Return Equipment” application and “LabelWriter” application with Use Case diagrams.

Chapter 5: Design

Design chapter describes detailed system description, system architecture and database design with database diagram and class diagram.

Chapter 6: Implementation

This chapter discusses implementation of Lab Inventory System with technical information of the system.

Chapter 7: Testing

Chapter seven describes testing methods used for Lab Inventory System and bug tracking with TFS during implementation and testing phases.

Chapter 8: Deployment

This chapter describes deployment methods and steps used for “Lab Inventory System” website, web service, “Loan/Return Equipment” application and “LabelWriter” application.

Chapter 9: Conclusion and Future Works

The last section is the conclusion. It also gives suggestions for improvements of the system and proposals for future work.

Part 2: Theory and Background

Part 2 contains “Programming Concepts” and “Database Design Concepts” chapters. It details concepts and theories behind programming and database design with software development tools.

2 Programming Concepts

Programming is the process of designing/writing computer applications. Thousands of programming languages have been designed/used with a specific purpose. Even though each programming language has their unique features or syntax, there are certain concepts common to all languages. There are different types of programming styles such as structured programming, object oriented programming, etc. In structured programming, blocks of programming statements (code) are executed one after another. Control statements change which blocks of code is executed next. In object oriented programming, data are contained in objects and is accessed using methods specific to the type of the object. There is no single “flow” of the program as objects can freely interact with one another by passing messages.

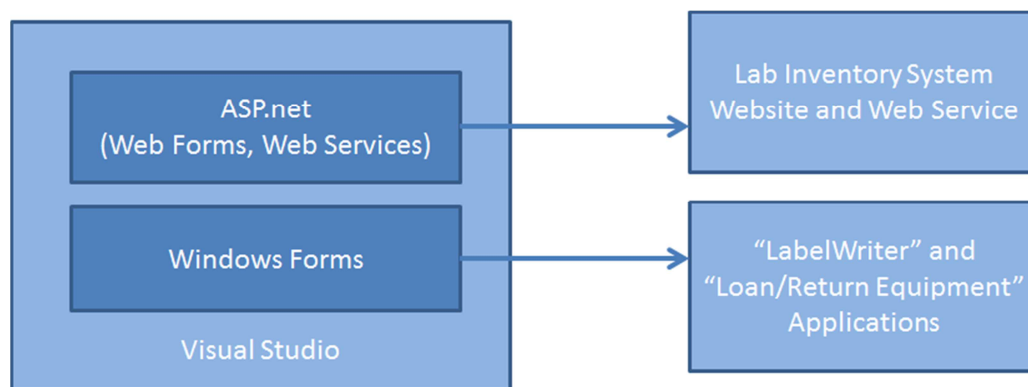


Figure 2-1 Lab Inventory System

This chapter mainly focuses on OOP concepts and pros and cons of OOP. Further it describes the fundamental theory behind web services. As shown in Figure 2-1, “Lab Inventory System” website and web service are implemented using ASP.NET and “LabelWriter” and “Loan/Return Equipment” applications are implemented using WinForms in Visual Studio. Therefore a detailed study has been carried out to understand and familiarize with the latest version of Visual Studio, ASP.NET and WinForms.

2.1 Object Oriented Programming Concepts

OOP is a programming methodology based on objects which is concerned to develop application on real time, so more emphasis is given on data unlike the other programming styles like structured or functional. The concept of having objects makes OOP more organized, reusable and speedy. Today most of the programming languages support OOP. Most of them have enhanced features to make OOP more easy and maintainable.

Object

An object consists of attributes and methods. Attributes define the properties of the object while methods define its behavior. In OOP, object is considered as an instance of a class. Considering Lab Inventory System, “Equipment” and “Customer” can be considered as objects and “Equipment Name” and “Customer Name” can be considered as their attributes.

Class

A class is simply a representation of a type of object. Using the blueprint analogy, a class is a blueprint of an object, and an object is a building made from that blueprint [1]. It consists of a name, attributes and methods.

Encapsulation, inheritance and polymorphism are the main fundamental concepts in OOP.

2.1.1 Encapsulation

Encapsulation normally refers to information hiding or treating a group of related properties, methods, and other members as a single unit or object [1]. Hence encapsulation guarantees integrity of object data and uses methods to access information as shown in Figure 2-2.

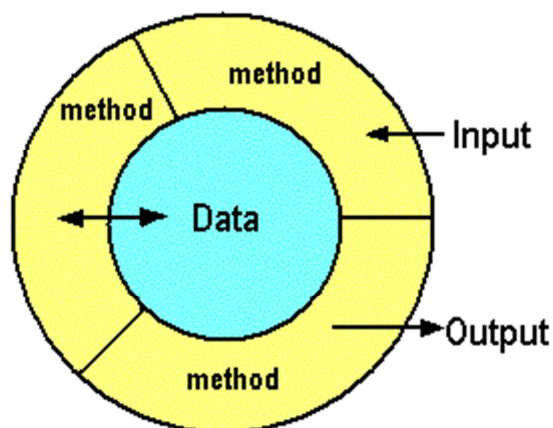


Figure 2-2 Encapsulation [2]

Encapsulation can be achieved using “private”, “public” and “protected” access modifiers.

“public” – access is not restricted

“private” – access limited to members of same class

“protected” – access limited to members of same class or descendants

2.1.2 Inheritance

Inheritance describes the ability to create new classes based on an existing class. So the object can inherit properties from another object while defining common codes only one place with enhancing maintainability of the software¹. Figure 2-3 shows an example of inheritance. According to the Figure 2-3 “Customer” is the parent class. Both “Student” and “Staff” inherit all the fields and methods of the parent class “Customer”. Hence inheritance guarantees code reusability.

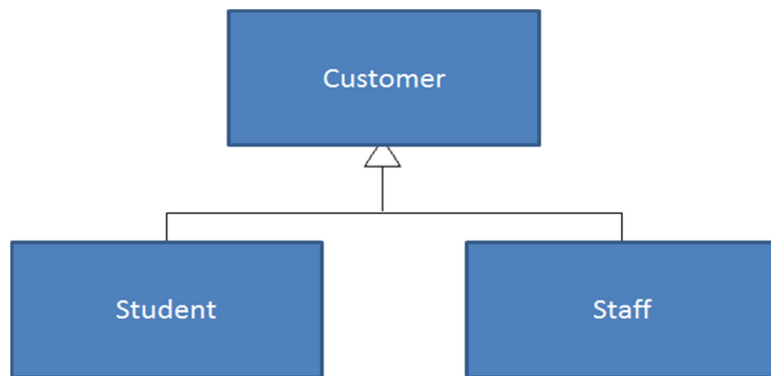


Figure 2-3 An example of Inheritance

2.1.3 Polymorphism

Polymorphism is the ability to take more than one forms depending on data type or class, so that multiple classes can be used interchangeably, even though each class implements the same properties or methods in different ways. Hence polymorphism guarantees maintainability of the software. Interface introduces polymorphism. It contains only definition of methods, properties and events, so the class that implements the interface, has the implementation or declaration to achieve polymorphism.

2.1.4 Advantages of Object Oriented Programming

- Easy to maintain and extend existing code
- Enhanced code reusability
- Object hiding can be achieved

¹ Referred from “Object-Oriented Analysis, Design, and Programming using UML and C#” lecture notes 2012 made by Nils Olav at Telemark University College

- Improved reliability and flexibility, as objects can be dynamically called and accessed, new objects may be created at any time
- Faster development, as reusing software modules lowers the time usage
- Cost effectiveness, as reusing software modules lowers the cost of development

2.1.5 Disadvantages of Object Oriented Programming

- Need extra time and effort for planning
- Not suitable for all types of problems
- OOP programs are generally larger and slower than normal programs

2.2 Web Services

A web service is implemented to handle interactions between both “LabelWriter” and “Loan/Return Equipment” applications and database as shown in Figure 2-4.

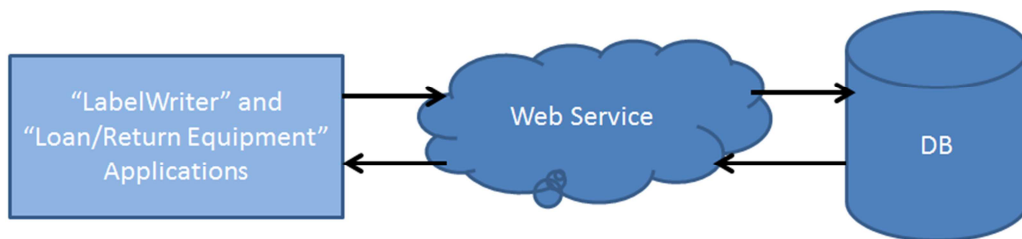


Figure 2-4 “LabelWriter” and “Loan/Return Equipment” applications interact with DB through web service

Web service is a software system designed to connect to other software applications. An application can publish its function or message to the rest of the world using web services. Web services use XML to code/decode data. Other applications can interact with web service using SOAP messages.

Web services offer many benefits over other types of such systems.

- Interoperability – This is the most important benefit of web services. Web Services allow different applications to interact with each other and share data and services among themselves. For example VB or .NET application can communicate with java web services and vice versa.
- Low cost communication as web services use SOAP over HTTP protocol.
- Reusability
- Usability – Other applications have the freedom to choose the web service they need.

Figure 2-5 shows a logical view of web services architecture which is based on three primary roles, service requester, service provider and service registry. Providers of the web services are known as service providers, users of the web services are known as clients or service requester. The essential part of web services is the interact relationship between a service provider and a service requester. These three roles interact using publish, find and bind operations. The service provider publishes the service description in a service registry. The service requester finds the service description in a service registry and uses the information in the description to bind to a service. The service registry provides a centralized location for storing service descriptions. They will become interface to a huge world of data and query services. UDDI, a platform independent, XML based registry is an example of service registry.

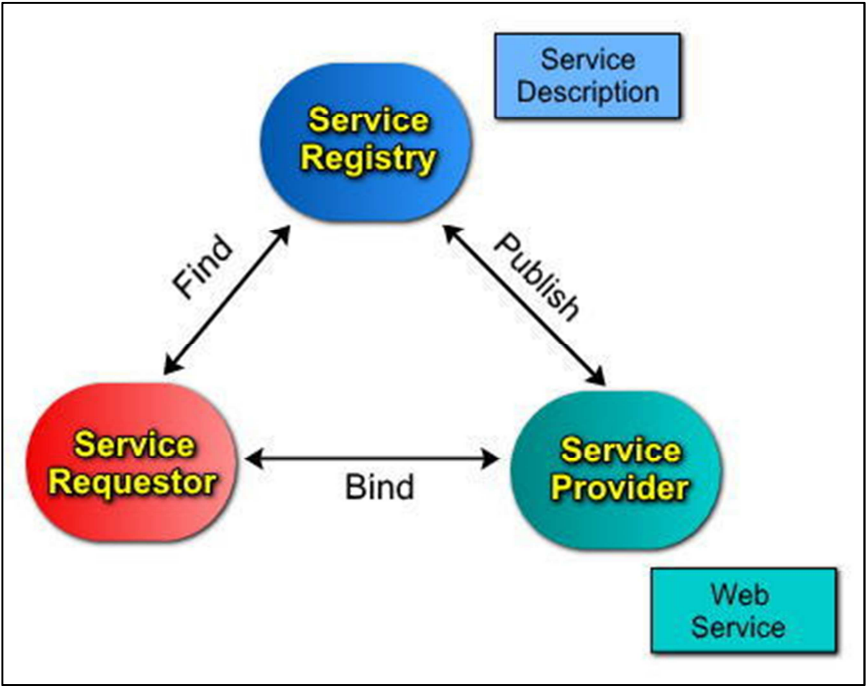


Figure 2-5 Web Services Architecture [3]

2.3 Visual Studio 2013

As per the task description, Lab inventory System is implemented using Visual Studio 2013. Visual Studio is a comprehensive collection of tools and services to help developers to create a wide variety of apps, both for the Microsoft platform and beyond [4]. It was developed by Microsoft. Microsoft has released several versions of Visual Studio as Visual Studio 2005, Visual Studio 2008, Visual Studio 2010, Visual Studio 2012 and the latest version is Visual

Studio 2013. It is also available in several editions as Visual Studio Express, Visual Studio Professional and Visual Studio Ultimate and is used to develop standalone applications, mobile applications, web sites, web applications and web services.

Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code. It supports different programming languages such as C, C++, C#.net, VB.net and F#. It also supports XML/XSLT, HTML/XHTML, JavaScript and CSS.

Code editor in Visual Studio supports syntax highlighting and code completion (using IntelliSense) as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger and allows stepping through the application line by line while executing. Visual Studio includes designers for windows applications, web applications and mobile applications.

2.3.1 Windows Forms

“LabelWriter” and “Loan/Return Equipment” applications are implemented using Windows Forms (WinForms). WinForms is the graphical application programming interface included as a part of Microsoft Visual Studio. It can display data, handle user input and deploy applications easily and with enhanced security. In Windows Forms, a form is a visual surface which displays information to the user. A control is a user interface element that displays data or accepts data input. A form can be enhanced with controls to create the appropriate user interface while adding code to manipulate data.

WinForms contains variety of controls such as TextBoxes, Radio Buttons, Buttons, ComboBoxes, GridViews, etc. It supports creating custom controls as well. It has rich user interface components to interact with high end applications like Microsoft Office.

2.3.2 ASP.NET

“Lab Inventory System” website and web service are implemented using ASP.NET. ASP.NET is a server-side web application framework developed by Microsoft. It is mainly designed for developing dynamic web sites, web applications and web services and is included as part of Visual Studio. ASP.NET takes the best from Active Server Pages as well as the rich services and features provided by the CLR [5]. The result is a robust, scalable, and

fast web development experience with great flexibility and little coding [5]. As it is built on CLR, it allows user to use any supported .NET language.

ASP.NET web pages are known as web forms. They are the main building blocks for application development. Web forms contain two components, the visual portion (the ASPX file which contains HTML), and the code behind the form, which resides in a separate class file. They are very similar to Windows Forms, so can enhance user interface by adding controls.

As indicated in Figure 2-6, ASP.NET web applications are published in IIS, so that the published websites can be accessed through web browsers.

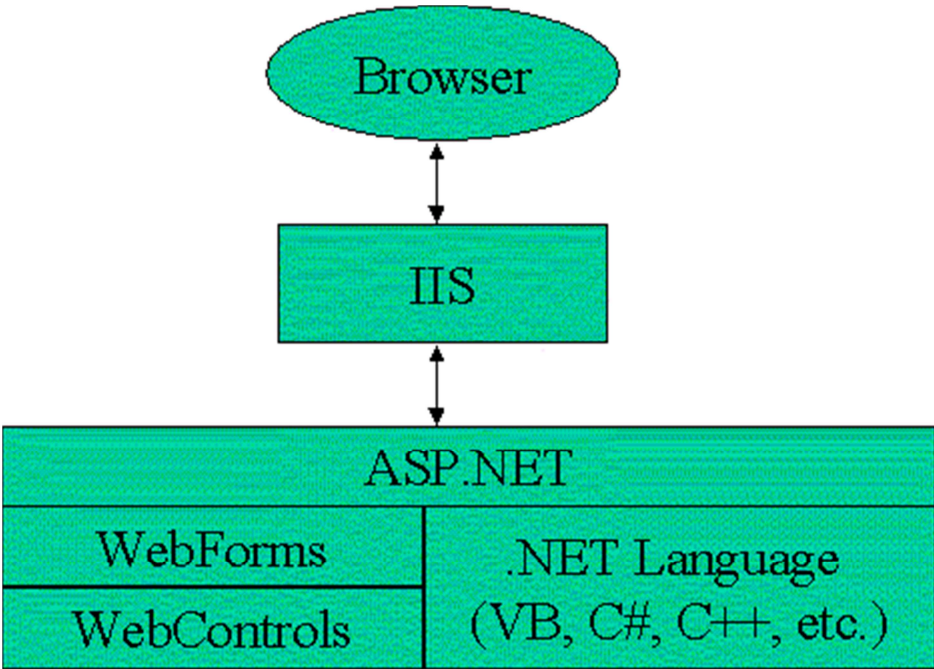


Figure 2-6 Web Forms are part of ASP.NET [5]

2.3.3 Team Foundation Server (TFS)

TFS is used for source code control and bug tracking of Lab Inventory System. It is the application life cycle management hub for Visual Studio [6]. It is a Microsoft product which provides facilities to manage repositories, build process, testing infrastructure, lab deployment and reporting while enabling all stakeholders to participate in the development process using a single solution. It helps developers to connect, collaborate and deliver solutions on time while managing heterogeneous projects. Therefore it is created to get more out of development teams with faster software delivery.

TFS can be used as a back end to numerous integrated development environments but is designed to provide the most benefit by serving as the back end to Microsoft Visual Studio or Eclipse (on Windows and non-Windows platforms) [7]. Figure 2-7 shows the whole usage of TFS. Team members on different platforms can collaborate by using TFS as it covers the whole life cycle of the product/software.

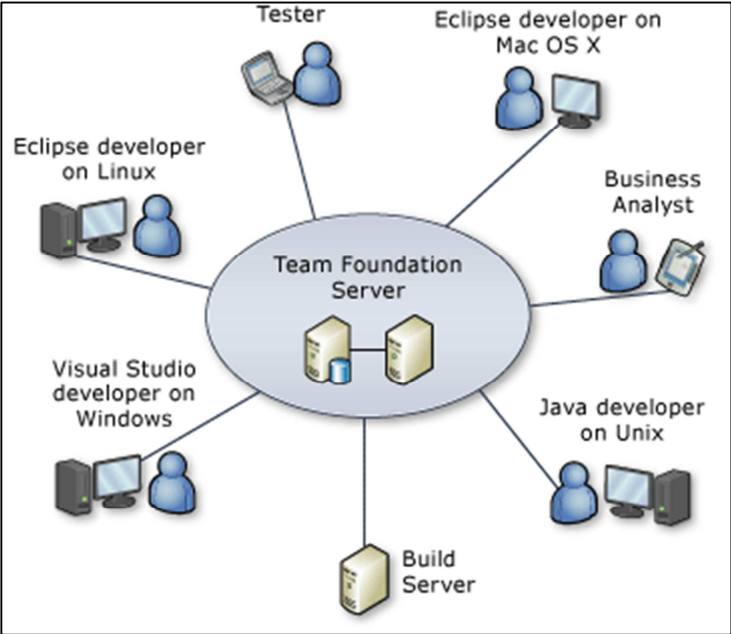


Figure 2-7 Usage of TFS [7]

3 Database Design Concepts

Database is a collection of logically related records (or data). It may consist of enormous number of data. Hence it is really important to manage and organize data within the database. Database Management Systems (DBMSs) are software applications that use to interact with database and the user by giving capability of managing data. It allows tagging, retrieving and manipulating data efficiently and quickly while ensuring the security and unauthorized access. SQL Server is one of the well-known DBMS used today.

Lab Inventory System uses a database for storing and managing records of loans and reservations, equipment details and user details. It uses basic database interactions such as insert, select, delete and update in different scenarios (for example when registering equipment or user, displaying equipment details, modifying and deleting equipment or user details, inserting loans and reservations). Figure 3-1 shows database interactions of Lab Inventory System.

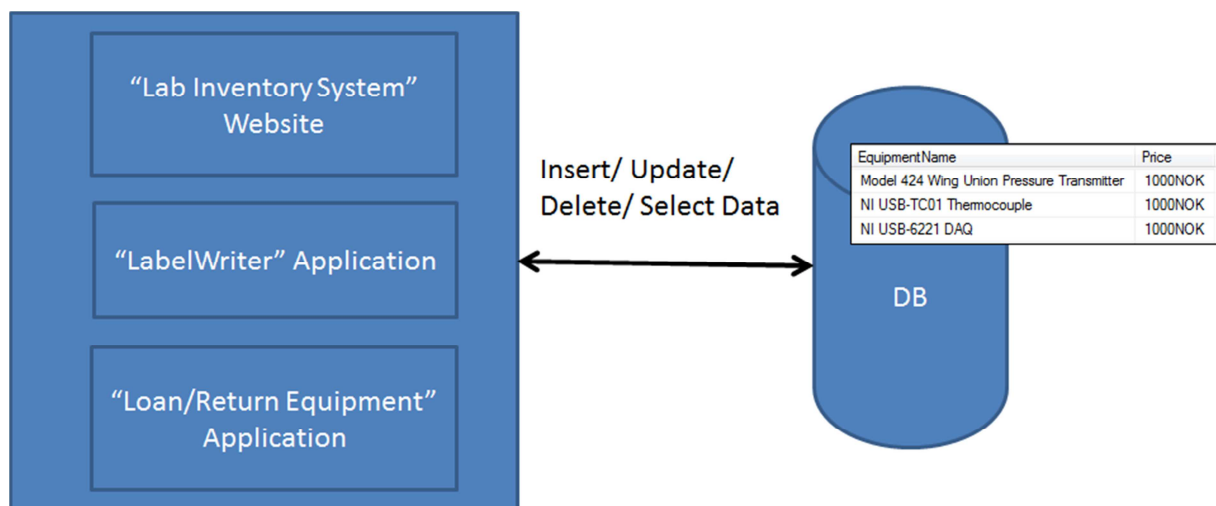


Figure 3-1 Database Interactions (Insert/Update/Delete/Select) of Lab Inventory System

Database model defines the logical structure of the database. Although there are several database models like “Network model”, “Flat Model”, “Hierarchy model” and “Tree model”, “Relational database model” is the most popular database model today because of its efficient and simple data handling.

This chapter details the concepts and methodologies related to database design and database models. As per the task description, the database and scripts of Lab Inventory System should be implemented using SQL Server. Hence a detailed study is carried out to understand and familiarize with the latest version of SQL Server and its tools.

3.1 Relational Database Model

The idea behind relational data model comes with organizing and storing data in tables. Relations, attributes and domains are the three main key terms used in relational database model². Tables of information is called relations, while attributes refer to the columns of the table and the domain is the set of values the attributes are allowed to take [8]. A database column can be defined as a primary key or a foreign key. Primary key is used to identify unique records while foreign key field describes some relationships between tables. One-to-one relationships associate one record in one table with a single record in the other table. One-to-many relationships associate one record in one table with many records in the other table. Figure 3-2 and Figure 3-3 show an example of a table design and a table structure respectively. It is always a good practice to reduce data redundancy (repetition of data) when designing tables to enhance the efficiency of the database.

Column Name	Data Type	Allow Nulls
CustomerTypeId	int	<input type="checkbox"/>
CustomerTypeName	nvarchar(50)	<input type="checkbox"/>
Description	nvarchar(MAX)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Figure 3-2 Table Design

² Referred from “Industrial Information Technology” lecture notes 2013 made by Nils Olav at Telemark University College

	CustomerTypeId	CustomerTypeName	Description
1	1	Student	Student at TUC
2	2	Staff	Staff at TUC

Figure 3-3 Structure of the table

3.2 Database Diagram

Database design diagram is a graphical representation of set of entities (tables) and their relationships among each other. Before designing the database diagram, it is necessary to understand the requirements, make assumptions, identify the entities, attributes, data types, limitations and their relationships. Software applications like SQL server, Microsoft Visio and ERWin provide easy and user friendly design package for drawing database diagrams.

3.2.1 ERWin

ERWin is a software application for designing databases and it is used to design database diagram of Lab Inventory System.

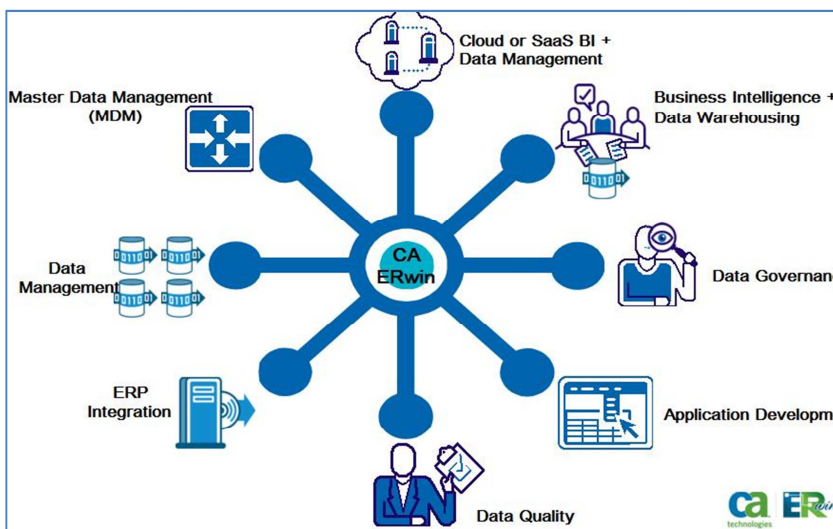


Figure 3-4 Features of ERWin [9]

It provides a collaborative data modeling environment to manage enterprise data through an intuitive, graphical interface [10]. Figure 3-4 shows supported features of ERWin.

3.3 SQL Server 2012

SQL Server which is developed by Microsoft is the most popular DBMS today. Microsoft has released several versions of SQL Server as SQL Server 2005, SQL Server 2008, SQL Server 2012 and etc. Its main functionality is storing and retrieving data in a secure manner as requested by other applications. T-SQL is the proprietary query language in SQL Server. Microsoft has released several SQL Server editions targeting different user groups such as Web, Express, Enterprise and Standard.

3.3.1 Data Retrieval and Manipulation in SQL Server

In a database, data is stored as a collection of tables with typed columns. SQL Server supports different data types such as Integer, Float, Decimal, Char (including character strings), Varchar (variable length character strings), binary, Text and etc. It supports user defined composite types as well.

Queries are used for data retrieval operations. The standard SQL commands such as SELECT, INSERT, UPDATE and DELETE are used for basic data manipulation operations. SELECT statement retrieves data from one or more tables. INSERT adds rows to an existing table while UPDATE modifies existing rows in a table. DELETE statement removes existing rows from the tables.

SQL Server also allows stored procedures to be defined. A stored procedure is a group of SQL statements that form a logical unit and perform a particular task, and they are used to encapsulate a set of operations or queries to execute on a database server [11]. Stored procedures are more quick and efficient than queries as they are compiled once and stored in executable form. Hence it requires low memory requirements. Stored procedures improves scalability (increase scalability by isolating application processing on the server), maintainability (can reuse in different applications) and security (users can be granted permission to execute a stored procedure even if they do not have permission to execute the procedure's statements directly [12]). Other than that it reduces network traffic with single call execution.

3.3.2 Concurrency and Locking

Concurrency refers to allowing multiple users to use same database simultaneously while ensuring data integrity when accessing shared data (for example when several users update same data or try to read data that is in the process of being changed by another user). It is one

of the main feature that differentiate database from other data sources like spreadsheets. SQL Server provides two modes of concurrency control as pessimistic concurrency and optimistic concurrency. Pessimistic concurrency is controlling concurrent access by using locks. Optimistic concurrency control mechanism allows a new version of a row to be created whenever the row is updated, as opposed to overwriting the row.

3.3.3 SQL Server Management Studio

SQL Server Management Studio is a GUI tool included with SQL Server 2005 and later for configuring, managing, and administering all components within Microsoft SQL Server. It includes both script editors and graphical tools that work with objects and features of the SQL server [13].

Part 3: Analysis and Design

Part 3 contains “Analysis” and “Design” chapters. It details system overview, functional requirements with Use Case diagrams, system architecture and design with database and class diagrams.

4 Analysis

Analysis is the process of determining user expectations for a software application. It is really important as requirements are the basis for software design, development and testing. Therefore this affects to the project cost. Requirements can be categorized into two groups as functional requirements and non-functional requirements. Functional requirements define functions or behaviors of the system while non-functional requirements specify quality of the system. But before dig into the analysis, it is also necessary to identify major components of the system with their interactions and behaviors.

A Use Case is a way to understand and describe the requirements³. Use Cases use “Extend” link to show that one Use Case may add functionality to another Use Case under certain circumstances and “Use/Include” relation to specify whether one Use Case describes some of the detail of another [14]. A Use Case diagram is normally an overview diagram of user functions, showing the system as a black box³. Fully dressed Use Case document has a detailed description of each Use Case.

Users of Lab Inventory System can be categorized as “End Users” and “Administrators”. Both of them have different accessibility to the system. Furthermore “End Users” can be categorized into two groups as “Staff” and “Student”.

This chapter details system overview with its major components and functional requirements of Lab Inventory System. Further these requirements have been documented using Use Case diagrams and fully dressed Use Case documents. It also indicates the UI sketches designed for Lab Inventory System at the initial stage.

4.1 System Overview

Lab Inventory System consists of five major components, a database, web service, “Lab Inventory System” website, “LabelWriter” application and “Loan/Return Equipment” application. The system overview diagram with its major components is shown in Figure 4-1.

³ Referred from “Object-Oriented Analysis, Design, and Programming using UML and C#” lecture notes 2012 made by Nils Olav at Telemark University College

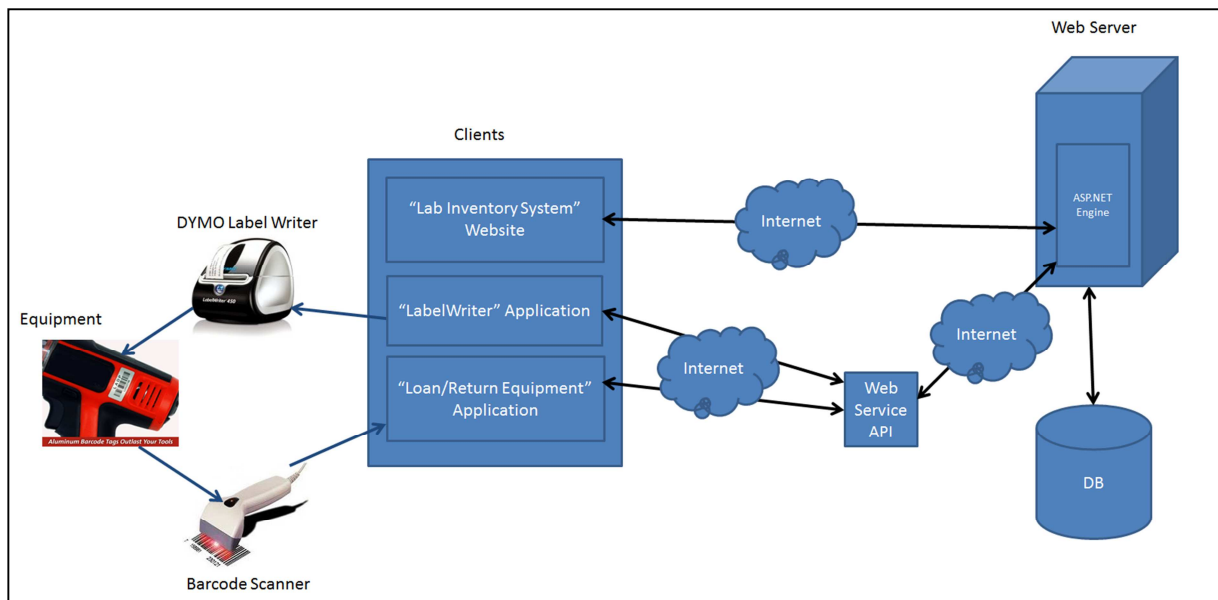


Figure 4-1 System Overview Diagram

4.1.1 Database

Database is used to store information of equipment, user profiles, loans and reservations. SQL Server is used as the DBMS for Lab Inventory System.

4.1.2 “Lab Inventory System” Website

Users can access “Lab Inventory System” website through the internet. It is mainly for managing user profiles, handling loans and reservations and registering equipment and user details. It allows search functionality as well. Before using the website, user has to register as a valid user and log into the website with valid email and password.

4.1.3 “LabelWriter” Application

“LabelWriter” application is a desktop application which is installed in administrator’s computer. DYMO Label writer should be connected and required drivers should be installed in the same computer before using “LabelWriter” application. Administrator can search for registered equipment or users and can print labels for selected items. The application has several default label templates. Hence administrators can use a default label template or their own label template when printing labels and attach the printed label to a particular equipment or user identity card.

4.1.4 “Loan/Return Equipment” Application

“Loan/Return Equipment” application is also a desktop application which is installed in each lab with a barcode scanner and a touch screen monitor. Users can scan equipment and user barcodes using barcode scanner and select whether they need to loan or return equipment. System will save loan/return equipment records in the database through a web service.

Designed UI sketches for “LabelWriter” and “Loan/Return Equipment” applications in analysis phase is shown in Figure 4-2.

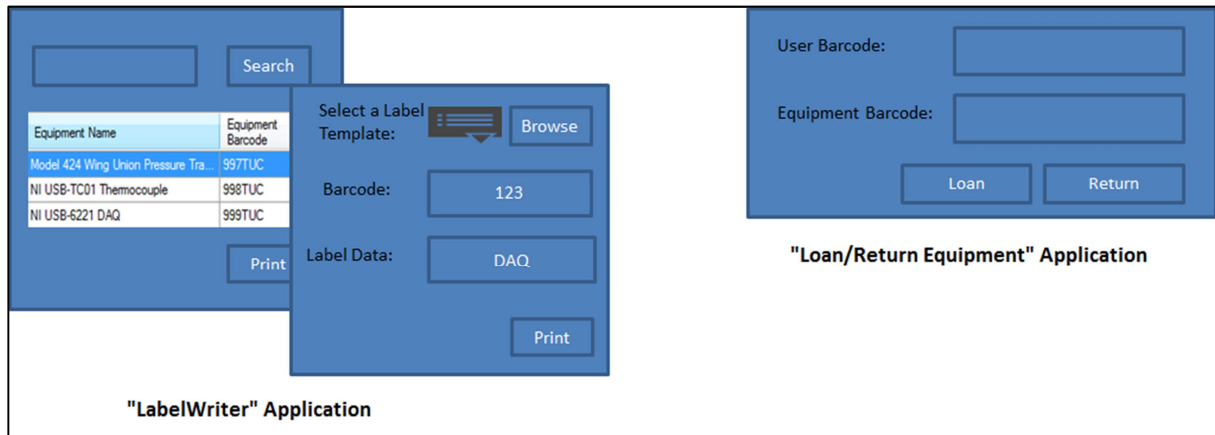


Figure 4-2 UI sketches for “LabelWriter” and “Loan/Return Equipment” applications

4.1.5 Web Service

Web service provides several web methods to retrieve equipment and user details for “LabelWriter” application and to update or save loan/return equipment records through “Loan/Return Equipment” application.

4.2 Requirements for “Lab Inventory System” Website

“Lab Inventory System” website is supposed to use by both “End Users” and “Administrators”.

End users should be able to:

- Create a user profile
- Login
- Log out
- Search for equipment (with filtered search)
- Reserve equipment
- Edit user profile

- View equipment details
- Loan equipment
- Return equipment

Administrators should be able to:

- Create a user profile
- Login
- Log out
- Search for equipment (with filtered search)
- Reserve equipment
- Edit user profile
- View equipment details
- Loan equipment
- Return equipment
- Register equipment
- Edit equipment
- Delete equipment
- Search for customer
- Edit user profile
- Delete user profile
- View existing loans

The requirements have been documented using UML Use Case formats as shown in Figure 4-3 and Figure 4-4 and have been divided according to the types of users that will interact with the system. Figure 4-3 and Figure 4-4 show Use Cases for “End User” and “Administrator” respectively.

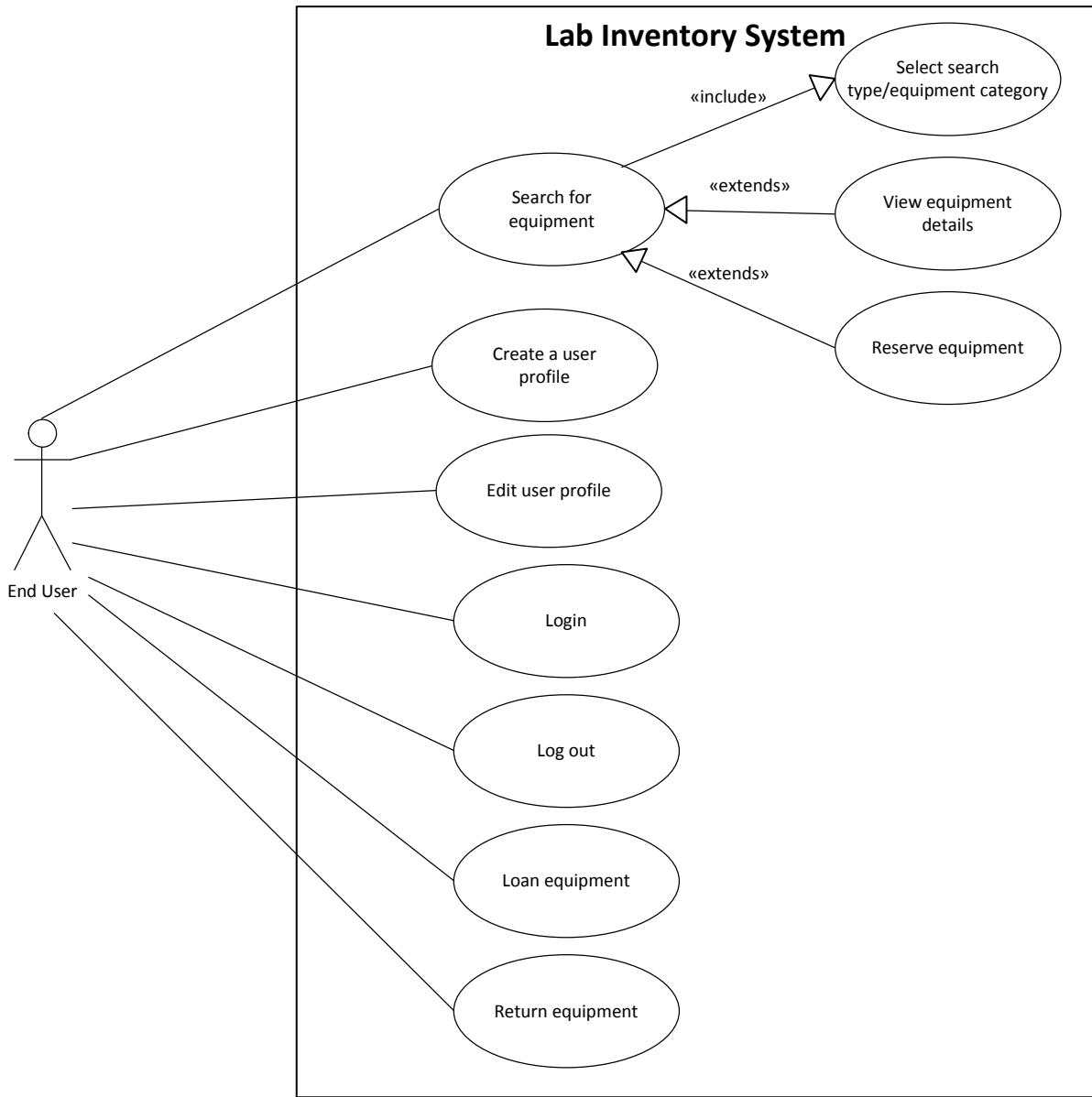


Figure 4-3 End User Use Cases

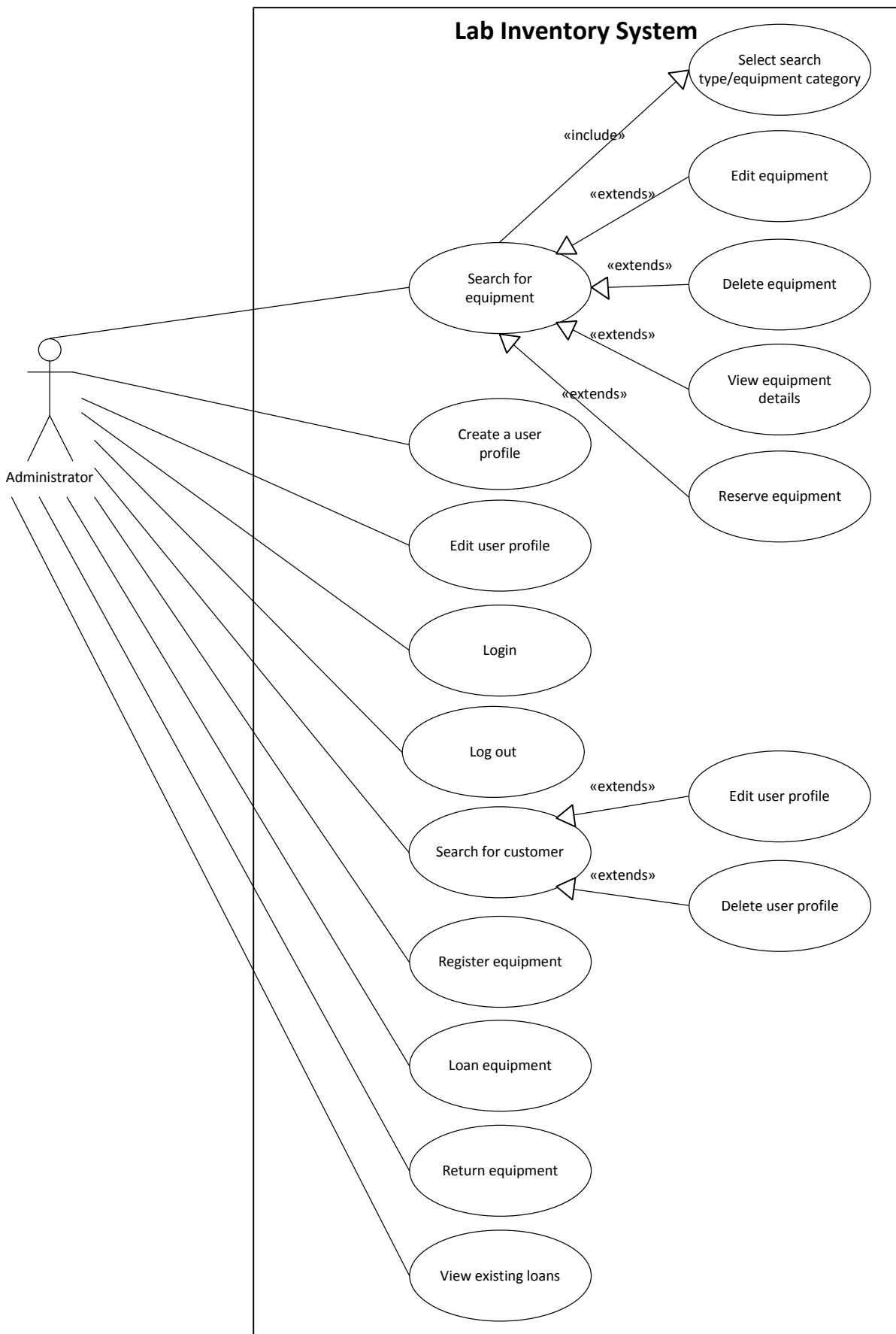


Figure 4-4 Administrator Use Cases

Table 4-1 Fully dressed Use Case document for " Create a user profile" Use Case

Use Case name	Create a user profile
Scope	"Lab Inventory System" website
Actor	End User, Administrator
Preconditions	None
Success Guarantee	Creating a new user profile successfully
Main success scenario	<ol style="list-style-type: none"> 1. Enter values for Email, Password, First Name, Last Name, Student Number, City of Residence, Street, Post Number, Phone Number, Position/ Title, Class, and Barcode fields 2. Select a user type 3. If needed upload an image 4. Click "Register" button 5. Save appropriate records in database successfully
Extensions	<p>5a. Email/ Password/ First Name/ Last Name/ Student Number/ Barcode fields are empty</p> <ol style="list-style-type: none"> 1. Give an error message <p>5b. Invalid email address</p> <ol style="list-style-type: none"> 1. Give an error message <p>5c. Invalid student number</p> <ol style="list-style-type: none"> 1. Give an error message <p>5d. Password mismatch with "Retype Password" field</p> <ol style="list-style-type: none"> 1. Give an error message <p>5e. Entered Email/ Barcode/ Student Number values are already in the database (not unique)</p> <ol style="list-style-type: none"> 1. Give an error message

Fully dressed Use Case document for “Create a user profile” Use Case is shown in Table 4-1. Fully dressed Use Case documents for “Search for equipment” and “View equipment details” Use Cases are in Appendix 2.

At the initial stage, several UI sketches have been designed for “Lab Inventory System” website before moving to design and implementation stages in the sense of easily identifying and describing requirements. These sketches are shown in Figure 4-5, Figure 4-6 and Appendix 3.



The image shows a UI sketch for the "Register New User" page of a "Lab Information System". The page has a dark blue header with the system name and a navigation bar with "Login" and "New User" buttons. The main content area is white and contains two sections: "Login Details" and "User Details". The "Login Details" section has three input fields for "User Name:", "Password:", and "Confirm Password:". The "User Details" section has four input fields for "First Name:", "Last Name:", "City:", and "Street:". A "Register User" button is located at the bottom right of the form.

Figure 4-5 UI Sketch for "Register New User" page

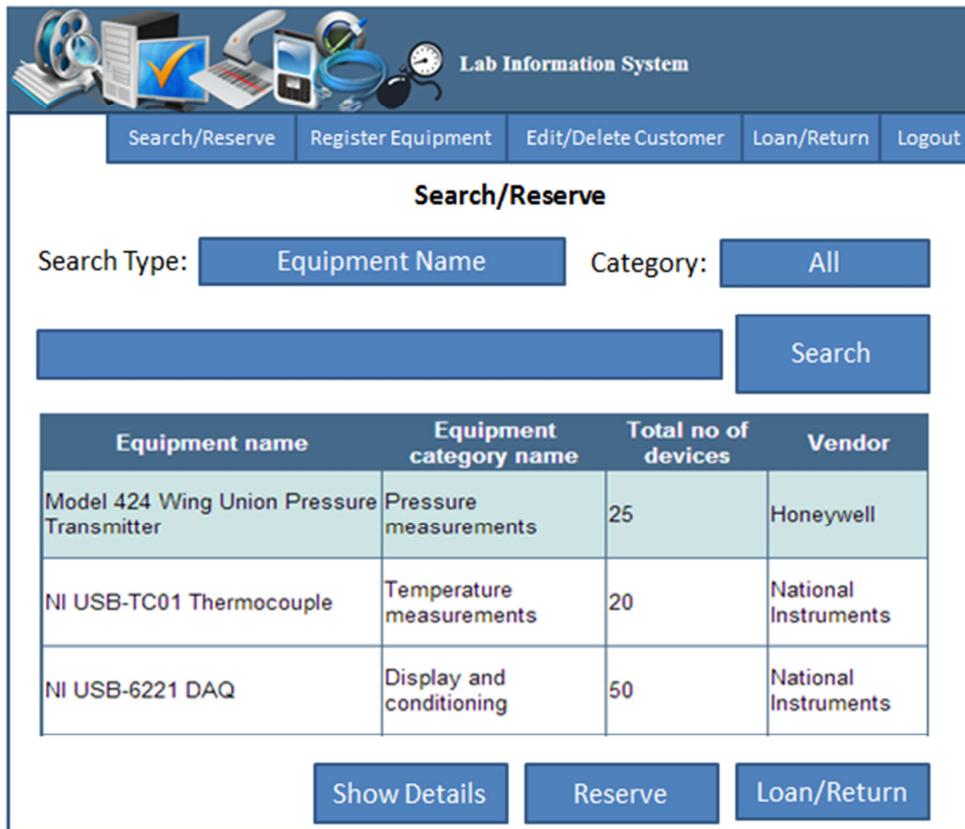


Figure 4-6 UI Sketch for "Search" page

4.3 Requirements for "LabelWriter" Application

"LabelWriter" application is supposed to use by "Administrators". They should be able to:

- Search for equipment or user
- Print label for selected search result

The requirements have been documented using a UML Use Case format as shown in Figure 4-7.

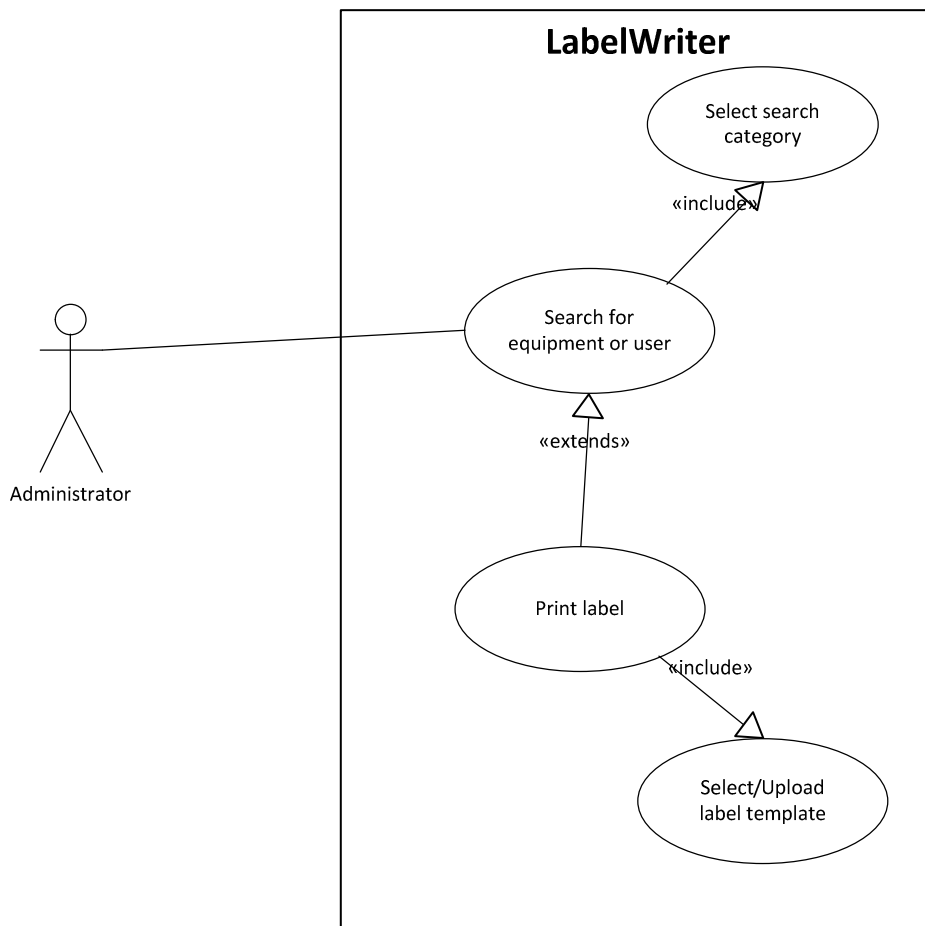


Figure 4-7 Use Case diagram for "LabelWriter" application

Fully dressed Use Case document for “Search for equipment or user” Use Case is shown in Table 4-2. Fully dressed Use Case documents for remaining Use Cases are in Appendix 4.

Table 4-2 Fully dressed Use Case document for " Search for equipment or user" Use Case

Use Case name	Search for equipment or user
Scope	"LabelWriter" application
Actor	Administrator
Preconditions	None
Success Guarantee	Display appropriate search results successfully
Main success scenario	<ol style="list-style-type: none"> 1. Select search category 2. Enter search text 3. Click "Search" button 4. Display appropriate search results successfully
Extensions	3a. If there are no search records, the system will display an appropriate message and allow the user to perform a new search.
Miscellaneous	None

4.4 Requirements for "Loan/Return Equipment" Application

"Loan/Return Equipment" application is supposed to use by both "End Users" and "Administrators", both of them have same accessibility to all the functionalities. The requirements for "Loan/Return Equipment" application can be listed as:

- Loan equipment
- Return equipment

The requirements have been documented using a UML Use Case format as shown in Figure 4-8.

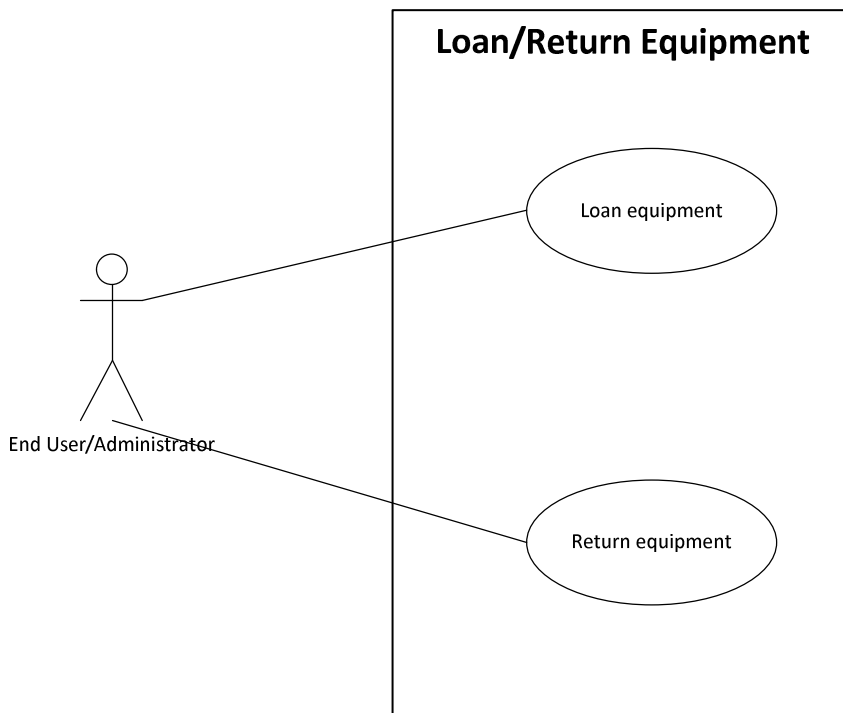


Figure 4-8 Use Case diagram for "Loan/Return Equipment" application

Fully dressed Use Case document for "Loan equipment" Use Case is shown in Table 4-3. Fully dressed Use Case documents for remaining Use Cases are in Appendix 5.

Table 4-3 Fully dressed Use Case document for "Loan equipment" Use Case

Use Case name	Loan equipment
Scope	"Loan/Return Equipment" application
Actor	End User, Administrator
Preconditions	None
Success Guarantee	Save appropriate records in database successfully
Main success scenario	<ol style="list-style-type: none"> 1. Scan equipment barcode 2. Scan user barcode 3. Display name of the user 4. Click "Loan Equipment" button 5. Save appropriate records in database successfully 6. Clear equipment barcode and user barcode textboxes and ready for next operation
Extensions	<p>3a. Invalid user</p> <ol style="list-style-type: none"> 1. Give an error message and clear user barcode textbox <p>4a. Equipment barcode and/or user barcode textboxes are empty</p> <ol style="list-style-type: none"> 1. Give an error message
Miscellaneous	None

5 Design

Analysis phase focuses on doing the “right” thing while design phase focuses on doing the “things” right⁴. The aim of design phase is to map functional requirements to hardware and software environment. In the design phase architecture is established and modules, system components, interaction among components, objects, external systems and interfaces are identified and defined. Hence successful completion of design phase compromises transformation of all requirements into detailed specifications.

During the design phase, following areas will be focused [15].

- Application Architecture
- Detailed Specification
- System Interface Design
- Finalize User Interface
- Test Plans

This chapter specifies database design and application design of Lab Inventory System. Further it details system architecture with detailed system description.

5.1 Application Design

During application design, system architecture is defined and the class diagram is documented for Lab Inventory System.

5.1.1 System Architecture

Figure 5-1 shows the architecture diagram of Lab Inventory System. It contains three tiers as Tier 1(presentation/user layer), Tier 2 (application layer) and Tier 3 (data layer). During an application's life cycle, this three-tier approach provides benefits such as reusability (can share and reuse the components and services), flexibility (because each tier can be managed or scaled independently, flexibility is increased), manageability, maintainability (because each

⁴ Referred from “Object-Oriented Analysis, Design, and Programming using UML and C#” lecture notes 2012 made by Nils Olav at Telemark University College

tier is independent of the other tiers, updates or changes can be carried out without affecting the application as a whole), and scalability [16].

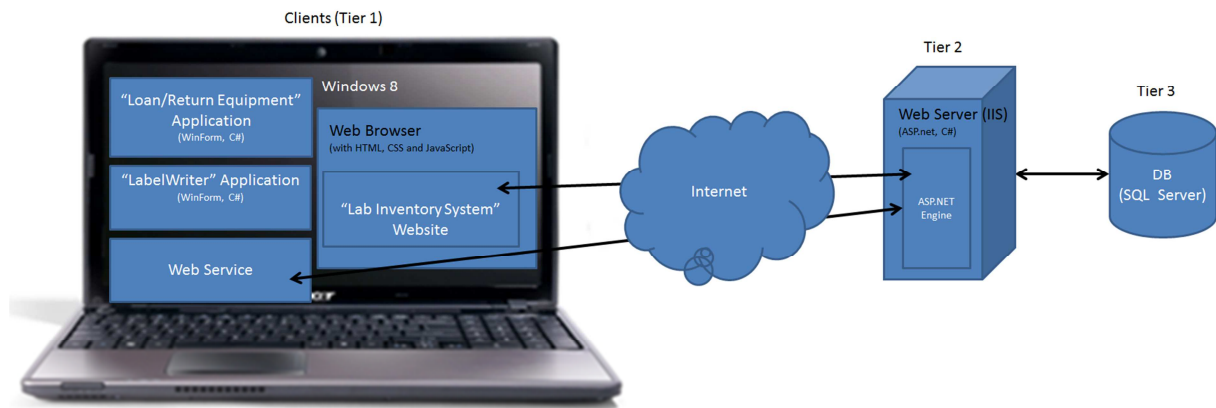


Figure 5-1 System Architecture Diagram [17]

Presentation Tier (Tier 1)

This is the top most tier which presents data to the user. For example in Windows application, WinForm is presentation layer, in Web application, WebForm is presentation layer. For “LabelWriter” and “Loan/Return Equipment” applications, WinForms are the presentation tier. The presentation tier for “Lab Inventory System” website consists of WebForms and web browsers with HTML, CSS and JavaScript.

Application Tier (Tier 2)

Application tier performs application logic. It contains both Business tier and Data Access tier (Data Access tier provides simplified access to data in the database) as shown in Figure 5-2. It processes clients requests from presentation layer and interacts with the database. It consists of the web server, web scripting language and the scripting language engine. Application layer or Tier 2 of Lab Inventory System consists of IIS as web server, C# as web scripting language and ASP.NET engine as scripting language engine. IIS will process the HTTP requests from the clients and the calculation or logic is done by the scripting language (C#) which runs on the scripting engine (ASP.NET engine).

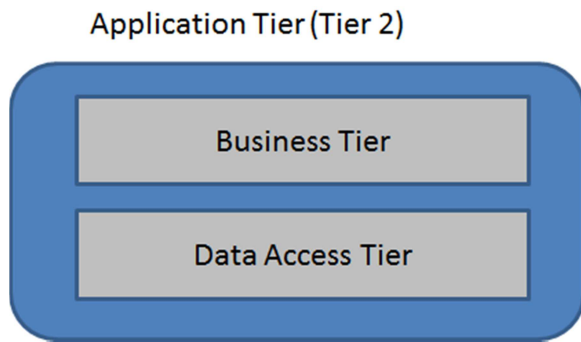


Figure 5-2 Application Tier

Data Tier (Tier 3)

Data tier or Tier 3 of Lab Inventory System consists of DBMS (SQL Server), SQL Management Studio, tables, stored procedures and views. It manages storage and retrieval of data.

Class diagram with its detailed description is in Appendix 7.

5.2 Database Design

ERWin database design diagram for Lab Inventory System is shown in Figure 5-3. The diagram shows identified tables, their columns and relationships among tables. According to the Figure 5-3, thirteen tables have been identified with their primary keys. Primary key columns set as auto increment fields which allow a unique number to be generated when a new record is inserted into a table. The core table of the database is "EQUIPMENT". This tracks information about particular equipment and each field holds a single piece of information about the equipment.

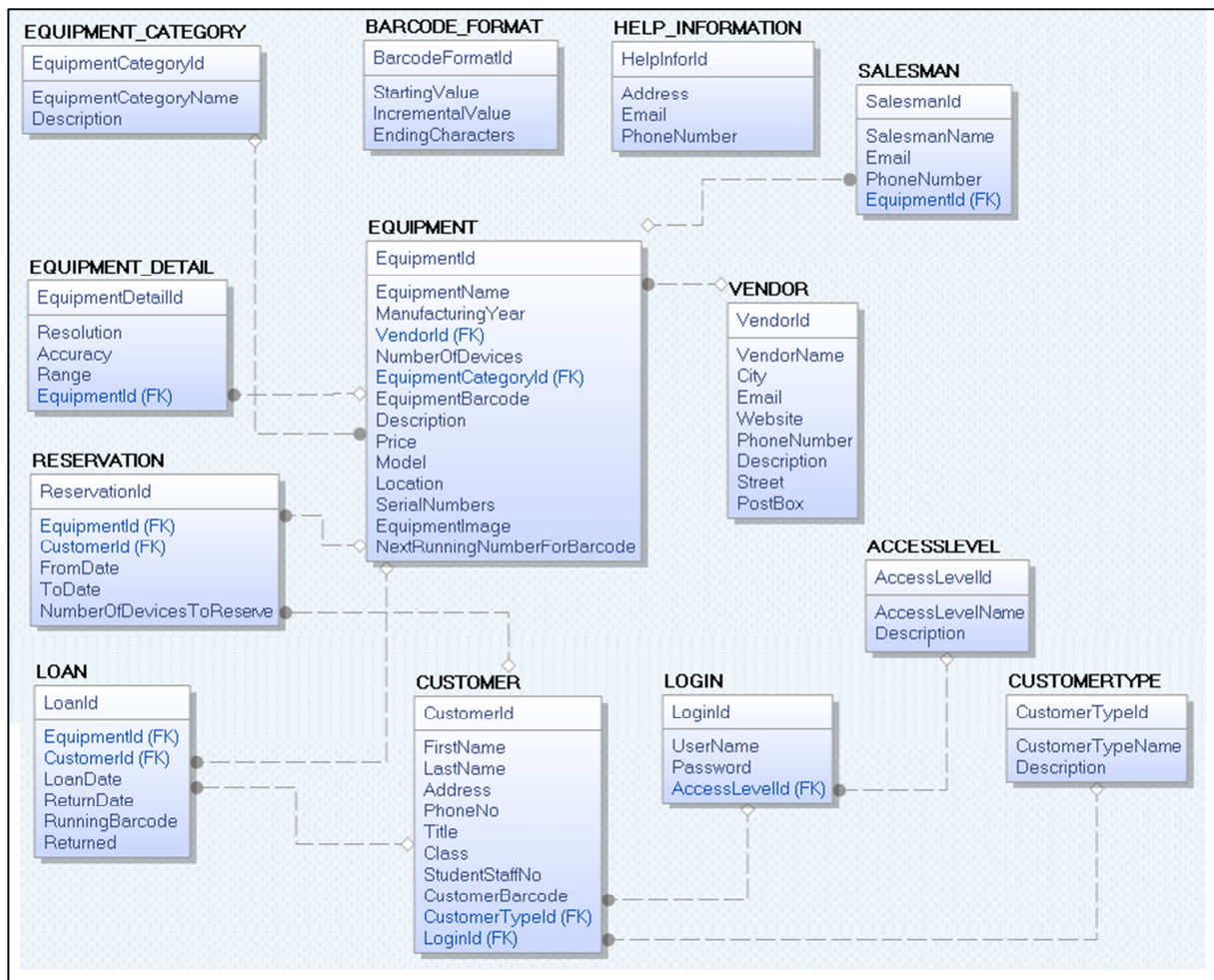


Figure 5-3 ERWin Database Diagram

FK standards for Foreign Key which indicates a relationship between two tables. “CUSTOMER” and “LOGIN” tables store registered user information and login information of particular user. “ACCESS_LEVEL” table stores information about different access levels, whether the customer has read, write or owner accessibility. “CUSTOMER_TYPE” table stores data related to different customer categories, whether the customer is a student, a staff member or an administrator. “LOAN” and “RESERVATION” tables track information related to loan and reservation of equipment. A customer/user can have zero or more reservations or loans, and all reservations/ loans must be linked to an existing customer in “CUSTOMER” table and an existing equipment in “EQUIPMENT” table. “BARCODE_FORMAT” table is used to generate equipment barcode during registration process of each equipment. “VENDOR” table stores details of different vendors while each record in “SALESMAN” table indicates salesman details of particular equipment. “EQUIPMENT_CATEGORY” table stores details of different equipment categories.

In order to allow for maximum readability, a standard naming practice is used during database design. Table name is always in capital letters and it can be multiple words where any spaces between words are separated with an underscore. For example, the table name to track equipment categories would be “EQUIPMENT_CATEGORY”. Column name always starts with a capital letter and if it has multiple words, a new word always begins with a capital letter without a space. For example, the column name to record the equipment name would be “EquipmentName”.

	Column Name	Data Type	Allow Nulls
▶	BarcodeFormatId	int	<input type="checkbox"/>
	StartingValue	int	<input type="checkbox"/>
	IncrementalValue	int	<input type="checkbox"/>
	EndingCharacters	nvarchar(50)	<input checked="" type="checkbox"/>

Figure 5-4 " BARCODE_FORMAT " table

As shown in Figure 5-4, acceptable data types have been assigned to table columns with indicating whether the column accepts null values or not. Choice of data type will determine what types of information can be stored in these columns. For example “StartingValue” might allow only integers whereas “EndingCharacters” might allow alphanumeric characters. Database diagram generated using SQL Server is in Appendix 6.

Part 4: Implementation, Testing and Deployment

Part 4 contains “Implementation”, “Testing”, “Deployment” and “Conclusion and Future Work” chapters. It details implementation process, testing and deployment methods of Lab Inventory System. Further it presents a short conclusion about thesis work and suggestions and improvements for future work of the system.

6 Implementation

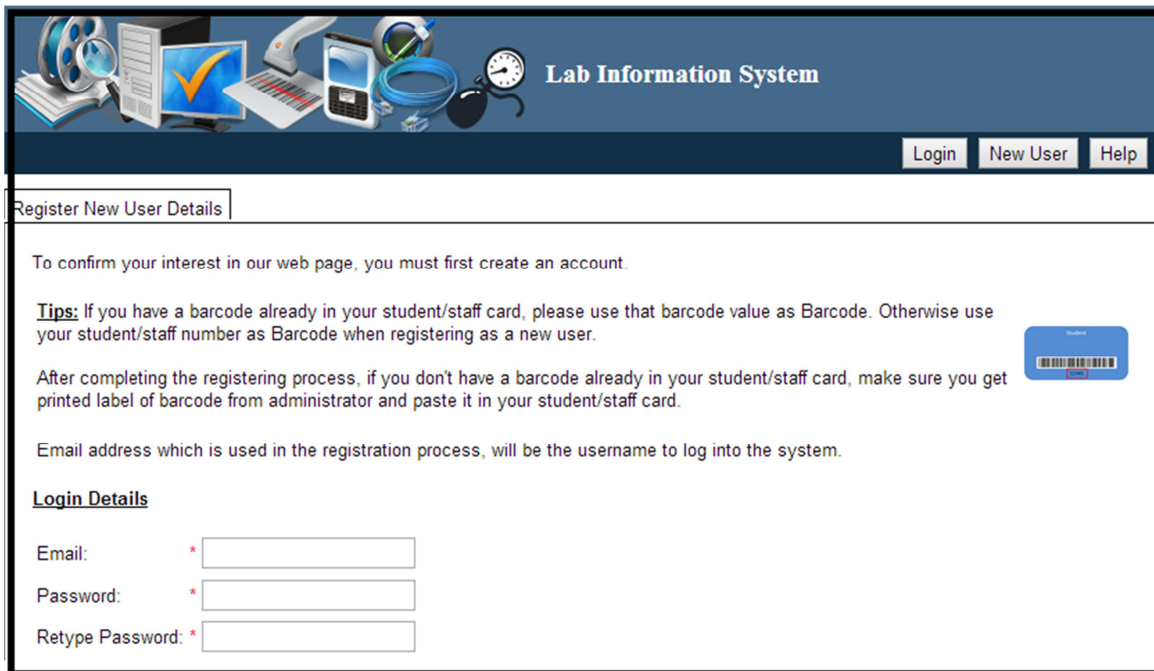
Implementation is the process of writing source code for a system. The objective is to transform the design into program and code modules. The final deliverables of implementation phase are source code and related documents.

This chapter details the implementation process of “Lab Inventory System” website, “LabelWriter” application, “Loan/Return Equipment” application and database scripts (scripts, stored procedures and views). Further it describes the different technologies, methods and patterns used during Lab Inventory System development. It also presents the best coding practices (naming conventions, commenting and exception handling) used during the implementation process and source code control in TFS.

6.1 Implementation of “Lab Inventory System” Website

“Lab Inventory System” website is implemented using ASP.NET. When using “Lab Inventory System” website for the first time, user has to register as New User by inserting user details and log into the system using valid Email address and Password as shown in Figure 6-1. The website uses “Session” objects for user authentication and validation. Hence it is not needed to recheck the user validation from the database, every time when the user redirects to a different page. Therefore user validation can be easily done with “Session” objects. After log into the web site, it will be redirected to “Loan/Return Equipment” page where user can borrow or return equipment by entering required details. Other than that user can search for equipment, register equipment, edit/delete user profiles, reserve equipment, view existing loans and view equipment details using top menu bar as shown in Figure 6-2. These functionalities are displayed depending on the user access level. A Help page is also available in the website with a proper user guide. Screen dumps of several other web pages are in Appendix 8.

During the implementation process of website, a separate JavaScript file is used for client side scripting. For example JavaScript is used to show/hide the calendar element in “Reservation” page as shown in Figure 6-3. JavaScript allows fast and easy web experience to the user as the code is executed on the browser instead of the web server. A separate CSS file is used for styling web pages because of its faster loading and easy maintenance than doing styling in raw HTML file. Figure 6-4 shows the Master Page used for “Lab Inventory System” website. This creates a consistent layout throughout all the pages in the website.



Lab Information System

Login New User Help

Register New User Details

To confirm your interest in our web page, you must first create an account.

Tips: If you have a barcode already in your student/staff card, please use that barcode value as Barcode. Otherwise use your student/staff number as Barcode when registering as a new user.

After completing the registering process, if you don't have a barcode already in your student/staff card, make sure you get printed label of barcode from administrator and paste it in your student/staff card.

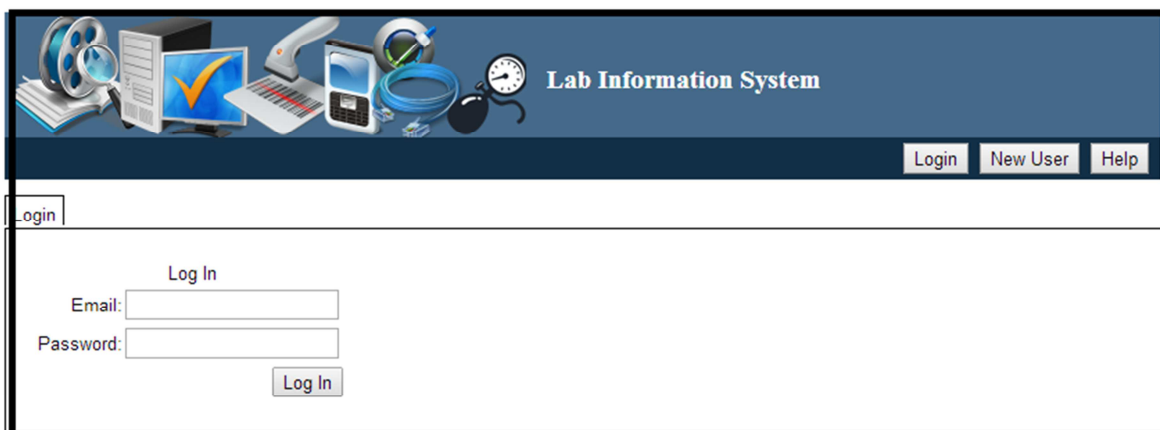
Email address which is used in the registration process, will be the username to log into the system.

Login Details

Email: *

Password: *

Retype Password: *



Lab Information System

Login New User Help

login

Log In

Email:

Password:

Log In

Figure 6-1 "Register New User" and "Login" pages

Logged in as: hans.p.halvorsen@hit.no

Lab Information System

Loan/Return Equipment
Search Equipment
Register Equipment
Edit/Delete User
Log Out
Help

Logged in as: hans.p.halvorsen@hit.no

Lab Information System

Loan/Return Equipment
Search Equipment
Register Equipment
Edit/Delete User
Log Out
Help

Loan/Return Equipment

You are ready to loan/return instruments/ lab equipment.

Tips: You can scan user barcode and/or equipment barcode using barcode scanner.

Equipment Barcode: *

User Barcode: *

Search/ Reserve

You are searching: The Lab Equipment Database

Search Type: Equipment Category:

Displaying top 100 records from the database....
7 search results found.
 Search Keyword: "", Search Type: "Equipment Name", Equipment Category: "All"

Equipment name	Equipment category name	Total no of devices	Vendor	Preview image
Model 424 Wing Union Pressure Transmitter	Pressure measurements	25	Honeywell	
NI USB-TC01 Thermocouple	Temperature measurements	20	National Instruments	
NI USB-6221 DAQ	Display and conditioning	50	National Instruments	
USB-6008	Temperature measurements	50	National Instruments	

Figure 6-2 "Loan/Return Equipment" and "Search Equipment" pages

Model 424 Wing Union Pressure Transmitter

You are ready to reserve an instrument/ lab equipment.

Equipment Barcode: *

Number of Devices to Reserve: *

User Barcode: *

From Date: *

mai 2014						
man	tir	ons	tor	fre	lør	søn
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

To Date: *

Figure 6-3 JavaScript is used to show/hide the calendar element



Figure 6-4 Master Page

“Lab Inventory System” website uses a “Web.config” file to save main settings and configurations of website. For example it contains database connection string and several application settings like default access level, default customer type, etc.

Validator controls have been used for input validation in Lab Inventory System. A validator is a visual ASP.NET control that checks a specific validity condition of another control and generally appears to the user as a piece of text that displays or hides depending on whether the control it is checking is in error [18]. “RequiredFieldValidator” has been used to avoid empty controls, “RegularExpressionValidator” has been used to validate data while “CompareValidator” has been used to compare values in two controls. As shown in Figure 6-5, “RequiredFieldValidators” have been used for “Email”, “Password” and “Retype Password” fields. For “Email” field, a “RegularExpressionValidator” has also been used. “CompareValidator” is used to compare “Password” and “Retype Password” fields to check whether there is a password mismatch.

The code structure of “Lab Inventory System” website is shown in Figure 6-6.

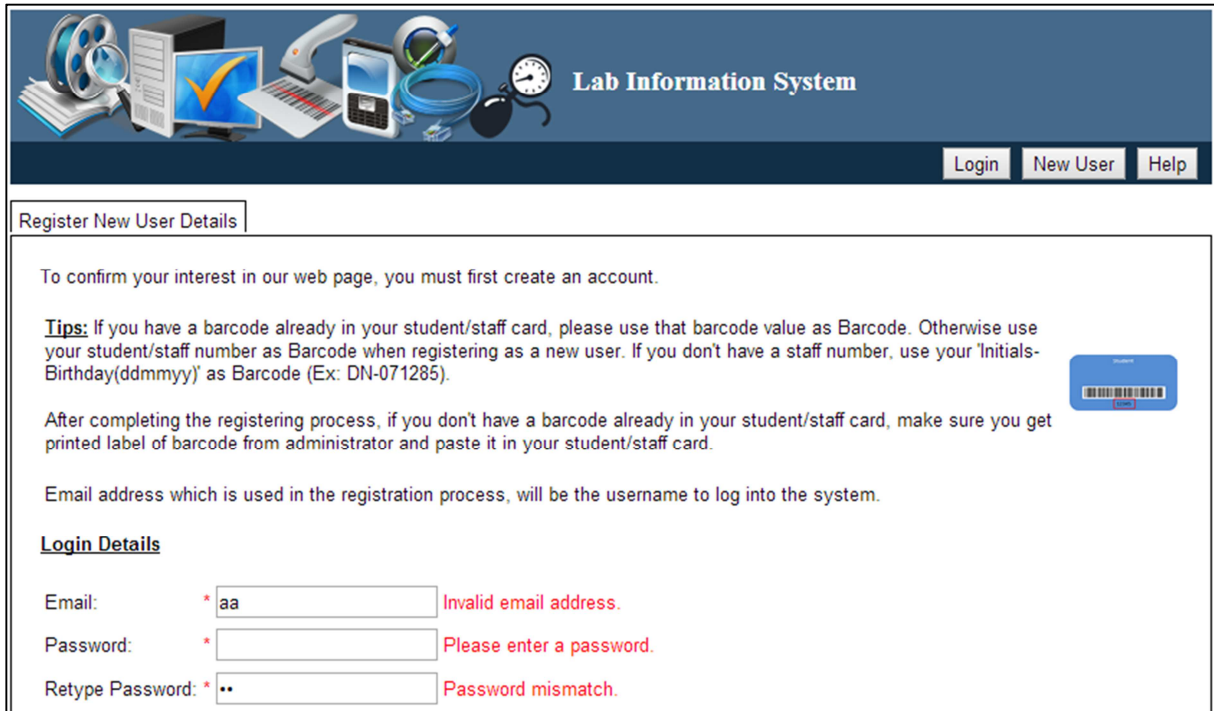


Figure 6-5 Validating Login Details

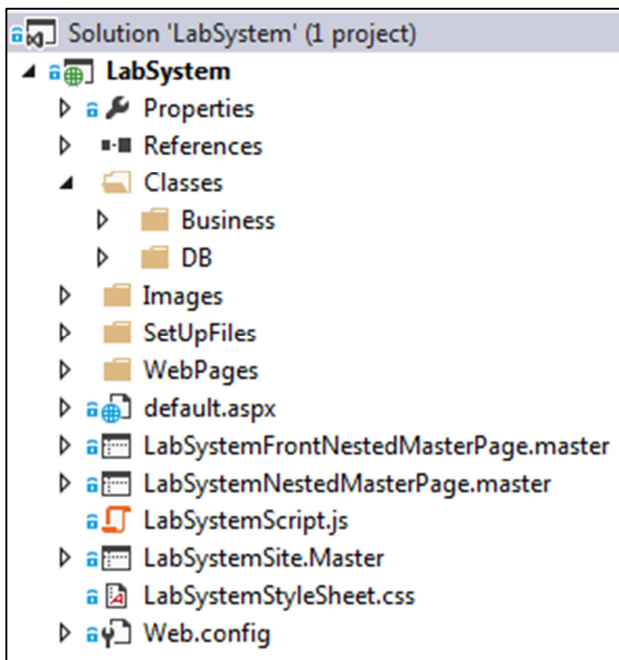


Figure 6-6 Code structure of "Lab Inventory System" website

6.2 Implementation of Web Service

A web service is implemented using ASP.NET to handle interactions between both “LabelWriter” and “Loan/Return Equipment” applications and database. This web service exposes several web methods such as:

- Retrieve equipment and user details from the database for “LabelWriter” application
- Insert/update loan/return equipment records in the database through “Loan/Return Equipment” application
- Retrieve user name for “Loan/Return Equipment” application
- Retrieve and update running barcode value of particular equipment through “LabelWriter” application

If several numbers of devices of same equipment are available, then the barcode value of each device is printed with a running number. For example,

Equipment Name – USB 6008

Equipment Barcode – 1000TUC

Number of devices – 3

Barcode values for devices – 1000TUC-0, 1000TUC-1 and 1000TUC-2

Hence latest running barcode value of particular equipment will be saved into the database through the web service.

Exposed web methods are shown in Figure 6-7. Web methods with their input and return parameters are in Appendix 9.



Figure 6-7 Web Service

6.3 Implementation of “LabelWriter” and “Loan/Return Equipment” Applications

“LabelWriter” and “Loan/Return Equipment” applications are implemented using WinForms in Visual Studio 2013. Implemented “Loan/Return Equipment” application is shown in Figure 6-8. Users can enter equipment and user barcodes as input values and these records will be saved to the database with clicking “Loan Equipment” or “Return Equipment” buttons.

Figure 6-9 shows “LabelWriter” application where users can search for both equipment and users and print labels with a barcode and label data. Default label templates of “LabelWriter” application (Small, Medium and Large label templates) are shown in Figure 6-10. Small label template has only the barcode field. Both medium and large templates have barcode and label data fields where user can enter text for label data field. If the search category is user, only the medium label template is available. For equipment search category, user can select one from default label templates or can upload their own label template.

Both applications use separate “App.config” files to save main settings and configurations of application. For example it contains web service url and several application settings like file paths of label templates, etc.

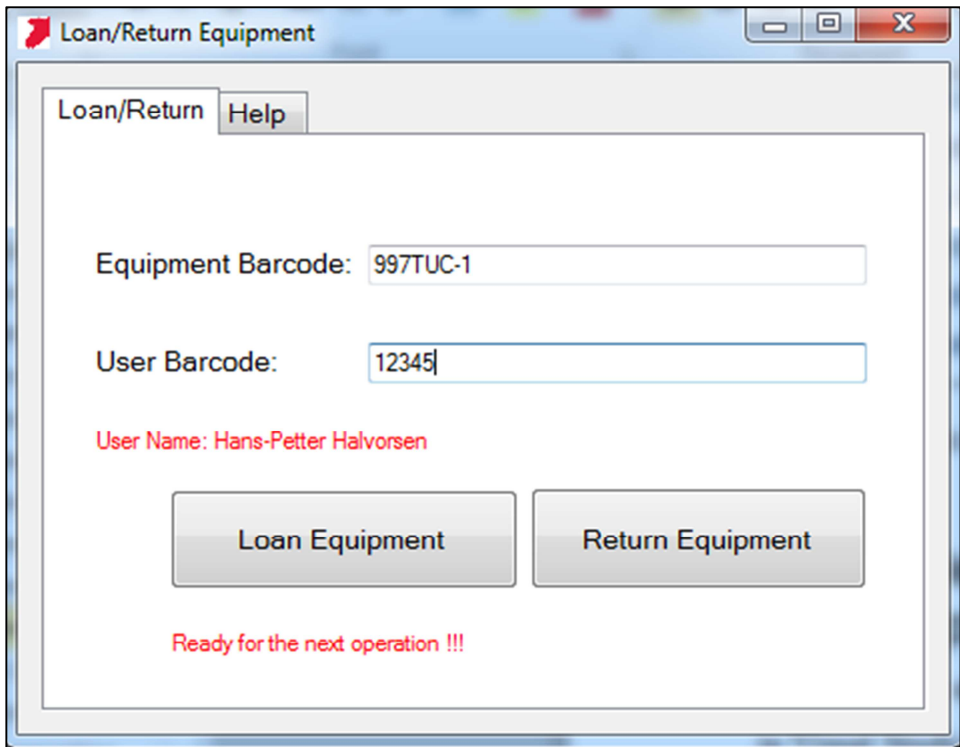


Figure 6-8 “Loan/Return Equipment” Application

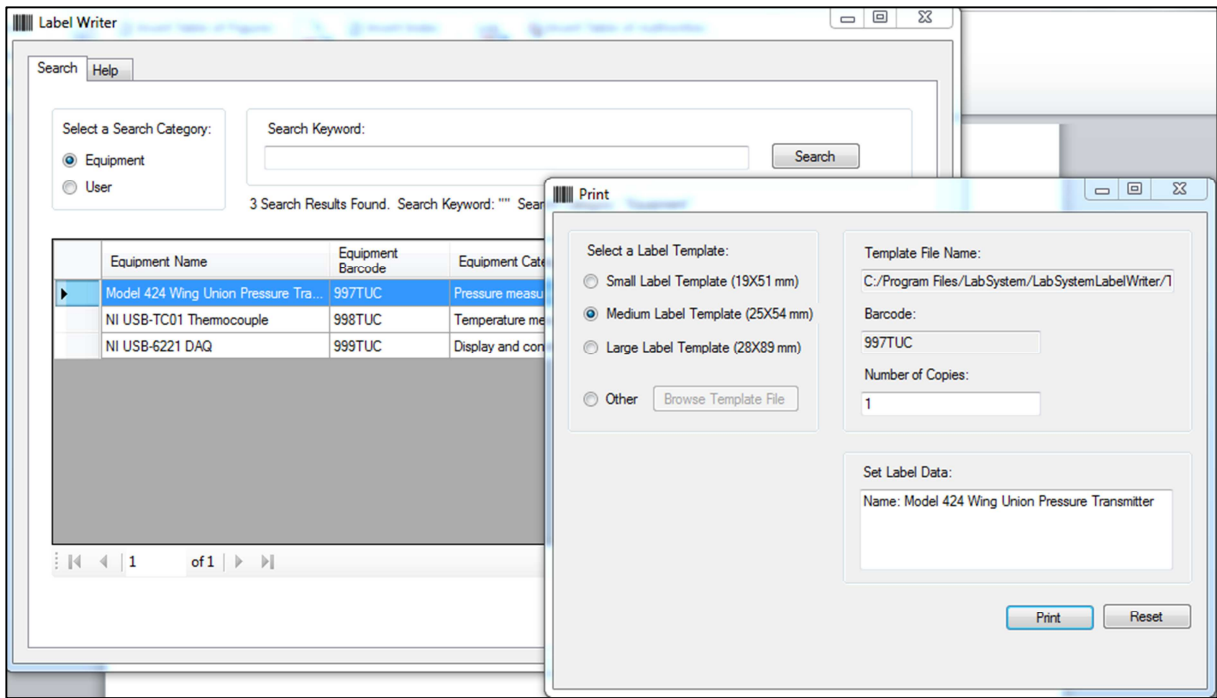


Figure 6-9 “LabelWriter” Application



Figure 6-10 Default label templates of “LabelWriter” Application

6.4 Database Scripts, Stored Procedures and Views

Two separate scripts have been created for generating tables and inserting dummy data. These scripts can be executed in SQL Management Studio. Though creating scripts take some time, the advantage is that it can be reused each and every time when creating a new database for Lab Inventory System without adding new tables and data manually. Several stored procedures have been created to implement logics for:

- Registering user details
- Registering equipment details
- Save reservations
- Retrieve equipment details
- Retrieve user details
- Update/delete user details
- Update/delete equipment details, etc.

A DATABASE VIEW has been implemented to retrieve data from “LOAN”, “CUSTOMER” and “EQUIPMENT” tables. A VIEW is a virtual table that collects data from several tables

and it can be used when implementing complex queries to retrieve some data from different tables. It has several advantages than tables as it reduces complexity and space usage.

INNER JOIN keyword is used to return rows when there is at least one match on “CustomerId” in “LOAN” and “CUSTOMER” tables and “EquipmentId” in “LOAN” and “EQUIPMENT” tables as shown in Figure 6-11.

```
-- Create date:17/4/2014
-- Description:Get Existing Loan Data
-- =====
IF EXISTS (SELECT name FROM sysobjects WHERE name = 'GetLoanData' AND type = 'V')
  DROP VIEW GetLoanData
GO

CREATE VIEW GetLoanData
AS
SELECT
  RunningBarcode,
  LoanDate,
  ReturnDate,
  (FirstName + ' ' + LastName) as Name,
  PhoneNumber,
  EquipmentName,
  Returned
FROM
  LOAN
  INNER JOIN CUSTOMER ON
  LOAN.CustomerId = CUSTOMER.CustomerId
  INNER JOIN EQUIPMENT ON
  LOAN.EquipmentId = EQUIPMENT.EquipmentId
GO
```

Figure 6-11 View "GetLoanData"

6.5 Source Code Control

TFS is used to manage source code of Lab Inventory System. The folder structure in TFS is shown in Figure 6-12. Database scripts, stored procedures and views are checked in to “Database” folder. The source code of Lab Inventory System is available in “Code/Application” folder. All the documents including thesis report are checked in to “Documents” folder and “LabelWriter” and “Loan/Return Equipment” setup files are available in “SetUpFiles” folder.

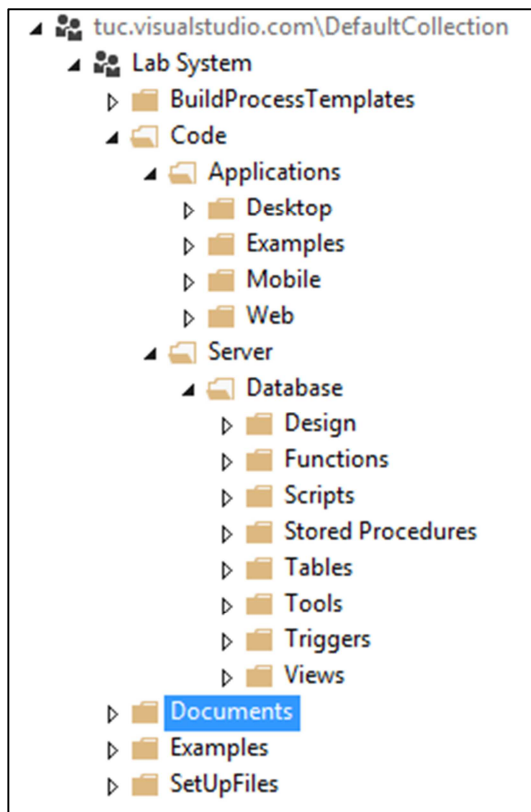


Figure 6-12 Folder Structure in TFS

6.6 Best Coding Practices

During the implementation phase of Lab Inventory System, several coding practices have been used to maximize the code readability. Proper naming conventions have been used throughout the implementation process with meaningful names. For example, function names or class names always start with a capital letter, while variable names start with a simple letter (the class name to handle customer operations would be “Customer” while a variable name to save customer name would be “name”). Other than proper naming conventions, commenting and exception handling can also be considered as best coding practices.

6.6.1 Commenting

During the implementation process of Lab Inventory System, code comments have been written for functions, classes and logics. Commenting describes the purpose of a code block. Proper use of commenting improves maintenance and readability of source code and helps to find bugs faster.

6.6.2 Exception Handling

Exception handling is a mechanism to detect and handle run time errors. During the implementation process of Lab Inventory System, “try” and “catch” blocks are used to handle exceptions as shown in Figure 6-13.

```
try
{
    Customer customer = new Customer();

    customer.StudentStaffNumber = int.Parse(txtStudentNo.Text);
    customer.CustomerBarcode = txtCustomerBarcode.Text;
    customer.CustomerTypeId = int.Parse(dropDownCustomerType.SelectedValue);

    customer.RegisterCustomer(customer);

    lblMsg.Text = "Successfully registerd in database!!!";
}
catch (SqlException exception)
{
    if (exception.Message.Contains("Cannot insert duplicate key in object 'dbo.LOGIN'"))
    {
        lblMsg.Text = "Cannot proceed the operation as Email is already exist.";
    }
    else if (exception.Message.Contains("Cannot insert duplicate key in object 'dbo.CUSTOMER'"))
    {
        lblMsg.Text = "Cannot proceed the operation as " + lblStudentNo.Text.Replace(":", "");
    }
}
catch (Exception ex)
{
    ClearRegUserData();
    lblMsg.Text = ex.Message;
}
```

Figure 6-13 Exception handling using "try" and "catch"

7 Testing

Testing is the process of executing a software system to determine whether it matches its specification or meets its expected results. It is a very effective way to show the presence of bugs (errors, faults or failures in the software system) and understand the risk of software implementation. Hence it is a process of validating and verifying software application. Typical goals of testing are:

- Check if critical functionalities work
- It helps to determine what is most important to end users
- Improve reliability and maintainability of the software application
- Ensure better quality software application

To achieve a reliable and perfect software system, testing is really important as it assesses the quality of the application.

This chapter describes testing methods used for Lab Inventory System and bug tracking with TFS during implementation and testing phases.

7.1 Testing of Lab Inventory System

Testing of Lab Inventory System is vital important as it ensures that the system meets its specified requirements without any failures. Therefore the application was deployed in test servers before going to production or live servers and was tested by several end users (several students and staff). Following test methods have been used during the testing process of Lab Inventory System.

- Integration Testing – test “Lab Inventory System” website, “Loan/Return Equipment” and “LabelWriter” applications independently with hardware.
- System Testing – test the whole system
- Regression Testing – test the system after doing any modifications to the code

As shown in Figure 7-1, “Lab Inventory System” website was hosted in a test server. It was tested with different scenarios in different web browsers. For example it was used to register new equipment and users. Further Lab Inventory System Station with a touch screen monitor, a barcode scanner, a DYMO Label Writer and a small size computer was configured in a lab to test “Loan/Return Equipment” and “LabelWriter” applications as shown in Figure 7-2.

Touch screen monitor was used to access “Loan/Return Equipment” and “LabelWriter” applications, for printing labels for registered equipment or user and loaning or returning equipment.

TFS is used to track and manage bugs, tasks and work items of Lab Inventory System during implementation and testing phases.

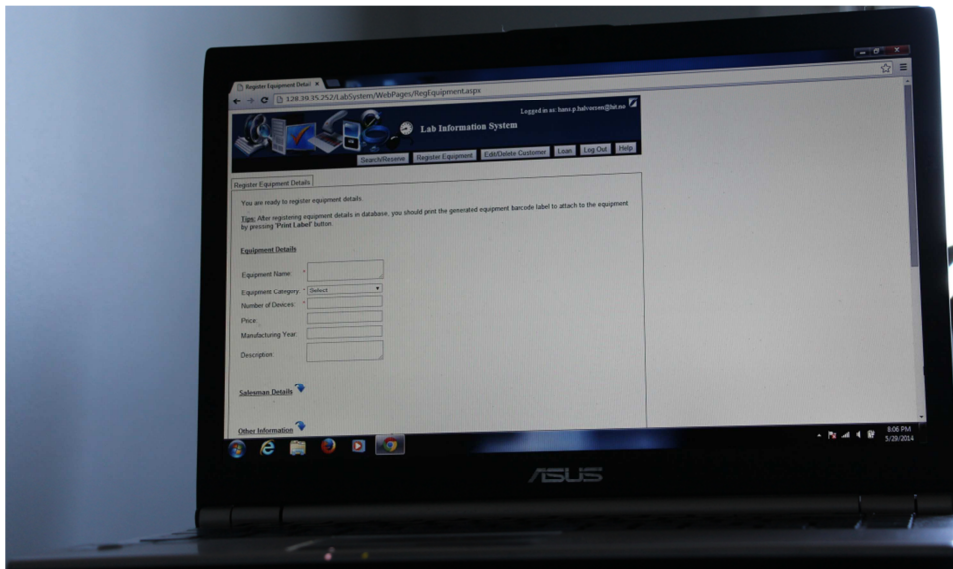


Figure 7-1 "Lab Inventory System" website is hosted in a test server



Figure 7-2 Lab Inventory System Station

7.1.1 Bugs/Tasks Tracking with Team Foundation Server

TFS provides facilities for testing and bug tracking. It uses different types of work items to track different types of work, such as customer requirements, product bugs, and development tasks [19].

Team Explorer is the interface to the TFS, connects Visual Studio to team projects that were created on TFS or Visual Studio Online [20]. It is used to manage source code, work items, and builds as shown in Figure 7-3.

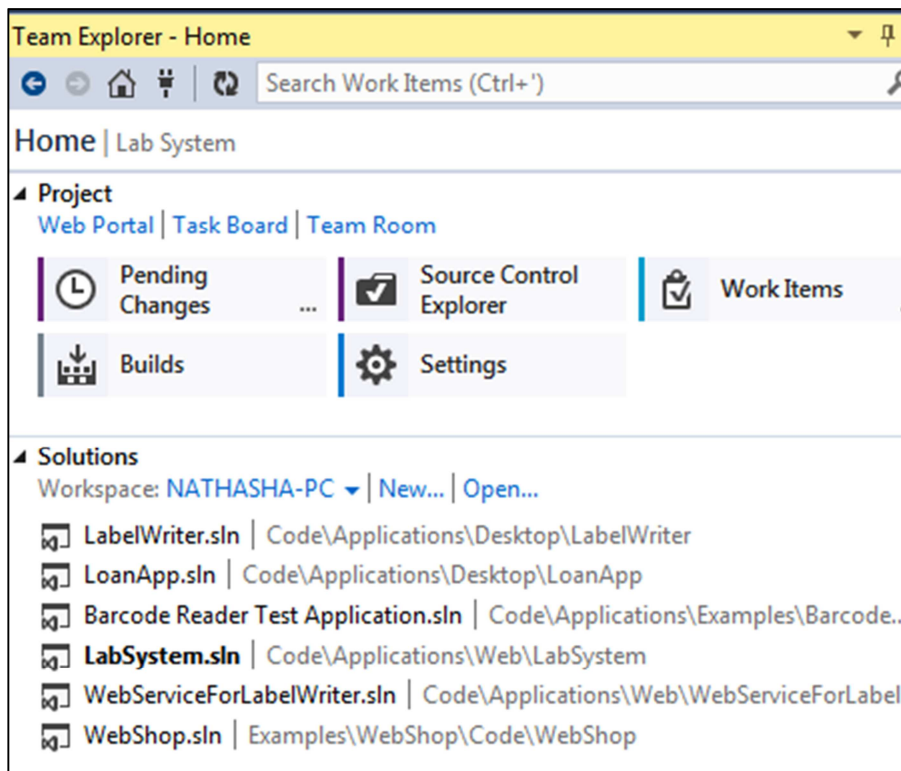


Figure 7-3 Team Explorer

Query Results: 5 items found (1 currently selected).				
Title	Priority	Severity	Created By	
Edit Equipment is not working	1	1 - Critical	Hans-Petter Halvorsen	
After Deleting: go back to list	2	3 - Medium	Hans-Petter Halvorsen	
Existing loans cannot be seen	2	3 - Medium	Hans-Petter Halvorsen	
Online Help missing	2	3 - Medium	Hans-Petter Halvorsen	
Improved Names for Loan and Printer Apps	2	3 - Medium	Hans-Petter Halvorsen	

Figure 7-4 Bug list of Lab Inventory System

Figure 7-4 and Figure 7-6 show the bug list and task list of Lab Inventory System. A bug consists of a title, a priority level, a severity level, a state (whether it is active or resolved), reason, a detailed description (steps to recreate the bug) and comments or improvements. Task has several additional fields such as effort (Original estimate, remaining hours and completed hours) and implementation. Overview of a bug is shown in Figure 7-5. Bugs or tasks were fixed according to their priority level. According to Figure 7-7, considerable

amount of reported bugs and tasks have been fixed, therefore the Lab Inventory System is good enough to use.

Bug 86 : Existing loans cannot be seen

Existing loans cannot be seen

STATUS
Assigned To <No one>
State Active
Reason New

CLASSIFICATION
Area Lab System
Iteration Lab System\Master Thesis 2014\May

PLANNING
Stack Rank <None>
Priority 2
Severity 3 - Medium

REPRO STEPS SYSTEM INFO TEST CASES

We need to have a place where we can see existing loans.
Options: See all loans, see loans for one specific equipment, see loans for one specific user

HISTORY ALL LINKS ATTACHMENTS

Type your comment here.

DISCUSSION ONLY ALL CHANGES
(no entries with comments)

Figure 7-5 Overview of a bug

Task 30 Tasks [Results] Bug 77 Loan.aspx Bug 86 Bugs [Results]

Save Results Save Query Open in Microsoft Office Edit Query Column Options

Query Results: 15 items found (1 currently selected).

Title	Priority	Created By
Encrypt Password in Database	1	Hans-Petter Halvorsen
Management Page where you can add/edit Equipment Categories, Access Levels, etc.	2	Hans-Petter Halvorsen
Vendor & Salesman: Should be possible to create and edit them independently of the Equipm...	2	Hans-Petter Halvorsen
Edit Equipment	1	Dona Nathasha Nakandalage
Add time to From/To date in reservation page	2	Dona Nathasha Nakandalage
Loan Black List: list with loans that has expired. Possible to send e-mail, extend deadline, etc	2	Hans-Petter Halvorsen
Online Help/User Guide when click the Help button. How to use the system!	2	Hans-Petter Halvorsen
More Hints & Tips in the different Web Pages in order make it easier to use	2	Hans-Petter Halvorsen
Register Equipment: More fields needed, ref. Eivind Fjeldalen, eg. Serial Number, etc	1	Hans-Petter Halvorsen
Make it more intuitive and userfriendly, improve GUI (images, fonts, colors, etc)	2	Hans-Petter Halvorsen
Loan/Return: Images of User and Equipment should appear when scanning barcode to make it...	2	Hans-Petter Halvorsen

Figure 7-6 Task list of Lab Inventory System

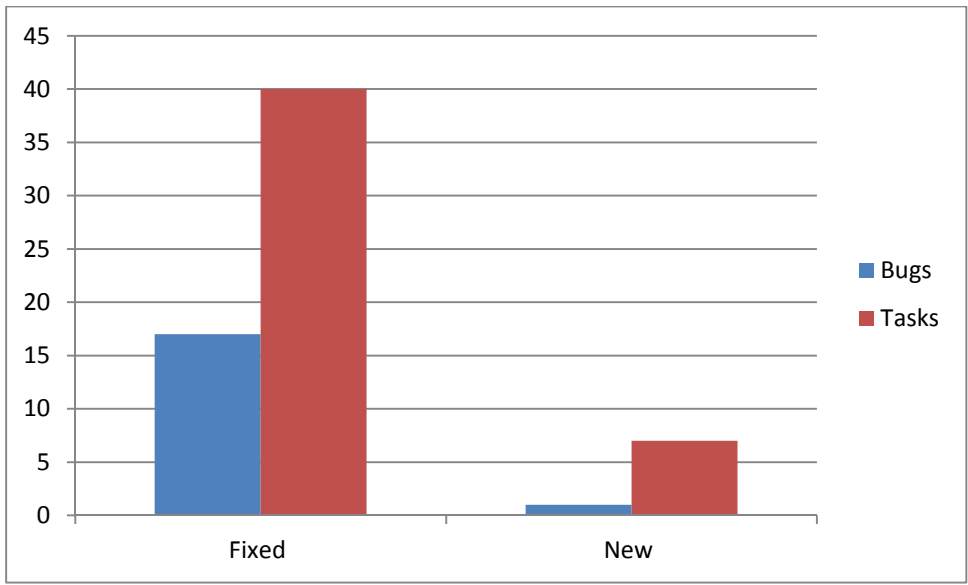


Figure 7-7 Graph of Fixed vs New “bugs and tasks” of the latest release

8 Deployment

Deployment is the process of making finished software available to all users, in other words placing the solution into production environment. It is one of the final phases of software development process. Deployment phase heavily depends on the type of application, IT infrastructure and users. It refers to all the activities that make the software system available to use such as [21]:

- Creating installation packages
- Documentation (Installation Guide)
- Installing the software

This chapter describes deployment methods and steps used for “Lab Inventory System” website, web service, “Loan/Return Equipment” application and “LabelWriter” application.

8.1 Deploy “Lab Inventory System” Website and Web Service

“Publish Web” tool which is part of Visual Studio 2013, is used to deploy “Lab Inventory System” website and web service to a target server. It precompiles the pages and code that are in the website project and writes the compiler output to a specified local folder, so that it is easy to copy this folder to the target server (test server or production server) and configure it in IIS. Figure 8-1 shows different publish methods in “Publish Web” tool.

It is also necessary to configure/update “Web.config” file with latest or real configurations before hosting the website in IIS. For example check the connection string of database is correct with production environment. Figure 8-2 shows the “Web.config” file of “Lab Inventory System” website.

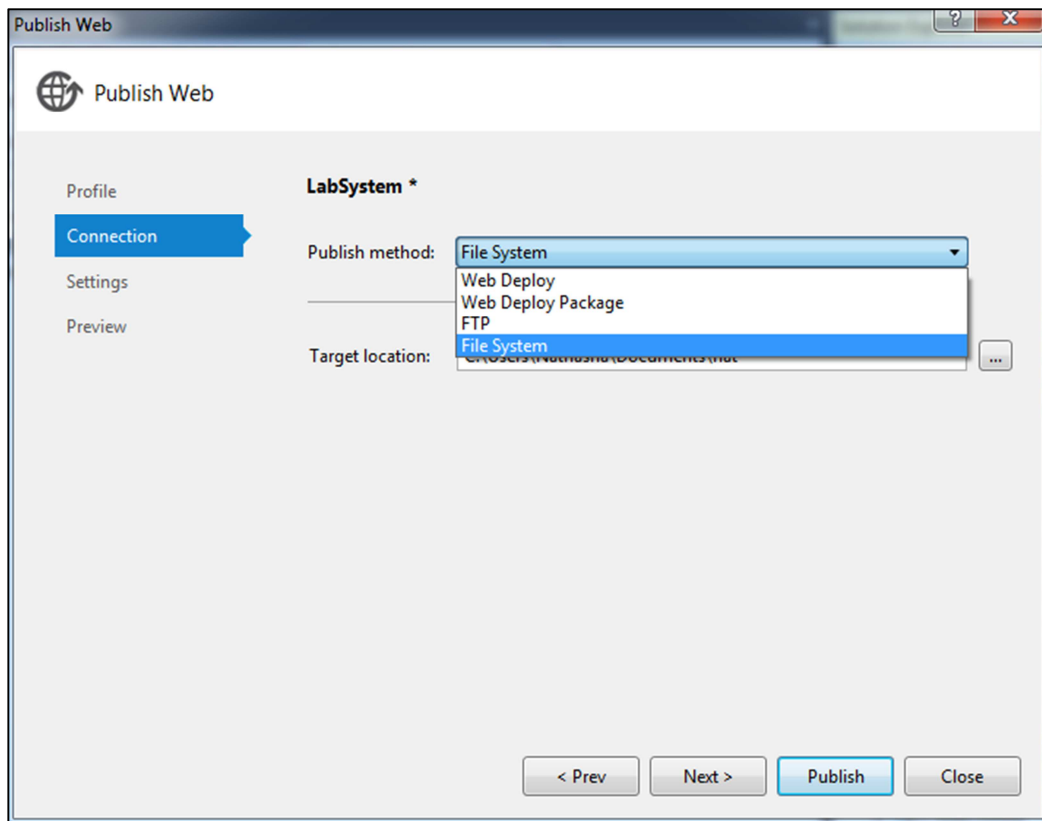


Figure 8-1 “Publish Web” tool

```

<<configuration>
  <appSettings>
    <!-- keys added by user-->
    <add key="DefaultAccessLevel" value="Read"/>
    <add key="DefaultCustomerType" value="Student"/>
    <add key="DefaultSearchType" value="Equipment Name"/>
    <add key="DefaultEquipmentCategory" value="All"/>
    <add key="DefaultEquipmentCategoryRegEquipment" value="Select"/>
    <add key="OtherVendorName" value="Other"/>
    <add key="HighestAccessLevel" value="Owner"/>
    <add key="EmailHost" value="smtp.live.com"/>
    <add key="EmailPort" value="587"/>
    <add key="FromEmailAddress" value="nathasha.nakandalage@hotmail.com"/>
    <add key="FromEmailAddressPassword" value="achini123"/>
    <!-- keys added by user-->
  </appSettings>
  <system.web>
    <compilation debug="true" targetFramework="4.0"/>
    <httpRuntime/>
    <pages controlRenderingCompatibilityVersion="4.0"/>
  </system.web>
  <connectionStrings>
    <!--add name="connectionString" connectionString="server=NATHASHA-PC\SOLEXPRESS;database=LAB_SYSTEM;uid=myUser;password=myPass;" /-->
    <add name="connectionString" connectionString="server=NATHASHA-PC\SOLEXPRESS;database=LABSYSTEM;Trusted_Connection=Yes;" />
  </connectionStrings>
</configuration>

```

Figure 8-2 “Web.config” file of “Lab Inventory System” website

8.1.1 IIS Configuration

“Lab Inventory System” website and web service need to be hosted on a target server to allow other users to access them and the concept of web servers comes in between. Web server is responsible for providing responses for all the requests that are coming from clients

(website or web service) as shown in Figure 8-3. Clients request for resources and web server processes the request and sends back to clients.

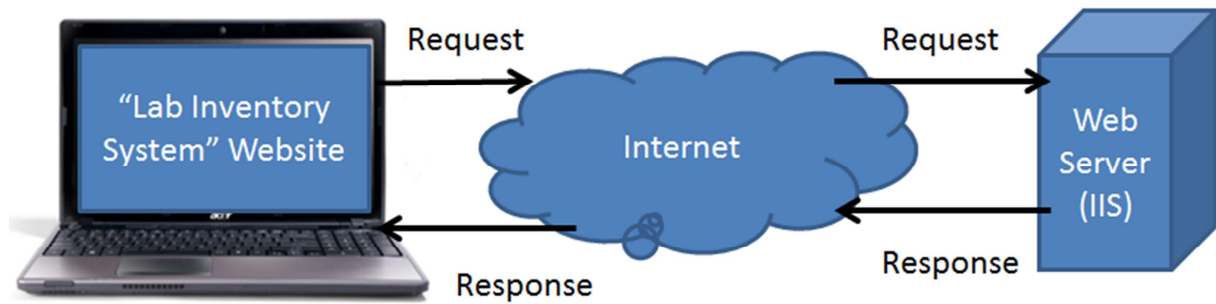


Figure 8-3 Deployment structure of “Lab Inventory System” website with IIS web server

IIS is an extensible web server created by Microsoft for use with Windows NT family [22]. IIS supports HTTP, HTTPS, FTP and SMTP. It provides a secure, easy-to-manage and extensible platform for reliably hosting websites, services and applications. IIS Manager is the graphical user interface of IIS and can be accessed through the Microsoft Management Console or Administrative Tools in the Control Panel. It is shown in Figure 8-4.

Steps to configure “Lab Inventory System” website or web service in IIS can be listed as follows:

- Open IIS Manager
- Right click on “Default Web Site”
- Click “Add Application”
- Insert values for “Alias” (for example set “LabSystem” as “Alias”) and “Physical Path” (Folder path that contains precompiled pages and code of website generated using “Publish Web” tool) fields
- Click “OK”
- Restart IIS
- Website can be accessed through “Machine IP Address/Alias value”

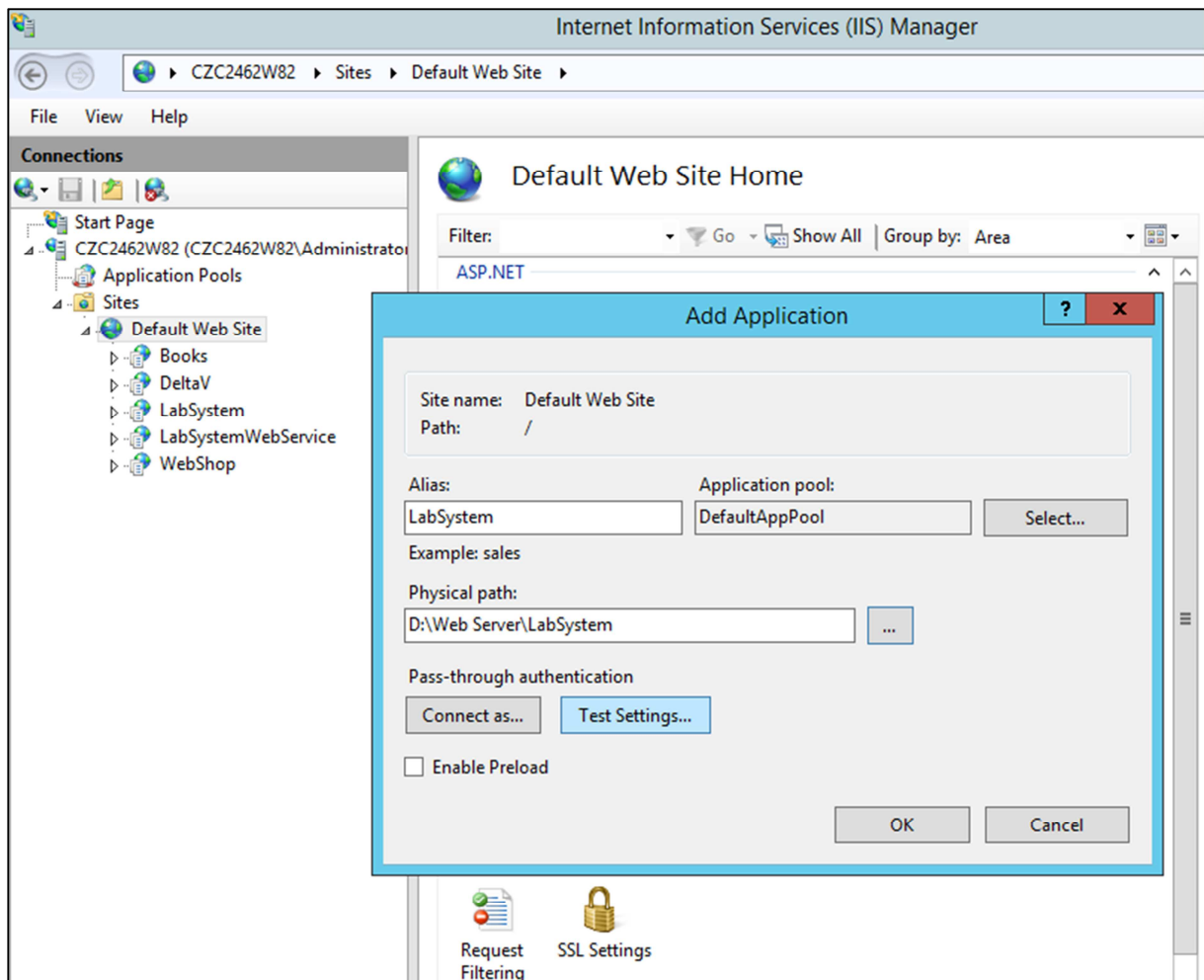


Figure 8-4 IIS Manager

8.1.2 Deploy Database Scripts

When deploying Lab Inventory System in test servers or production servers, it is necessary to deploy created database scripts in a target server. Database of Lab Inventory System has several scripts and stored procedures. “Database Script Generator” tool which is shown in Figure 8-5, is used to create a single script with combining all database scripts, views, stored procedures and triggers. Steps to deploy Lab Inventory System database scripts can be listed as follows:

- Open SQL Server Management Studio
- Create an empty database (Right click on “Databases”, Select “New Database”, give a name to the database and Click “OK”)
- Open and execute the script generated from “Database Script Generator” tool
- Check whether the database has its tables, data and stored procedures

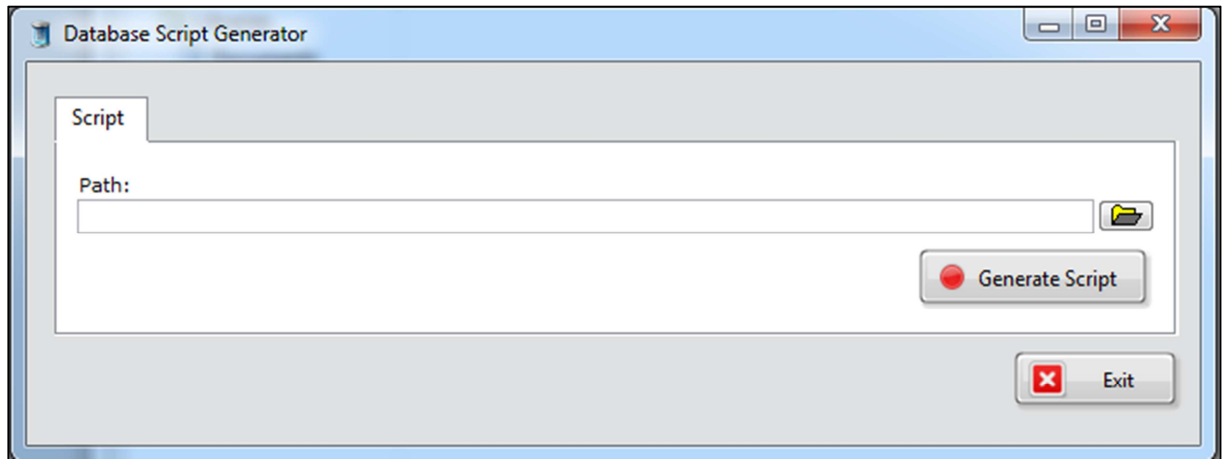


Figure 8-5 “Database Script Generator” tool

8.2 Deploy “Loan/Return Equipment” Application and “LabelWriter” Application

InstallShield is used to create installation packages for both “Loan/Return Equipment” application and “LabelWriter” application. It is a powerful and easy-to-use installation development solution for creating windows installations [23]. It makes it easy for customers to deploy and manage the applications, for developer to create installation packages. InstallShield provides a Project Assistant to help quickly and easily build a basic installation project as shown in Figure 8-6. The Project Assistant provides a framework of installation project tasks to guide through the project creation process and provides pertinent information along the way [24].

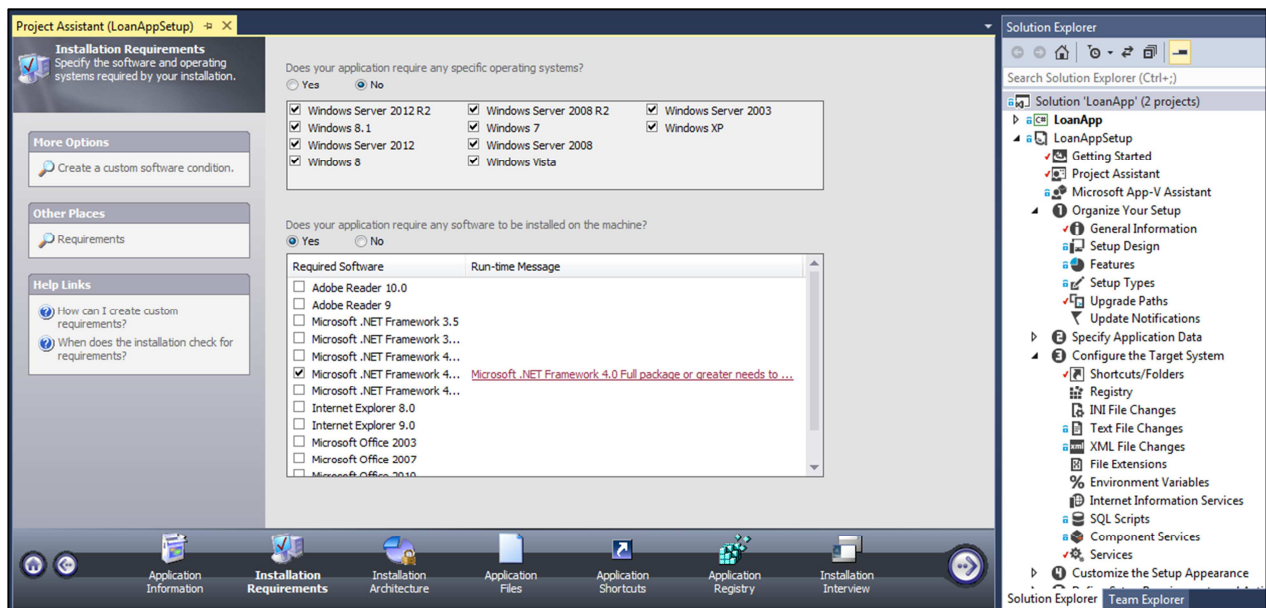


Figure 8-6 Project Assistant of InstallShield

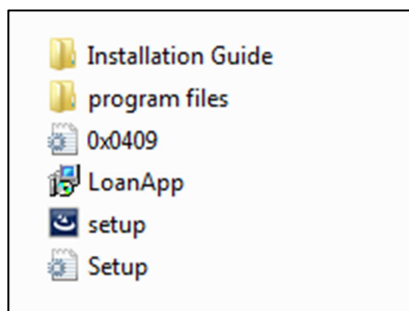
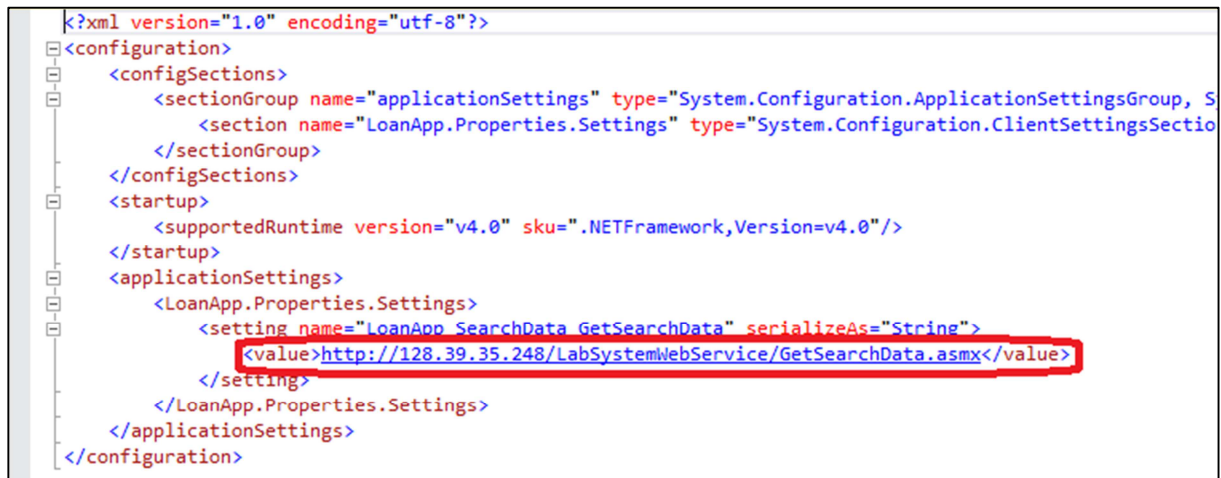


Figure 8-7 Created installation package for “Loan/Return Equipment” application using InstallShield

For both “Loan/Return Equipment” application and “LabelWriter” application, separate InstallShield projects have been created with their source code. So whatever changes have been done to the source code, are automatically added to the installation package by rebuilding the InstallShield project. General application information (application name, application version and icon), installation requirements (software and operating system requirements) and application files with installation guide and other required documents and templates can be added to the InstallShield project, so that it will create the installation package with required files as shown in Figure 8-7.

As shown in Figure 8-7, “setup.exe” can be used to install the software application. But before installing the software it is always important to read the installation guide, check and install pre-requests. It is also necessary to configure/update “App.config” file with latest or real configurations before actually using the application. For example check the web service

values are correct with production environment. Figure 8-8 shows the “App.config” file of “Loan/Return Equipment” application.



```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <configSections>
    <sectionGroup name="applicationSettings" type="System.Configuration.ApplicationSettingsGroup, S
      <section name="LoanApp.Properties.Settings" type="System.Configuration.ClientSettingsSectio
    </sectionGroup>
  </configSections>
  <startup>
    <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.0"/>
  </startup>
  <applicationSettings>
    <LoanApp.Properties.Settings>
      <setting name="LoanApp_SearchData_GetSearchData" serializeAs="String">
        <value>http://128.39.35.248/LabSystemWebService/GetSearchData.asmx</value>
      </setting>
    </LoanApp.Properties.Settings>
  </applicationSettings>
</configuration>
```

Figure 8-8 "App.config" file of “Loan/Return Equipment” application

DYMO Label Writer is used for printing labels. Before using “LabelWriter” application, DYMO Label Writer should be connected, drivers should be installed in the computer and make sure it is working properly.

9 Conclusion and Future Work

The report explains the design and implementation of Lab Inventory System in detail. Whole system was implemented using Visual Studio 2013, SQL Server was used for database design according to the task description. Therefore the theories and concepts behind programming, web services and database design were studied beforehand with Visual Studio, SQL Server and their tools.

In task one, a database was designed using SQL Server and ERWin software applications for handling and storing records of Lab Inventory System. Required entities, their attributes and relationships among entities were identified beforehand.

“LabelWriter” application was designed and developed as a desktop application as in task two, for printing out labels for registered equipment, so that administrator can install and configure it in their personal computers with DYMO Label Writer and its software. A web service was implemented for retrieving and storing information in database when using desktop applications.

Under task three “Loan/Return Equipment” application was designed and developed instead of implementing smartphone application as “Loan/Return Equipment” application is more important than smartphone application. This was implemented to configure in each laboratory for loaning and returning equipment easily.

In task four, “Lab Inventory System” website was designed and implemented for handling loans and reservations, managing user profiles and allowing search facilities for users to find out information of equipment in detail.

Use Case diagrams were designed with fully dressed Use Case documents for all three applications in analysis phase. In design phase, architecture was defined, a database diagram was designed using ERWin and a class diagram which describes the objects and information structure was documented. During implementation the appearance of web pages is defined using CSS while JavaScript is used for client side scripting. Whole system was tested by several users and bug tracking and source control were handled in TFS. Separate setup projects have been created for both “LabelWriter” application and “Loan/Return Equipment” application using InstallShield in deployment phase. “Lab Inventory System” website and web service were hosted in a test server and a production server for easy access.

9.1 Suggestions for future work

This Lab Inventory System needs some future work and correlations. Design and implementation of smartphone application is remained as a future work. With smartphone application, users can access to the application anytime using their mobile phones, even though they don't have access to desktop applications. By creating an application for phones, a better experience can be delivered to the user as screen size and touchscreen make traditional webpages much more difficult to use through mobile phones. Implement all the functionalities to use access card with RFID chips instead of barcode on the card is another suggestion for future work as Telemark University College is discussing the possibilities of using RFID chips in user identity cards in the future. Replace existing "LabelWriter" and "Loan/Return Equipment" applications with new Windows Store Apps is another suggestion. Extend Lab Inventory System with more functionalities such as implementing "Add to Basket" feature in search page of website can be considered as a future work, as this feature seems really helpful for users as they can add equipment anytime to their own basket and loan/reserve them later.

References

- [1] Microsoft. (2014). *Object-Oriented Programming (C# and Visual Basic)*. Available: <http://msdn.microsoft.com/en-us/library/dd460654.aspx#Classes>
- [2] (1995). *Encapsulation*. Available: <http://faculty.orangecoastcollege.edu/sgilbert/book/11-3-Encapsulation/index.html>
- [3] IBM. (2002). *An overview of the Web Services Inspection Language*. Available: <http://www.ibm.com/developerworks/library/ws-wslover/>
- [4] Microsoft. (2014). *Visual Studio*. Available: <http://www.visualstudio.com/>
- [5] Microsoft. (2014). *Introduction to ASP.NET and Web Forms*. Available: http://msdn.microsoft.com/en-us/library/ms973868.aspx#introwebforms_topic1
- [6] Microsoft. (2014). *Team Foundation Server*. Available: <http://www.visualstudio.com/en-us/products/tfs-overview-vs.aspx>
- [7] Microsoft. (2014). *Adopting Team Explorer Everywhere*. Available: <http://msdn.microsoft.com/en-us/library/gg413285.aspx>
- [8] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts*, 6 ed.: McGraw-Hill, 2010.
- [9] B. T. B. Technology. (2013). *ERWin Data Modeler*. Available: <http://www.bitalent.com/urunler/1010/erwin-data-modeler/1004/erwin-data-modeler.aspx>
- [10] C. Technologies. (2012). *how can I manage data complexity and improve business agility?* Available: <http://erwin.com/content/products/CA-ERwin-r9-Modeling-Solution-Brief-na.pdf>
- [11] Oracle. (1995). *Using Stored Procedures*. Available: <http://docs.oracle.com/javase/tutorial/jdbc/basics/storedprocedures.html>
- [12] Microsoft. (2014). *Stored Procedures*. Available: <http://technet.microsoft.com/en-us/library/aa214299%28v=sql.80%29.aspx>
- [13] Microsoft. (2014). *Use SQL Server Management Studio*. Available: <http://msdn.microsoft.com/en-us/library/ms174173.aspx>
- [14] Microsoft. (2014). *UML Use Case Diagrams: Guidelines*. Available: <http://msdn.microsoft.com/en-us/library/dd409432.aspx>
- [15] H. P. Halvorsen. (2014). *The Software Development Process*. Available: http://home.hit.no/~hansha/documents/software/software_development/topics/resources/SDLC%20Overview.pdf
- [16] Microsoft. (2014). *Using a Three-Tier Architecture Model*. Available: <http://msdn.microsoft.com/en-us/library/windows/desktop/ms685068%28v=vs.85%29.aspx>
- [17] (2014). *Create and Implement 3-Tier Architecture in ASP.Net*. Available: <http://www.c-sharpcorner.com/UploadFile/4d9083/create-and-implement-3-tier-architecture-in-Asp-Net/>
- [18] Microsoft. (2014). *User Input Validation in ASP.NET*. Available: <http://msdn.microsoft.com/en-us/library/ms972961.aspx>

- [19] Microsoft. (2014). *Tracking Bugs, Tasks, and Other Work Items (Team Explorer Everywhere)*. Available: <http://msdn.microsoft.com/en-us/library/gg490747%28v=vs.100%29.aspx>
- [20] Microsoft. (2014). *Work in Team Explorer*. Available: <http://msdn.microsoft.com/en-us/library/hh500420.aspx>
- [21] H. P. Halvorson. *Software Deployment & Maintenance*. Available: http://home.hit.no/~hansha/documents/software/software_development/topics/resources/Software%20Deployment%20Overview.pdf
- [22] Microsoft. (2014). *Running IIS 6.0 as an Application Server (IIS 6.0)*. Available: <http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/ddfd92f-3e6e-423f-b024-35cefc10a22f.mspx?mfr=true>
- [23] F. S. LLC. (2014). *InstallShield*. Available: <http://www.installshield.com/>
- [24] F. S. LLC. (2009). *Project Assistant*. Available: http://helpnet.installshield.com/installshield16helplib/PA_Overview.htm

Appendix 1: Task Description

Task description

The project consists in producing a detailed document with the following content.

- Design a Database for such a system. Implement the Database using SQL Server. The database should include tables for storing information about the different lab equipment, handling reservations and loans, etc.
- Implement a Label Writer for printing out barcode labels that can be attached to the different equipment.
- Create a Web Application (ASP.NET) for Management of the Inventory.
- Design and Implement a Smartphone App for easy Management of the Lab Equipment. The App should be able to read Barcodes for easy check-in and check-out of the Laboratory Equipment.
- Delivery of written report following guidelines from the Faculty.

Task background

TUC/TF has lots of expensive laboratory equipment that are used in different laboratory work and project. The equipment is used by the staff and the students.

Appendix 2: Fully Dressed Use Case Documents for “Lab Inventory System” Website

“Search for equipment” Use Case

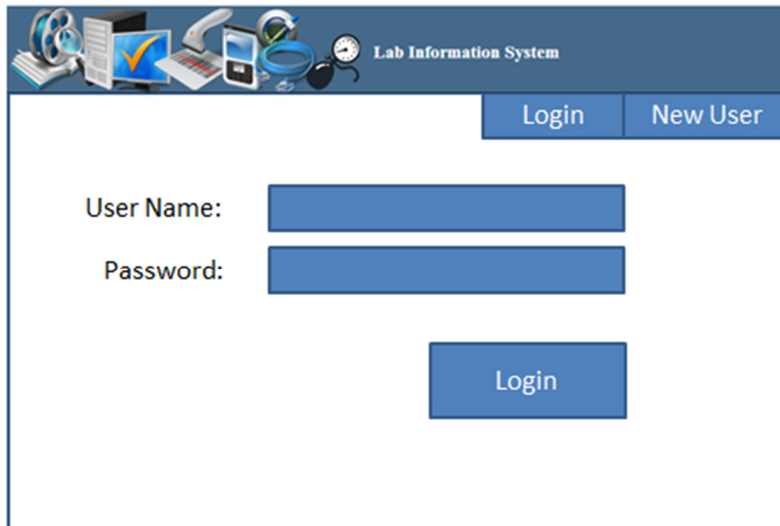
Use Case name	Search for equipment
Scope	“Lab Inventory System” website
Actor	End User, Administrator
Preconditions	The user should have completed the Use Case “Login” and clicked “Search Equipment” button
Success Guarantee	Display appropriate search results successfully
Main success scenario	<ol style="list-style-type: none"> 1. Select search type/equipment category 2. Enter search text 3. Click “Search” button 4. Display appropriate search results successfully
Extensions	3a. If there are no search records, the system will display an appropriate message and allow the user to perform a new search.
Miscellaneous	None

“View equipment details” Use Case

Use Case name	View equipment details
Scope	“Lab Inventory System” website
Actor	End User, Administrator
Preconditions	The user should have completed the Use Case “Search for equipment” and selected a search record and clicked “Show Equipment Details” button
Success Guarantee	Display equipment details successfully
Main success scenario	1. Display equipment details
Extensions	None
Miscellaneous	None

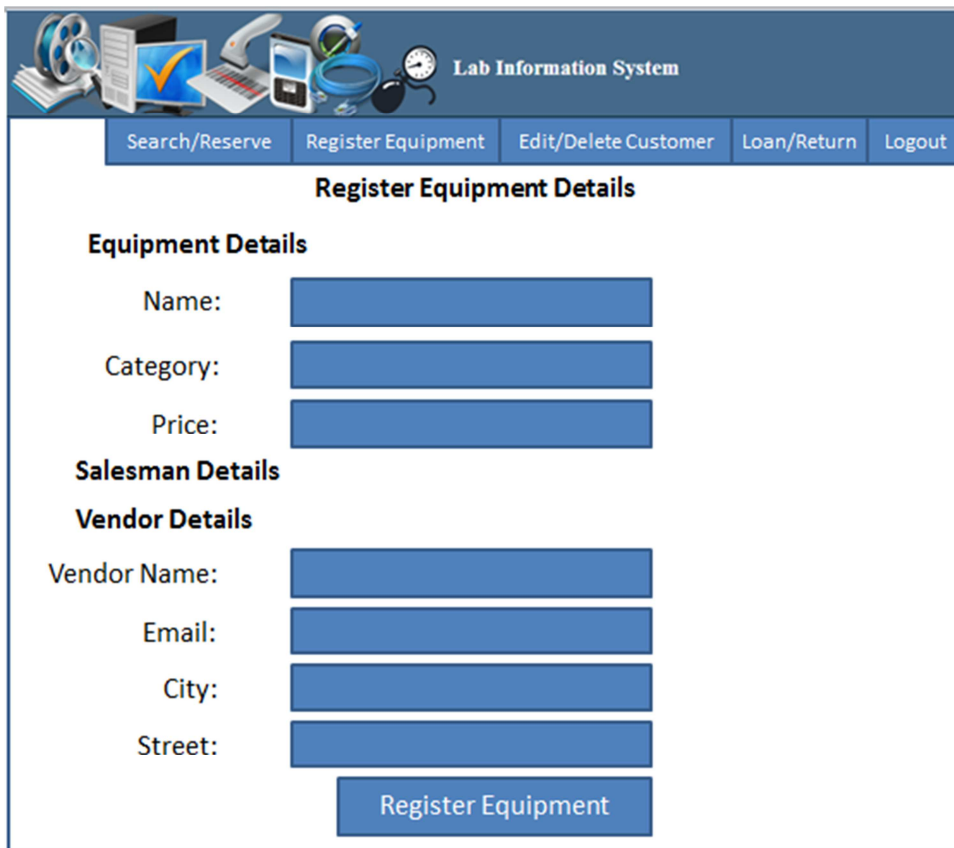
Appendix 3: User Interface Sketches for “Lab Inventory System” Website

UI Sketch for “Login” page



The sketch shows a login page for the "Lab Information System". At the top, there is a header with the system name and a navigation bar with "Login" and "New User" buttons. The main content area contains a "User Name:" label followed by a text input field, a "Password:" label followed by a text input field, and a "Login" button centered below the fields.

UI Sketch for “Register Equipment” page



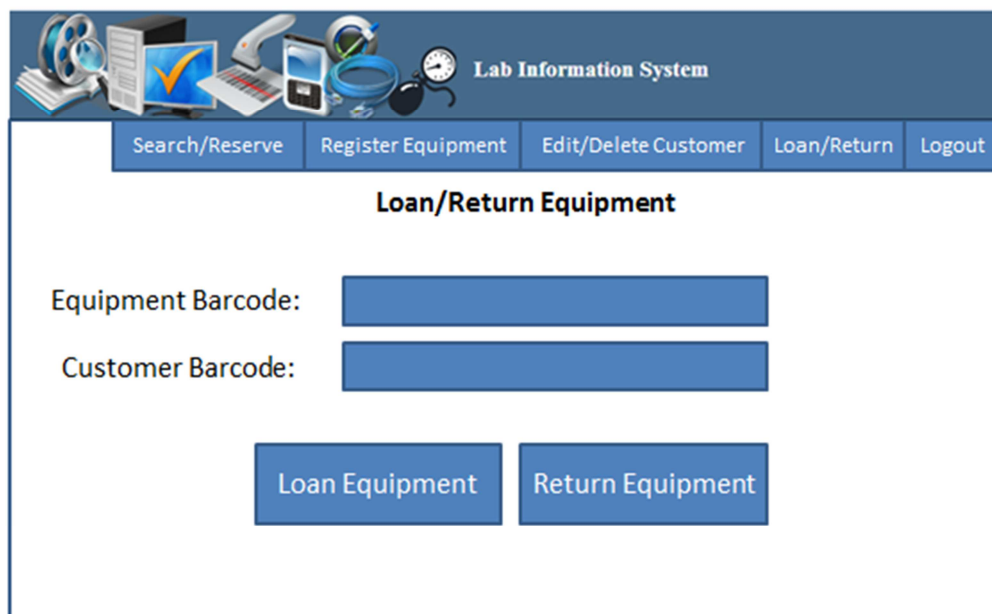
The sketch shows a "Register Equipment" page for the "Lab Information System". The header includes the system name and a navigation bar with buttons for "Search/Reserve", "Register Equipment", "Edit/Delete Customer", "Loan/Return", and "Logout". The main content area is titled "Register Equipment Details" and is divided into three sections: "Equipment Details" with fields for "Name:", "Category:", and "Price:"; "Salesman Details"; and "Vendor Details" with fields for "Vendor Name:", "Email:", "City:", and "Street:". A "Register Equipment" button is located at the bottom of the form.

UI Sketch for “Edit/Delete Customer” page



The UI sketch for the "Edit/Delete Customer" page features a dark blue header with the text "Lab Information System" and a navigation bar with buttons for "Search/Reserve", "Register Equipment", "Edit/Delete Customer", "Loan/Return", and "Logout". The main content area is titled "Edit/Delete Customer" and contains a "Customer Barcode" input field with a "Find" button. Below this are sections for "Login Details" (User Name, Password, Confirm Password) and "User Details" (First Name, Last Name, City, Street), each with corresponding input fields. At the bottom are "Save" and "Delete" buttons.

UI Sketch for “Loan/Return Equipment” page



The UI sketch for the "Loan/Return Equipment" page features a dark blue header with the text "Lab Information System" and a navigation bar with buttons for "Search/Reserve", "Register Equipment", "Edit/Delete Customer", "Loan/Return", and "Logout". The main content area is titled "Loan/Return Equipment" and contains two input fields for "Equipment Barcode" and "Customer Barcode". At the bottom are "Loan Equipment" and "Return Equipment" buttons.

Appendix 4: Fully Dressed Use Case Documents for “LabelWriter” Application

“Print label” Use Case

Use case name	Print label
Scope	“LabelWriter” application
Actor	Administrator
Preconditions	The user should have completed the Use Case “Search for equipment or user” and selected a search record and clicked “Print” button
Success Guarantee	Print label successfully
Main success scenario	<ol style="list-style-type: none"> 1. Select/Upload a label template 2. If needed set/modify “Number of Copies” and “Label Data” fields 3. Click “Print” button 4. Print label successfully
Extensions	<p>3a. “Number of Copies” field is empty</p> <ol style="list-style-type: none"> 1. Give an error message <p>3b. Incorrect label template/ printer is not working properly / printer is not connected properly/ printer drivers are not installed</p> <ol style="list-style-type: none"> 1. Give an error message
Miscellaneous	None

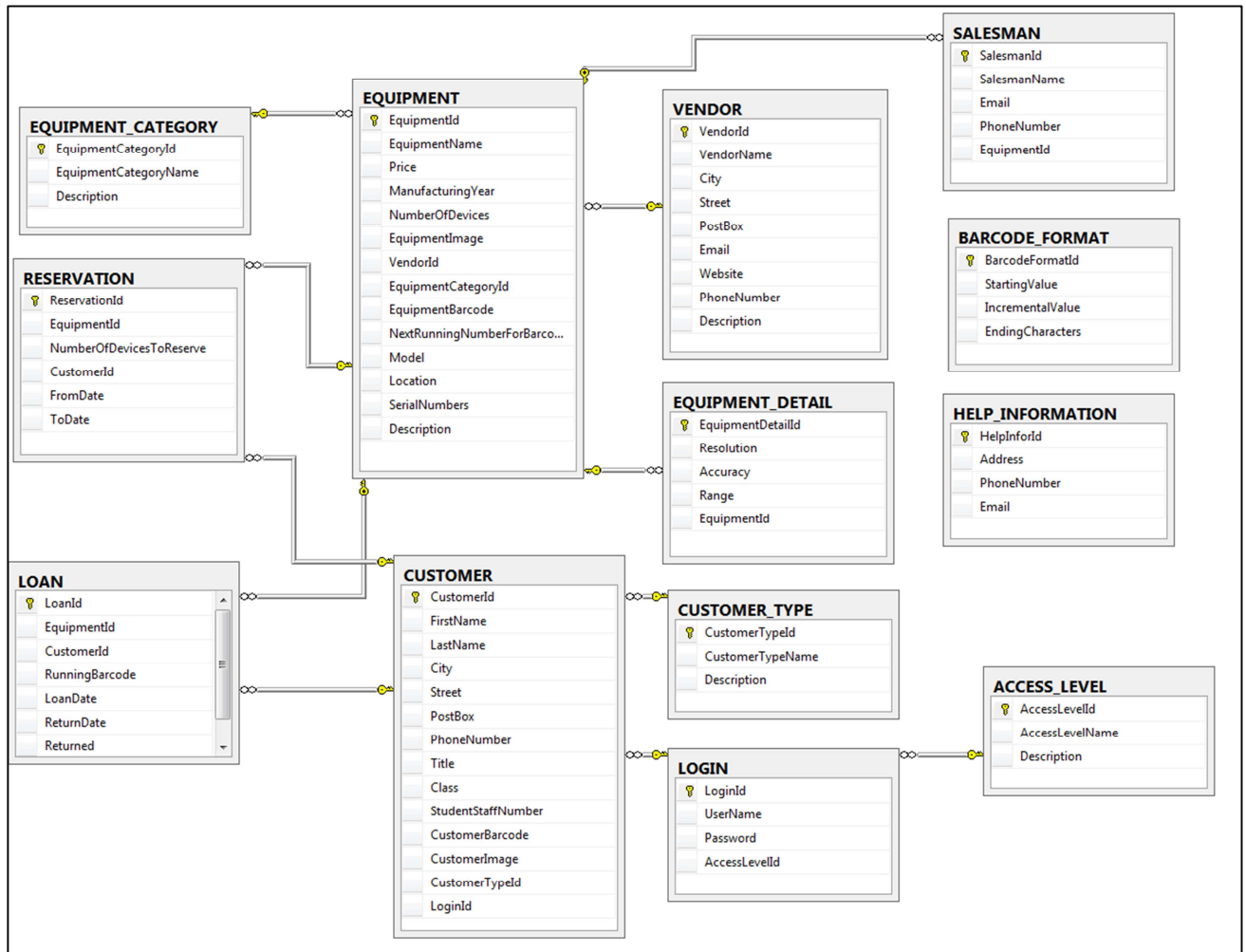
Appendix 5: Fully Dressed Use Case Documents for “Loan/Return Equipment” Application

“Return equipment” Use Case

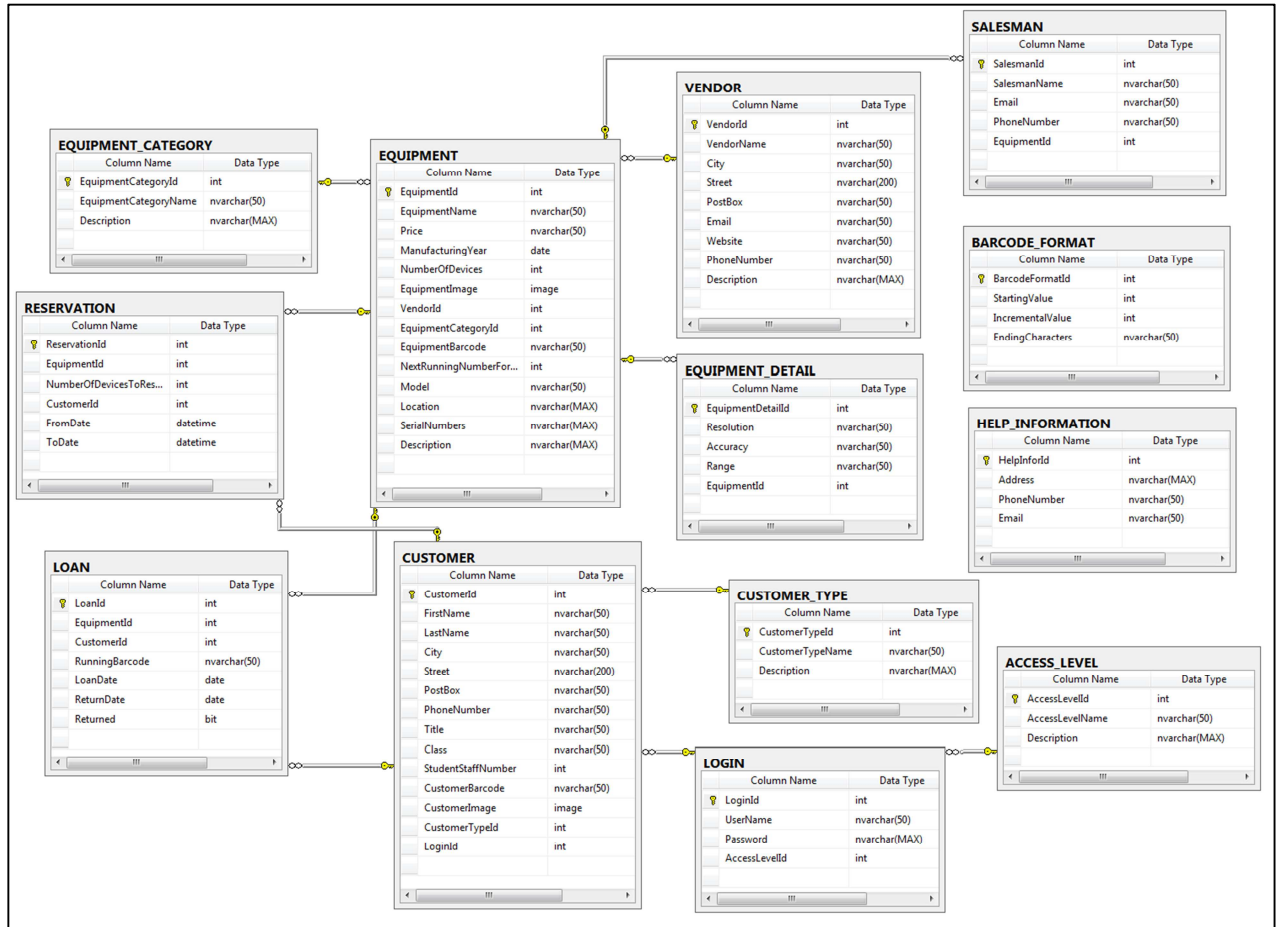
Use Case name	Return equipment
Scope	“Loan/Return Equipment” application
Actor	End User, Administrator
Preconditions	None
Success Guarantee	Update appropriate records in database successfully
Main success scenario	<ol style="list-style-type: none"> 1. Scan equipment barcode 2. Scan user barcode 3. Display name of the user 4. Click “Return Equipment” button 5. Update appropriate records in database successfully 6. Clear equipment barcode and user barcode textboxes and ready for next operation
Extensions	<p>3a. Invalid user</p> <ol style="list-style-type: none"> 1. Give an error message and clear user barcode textbox <p>4a. Equipment barcode and/or user barcode textboxes are empty</p> <ol style="list-style-type: none"> 1. Give an error message
Miscellaneous	None

Appendix 6: SQL Server Database Diagrams

SQL Server database diagram

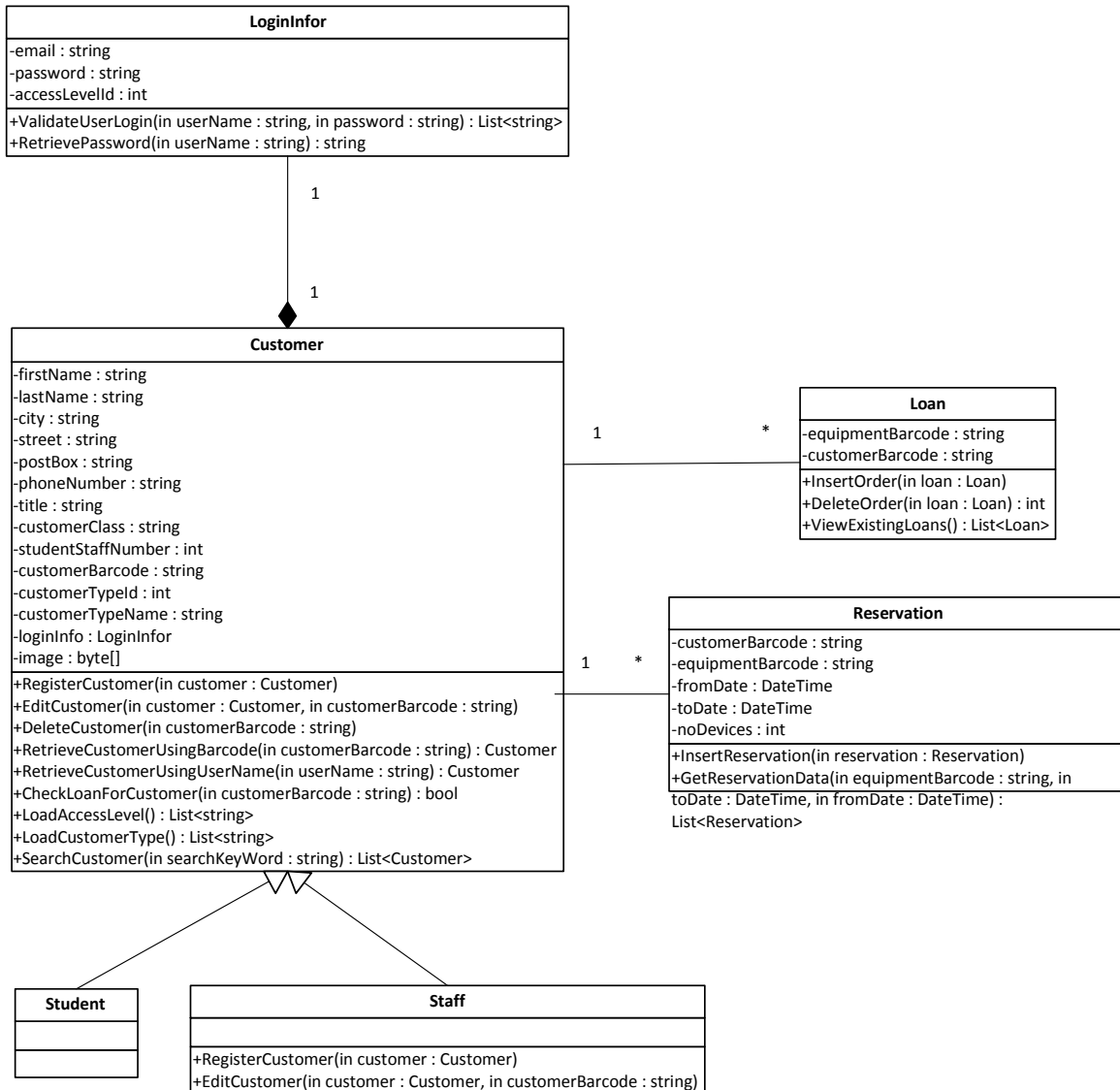


SQL Server database diagram with data types

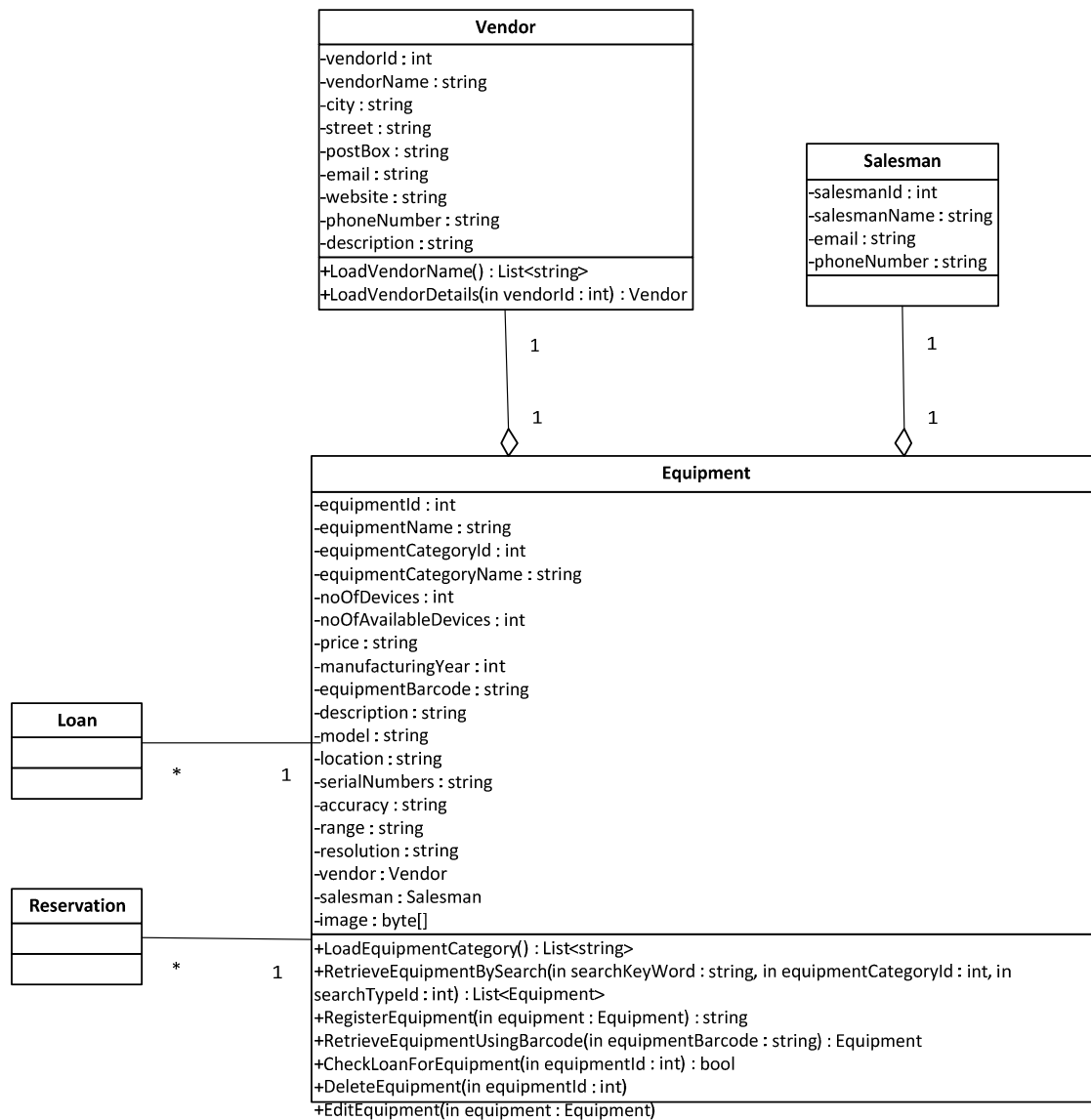


Appendix 7: Class Diagram

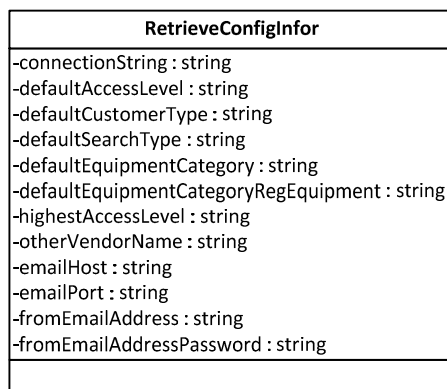
Class diagram – Part 1



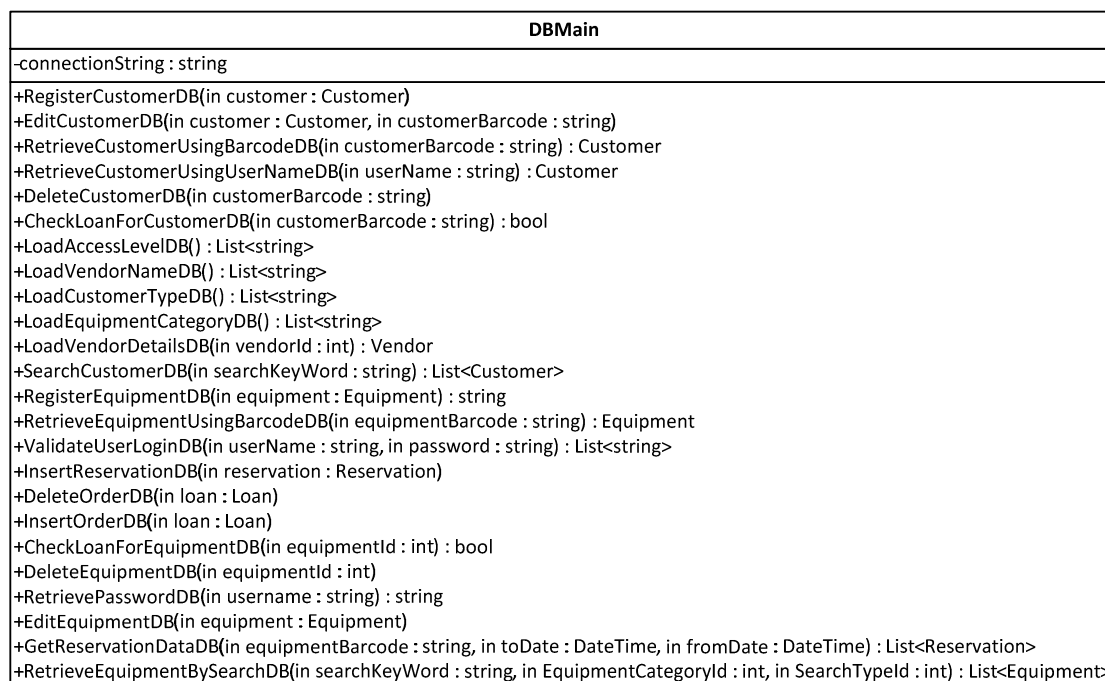
Class diagram – Part 2



“RetrieveConfigInfor” Class



“DBMain” Class



Class diagram describes objects and information structure of a software application. It displays identified classes, their attributes, methods and relationships.

“Customer” class handles customer operations such as “Register Customer”, “Edit Customer”, “Delete Customer”, “Search Customer” etc. Both “Staff” and “Student” classes inherit attributes and methods of “Customer” class. The classes “Loan” and “Reservation” are associated with “Customer” class. The multiplicity of the association denotes the number of objects that can participate in the relationship. For example, a Loan object can be associated

to only one Customer, but a Customer can be associated to many Loans. In the same way, a Reservation object can be associated to only one Customer, but a Customer can be associated to many Reservations. “LoginInfor” class handles login functionalities. The relationship between “LoginInfor” and “Customer” class specifies a composition or strong type whole-part relationship.

“Equipment” class handles operations related to equipment such as “Register Equipment”, “Edit Equipment”, “Delete Equipment” etc. According to the class diagram (part 2) , a Loan object can be associated to only one Equipment, but an Equipment can be associated to many Loans. In the same way, a Reservation object can be associated to only one Equipment, but Equipment can be associated to many Reservations. The relationship between “Vendor/Salesman” and “Equipment” class specifies an aggregation relationship.

“RetrieveConfigInfor” Class is a static class. It reads all keys in “appSettings” section in Web.config or App.config file. “DBMain” class handles all database operations.

Appendix 8: Screen dumps of “Lab Inventory System” Website

“View Existing Loans” and “Help” pages



View Existing Loans

You are ready to view existing loans of Lab Information System. You can search loans for particular equipment or user or both by entering values for 'Equipment Name' and 'User Name' textboxes.

Equipment Name: User Name:

Displaying top 100 records from the database....
4 search results found.
 Equipment Name: "", User Name: ""

Equipment barcode	Equipment name	Loan Date	Return Date	Customer Name	Phone Number
997TUC-0	Model 424 Wing Union Pressure Transmitter	17.01.2014	31.01.2014	Nathasha Nakandalage	96867508
998TUC-0	NI USB-TC01 Thermocouple	17.01.2014	31.01.2014	Deshaka Kottage	96867509
999TUC-0	NI USB-6221 DAQ	17.01.2014	31.01.2014	Hans-Petter Halvorsen	96867507
997TUC-1	Model 424 Wing Union Pressure Transmitter	31.05.2014	14.06.2014	Hans-Petter Halvorsen	96867507




Help

Browse Lab Information System Help

Loan/Return Equipment

1. Scan equipment barcode
2. Scan user barcode
3. Click "Loan Equipment" or "Return Equipment"

A desktop application is also available for borrowing and returning equipments.





Please [click here](#) to download 'Loan/Return Equipment' desktop application.

Search Equipment

1. Click "Search Equipment" button in Top Menu Bar
2. Select "Equipment Type" and/or "Search Type" from dropdown menus
3. Enter search keyword
4. Click "Search"

“Register Equipment Details” and “Edit/Delete User” web pages


Logged in as: hans.p.halvorsen@hit.no 

Lab Information System

Loan/Return Equipment
Search Equipment
Register Equipment
Edit/Delete User
Log Out
Help

Register Equipment Details

You are ready to register equipment details.

Tips: After registering equipment details in database, you should print the generated equipment barcode label to attach to the equipment by using 'LabelWriter' application and 'DYMO LabelWriter' equipment. 'LabelWriter' application is a desktop application that you need to install and configure in your personal computer.

Please [click here](#) to download 'LabelWriter' application.

Equipment Details

Equipment Name: *

Equipment Category: *



Number of Devices: *

Price:

Manufacturing Year:

Model:

Location:


Logged in as: hans.p.halvorsen@hit.no 

Lab Information System

Loan/Return Equipment
Search Equipment
Register Equipment
Edit/Delete User
Log Out
Help

Edit/ Delete User Details

You are ready to edit/ delete user details which are saved in Lab Equipment Database. Please enter name to retrieve user details.

Name: *

Displaying top 100 records from the database....
7 search results found.
 Search Keyword: "a"

First Name	Last Name	Title	City	Phone Number	Type
Nathasha	Nakandalage	Student	Porsgrunn	96867508	Student
Deshaka	Kottage	Student	Porsgrunn	96867509	Student
Hans-Petter	Halvorsen	Teacher	Porsgrunn	96867507	Staff
Hakon	Lillehammer				Student
Kristin	Lillehammer				Student
Kari	Nordmann				Staff
Ola	Nordmann				Staff

Login Details

Email: * [Send Email](#)

Password: *

Appendix 9: Web Methods of “Lab Inventory System”

Web Service

Name	Input Parameters	Return Data Type
GetCustomerData	string searchKeyword	List<Customer>
GetEquipmentData	string searchKeyword	List<Equipment>
LoanEquipment	string equipmentBarcode, string customerBarcode	bool
DeleteEquipmentOrder	string equipmentBarcode, string customerBarcode	bool
RetrieveNextRunningBarcode	string equipmentBarcode	int
UpdateNextRunningBarcode	string equipmentBarcode, int nextRunningBarcode	void
RetrieveUserName	string userBarcode	string