

ABSTRACT

A Web-Based Information Solution for an Educational Institution

i-Solution is:

- A collaborative web-based platform for information exchange between the Administrators, Students and Teachers of an institution which allows efficient and purposeful integration between the users of the system.

Goals of the System:

- To allow the Administrators of an Institution to manage information regarding the students, teachers, subjects and share the information with other users of the system.
- To allow the Class Teachers to manage student profiles, subject teachers and attendance.
- To allow the Class Teachers to gather attendance information from the subject teacher with ease.
- To generate extensive attendance reports and provide data in various formats.
- To allow library staff to manage Accession Registers and Circulation of books and also share the information with end users like teachers, students through a web based platform.
- To allow teachers and students to share study materials by letting them upload various study material to the system.
- To allow information exchange between Administrators, Teachers and Students regarding everything using notices.

Goals of the Project:

- To implement a real-world project on a widely used web development platform – PHP.
- To learn and achieve a higher degree of abstraction between HTML and PHP code.
- To learn the art of developing web applications with a User Interface that is pleasing to the eye and easy to use with the use of JS and CSS.
- Design an interactive UI using the most talked about and emerging technology right now, AJAX.

1. Introduction

1.1 Problem Statement

Problem Statement:

To develop a web based “Information Solution” for an institution.

Problem Definition in Detail:

Most educational institutes at present have full-fledged software developed aimed at administrative purposes, like student registration, library management services, accounts and others. This way, they are not utilizing the power and the facilities of the internet/Web. The primary requirement of an institute to go forward in this world of internet is to create a framework for students to learn and exchange views through communication in different modes. I-Solution will be a technique to use a web based interactive information system for an institution focusing mainly on the need of the systems who are end users, providing them different functionalities from within and outside the institute.

Need:

i-Solution is needed because such a solution is not yet available for an education system like ours. The existing solutions (based on the western educational system) cannot be implemented for our system. I-Solution is aimed at designing an information solution which will be easy to implement for any institution with little or no modification.

Purpose:

To provide a complete solution for the student, teachers and administrators of an Institution which will be easy to use.

2. Technical Research

2.1 Technology

Service	Technology Used
Language	PHP
Database	mySQL
HTTP Server	Apache
AJAX Framework	xajax Framework
Testing	simpleTest

Table 2.1 Technologies Used

2.1.1 PHP

PHP (recursive acronym for "PHP: Hypertext Preprocessor") is a widely-used Open Source general-purpose scripting language that is especially suited for Web development and can be embedded into HTML.

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server. If we were to have a script similar to the above on our server, the client would receive the results of running that script, with no way of determining what the underlying code may be. We can even configure our web server to process all our HTML files with PHP.

The best thing about using PHP is that it is extremely simple for a newcomer, yet offers many advanced features for a professional programmer

PHP was written as a set of CGI binaries in the C programming language by the Danish-Canadian programmer Rasmus Lerdorf in 1994, to replace a small set of Perl scripts he had been using to maintain his personal homepage. Lerdorf initially created PHP to display his résumé and to collect certain data, such as how much traffic his page was receiving. "Personal Home Page

Tools" was publicly released on June 8, 1995 after Lerdorf combined it with his own Form Interpreter to create PHP/FI.

Usage

PHP generally runs on a web server, taking PHP code as its input and creating Web pages as output, but command-line scripting and client-side GUI applications are part of the three primary uses of PHP as well.

Server-side scripting

Originally designed to create dynamic web pages, server-side scripting is the principal focus for PHP. While running the PHP parser with a web server and web browser, the PHP model can be compared to other server-side scripting languages such as Microsoft's ASP.NET system, Adobe ColdFusion, Sun Microsystems' JavaServer Pages, Zope, mod_perl and the Ruby on Rails framework, as they all provide dynamic content to the client from a web server. To more directly compete with the "framework" approach taken by these systems, Zend is working on the Zend Framework - an emerging (as of June 2006) set of PHP building blocks and best practices; other PHP frameworks along the same lines include CakePHP and Symfony.

The LAMP architecture has become popular in the Web industry as a way of deploying inexpensive, reliable, scalable, secure web applications. PHP can be used with a large number of relational database management systems, runs on all of the most popular web servers and is available for many different operating systems. This flexibility means that PHP has a wide installation base across the Internet; over 18 million Internet domains are currently hosted on servers with PHP installed.

Examples of popular server-side PHP applications include phpBB, PHP-Nuke, Joomla, Wordpress and MediaWiki.

Command-line scripting

PHP also provides a command line interface SAPI for developing shell and desktop applications, log parsing, or other system administration tasks. It is increasingly used on the command line for tasks which have traditionally been the domain of Perl, awk, or shell scripting.

Client-side GUI applications

PHP provides bindings to GUI libraries such as GTK+ and text mode libraries like ncurses in order to facilitate development of a broader range of cross-platform GUI applications.

Data types

PHP stores whole numbers in a platform-dependent range. This range is typically that of 32-bit signed integers. Integer variables can be assigned using decimal (positive and negative), octal and hexadecimal notations. Real numbers are also stored in a platform-specific range. They can be specified using floating point notation, or two forms of scientific notation.

PHP has a native Boolean type, named "boolean", similar to the native Boolean types in Java and C++. Using the Boolean type conversion rules, non-zero values can be interpreted as true and zero as false, as in Perl.

The null data type represents a variable that has no value. The only value in the null data type is NULL.

Arrays are heterogeneous, meaning a single array can contain objects of more than one type. They can contain any type that PHP can handle, including resources, objects, and even other arrays. Order is preserved in lists of values and in hashes with both keys and values, and the two can be intermingled.

Variables of the "resource" type represent references to resources from external sources. These are typically created by functions from a particular extension, and can only be processed by functions from the same extension. Examples include file, image and database resources.

Objects

Basic object-oriented functionality was added in PHP 3. However, handling of objects was completely rewritten for PHP 5, allowing for better performance and more features. In previous versions of PHP, objects were handled like primitive types. The drawback of this method was that semantically the whole object was copied when a variable was assigned, or passed as a parameter to a method. In the new approach, objects are referenced by handle, and not by value. PHP 5 introduced private and protected member variables and methods, along with abstract classes and abstract methods. It also introduced a standard way of declaring constructors and destructors similar to that of other object-oriented languages, such as C++, and an exception handling model similar to that of other programming languages.

Libraries

PHP includes a large number of free and open source libraries with the core build. PHP is a fundamentally Internet-aware system with modules built in for accessing FTP servers, many database servers, embedded SQL libraries like embedded MySQL and SQLite, LDAP servers, and others. Many functions familiar to C programmers such as the printf family are available in the standard PHP build.

PHP extensions have been written to add support for the Windows API, process management on Unix-like operating systems, multibyte strings (Unicode), cURL, and several popular compression formats. Some more unusual features include integration with Internet relay chat, and dynamic generation of images and Adobe Flash content. Some additional extensions are available via the PHP Extension Community Library.

2.1.2 MySQL

Top Ten Reasons to Use MySQL

1. Scalability and Flexibility

The MySQL database server provides the ultimate in scalability, sporting the capacity to handle deeply embedded applications with a footprint of only 1MB to running massive data warehouses holding terabytes of information. Platform flexibility is a stalwart feature of MySQL with all flavors of Linux, UNIX, and Windows being supported. And, of course, the open source nature of MySQL allows complete customization for those wanting to add unique requirements to the database server.

2. High Performance

A unique storage-engine architecture allows database professionals to configure the MySQL database server specifically for particular applications, with the end result being amazing performance results. Whether the intended application is a high-speed transactional processing system or a high-volume web site that services a billion queries a day, MySQL can meet the most demanding performance expectations of any system. With high-speed load utilities, distinctive memory caches, full text indexes, and other performance-enhancing mechanisms, MySQL offers all the right ammunition for today's critical business systems.

3. High Availability

Rock-solid reliability and constant availability are hallmarks of MySQL, with customers relying on MySQL to guarantee around-the-clock uptime. MySQL offers a variety of high-availability options from high-speed master/slave replication configurations, to specialized Cluster servers offering instant failover, to third party vendors offering unique high-availability solutions for the MySQL database server.

4. Robust Transactional Support

MySQL offers one of the most powerful transactional database engines on the market. Features

include complete ACID (atomic, consistent, isolated, durable) transaction support, unlimited row-level locking, distributed transaction capability, and multi-version transaction support where readers never block writers and vice-versa. Full data integrity is also assured through server-enforced referential integrity, specialized transaction isolation levels, and instant deadlock detection.

5. Web and Data Warehouse Strengths

MySQL is the de-facto standard for high-traffic web sites because of its high-performance query engine, tremendously fast data inserts capability, and strong support for specialized web functions like fast full text searches. These same strengths also apply to data warehousing environments where MySQL scales up into the terabyte range for either single servers or scale-out architectures. Other features like main memory tables, B-tree and hash indexes, and compressed archive tables that reduce storage requirements by up to eighty-percent make MySQL a strong standout for both web and business intelligence applications.

6. Strong Data Protection

Because guarding the data assets of corporations is the number one job of database professionals, MySQL offers exceptional security features that ensure absolute data protection. In terms of database authentication, MySQL provides powerful mechanisms for ensuring only authorized users have entry to the database server, with the ability to block users down to the client machine level being possible. SSH and SSL support are also provided to ensure safe and secure connections. A granular object privilege framework is present so that users only see the data they should, and powerful data encryption and decryption functions ensure that sensitive data is protected from unauthorized viewing. Finally, backup and recovery utilities provided through MySQL and third party software vendors allow for complete logical and physical backup as well as full and point-in-time recovery.

7. Comprehensive Application Development

One of the reasons MySQL is the world's most popular open source database is that it provides comprehensive support for every application development need. Within the database, support

can be found for stored procedures, triggers, functions, views, cursors, ANSI-standard SQL, and more. For embedded applications, plug-in libraries are available to embed MySQL database support into nearly any application. MySQL also provides connectors and drivers (ODBC, JDBC, etc.) that allow all forms of applications to make use of MySQL as a preferred data management server. It doesn't matter if it's PHP, Perl, Java, Visual Basic, or .NET, MySQL offers application developers everything they need to be successful in building database-driven information systems.

8. Management Ease

MySQL offers exceptional quick-start capability with the average time from software download to installation completion being less than fifteen minutes. This rule holds true whether the platform is Microsoft Windows, Linux, Macintosh, or UNIX. Once installed, self-management features like automatic space expansion, auto-restart, and dynamic configuration changes take much of the burden off already overworked database administrators. MySQL also provides a complete suite of graphical management and migration tools that allow a DBA to manage, troubleshoot, and control the operation of many MySQL servers from a single workstation. Many third party software vendor tools are also available for MySQL that handle tasks ranging from data design and ETL, to complete database administration, job management, and performance monitoring.

9. Open Source Freedom and 24 x 7 Support

Many corporations are hesitant to fully commit to open source software because they believe they can't get the type of support or professional service safety nets they currently rely on with proprietary software to ensure the overall success of their key applications. The questions of indemnification come up often as well. These worries can be put to rest with MySQL as complete around-the-clock support as well as indemnification is available through MySQL Network. MySQL is not a typical open source project as all the software is owned and supported by MySQL AB, and because of this, a unique cost and support model are available that provides a unique combination of open source freedom and trusted software with support.

10. Lowest Total Cost of Ownership

By migrating current database-drive applications to MySQL, or using MySQL for new development projects, corporations are realizing cost savings that many times stretch into seven figures. Accomplished through the use of the MySQL database server and scale-out architectures that utilize low-cost commodity hardware, corporations are finding that they can achieve amazing levels of scalability and performance, all at a cost that is far less than those offered by proprietary and scale-up software vendors. In addition, the reliability and easy maintainability of MySQL means that database administrators don't waste time troubleshooting performance or downtime issues, but instead can concentrate on making a positive impact on higher level tasks that involve the business side of data.

2.1.3 HTTP Server - Apache

The Apache HTTP Server is a web server for Unix-like systems, Microsoft Windows, Novell NetWare, Mac OS X and other operating systems. Apache is notable for playing a key role in the initial growth of the World Wide Web.

When first released, Apache was the only viable free/open source alternative to the Netscape Communications Corporation web server (currently known as Sun Java System Web Server). It has since evolved to rival other Unix-based web servers in terms of functionality and performance. Since April 1996 Apache has been the most popular HTTP server on the World Wide Web; as of March 2007 Apache served 58% of all websites.

Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. Released under the Apache License, the Apache HTTP Server is free/open source software.

Usage

Apache is primarily used to serve both static content and dynamic Web pages on the World Wide Web. Many web applications are designed expecting the environment and features that Apache provides.

Apache is the web server component of the popular LAMP web server application stack, alongside Linux, MySQL, and the PHP/Perl/Python programming languages.

Apache is redistributed as part of various proprietary software packages including the Oracle RDBMS or the IBM WebSphere application server. Mac OS X integrates Apache as its built-in web server and as support for its WebObjects application server. It is also supported in some way by Borland in the Kylix and Delphi development tools. Apache is included with Novell NetWare 6.5, where it is the default web server.

Apache is used for many other tasks where content needs to be made available in a secure and reliable way. One example is sharing files from a personal computer over the Internet. A user who has Apache installed on their desktop can put arbitrary files in the Apache's document root which can then be shared.

Programmers developing web applications often use a locally installed version of Apache in order to preview and test code as it is being developed.

Microsoft Internet Information Services (IIS) is the main competitor to Apache, trailed by Sun Microsystems' Sun Java System Web Server and a host of other applications such as Zeus Web Server.

Features

Apache supports a variety of features, many implemented as compiled modules which extend the core functionality. These can range from server-side programming language support to authentication schemes. Some common language interfaces support mod_perl, mod_python, Tcl, and PHP. Popular authentication modules include mod_access, mod_auth, and mod_digest. A sample of other features include SSL and TLS support (mod_ssl), a proxy module, a useful URL rewriter (also known as a rewrite engine, implemented under mod_rewrite), custom log files (mod_log_config), and filtering support (mod_include and mod_ext_filter). Apache logs can be analyzed through a web browser using free scripts such as AWStats/W3Perl or Visitors.

Virtual hosting allows one Apache installation to serve many different actual 'websites'. For example, one machine, with one Apache installation could simultaneously serve www.example.com, www.test.com, test47.test-server.test.com, etc.

Apache features configurable error messages, DBMS-based authentication databases, and content negotiation. It is also supported by several graphical user interfaces (GUIs) which permit easier, more intuitive configuration of the server.

2.1.4 AJAX

Ajax (also known as AJAX), shorthand for "Asynchronous JavaScript and XML," is a web development technique for creating interactive web applications. The intent is to make web pages feel more responsive by exchanging small amounts of data with the server behind the scenes, so that the entire web page does not have to be reloaded each time the user requests a change. This is intended to increase the web page's interactivity, speed, and usability.

Constituent technologies

The Ajax technique uses a combination of:

- XHTML (or HTML) and CSS, for marking up and styling information.
- The DOM accessed with a client-side scripting language, especially ECMAScript implementations such as JavaScript and JScript, to dynamically display and interact with the information presented.
- The XMLHttpRequest object is used to exchange data asynchronously with the web server. In some Ajax frameworks and in certain situations, an IFrame object is used instead of the XMLHttpRequest object to exchange data with the web server, and in other implementations, dynamically added <script> tags may be used.
- XML is sometimes used as the format for transferring data between the server and client, although any format will work, including preformatted HTML, plain text, JSON and even EBML. These files may be created dynamically by some form of server-side scripting.

Like DHTML, LAMP and SPA, Ajax is not a technology in itself, but a term that refers to the use of a group of technologies.

Advantages and disadvantages

Advantages

- **User interface**

The most obvious reason for using Ajax is an improvement to the user experience. Pages using Ajax behave more like a standalone application than a typical web page. Clicking on links that cause the entire page to refresh feels like a "heavy" operation. With Ajax, the page often can be updated dynamically, allowing a faster response to the user's interaction. While the full potential of Ajax has yet to be determined, some believe it will prove to be an important technology, helping make the Web even more interactive and popular than it currently is.

- **Bandwidth usage**

By generating the HTML locally within the browser, and only bringing down JavaScript calls and the actual data, Ajax web pages can appear to load relatively quickly since the payload coming down is much smaller in size. An example of this technique is a large result set where multiple pages of data exist. With Ajax, the HTML of the page, e.g., a table structure with related TD and TR tags can be produced locally in the browser and not brought down with the first page of the document.

In addition to "load on demand" of contents, some web based applications load stubs of event handlers and then load the functions on the fly. This technique significantly cuts down the bandwidth consumption for web applications that have complex logic and functionality.

- **Separation of data, format, style, and function**

A less specific benefit of the Ajax approach is that it tends to encourage programmers to clearly separate the methods and formats used for the different aspects of information delivery via the web. Although Ajax can appear to be a jumble of languages and techniques, and programmers are free to adopt and adapt whatever works for them, they are generally propelled by the development motive itself to adopt separation between the following:

- Adopt separation between the raw data or content to be delivered - which is normally embedded in XML and sometimes derived from a server-side database.
- Adopt separation between the format or structure of the webpage - which is almost always built in HTML (or better, XHTML) and is then reflected and made available to dynamic manipulation in the DOM.
- Adopt separation between the style elements of the webpage: everything from fonts to picture placement are derived by reference to embedded or referenced CSS.
- Adopt separation between the functionality of the web page which is provided by a combination of
 - Javascript on the client browser (also called DHTML),
 - Standard HTTP and XMLHttpRequest for client-to-server communication, and
 - Server-side scripting and/or programs using any suitable language preferred by the programmer to receive the client's specific requests and respond appropriately.

Disadvantages

- **Browser integration**

The dynamically created page does not register itself with the browser history engine, so triggering the "Back" function of the users' browser might not bring the desired result.

Developers have implemented various solutions to this problem. These solutions can involve using invisible IFRAMEs to invoke changes that populate the history used by a browser's back button. Google Maps, for example, performs searches in an invisible IFRAME and then pulls results back into an element on the visible web page. The World Wide Web Consortium (W3C) did not include an iframe element in its XHTML 1.1 Recommendation; the Consortium recommends the object element instead.

Another issue is that dynamic web page updates make it difficult for a user to bookmark a particular state of the application. Solutions to this problem exist, many of which use the URL fragment identifier (the portion of a URL after the '#'[6][7]) to keep track of, and

allow users to return to, the application in a given state. This is possible because many browsers allow JavaScript to update the fragment identifier of the URL dynamically, so that Ajax applications can maintain it as the user changes the application's state. This solution also improves back-button support. It is, however, not a complete solution.

- **Response-time concerns**

Network latency — or the interval between user request and server response — needs to be considered carefully during Ajax development. Without clear feedback to the user,[8] smart preloading of data and proper handling of the XMLHttpRequest object, users might experience delay in the interface of the web application, something which they might not expect or understand. Additionally, when an entire page is rendered there is a brief moment of re-adjustment for the eye when the content changes. The lack of this re-adjustment with smaller portions of the screen changing makes the latency more apparent. The use of visual feedback (such as throbbers) to alert the user of background activity and/or preloading of content and data are often suggested solutions to these latency issues.

- **Search engine optimization**

Websites that use Ajax to load data which should be indexed by search engines must be careful to provide equivalent data at a public, linked URL and in a format that the search engine can read, as search engines do not generally execute the JavaScript code required for Ajax functionality. This problem is not specific to Ajax, as the same issue occurs with sites that provide dynamic data as a full-page refresh in response to, say, a form submit (the general problem is sometimes called the hidden web).

- **Reliance on JavaScript**

Ajax relies on JavaScript, which may be implemented differently by different browsers or versions of a particular browser. Because of this, sites that use JavaScript may need to be tested in multiple browsers to check for compatibility issues. It's not uncommon to see a JavaScript code written twice, one part for IE, another part for Mozilla compatibles. (see

also Cross-platform web design). The level of IDE support for JavaScript is exceptionally poor.

An issue also arises if the user has switched off JavaScript support in the browser, thus disabling the functionality of the page.

- **Web Analytics**

Many web analytics solutions are based on the paradigm of a new page being loaded whenever new or updated content is displayed to the user, or to track a series of steps in a process such as a check-out. Since AJAX alters this process, care must be taken to account for how to instrument a page or a portion of a page so that it can be accurately tracked. Analytics systems which allow for the tracking of events other than a simple page view, such as the click of a button or link, are the ones most likely to be able to accomodate a site which heavily utilizes AJAX.

xajax Framework for AJAX

xajax is an open source PHP class library that allows you to easily create powerful, web-based, Ajax applications using HTML, CSS, JavaScript, and PHP. Applications developed with xajax can asynchronously call server-side PHP functions and update content without reloading the page.

How does xajax work?

The xajax PHP object generates JavaScript wrapper functions for the PHP functions you want to be able to call asynchronously from your application. When called, these wrapper functions use JavaScript's XMLHttpRequest object to asynchronously communicate with the xajax object on the server which calls the corresponding PHP functions. Upon completion, an xajax XML response is returned from the PHP functions, which xajax passes back to the application. The XML response contains instructions and data that are parsed by xajax's JavaScript message pump and used to update the content of your application.

xajax's unique XML response / javascript message-pump system does the work for us, automatically handling the data returned from our functions and updating our content or state according to the instructions we return from our PHP functions. Because xajax does the work, we don't have to write javascript callback handler functions.

3. PROJECT PLAN

3.1 Purpose

To provide a complete solution for the student, teachers and administrators of an Institution which will be easy to use and can be implemented on Internet or Intranet.

There are many e-learning solutions such as BlackBorad and Moodle which are successfully being used in other coutries but cannot be used in our country.

3.2 Project Scope

“i-Solution” is designed to be implemented on the web which will provide the following solutions to the students, teachers and administrators:

Students:

- Online Library Information
- Check Attendance
- Archive for Study materials
- Notices

Teaching Staff:

- Attendance Management
- Notices
- Manage Study Material Archive

Administrative Staff:

- Staff Management
- Student Management
- Library Management
- Class Management

3.3 Feasibility study

Similar to all systems, while considering proposed system many aspects and factors were taken into consideration, which we may call feasibility study. Following points were considered so that proposed system will be easily taken over after completion with current setup and will be economical. The results were as follows:

Technical feasibility:

The technical requirements of the system are

Service	Technology Used
Language	PHP
Database	mySQL
HTTP Server	Apache
AJAX Framework	xajax Framework

Table 3.1 Technical Feasibility

Operational feasibility:

The users of this web application are disturbed with the web site environment, so there is no real necessity of training them to use this system. Utmost care has been taken to make the site extremely user-friendly by designing great UIs with the help of JS ad AJAX.

Economical feasibility:

For the economic feasibility the cost benefit analysis was done to verify whether the proposed system is economically feasible or not. This means that the cost incurred on the installation of the hardware and software should be minimum, but profits gained must be maximum.

The proposed hardware is already present with the users and with company as well, only cost of domain name registration needs to be considered IF the solution is deployed on the internet.

With the consideration of the following factors the proposed system is economically feasible.

3.4 Resources

3.4.1 Human Resources

The democratic decentralized approach has been followed to complete the project. The development team consists of 2 people. The team didn't have any permanent leader, rather there were task coordinators allocated for short durations. Decisions on problems and approach were made by group consensus. Communication among team members was horizontal.

The problem to be solved was modularized and all the modules were divided between the two team members. Each of these parts was taken up by one team member who looked after his own module and little combined work. Only later, during coordination and integration of modules, the team members worked together.

3.4.2 Software Tools

- PHP
- MySql
- Altova Umodel

3.5 Cost Estimation:

Basic COCOMO Model:

$$E = a_b * (\text{KLOC})^{b_b}$$

$$D = c_b * (E)^{d_b}$$

Where

E = efforts applied in person months

D = development time in chronological months

For a semi-detached project

$$a_b = 3.0, b_b = 1.12, c_b = 2.5, d_b = 0.35$$

Therefore,

$$E = 7.585 \text{ person months}$$

$$D = 6.564 \text{ chronological months}$$

3.6 Schedule

1	2	3	4
Requirement Specifications	Learning the Technologies Involved	Functional Spec Database Design	Implementation Testing, Reviews
July - Sept	August - October	October - Dec	January – April
Gathered requirements by discussing with Guide, Teachers, Library Staff and our own experience in the Institution	Collected material on PHP, Created sample web applications, Read material on object programming, Searched for feasible AJAX Framework	Table Design Design Security rules Create schema files UI design	Implement classes for functionality Implement modules: Implement views in HTML Testing

Table 3.2 Project Schedule

3.7 Risk Management:

Risk is the probability that a loss will occur, "a weighted pattern of possible outcomes and their associated consequences." It indicates the probability that a software project will experience undesirable events, such as schedule delays, cost overruns, or outright cancellation. Risk is proportional to size and inversely proportional to skill and technology levels. Therefore the larger the project, the greater is the risk.

The risks involved in any software development can be condensed down to a set of *three* very critical issues that occur over and over again in hundreds of corporations:

1. Software projects are not estimated or planned with acceptable accuracy.
2. Software project status reporting is often wrong and misleading.
3. Software quality and reliability are often unacceptably poor.

Hence, it is crucial to examine the root causes of each of these key risk factors and explore the current state of the art for minimizing their harmful effects.

3.7.1 Risk Management Process:



Figure 3.1 Risk Management Process

Identify

The purpose of *identification* is to consider risks before they become problems and to incorporate this information into the project management process. Anyone in a project can identify risks to the project. Each individual has particular knowledge about various parts of a project. During Identify, uncertainties and issues about the project are transformed into distinct (tangible) risks that can be described and measured.

Analyze

The purpose of *analysis* is to convert the data into decision-making information. Analysis is a process of examining the risks in detail to determine the extent of the risks, how they relate to each other, and which ones are the most important. Analyzing risks has *three* basic activities: evaluating the attributes of the risks (impact, probability, and timeframe), classifying the risks, and prioritizing or ranking the risks.

Plan

Planning is the function of deciding what, if anything, should be done about a risk or set of related risks. In this function, decisions and mitigation strategies are developed based on current knowledge of project risks. Hence, the purpose of planning is to:

- make sure the consequences and the sources of the risk are known
- develop effective plans
- plan efficiently (only as much as needed or will be of benefit)
- produce, over time, the correct set of actions that minimize the impacts of risks (cost and schedule) while maximizing opportunity and value
- plan important risks first

Track

Tracking is the process by which risk status data are acquired, compiled, and reported. The purpose of tracking is to collect accurate, timely, and relevant risk information and to present it in a clear and easily understood manner to the appropriate people/group. Tracking is done by those responsible for monitoring "*watched*" or "*mitigated*" risks. Tracking status information becomes critical to performing the next function in the Continuous Risk Management paradigm, *i.e.* Control. Supporting information, such as schedule and budget variances, critical path changes, and project/performance indicators can be used as triggers, thresholds, and risk- or plan-specific measures where appropriate.

Control

The purpose of the *Control* function is to make informed, timely, and effective decisions regarding risks and their mitigation plans. It is the process that takes in tracking status information and decides exactly what to do based on the reported data. Controlling risks involves analyzing the status reports, deciding how to proceed, and then implementing those decisions.

3.7.2 Project Risks Factors:

Business Impact Risks:

This is the risk where concern is that of the not being able to come up or produce the product that has an impact on the client's business. If the software produced does not achieve its goals or if it fails to help the business of clients improve in special ways, the software development basically fails.

Customer Risks:

This is the risk where concern is client's motivation or willingness in helping the software development team. If the client fails to attend meeting regularly and fails to describe the real need of the business the produces software will not be one that helps the business.

Development Risks:

If client fails to provide all the necessary equipment for the development and execution of the software this will cause the software to become a failure. So in other words customer has to be able to provide time and resource for the software development team. If all the requested resources are not provided to the software development team odds for the software development to fail rises greatly.

Employee Risks:

This risk is totally dependent on the ability, experience and willingness of the software development team members to create the working product. If the team members are not experienced enough to use the application necessary to develop the software It will keep pushing the development dates until it's too late to save the project. If one or more members of the software development team are not putting in all the effort required to finish the project it will cause the project to fail. Employee risk is one of the major risk to consider while designing the software.

Process Risks:

Process risk involves risks regarding the product quality. If the product developed does not meet the standards set by the customer or the development team it is a failure. This can happen because of the customer's failure to describe the true business need or the failure of the software development team to understand the project and then to use proper equipment and employees to finish the project.

Product Size:

This risk involves misjudgment on behalf of the customer and also the software development team. If the customer fails to provide the proper size of the product that is to be developed it will cause major problems for the completion of the project. If the software development team misjudges the size and scope of the project, the team may be too small or large for the project thus spending too much money on project or not finishing project at all because of shortage of finances.

Technology Risk:

Technology risk involves of using technology that already is or is soon to be obsolete in development of the software. Such software will only be functional for short period of time thus taking away resources from the customer. Since the technology changes rapidly these days it is important to this risk. If customer requests use of software that soon to be obsolete software development team must argue the call and have to pursue customer to keep up with current technology.

3.7.3 RISK TABLE:

Sr. No.	Risks	Category	Probability (%)	Impact
1	Failure to meet the requirements	Project risks	10	Critical
2	Non Responsive or unsupportable software	Development Environment Risk	20	Marginal
3	Reduction in technical performance	Staff risks	20	Critical
4	Size estimate may be significantly low	Process risk	60	Marginal
5	Larger no. of users than planned	Product Risk	30	Negligible
6	Less reuse than planning	Product risk	40	Marginal
7	End users resist system	Customer characteristics	60	Catastrophic
8	Delivery deadline will be tightened	Business impact	40	Critical
9	Customer will change requirements	Customer Characteristics	60	Critical
10	Lack of training on tools	Technology risk	60	Critical
11	Staff irregularity	Staff risk	2	Marginal

12	Lack of adequate guidance	Business risk	50	Marginal
13	Insufficient staff	Staff risk	20	Negligible
14	Lack of coordination	Staff risk	30	Fair
15	Lack of budget	Business risk	10	Negligible
16	Difficulty in obtaining software to be used for code development	Development environment risk	30	Marginal
17	Technology will not meet expectation	Technology risk	10	Critical

Table 3.3 Risk Table

3.7.4 RMMM plan:

Risk:	Someone else comes up with a similar concept, before we finish our project.
Mitigation:	Keep all the work done documented.
Monitor:	The latest developments in the market.
Management:	Distribute work evenly among team members, and assure dead lines are met.

Table 3.4 Risk #1

Risk:	Site may temporarily crash if too many users try to access the system.
Mitigation:	Keep cleaning and maintaining database.
Monitor:	Do load testing at various phases of development.
Management:	Recover from any technical loss as soon as possible.

Table 3.5 Risk #2

Risk:	Schedules might slip as team members are relatively inexperienced.
Mitigation:	Keep all the work done documented.
Monitor:	Take timely reports from all team members about their progress.
Management:	Review schedules and consult internal and external guides for technical help.

Table 3.6 Risk #3

Risk:	Communication gaps between internal guides and team members.
Mitigation:	Keep informing all relevant members about the progress of the project.
Monitor:	Keep taking invaluable tips from people from the same field.
Management:	Reduce level of expectations and be more understanding & reasonable.

Table 3.7 Risk #4

Risk:	There is lack of continuity of effort.
Mitigation:	Give deadlines to each team member.
Monitor:	Keep and account of all the weekly progress made.
Management:	Figure out reasons for slow progress; take guidelines from people experienced in the field. Increase levels of pressure.

Table 3.8 Risk #5

Risk:	There is a lack of resources for learning a relatively new language.
Mitigation:	Gather as much information as possible on the technology.
Monitor:	Keep searching on the inter net for various books.
Management:	Consult external guides and request them to arrange for some books on the technology.

Table 3.9 Risk #6

Risk:	Quality of product documentation and coding that must be produced may be low.
Mitigation:	Decide a uniform format to be followed.
Monitor:	Keep getting the codes and document reviewed from time to time.
Management:	Review the quality of product by taking external help and maintain industry standards.

Table 3.10 Risk #7

Risk:	The debugging phase might take longer than expected.
Mitigation:	Practice debugging and keep in touch with testing strategies.
Monitor:	Project Schedules.
Management:	Review project schedules and put more effort for debugging.

Table 3.11 Risk #8

3.8 SQA PLAN

This part of the document describes the Software Quality Assurance Plan for Talent Capture project. It mainly focuses on the construction phase of development. The analysis and design phases are also taken into consideration.

Definition

SQA is the planned and systematic set of activities that ensure that software process and products conform to requirements, standards, and procedures. "Processes" include all activities involved in designing, coding, testing and maintaining; "products" include software, associated data, documentation, and all supporting and reporting paperwork.

Goals of SQA

The software development is a complex process full of risks. There are technical risks such as software will not perform as intended or be too hard to operate, modify, and/or maintain; there are programmatic risks such as the project will overrun cost or schedule. The goals of SQA are to reduce these risks by:

- Appropriately monitoring the software and the development process.
- Ensuring full compliance with standards and procedures for software and process.
- Ensuring that inadequacies in product, process, or standards are brought to management's attention so that they can be fixed.

Organization role in SQA

Since the project team consists of only three members, there will not be a specially created SQA group. Instead, the project team will be responsible for maintaining software quality at every step of development lifecycle. Timely formal reviews will be conducted with the external and internal guides to ensure that requirements of the customer are given first priority.

SQA tasks:*TASK 1:*

The first SQA task will be development of the project's software process description. A process for the work to be performed will be selected. Then, the team will review the process description for compliance with organizational policy, internal software standards, externally imposed standards (e.g. ISO-9001), and other parts of the software project plan.

TASK 2:

The second SQA task will be to conduct formal reviews of software engineering activities to verify compliance with the defined software process. The deviations from the process will be identified, documented and tracked. The corrections that have been made will be verified.

TASK 3:

The third SQA task will be to audit designated software work products to verify compliance with those defined as part of the software process. The project team will review selected work products; identify, document and track deviations; and verify that corrections have been made.

TASK 4:

The fourth SQA task will be to ensure that deviations in software work and work products are documented and handled according to a documented procedure. These deviations may be encountered in the project plan, process description, applicable standards, or technical work products.

TASK 5:

The final task will be to record any noncompliance.

3.8.1 Work Products and Documentation:

As a result of the above SQA activities, changes to the initial design were made in order to take into consideration, the exact requirements of the customer. The corresponding changes were also incorporated into the architectural design of the software.

Also, all defects encountered in the software requirements, design or codes were documented accurately. After every formal technical review, a formal technical review summary report was created.

3.8.2 Reviews and Audits:

This section discusses major project reviews conducted by the project team members.

Generic review guidelines:

The following set of guidelines for the conduct of Formal Technical Reviews was established:

1. Set an agenda and maintain it.
2. Limit debate and rebuttal.
3. Enunciate problem areas, but don't attempt to solve every problem noted.
4. Take written notes.
5. Allocate resources and schedule time for formal technical reviews.
6. Review your early reviews.

Formal Technical Reviews:

A description of the specific character and intent of each major FTR conducted during the software process.

Review 1- Requirements gathering phase:

This review was conducted after the SRS of the product had been jotted down. The participants included the external guide and two team members.

Description and focus of the review:

The review was mainly focused on ensuring that there is no communication gap between the development team and the customer. Through this review, we aimed to discard the probability of wrongly conceived customer requirements. This was an extremely crucial review because the success of any project depends on the accuracy of the requirements specification and the compliance of the requirements specification with the requirements of the customer.

Review 2- Design Phase

This review was focused on identifying deficiencies in the design document.

This review was conducted after the basic UI design and a rough draft of the database design had been created. The participants included the internal guide, Prof. Mrs. Patankar and two team members. Later, the design document was also discussed with the college internal guide for approval.

Description and focus of the review:

The main focus of this FTR was to make sure that the visual aspects of the product as conceived by the development team comply with the customer's view of the product. Also, it was to be made sure that the database design ensures optimum storage of data with minimal redundancy. The database design was tested thoroughly to see if it satisfies maximum number of normalization rules.

Review 3- Code development Phase

This review was focused at uncovering errors in the working model of the product.

This review was conducted after the code for the product had been created. Now, the actual working model was in front of the review team, instead of just a prototype or paper documents. The participants included the internal guide, H.O.D. and two team members.

Description and focus of the review:

The review was mainly focused on ensuring that the navigation across pages is smooth and intuitive. Thorough testing of all modules was done to ensure that the product performs according to what is expected of it. It was seen to that the output is displayed to the user in an understandable format and that the user interface is easy to comprehend and use. Aspects concerning the visual appeal of the product were also considered.

4. SOFTWARE REQUIRMENTS SPECIFICATION

4.1: Introduction:

This Software Requirements Specification provides a brief description of all the functions and specifications of the product "i-solution".

4.2: Purpose:

To provide a complete solution for the student, teachers and administrators of an Institution which will be easy to use and can be implemented on Internet or Intranet.

There are many e-learning solutions such as BlackBorad and Moodle which are successfully being used in other coutries but cannot be used in our country.

4.3: Scope Of The Problem In Detail:

“i-Solution” is designed to be implemented on the web which will provide the following solutions to the students, teachers and administrators:

Students:

- Online Library Information
- Check Attendance
- Archive for Study materials
- Notices

Teaching Staff:

- Attendance Management
- Manage Study Material Archive

- Notices

Administrative Staff:

- Staff Management
- Student Management
- Library Management
- Class Management

Characteristics of a good application:

An interface between a buyer and a seller should exist that is:

1. Clean.
2. Reachable.
3. Usable without much effort/Convenient.
4. Effective/Works.
5. Feature rich.
8. Inexpensive.

Reach:

All the students, teachers and administrators have access to the college internet/intranet facility. This solution can also be deployed over the internet which can be accessed by the end users from anywhere.

Convenient:

We are not introducing any new technology which a person will spend days getting used to, we will be using current prevalent technologies to solve the existing problem.

Content Centric/Effective:

By implementing an exhaustive information system via a web application. The web application is designed with the focus being content. The information system offers bottom-up approach

Feature rich:

The web interface provides more functionality to our service.

Inexpensive:

There is no cost involved for the end users to use this solution. The only cost involved is in the deployment and maintenance of the system that will be covered by the institution.

4.4 Overall description:

User Characteristics:

The Functional API is a table that describes the interface via which we can access the functions out system offers:

Fnc ID	Description	Requester	Possible Response
1	Creating Batches	Administrator	Batches created with appropriate start and end dates.
2	Managing Branches	Administrator	Branch is Created.
3	Staff and Student Management	Administrator	Staff and Student profiles added to appropriate branches.
4	Manage classes, levels, subjects, subject teachers.	Administrator	All the above are added showing the relationship between all of them.
5	Change Letter Format	Administrator	A proper letter format is set to be printed later.
6	Manage Libraries	Administrator	Different Libraries of an institution are added along with the users to manage those libraries.
7	Request for attendance	Class Teacher	The attendance request is sent to all the subject teachers for that class.
8	Generate Reports	Class Teachers	Appropriate and Extensive Reports are generated for the attendance.
9	Print Letters	Class Teachers	Letters generated to be sent to the guardian's address. Using the letter format set by the

			administrator.
10	Search Books	Teachers, Students	Information regarding the book and its availability is provided
11	Check Attendance	Teachers	Students get to see their attendance in real time.
12	Upload Study Material	Teachers, Students	Study material is made available to the students and teachers.
13	Send Notices	Administrators, Teachers	Notices can be sent to teachers or students.
14	Accession Register, Book Circulation	Library Staff	Management of all the books.

Table 4.1: Functional API

4.5 External interfaces required

1. Hardware requirements
 - Processor: Pentium 4 onwards
 - RAM: recommended above 128 mb
 - Server: Apache
2. Software requirements
 - Language: PHP
 - Database: MySQL
 - Internet explorer 5 above, Firefox

4.6 System features

- Portable
- Efficient
- Scalable
- Easy to use(User friendly)
- Navigable
- Authenticated

4.7 Graphical User Interfaces

- HTML
- Javascript
- Used CSS file for maintaining consistency in GUI screens

5. DESIGN SPECIFICATIONS

5.1 Development Method

While designing i-solution we followed an iterative and incremental development approach. In this approach a program is developed layer by layer in a prototype mode. We first developed a prototype application and then added features onto it, re-assessing requirements and risk at every stage.

5.2 System Architecture

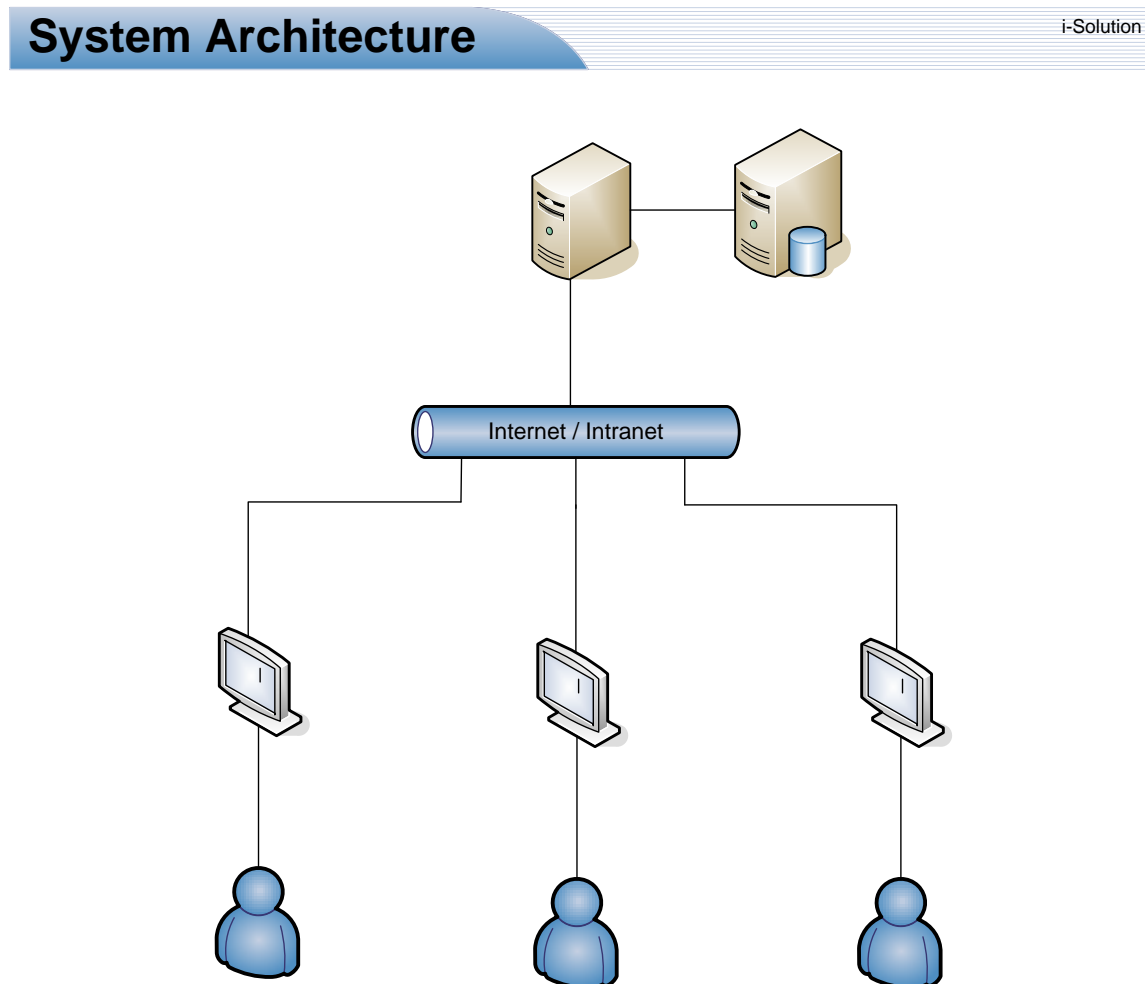


Figure 5.1: System Architecture

5.3 Use Case Diagrams

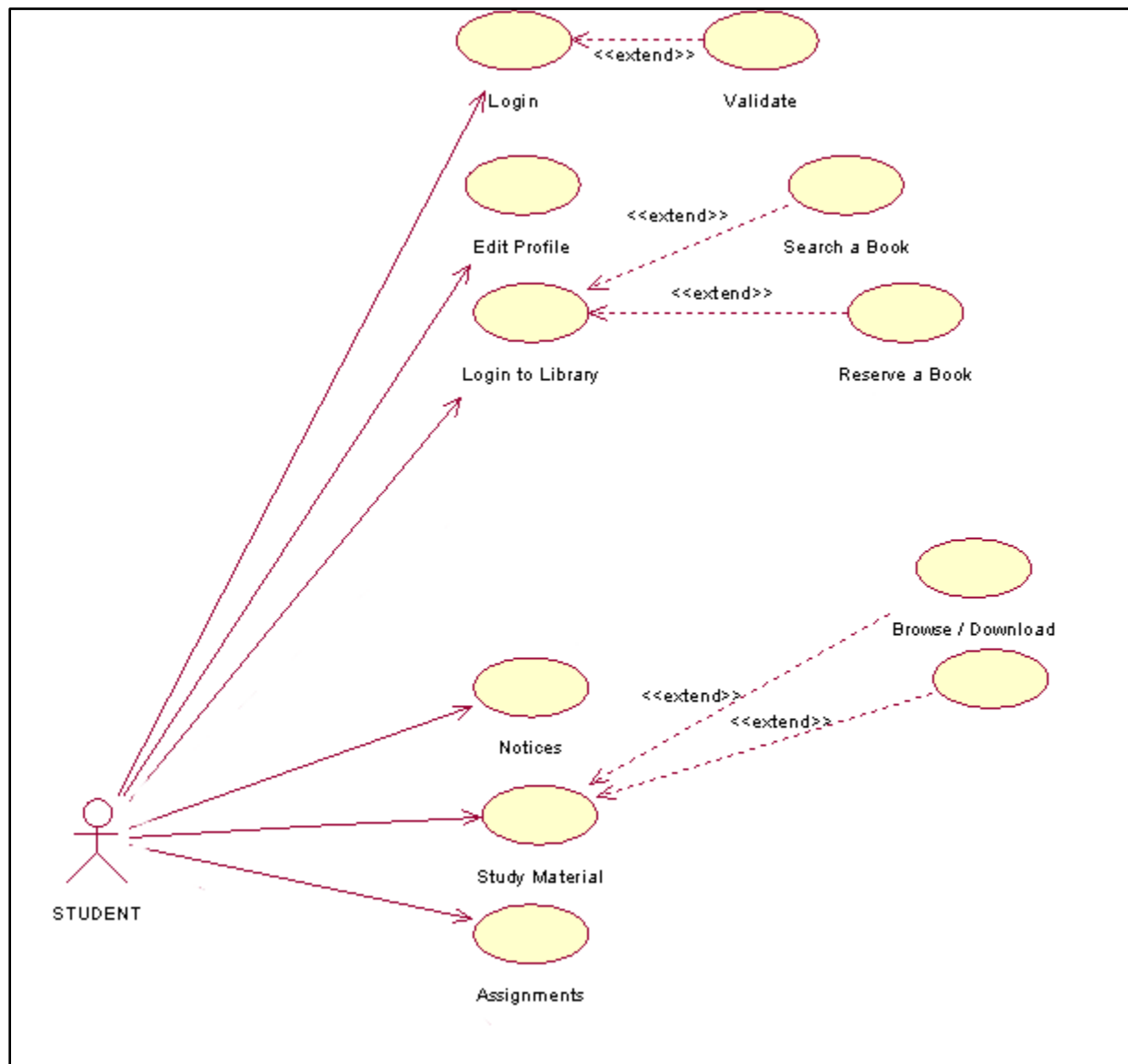


Figure 5.2: Use Case (Student)

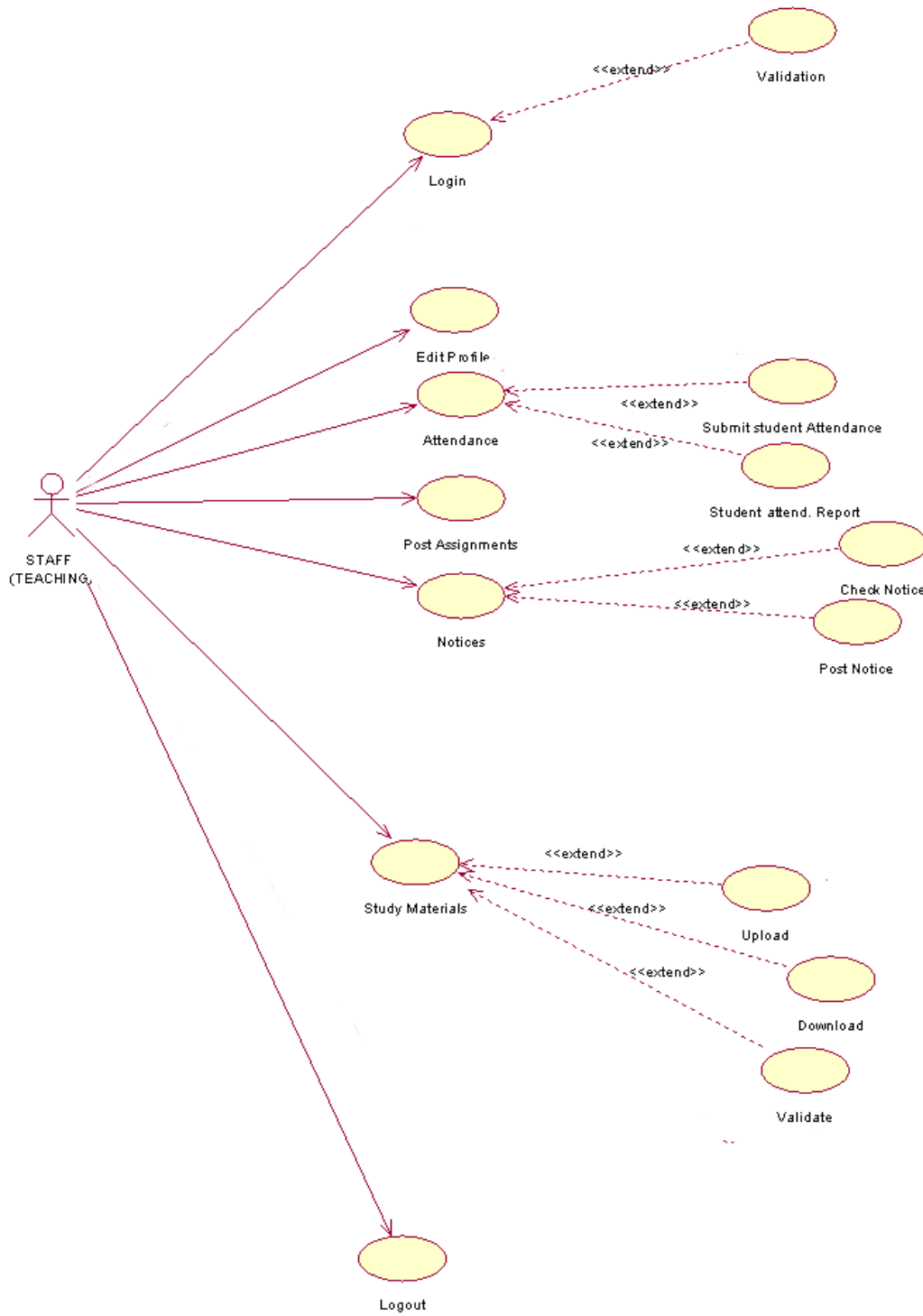


Figure 5.3: Use Case Teachers

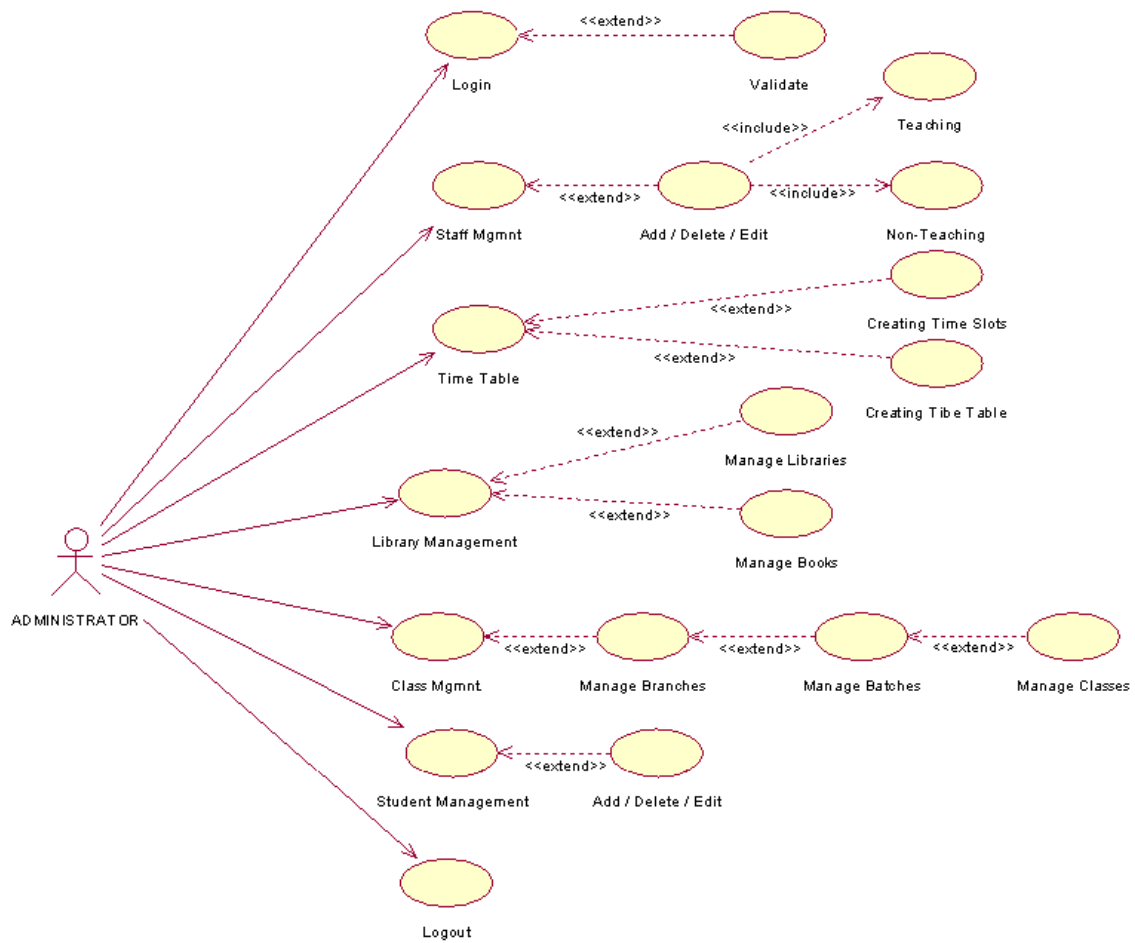


Figure 5.4: Use Case (Administrator)

5.4 Sequence Diagrams

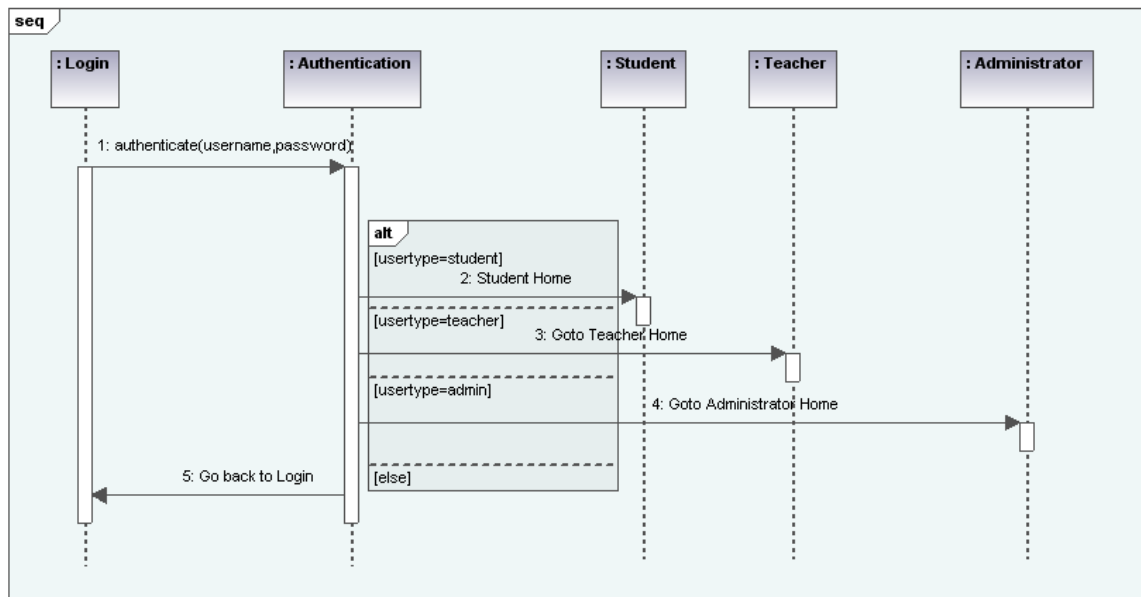


Figure 5.5 Sequence Diagram for Login

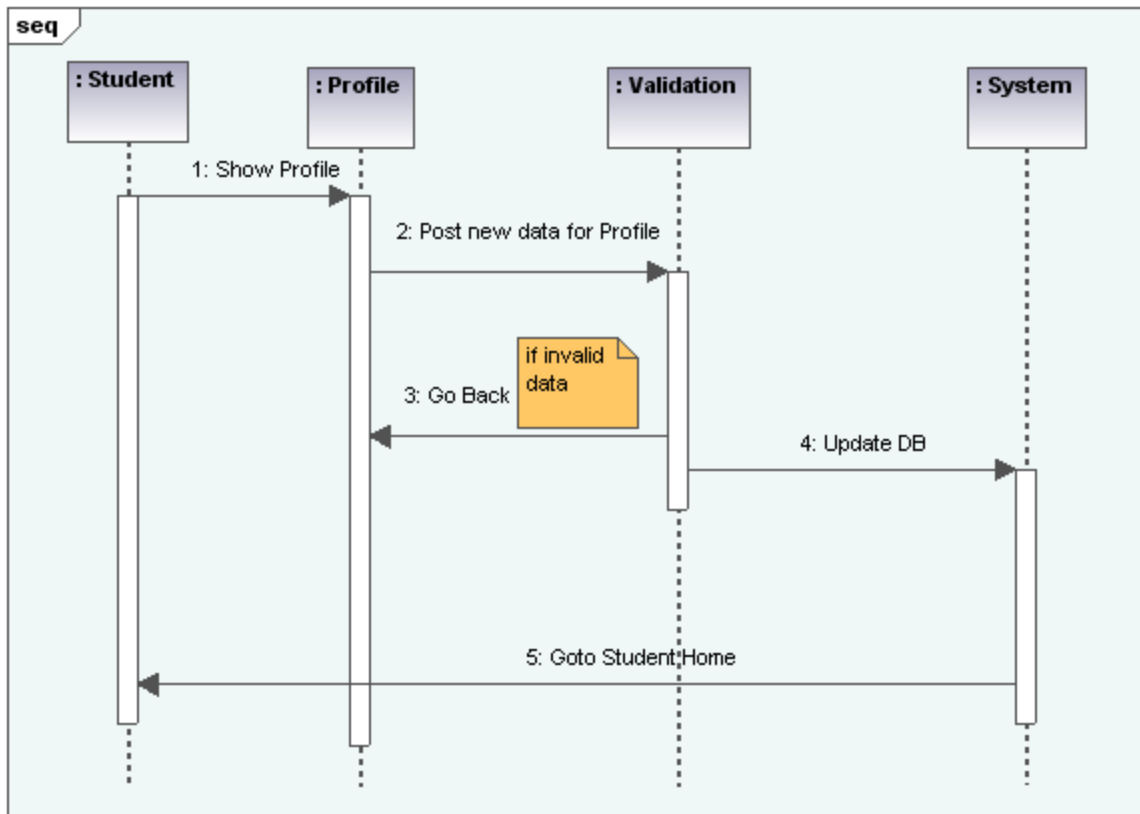


Figure 5.6 Sequence Diagram for Profile (Student)

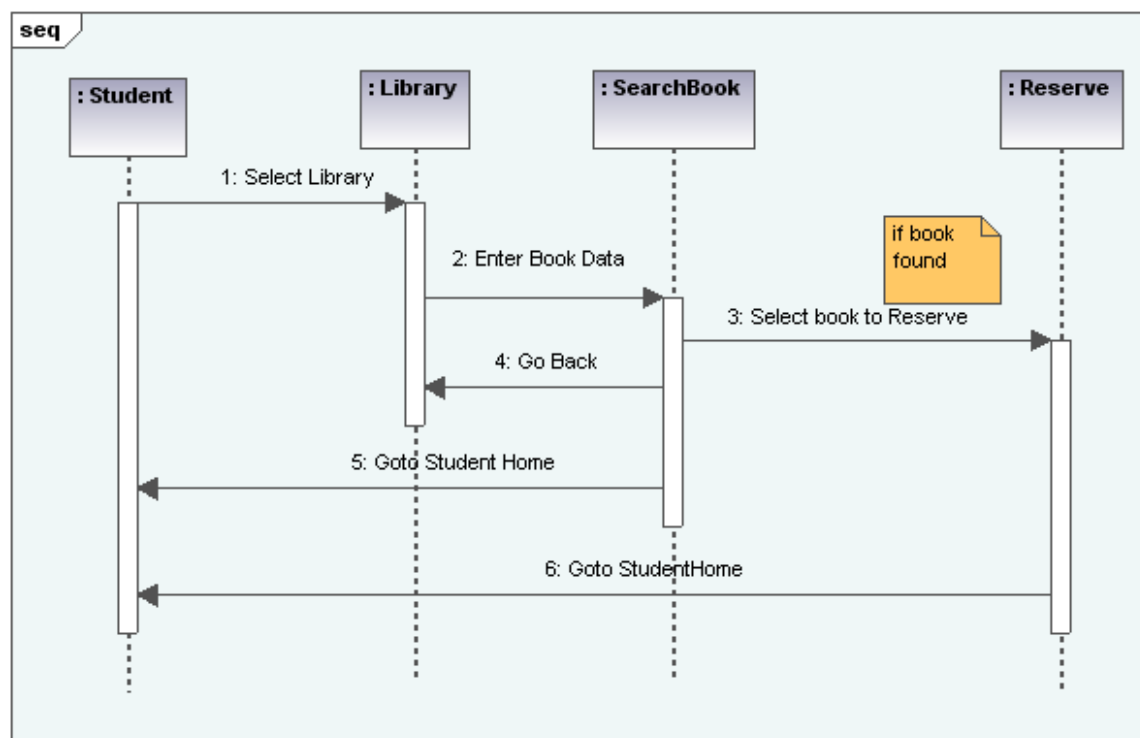


Figure 5.7 Sequence Diagram for Library (Student)

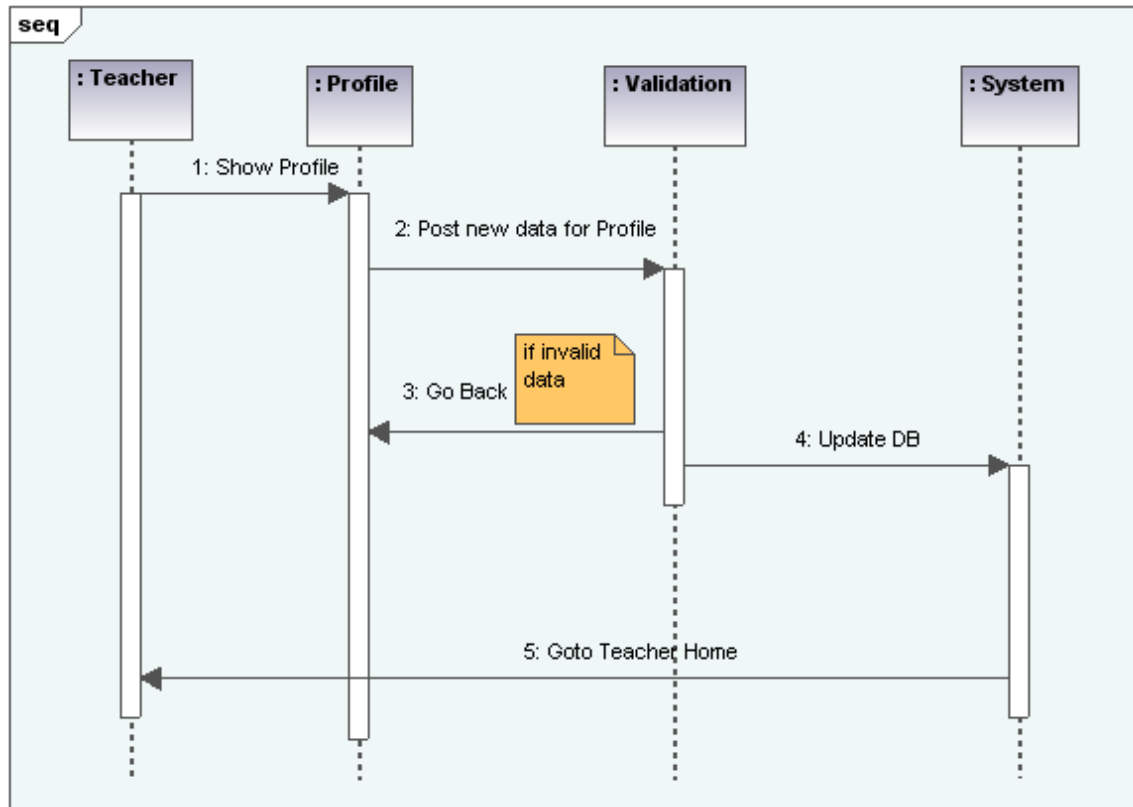


Figure 5.8 Sequence Diagram for Profile (Teachers)

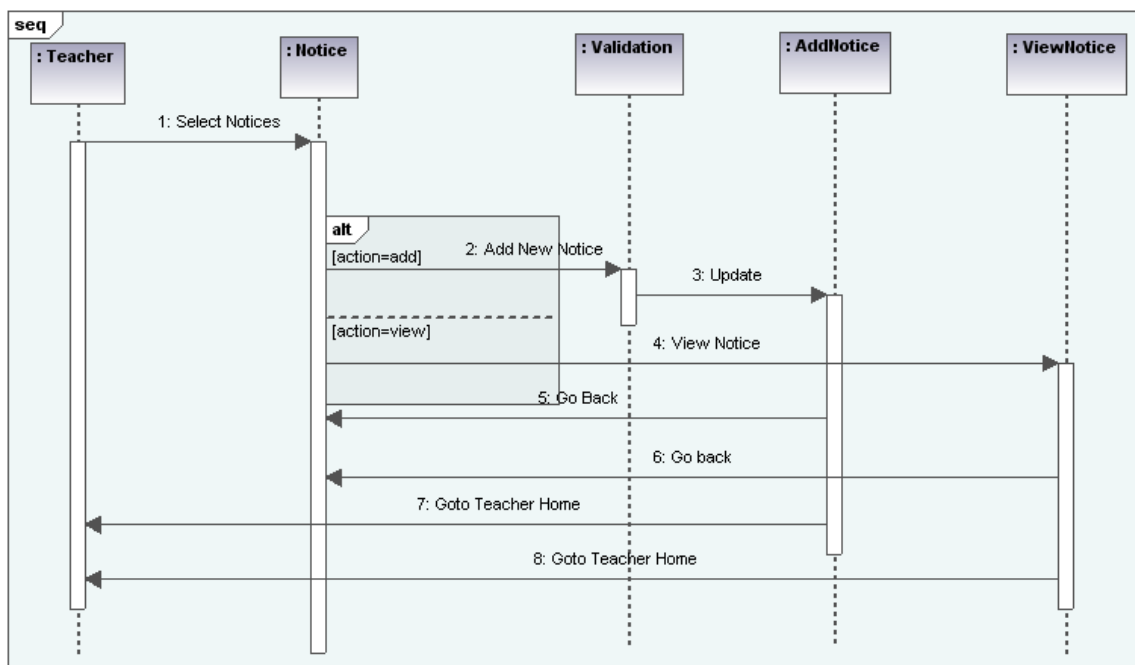


Figure 5.9 Sequence Diagram for Notices (Teachers)

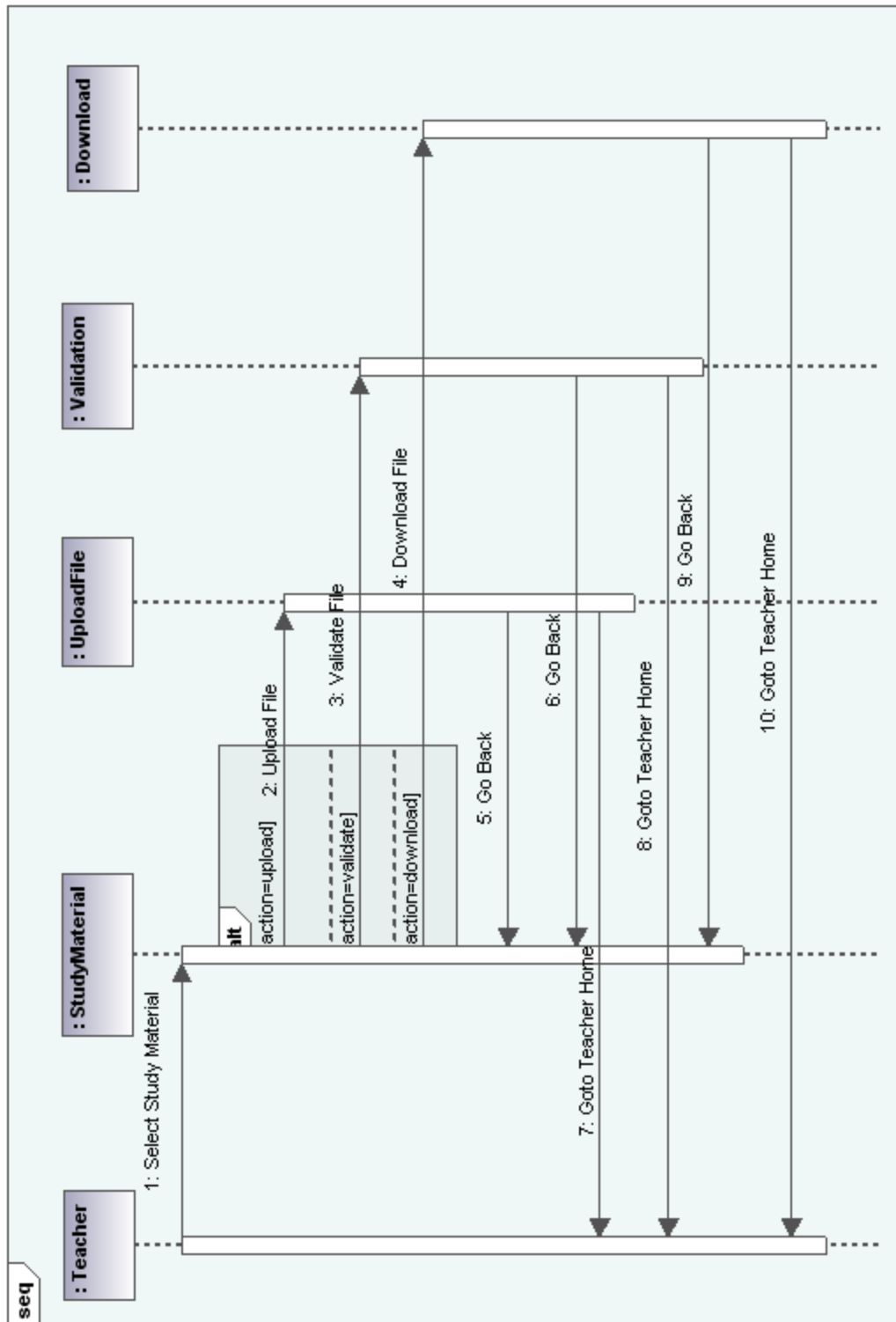


Figure 5.10 Sequence Diagram for Study Material (Teachers)

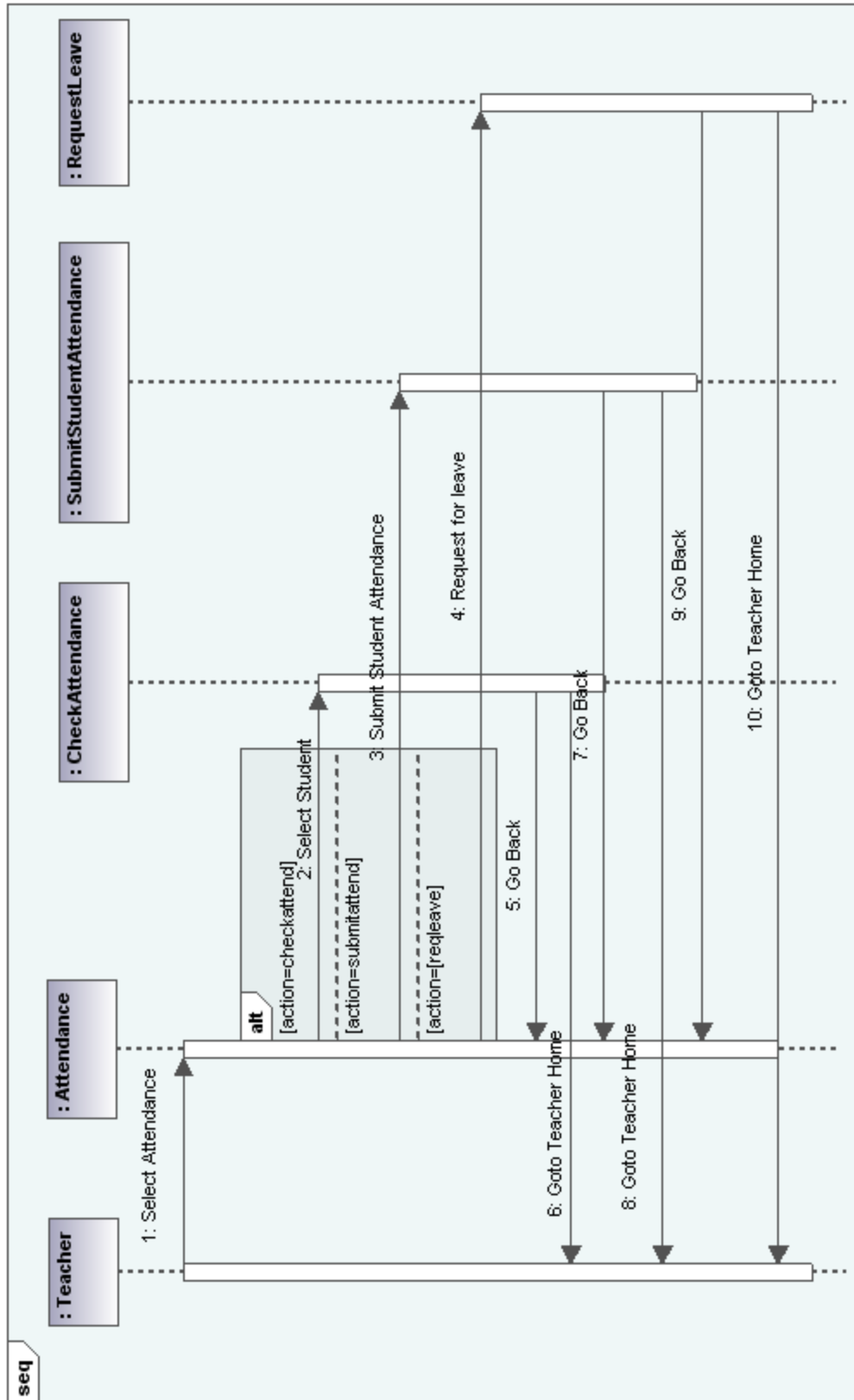


Figure 5.11 Sequence Diagram for Attendance (Teachers)

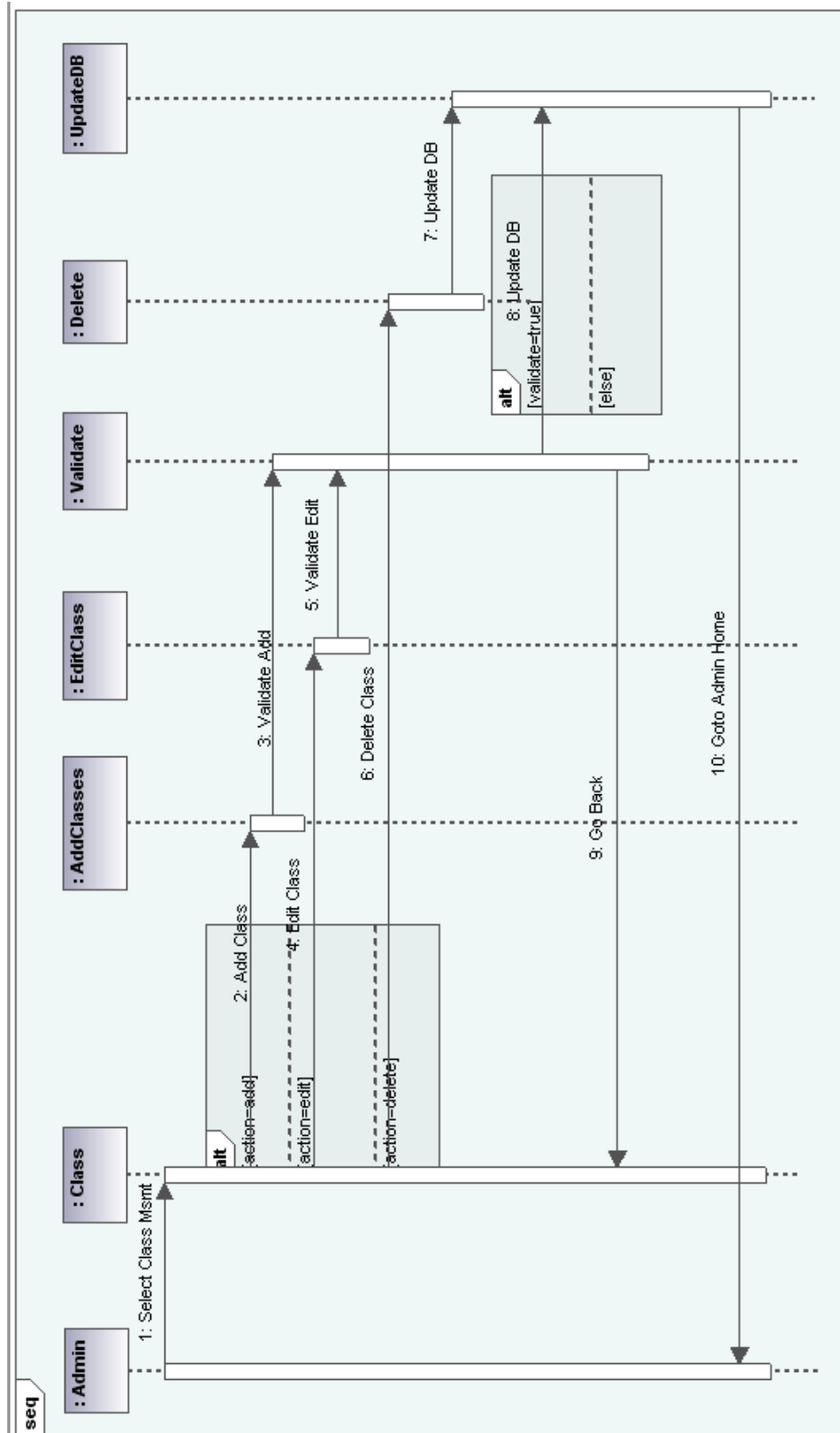


Figure 5.12 Sequence Diagram for Class Management (Admin)

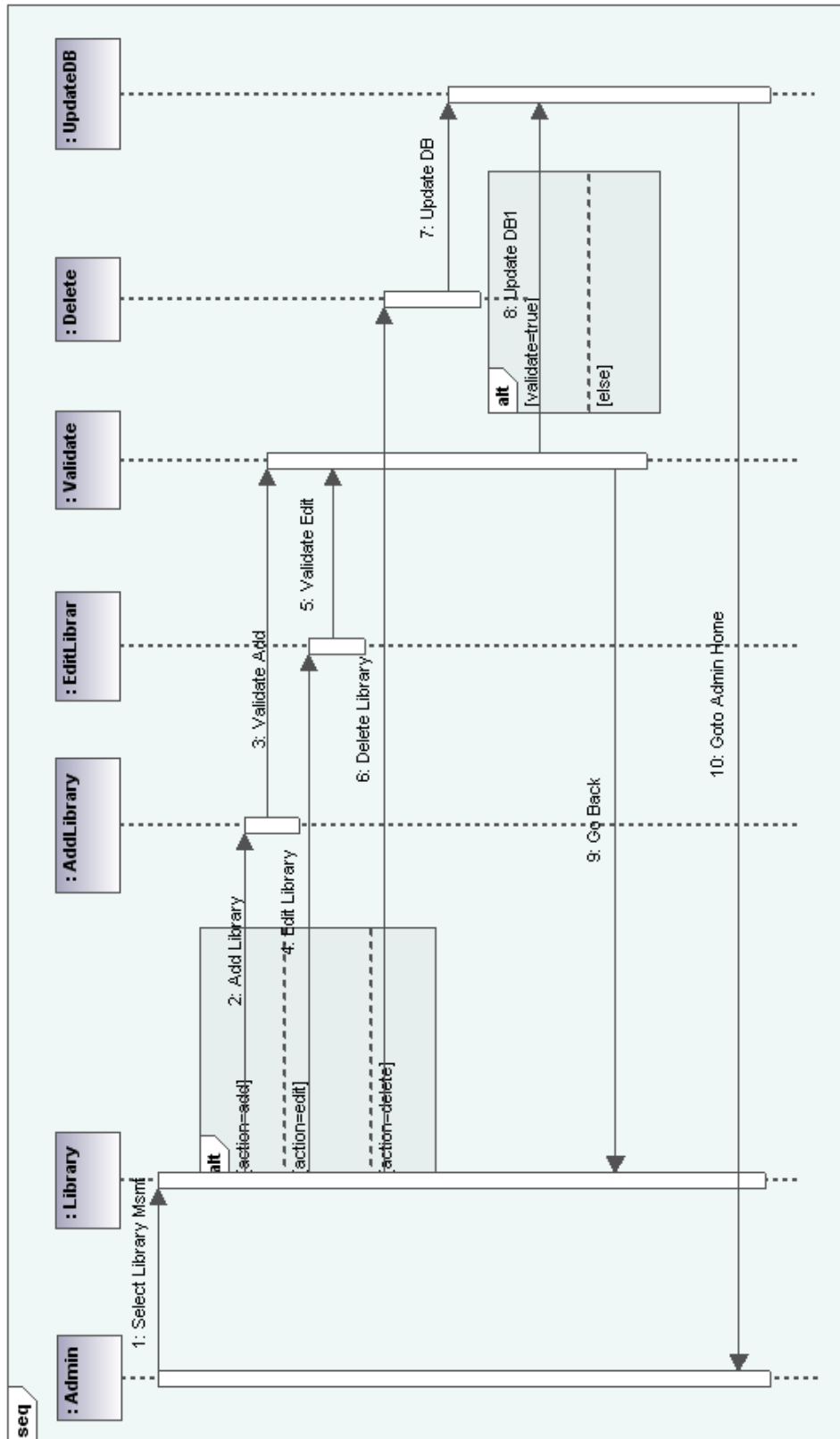


Figure 5.13 Sequence Diagram for Library Management (Admin)

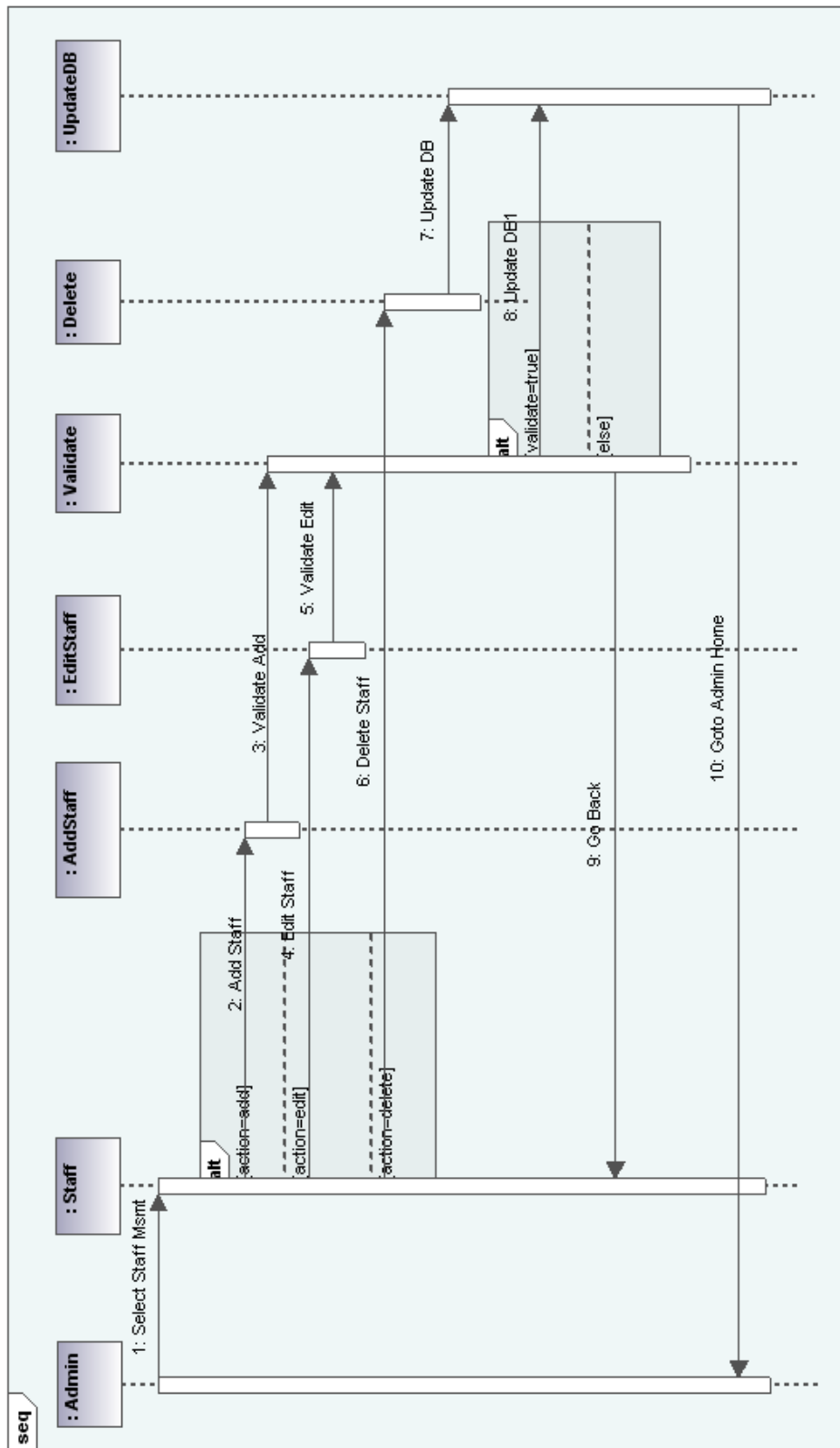


Figure 5.14 Staff Management (Admin)

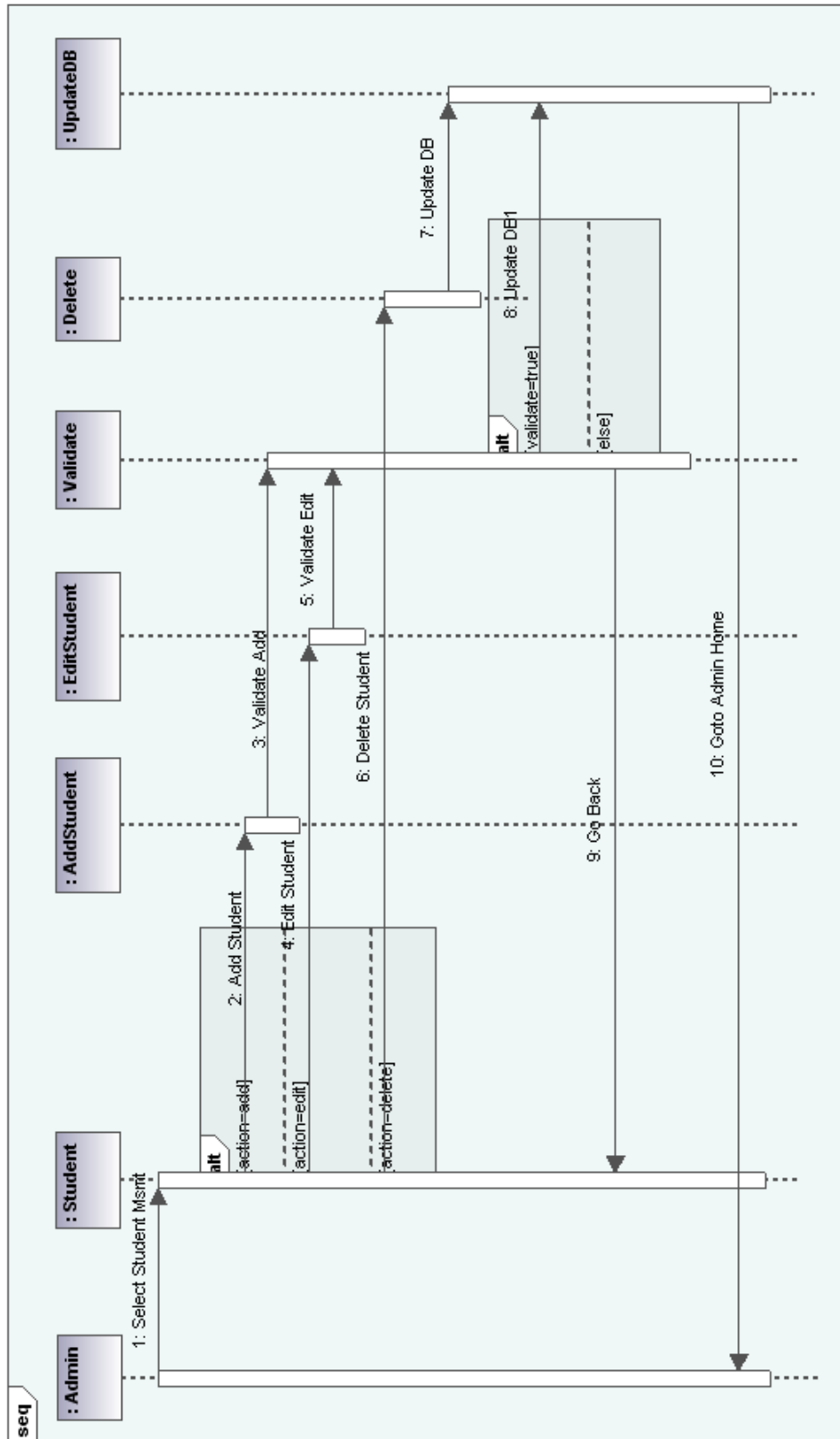


Figure 5.15 Student Management (Admin)

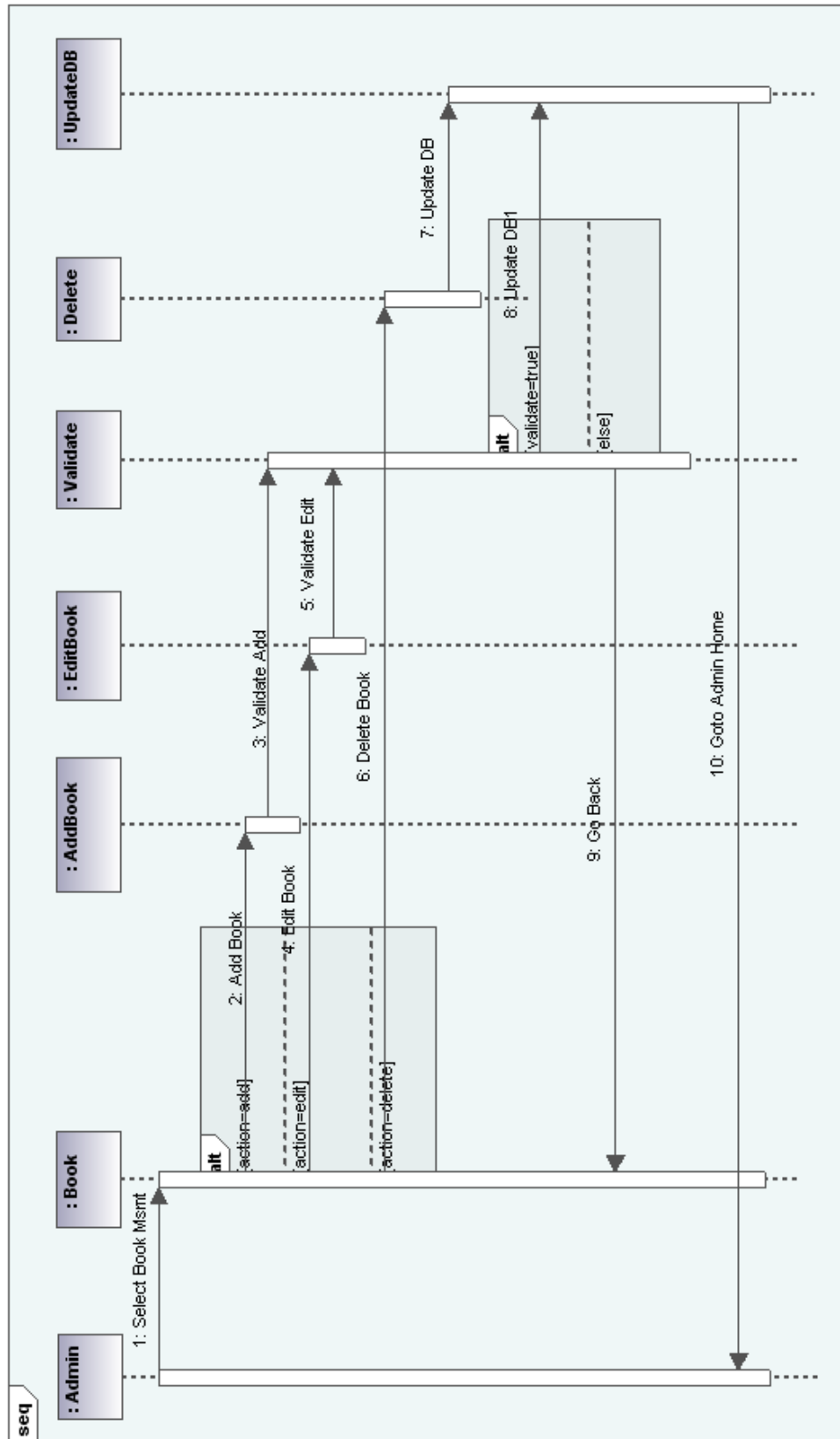


Figure 5.16 Book Management (Library)

5.5 Activity Diagram

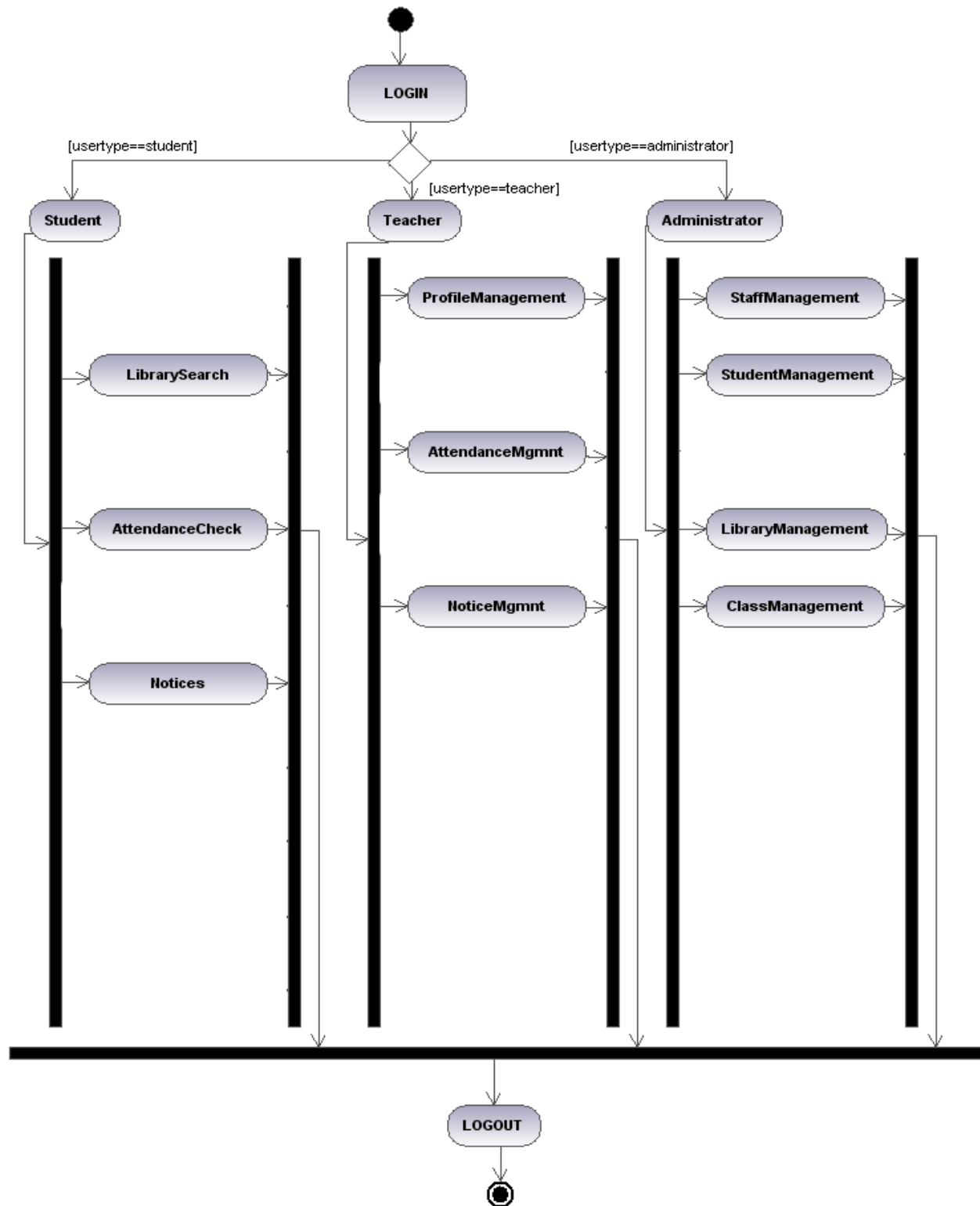


Figure 5.17 Activity Diagram

5.6 Package Diagram

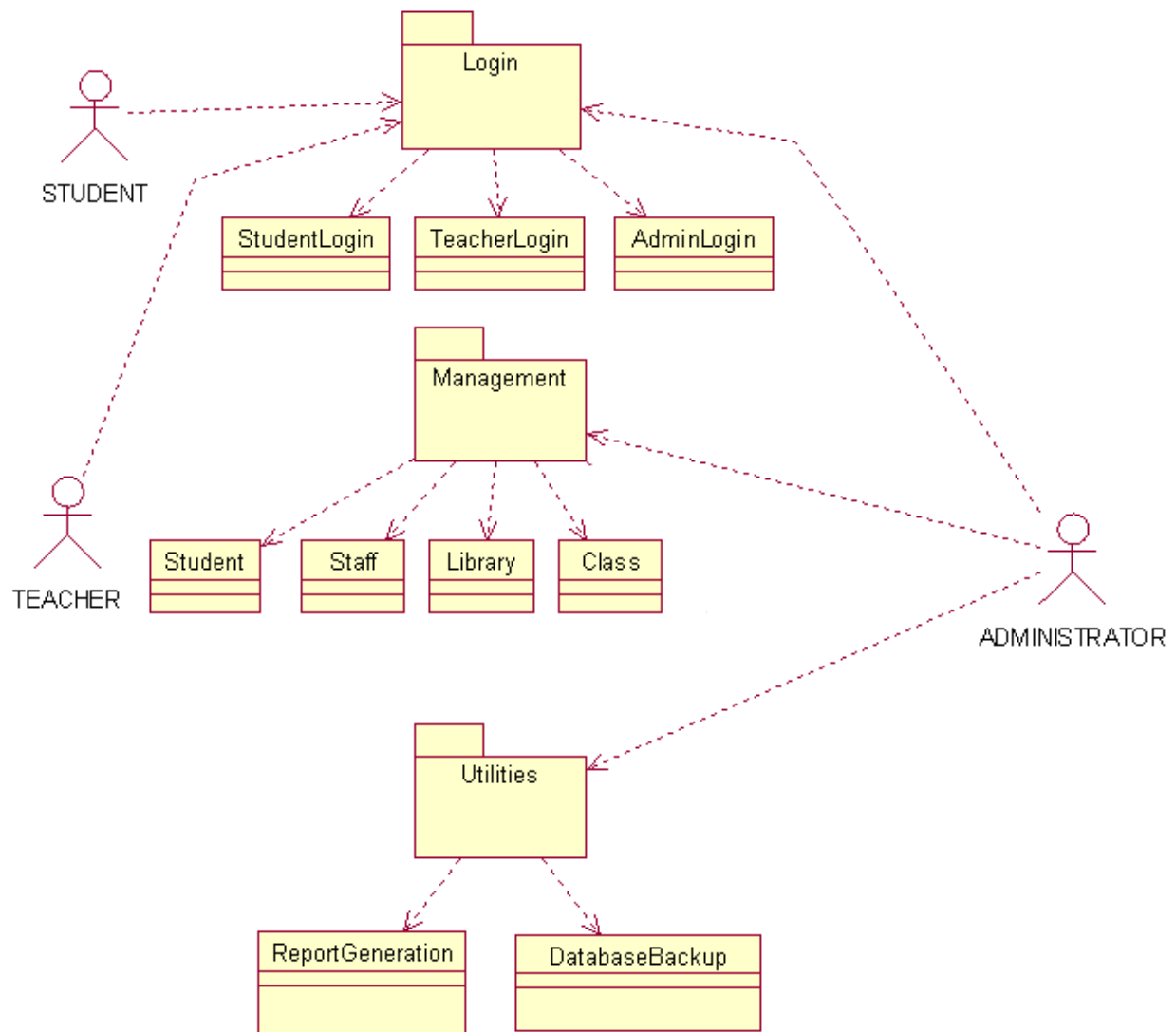


Figure 5.18 Package Diagram

5.7 Component Diagram

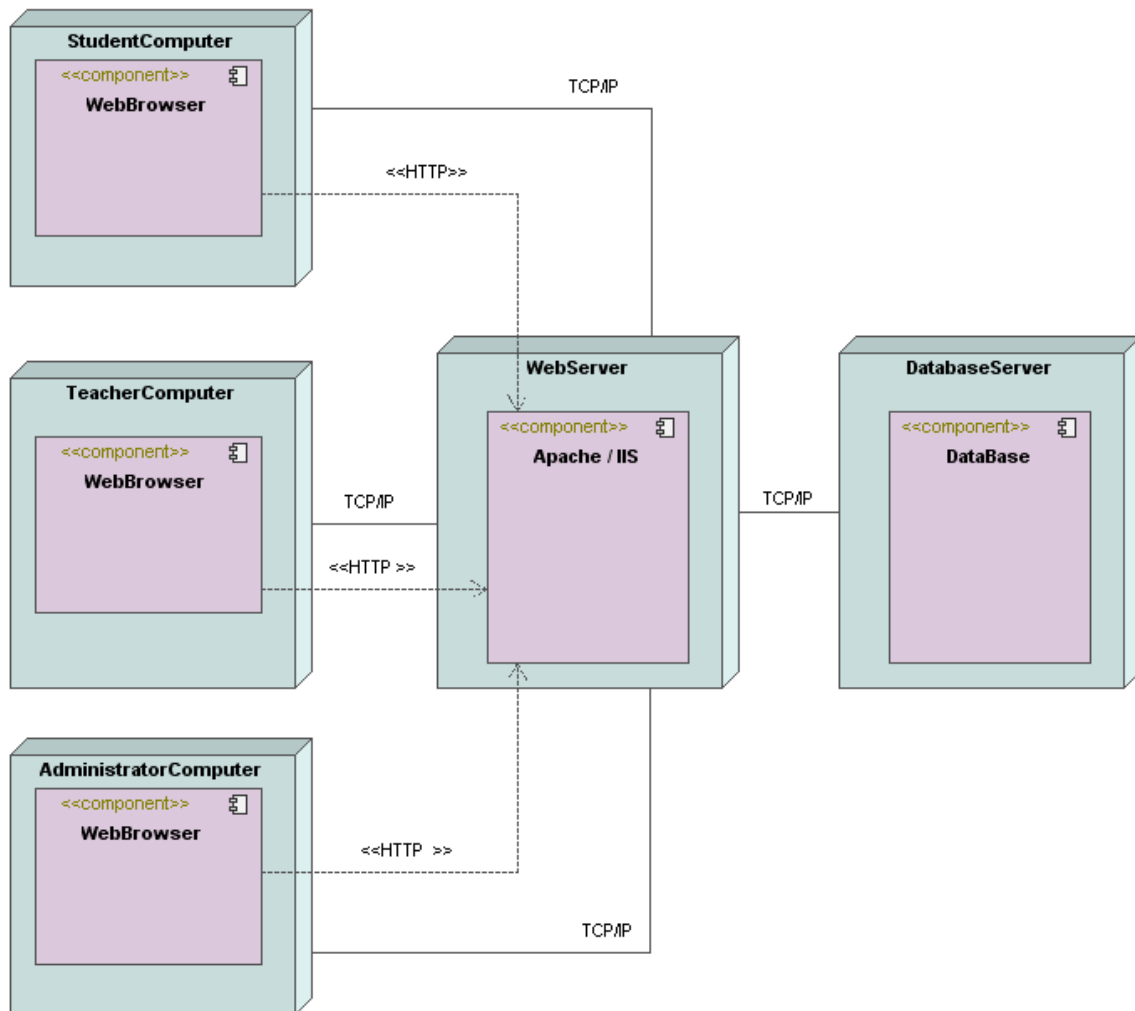


Figure 5.19 Component Diagram

5.8 Database Design

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
addr1	varchar (255)	
addr2	varchar (255)	
city	varchar (255)	
zip	varchar (15)	
state	int (11)	Foreign Key References list_state (id)
country	int (11)	Foreign Key References list_country (id)
phno	varchar (15)	
mobno	varchar (15)	

Table 5.1 Database address

Field	Type	Key
id	int(11)	Primary Key (Auto Increment)
class_id	int(11)	Foreign Key References class(id)
start_date	int(11)	
end_date	int(11)	

Table 5.2 Database attendance

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
attendance_id	int (11)	Foreign Key References attendance (id)
subject_id	int (11)	Foreign Key References subjects (id)
total_lect	int (11)	

Table 5.3 Database attendance_total

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
name	varchar (255)	
sdate	int (11)	
edate	int (11)	

Table 5.4 Database batch

Field	Type	Key
branch_id	int (11)	Primary Key (auto-increment)
branch_name	varchar (255)	

Table 5.5 Database branch

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
batch_id	int (11)	Foreign Key References brach (id)
name	varchar (255)	
teacher_id	int (11)	Foreign Key References teacher_info (id)
branch_id	int (11)	Foreign Key References branch (id)
level_id	int (11)	Foreign Key References level(id)

Table 5.6 Database class

Field	Type	Key
class_id	int (11)	Foreign Key References class (id)
student_id	int (11)	Foreign Key References student_info (id)

Table 5.7 Database class_students

Field	Type	Key
param	varchar (255)	
value	varchar (255)	

Table 5.8 Database config

Field	Type	Key
format	text	

Table 5.9 Database letter_format

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
name	varchar (255)	

Table 5.10 Database level

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
name	varchar (255)	
desc	text	
username	varchar (255)	
password	varchar (255)	

Table 5.11 Database library

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
library_id	int (11)	Foreign Key References library (id)
title	varchar (255)	
author	varchar (255)	
subject	int (11)	Foreign Key References library_subject (id)
keywords	varchar (255)	
publisher	varchar (255)	
isbn	varchar (15)	
cover_type	int (11)	
price	double	

Table 5.12 Database library_book

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
library_id	int (11)	Foreign Key References library (id)
borrower_type	tinyint(4)	
borrower_id	int (11)	Foreign Key References
book_id	int (11)	Foreign Key References library_book (id)
doi	int (11)	
dor	int (11)	

Table 5.13 Database library_book_issue

Field	Type	Key
id	int (11)	Foreign Key library (id)
student_days	int (11)	
teacher_days	int (11)	
student_fine	double	
teacher_fine	double	
student_books	int (11)	
teacher_books	int (11)	

Table 5.14 Database library_config

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
library_id	int (11)	Foreign Key References library (id)
name	varchar (255)	

Table 5.15 Database library_subject

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
name	varchar (255)	

Table 5.16 Database list_country

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
name	varchar (255)	

Table 5.17 Database list_state

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
prn	varchar(20)	
doj	int (11)	
first_name	varchar (100)	
middle_name	varchar (100)	
last_name	varchar (100)	
father_name	varchar (100)	
mother_name	varchar (100)	
dob	int (11)	
blood_group	smallint (6)	
nationality	varchar (255)	
email	varchar (255)	
perc_10 th	double	
perc_12 th	double	
localaddr_id	int (11)	Foreign Key References address (id)
permaddr_id	int (11)	Foreign Key References address (id)
lgaddr_id	int (11)	Foreign Key References address (id)
branch_id	int (11)	Foreign Key References branch (id)
first_login	tinyint (4)	
password	varchar (255)	
added_on	int (11)	
updated_on	int (11)	

Table 5.18 Database student_info

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
batch_id	int (11)	Foreign Key References batch (id)
branch_id	int (11)	Foreign Key References branch (id)
level_id	int (11)	Foreign Key References level (id)
name	varchar (255)	
type	tinyint (4)	

Table 5.19 Database subjects

Field	Type	Key
attendance_id	int (11)	Foreign Key References attendance (id)
subject_id	int (11)	Foreign Key References subjects (id)
student_id	int (11)	Foreign Key References student_id (id)
attended	int (11)	

Table 5.20 Database subject_attendance

Field	Type	Key
subject_id	int (11)	Foreign Key References subjects (id)
class_id	int (11)	Foreign Key References class (id)
teacher_id	int (11)	Foreign Key References teacher_info (id)

Table 5.21 Database subject_teachers

Field	Type	Key
id	int (11)	Primary Key (auto-increment)
username	varchar(20)	
doj	int (11)	
first_name	varchar (100)	
middle_name	varchar (100)	
last_name	varchar (100)	
father_name	varchar (100)	
mother_name	varchar (100)	
dob	int (11)	
blood_group	smallint (6)	
nationality	varchar (255)	
email	varchar (255)	
teach_exp	double	
work_exp	double	
addr_id	int (11)	Foreign Key References address (id)
branch_id	int (11)	Foreign Key References branch (id)
first_login	tinyint (4)	
password	varchar (255)	
added_on	int (11)	
updated_on	int (11)	

Table 5.22 Database teacher_info

6. CODING

6.1 Code Specification

Softwares Used:

1. Platform: PHP
2. Front End: HTML WebPages.
3. Back End: MySQL

6.2 Coding Style Followed

The rules here are the conventions used by PHP developers.

We often name variables and classes using short phrases. In PHP, the convention is to have variable names where the letters are all lowercase, and words are separated by underscores or by capitalized first letter of the word.

Classes and modules are named differently: there are no underscores, and each word in the phrase is capitalized.

6.3 Code layout

Including braces

Braces should always be included when writing code using if, for, while etc. blocks. There are no exceptions to this rule, even if the braces could be omitted. Leaving out braces makes code harder to maintain in the future and can also cause bugs that are very difficult to track down.

Braces should always be placed on a line on their own; again there are no exceptions to this rule. Braces should also align properly (use tabs to achieve this) so a closing brace is always in the same column as the corresponding opening brace.

Quoting strings

Strings in PHP can either be quoted with single quotes (") or double quotes ("). The difference between the two is that the parser will use variable-interpolation in double-quoted strings, but not with single-quoted strings. So if your string contains no variables, use single quotes and save the parser the trouble of attempting to interpolate the string for variables,

Use constants where possible

If a value is not going to change throughout the execution of your script, then use a constant rather than a variable to define it. That way, if you do change the value by accident, the PHP parser will output an error and allow you to fix the problem, without it causing any unforeseen side effects.

Remember that constants should never be enclosed in strings, either single or double. You must always use concatenation if you wish to include a constant's value in a string.

7. Testing

7.1 Test Specification

Introduction:

Software testing is important element of software quality assurance and represents ultimate review of specification, design and coding. A successful test is one that uncovers as yet undiscovered error. Testing demonstrates that software functions appear to be working according to specification that performance requirements appear to have been met. Also data collected as testing is conducted provides a good indication of software reliability and quality. It is general principle of testing that all tests should be traceable to customer requirements. Also tests should be planned long before testing begins.

Testing Plan:

Once code has been generated, program unit testing begins. The testing process focuses on the logical internals of the software, ensuring that all statements have been tested, and on the functional externals; that is, conducting tests to uncover errors and ensure that defined input will produce actual results that agree with required results.

After completing code generation, the system integration testing is done. It checks the complete behavior of the system, a data flow between each unit in the system.

Unit Test Documents (UTD) and System Test Documents (STD) are developed for both unit testing and system integrity respectively.

Unit Testing

Unit testing focuses on verification of the smallest unit of software design i.e. the software component or module. The unit test is white box oriented and can be conducted in parallel for multiple components.

White Box Testing

In this individual programs and screens were put to test using the most erroneous data possible. This went on to ensure the validity of each field in the screen and affirmed the correctness of the data which would reside in the database.

Alpha Testing: This test is conducted at the developer's site by customer. The software is used in a natural setting with the developer looking over the shoulder of the user and recording errors and usage problems.

Beta Testing: This test is conducted by one or more customer sites by end user of the software. This test is a live application of the software in an environment that cannot be controlled by the developer.

Integration Testing

Having gone through the prior test with satisfaction, the whole system was scrutinized for proper symbols of the user's module and individual screens. All the modules were integrated and then tested.

Validation testing

Software validation is achieved through a series of black box tests that demonstrate conformity with the requirements. Validation succeeds when the software functions in a manner expected by the customer.

System testing

In System Testing software and other elements are tested as a whole. This test checks whether the entire system works in accordance with the customers requirements.

There are four types of system testing

1. Recovery testing
2. Security Testing
3. Stress Testing
4. Performance Testing

7.2 Test Scripts

Test for Invalid Login

```
<?php
    if (! defined('SIMPLE_TEST')) {
        define('SIMPLE_TEST', '../simpletest/');
    }
    require_once(SIMPLE_TEST . 'unit_tester.php');
    require_once(SIMPLE_TEST . 'web_tester.php');
    require_once(SIMPLE_TEST . 'reporter.php');

    class SimpleFormTests extends WebTestCase
    {

        function testDefaultValue()
        {
            $this->post(
                'http://isolution/login.php',
                array('user_type' => 'admin', 'username' => "admin", 'password1' =>
'check1'));
            $this->assertTitle('i-Solution - Login');
            $this->assertText('Please enter the correct password.');
```

```
        }
    }

    $test = &new SimpleFormTests();
    $test->run(new HtmlReporter());
?>
```

Test for Invalid User Type

```
<?php
    if (! defined('SIMPLE_TEST')) {
        define('SIMPLE_TEST', '../simpletest/');
    }
    require_once(SIMPLE_TEST . 'unit_tester.php');
    require_once(SIMPLE_TEST . 'web_tester.php');
    require_once(SIMPLE_TEST . 'reporter.php');
    class SimpleFormTests extends WebTestCase
    { function testDefaultValue()
        {
            $this->post(
                'http://isolution/login.php',
                array('user_type' => 'admin1', 'username' => "admin", 'password1' =>
'check1'));
            $this->assertTitle('i-Solution - Login');
```

```
        }
    }
}
```

```

    $test = &new SimpleFormTests();
    $test->run(new HtmlReporter());
?>

```

Test for Unsuccessful Login of Teachers and Students when Batch Not Set.

```

<?php
    if (! defined('SIMPLE_TEST')) {
        define('SIMPLE_TEST', '../simpletest/');
    }
    require_once(SIMPLE_TEST . 'unit_tester.php');
    require_once(SIMPLE_TEST . 'web_tester.php');
    require_once(SIMPLE_TEST . 'reporter.php');
    class SimpleFormTests extends WebTestCase
    { function testDefaultValue()
      {
          $this->post(
              'http://isolution/login.php',
              array('user_type' => 'teacher', 'username' => "teacher5", 'password1'
=> 'check'));
          $this->assertTitle('i-Solution - Login');
          $this->post(
              'http://isolution/login.php',
              array('user_type' => 'student', 'username' => "3070666F", 'password1'
=> 'check'));
          $this->assertTitle('i-Solution - Login');

          }
      }
    $test = &new SimpleFormTests();
    $test->run(new HtmlReporter());
?>

```

Test for Subject Teacher Requesting for Attendance

```

<?php
    if (! defined('SIMPLE_TEST')) {
        define('SIMPLE_TEST', '../simpletest/');
    }
    require_once(SIMPLE_TEST . 'unit_tester.php');
    require_once(SIMPLE_TEST . 'web_tester.php');
    require_once(SIMPLE_TEST . 'reporter.php');
    class SimpleFormTests extends WebTestCase
    { function testDefaultValue()
      {
          $this->post(
              'http://isolution/login.php',
              array('user_type' => 'teacher', 'username' => "teacher6", 'password1'
=> 'check'));
          $this->assertTitle('i-Solution - Index');
          $this->clickLink('Reports and Notices');
          $this->assertText('Only a class teacher may request for attendance.');
```

```

    $test = &new SimpleFormTests();
    $test->run(new HtmlReporter());
?>

```

Test for Class Teacher Requesting for Attendance

```

<?php
    if (! defined('SIMPLE_TEST')) {
        define('SIMPLE_TEST', '../simpletest/');
    }
    require_once(SIMPLE_TEST . 'unit_tester.php');
    require_once(SIMPLE_TEST . 'web_tester.php');
    require_once(SIMPLE_TEST . 'reporter.php');
    class SimpleFormTests extends WebTestCase
    { function testDefaultValue()
      {
          $this->post(
              'http://isolution/login.php',
              array('user_type' => 'teacher', 'username' => "teacher5", 'password1'
=> 'check'));
          $this->assertTitle('i-Solution - Index');
          $this->clickLink('Reports and Notices');
          $this->assertText('Print Notices');
      }
    }
    $test = &new SimpleFormTests();
    $test->run(new HtmlReporter());
?>

```

Test for a case where Active Batch is not Set.

```

<?php
    if (! defined('SIMPLE_TEST')) {
        define('SIMPLE_TEST', '../simpletest/');
    }
    require_once(SIMPLE_TEST . 'unit_tester.php');
    require_once(SIMPLE_TEST . 'web_tester.php');
    require_once(SIMPLE_TEST . 'reporter.php');
    class SimpleFormTests extends WebTestCase
    { function testDefaultValue()
      {
          $this->post(
              'http://isolution/login.php',
              array('user_type' => 'admin', 'username' => "admin", 'password1' =>
'check'));
          $this->assertTitle('i-Solution - Index');
          $this->clickLinkById('setactive');
          $this->assertText('No Active Batch set.');
```

Test for Setting an Active Batch

```
<?php
    if (! defined('SIMPLE_TEST')) {
        define('SIMPLE_TEST', '../simpletest/');
    }
    require_once(SIMPLE_TEST . 'unit_tester.php');
    require_once(SIMPLE_TEST . 'web_tester.php');
    require_once(SIMPLE_TEST . 'reporter.php');
    class SimpleFormTests extends WebTestCase
    { function testDefaultValue()
      {
          $this->post(
              'http://isolution/login.php',
              array('user_type' => 'admin', 'username' => "admin", 'password1' =>
'check'));
          $this->assertTitle('i-Solution - Index');
          $this->clickLinkById('setactive');
          $this->assertText('No Active Batch set. ');
          $this->clickLink('Click Here. ');
          $this->assertNoText('No Active Batch set. ');
      }
    }
    $test = &new SimpleFormTests();
    $test->run(new HtmlReporter());
?>
```

All the above tests passed.

8. PERFORMANCE MEASUREMENTS

When we talk about system needing good performance, we're mainly talking about raw speed (the ability of the virtual machine or interpreter to churn through statements in our code) and scalability (the ability to scale our system by throwing in more hardware). A further factor to take into account with web applications is the platforms' support for effective caching between requests.

8.1 Raw speed

Raw speed may actually not matter all that much for a significant portion of enterprise applications. In fact for most enterprise developers a far more important non-functional requirement is that of scalability rather than raw speed.

8.2 Scalability

Scalability is also a somewhat ambiguous word. Scalability comes in two flavors, horizontal and vertical scalability. Horizontal scalability means scaling "out" by adding more boxes next to the existing ones. Vertical scalability means scaling "up" by adding more memory, more CPUs, faster disks, and so forth, into the existing boxes. Scaling up is really somewhat trivial with PHP. Being based on multi-process execution through FastCGI on top of web servers like Apache and lighttpd, PHP is well adept to making use of added memory, added CPU speed, and faster I/O. PHP scales out quite nicely up to a point. PHP delegates clustering, fail-over, load balancing, and so forth, to the web server infrastructure and thus supports whatever Apache, for example, supports in terms of high availability and high throughput properties. Generally speaking, a multi-threaded architecture like what we have in J2EE has a more scalable approach. In practice, however, these theoretical limits are far from the reality for a significant number of what we call "enterprise applications."

8.3 Security

MySQL uses security based on Access Control Lists (ACLs) for all connections, queries, and other operations that users can attempt to perform. There is also support for SSL-encrypted connections between MySQL clients and servers. Many of the concepts discussed here are not specific to MySQL at all; the same general ideas apply to almost all applications.

Security can be provided to the database by implementing the following

- Do not ever give anyone (except MySQL root accounts) access to the user table in the mysql database! This is critical.
- Learn the MySQL access privilege system. The GRANT and REVOKE statements are used for controlling access to MySQL. Do not grant more privileges than necessary. Never grant privileges to all hosts.
- Do not store any plain-text passwords in your database. If your computer becomes compromised, the intruder can take the full list of passwords and use them. Instead, use MD5(), SHA1(), or some other one-way hashing function and store the hash value.
- Do not choose passwords from dictionaries. Special programs exist to break passwords. Even passwords like “xfish98” are very bad. Much better is “duag98” which contains the same word “fish” but typed one key to the left on a standard QWERTY keyboard. Another method is to use a password that is taken from the first characters of each word in a sentence (for example, “Mary had a little lamb” results in a password of “Mhall”). The password is easy to remember and type, but difficult to guess for someone who does not know the sentence.
- Invest in a firewall. This protects you from at least 50% of all types of exploits in any software. Put MySQL behind the firewall or in a demilitarized zone (DMZ).

8.4 Testing

PHP does not have any built in testing framework. We have used the simpleTest testing framework to perform the testing of our application wherever possible.

simpleTest testing framework does not support XMLHTTPPrequest and as most of our application is based on AJAX we had to revert to manual testing of the other features.

8.5 User Interface

The user interface has been made in a way that is very easy to use, easy to get used to and time saving; which in turn saves bandwidth and server utilization.

The whole design of the site is governed by a single CSS file. There are separate HTML files which have the design of the pages. The PHP code is completely separated from the HTML files, providing a level of abstraction.

9. Conclusion and Future Enhancements

9.1 Conclusion

Today, students and teachers do have a problem with information sharing regarding administration. This system will help the teachers, students and the administrative staff to share, store data easily and generate extensive reports.

9.2 Future Enhancements

1. An SMS gateway could be used to send notices to the students and teachers.
2. Web 2.0 can be implemented by adding a module for the forums.
3. Placement Management which is an exhausting task in the institutions can be simplified if there was a better information management system.
4. A module for committee management could be setup.
5. An internal messaging system within the application.

10. Screenshots

i-Solution

Login

Login Error

- Please enter the username and the password.

User Type :

Username :

Password :

If you are logging in for the first time, please enter the password as your DOB in this format : **DDMMYYYY**

Figure 10.1 Login Screen

i-Solution [Administrator]

Admin Home | Batch | Branch | Class | Attendance | Staff | Student | Library | Logout

Edit Branch

Batch Description :

Start Date : / /

End Date : / /

List Batch

S.No.	Name	Actions
1	2006/2007 Semester I	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
2	2006/2007 Semester II	<input type="button" value="Edit"/> <input type="button" value="Delete"/>

Figure 10.2 Batch Management

The screenshot shows the 'i-Solution [Administrator]' interface. At the top, there is a navigation menu with the following items: Admin Home, Batch, Branch, Class, Attendance, Staff, Student, Library, and Logout. Below the menu, the 'List Batch' section is displayed. It features a green notification box titled 'Current Active Batch' with a bullet point stating 'Current Active Batch is 2006/2007 Semester II.' Below the notification is a table with the following data:

S.No.	Name	Actions
1	2006/2007 Semester I	<input type="button" value="Activate"/>
2	2006/2007 Semester II	Active Batch

Figure 10.3 Set Active Batch

i-Solution [Administrator]

Admin Home | Batch | Branch | Class | Attendance | Staff | Student | Library | Logout

Edit Branch X

Branch Name :

List Branch

S.No.	Name	Actions	
1	Chatting	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
2	Civil	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
3	Computer	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
4	Electronics and Telecommunication	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
5	Information Technology	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
6	Mechanical	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
7	Mining	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
8	Petrochemical	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
9	Petroleum	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>

Figure 10.4 Branch Management

i-Solution [Administrator]

Admin Home | Batch | Branch | Class | Attendance | Staff | Student | Library | Logout

Edit Level X

Level Name :

List Level

S.No.	Name	Actions
1	1st Year	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
2	2nd Year	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
3	3rd Year	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
4	4th Year	<input type="button" value="Edit"/> <input type="button" value="Delete"/>

Figure 10.5 Level Management

i-Solution [Administrator]

[Admin Home](#) | [Batch](#) | [Branch](#) | [Class](#) | [Attendance](#) | [Staff](#) | [Student](#) | [Library](#) | [Logout](#)

List Classes

[Add New Class](#)

Batch :

Branch :

Level :

[List Classes](#)

S.No.	Name	Batch	Branch - Level	Teacher Name	Actions
1	Division I	2006/2007 Semester II	Computer - 4th Year	TFirst1 TMiddle1 TLast1	Edit
2	Division II	2006/2007 Semester II	Computer - 4th Year	TFirst2 TMiddle2 TLast2	Edit
3	Division I	2006/2007 Semester II	Electronics and Telecommunication - 4th Year	TFirst5 TMiddle5 TLast5	Edit

Figure 10.6 Class Management

i-Solution [Administrator]

Admin Home Batch Branch Class Attendance Staff Student Library Logout

Add Subjects X

Select Batch :

Select Level :

Select Branch :

Subject Name :

Subject Type :

List Subjects

S.No.	Branch - Level	Subject Name
No Subjects listed yet.		

Figure 10.7 Subject Management

i-Solution [Administrator]

Admin Home
Batch
Branch
Class
Attendance
Staff
Student
Library
Logout

List Subjects

2006/2007 Semester II ▾

S.No.	Branch - Level	Class	Subject Name	Subject Teacher
1	Computer - 4th Year	Division I	ACA (Theory)	TFirst25 TMiddle25 TLast25 ▾
2	Computer - 4th Year	Division II	ACA (Theory)	TFirst2 TMiddle2 TLast2 ▾
3	Computer - 4th Year	Division I	Distributed Systems (Theory)	TFirst25 TMiddle25 TLast25 ▾
4	Computer - 4th Year	Division II	Distributed Systems (Theory)	TFirst26 TMiddle26 TLast26 ▾
5	Computer - 4th Year	Division I	DS (Practical)	TFirst27 TMiddle27 TLast27 ▾
6	Computer - 4th Year	Division II	DS (Practical)	TFirst28 TMiddle28 TLast28 ▾
7	Computer - 4th Year	Division I	Network Information and Security (Theory)	TFirst29 TMiddle29 TLast29 ▾

Figure 10.8 Assign Subject Teachers

i-Solution [Administrator]

[Admin Home](#) | [Batch](#) | [Branch](#) | [Class](#) | [Attendance](#) | [Staff](#) | [Student](#) | [Library](#) | [Logout](#)

Set Letter Format

Variable List

1. *TODAY_DATE* - Current Date
2. *STUDENT_NAME* - Student's Name
3. *ADDRESS* - Student's Permanent Address
4. *START_DATE* - Starting date of the period of attendance
5. *END_DATE* - Ending date of the period of attendance
6. *ATTENDANCE* - Attendance Percentage of the Student
7. *FATHER_NAME* - Attendance Percentage of the Student

Letter Format :

```

<!--
.style1 {
    font-family: Verdana, Arial, Helvetica, sans-serif;
    font-size: 12px;
}
-->
</style>
<p class="style1">Date : *TODAY_DATE* </p>
<p class="style1">To,<br />
    *FATHER_NAME*,<br />
    *ADDRESS*</p>

```

Figure 10.9 Letter Format

View Reports

Name	Batch	Branch - Level	Teacher Name
Division I	2006/2007 Semester II	Electronics and Telecommunication - 4th Year	TFirst5 TMiddle5 TLast5

S.No.	From	To	Reports
1	8 Apr, 2007	23 Apr, 2007	View Report Print Notices
2	5 Apr, 2007	20 Apr, 2007	View Report Print Notices
3	2 Apr, 2007	17 Apr, 2007	View Report Print Notices

Attendance Records From 2 Apr, 2007 to 17 Apr, 2007

S.No.	Name	ACA - Theory (19)	ACA - Practical (3)	NS - Theory (20)	NS - Practical (5)	STQA - Theory (22)	STQA - Practical (4)	Total (61)
1.	First104 Middle104 Last104	21.05% (4)	0	75% (15)	4	63.64% (14)	2	54.1% (33)
2.	First105 Middle105 Last105	26.32% (5)	0	5% (1)	5	45.45% (10)	1	26.23% (16)
3.	First106 Middle106 Last106	42.11% (8)	0	20% (4)	0	27.27% (6)	1	29.51% (18)
4.	First107 Middle107 Last107	36.84% (7)	0	40% (8)	0	9.09% (2)	4	27.87% (17)
5.	First108 Middle108 Last108	31.58% (6)	0	0% (0)	0	0% (0)	0	9.84% (6)
6.	First109 Middle109 Last109	26.32% (5)	3	30% (6)	3	0% (0)	0	18.03% (11)
7.	First117 Middle117 Last117	100% (19)	2	45% (9)	0	9.09% (2)	0	49.18% (30)
8.	First118 Middle118 Last118	94.74% (18)	1	20% (4)	0	36.36% (8)	0	49.18% (30)
9.	First144 Middle144 Last144	5.26% (1)	0	45% (9)	2	0% (0)	4	16.39% (10)
10.	First145 Middle145 Last145	0% (0)	2	55% (11)	1	0% (0)	4	18.03% (11)
11.	First198 Middle198 Last198	10.53% (2)	2	60% (12)	0	0% (0)	4	22.95% (14)

Figure 10.10 Attendance Report

i-Solution [Main Library]

Library Home | Accession Register | Circulation | **Configure Rules** | Logout

Library Options

No. of Days Allotted for Students :

No. of Days Allotted for Teachers :

Fine for Students per day :

Fine for Teachers per day :

Max. No. of Books Allotted for Students at a time :

Max. No. of Books Allotted for Teachers at a time :

Figure 10.11 Configure Library Rules

i-Solution [Main Library]

Library Home | Accession Register | Circulation | **Configure Rules** | Logout

Edit Book

Title :

Author :

Subject : ▼

Keywords :

Publisher :

ISBN :

Cover Type : ▼

Price :

Figure 10.12 Book Management

i-Solution [Main Library]

Library Home | Accession Register | Circulation | Configure Rules | Logout

Issue Book

Borrower :

teacher3 - TFirst3 TMiddle3 TLast3
 Select Teacher : [Search Teacher](#)

Teacher Name :

S.No.	Name	Books Borrowed	Actions
1	teacher10 - TFirst10 TMiddle10 TLast10	0 / 1	<input type="button" value="Select"/>

Select Book : Electronic Measurement Systems by U. A. Bakshi
[Search Book](#)

Figure 10.13 Issue Book

i-Solution [Main Library]

Library Home | Accession Register | Circulation | Configure Rules | Logout

Return Book

Borrower :

Borrower: teacher10 - TFirst10 TMiddle10 TLast10
 Borrower and Book Informatio [\[Search Teacher\]](#) : Book : Electronic Measurement Systems, U. A. Bakshi
 DOI : 26 Apr, 2007

Teacher Username :

Teacher Name :

S.No.	Name	Book Borrowed	Date of Issue	Actions
1	teacher10 - TFirst10 TMiddle10 TLast10	Electronic Measurement Systems by U. A. Bakshi - Electronic Measurement Systems	26 Apr, 2007	<input type="button" value="Select"/>

Figure 10.14 Return Book

i-Solution [Administrator]

Admin HomeBatchBranchClassAttendanceStaffStudentLibraryLogout

Student List

Name	Batch	Branch - Level	Teacher Name
Division I	2006/2007 Semester II	Computer - 4th Year	TFirst1 TMiddle1 TLast1

S.No.	Name	Actions
No results meeting your search criteria.		

Search Students

Permanent Registration Number :

Name :

Joined Between: / / and / /

Figure 10.15 Student – Class Management

References

- Guide to PHP Design Patterns. PHP|architect
- PHP and MySQL for Dynamic Web Sites, Peachpit Press
- PHP Website (<http://www.php.net>)
- Google Groups - comp.lang.php
- MySQL database website
- Planet MySQL - an aggregation of MySQL-related blogs
- MySQL Dev Forum @ MySQL Database Website
- xajax Homepage
- xajax Community Forum
- PHP Web Forums
- AJAX Forums
- xajax Wiki (<http://wiki.xajaxproject.org/>)