Project Report

On

"INSTITUTION AUTOMATION SYSTEM" VERSION 1.0 (SCHOOL)

Submitted for partial fulfillment of the degree of

BACHELOR OF ENGINEERING

(Computer Technology)

By:

Amit Prakash
Gaurangi Wanjari
Kawaljot Singh Bagga

B.E. 6th Semester
Department of Computer Technology

Under the Guidance of

Mrs. Manali Kshirsagar	**Mrs. Asmita Mulay**
Head of Department	**Technical Trainer**
Y.C.C.E, Nagpur	**HCL CDC, Nagpur**

Department of Computer Technology,
Yeshwantrao Chavan College of Engineering, Nagpur
2009-2010

<u>CERTIFICATE</u>

Institution Automation System is a bona-fide work and is submitted to the Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur.

By:

Amit Prakash
Gaurangi Wanjari
Kawaljot Singh Bagga

In the partial fulfillment of the degree of BACHELOR OF ENGINEERING in Computer Technology, during the academic year 2009-2010 under my guidance.

Mrs. Manali Kshirsagar
Department of Computer Technology
Yeshwantrao Chavan College of Engineering,
Nagpur.

Department of Computer Technology,
Yeshwantrao Chavan College of Engineering, Nagpur
Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur.
2009-2010

CERTIFICATE

*This is to certify that below mentioned candidates have worked on a Software project in HCL Infosystems Ltd. They are Students of esteemed "Yeshwantrao Chavan College of Engineering". They worked on "**INSTITUTION AUTOMATION SYSTEM**" project.*

Student Details:

1) Mr. Amit Prakash
2) Mr. Kawaljot Singh Bagga
3) Ms. Gaurangi Wanjari
4) Ms. Kirti

Project Details:

Project name: Institution Automation System
Duration: 6 months
Team size: 4
Start date: 10-July-2009
Software requirements:
Microsoft Windows XP or above/Linux, Java, Apache Tomcat 6.0

Ms. Radha Deshpande,
Centre Technical Head
HCL CDC, Nagpur

Mrs. Asmita Mulay,
Project Guide
HCL CDC, Nagpur

HCL INFOSYSTEMS LTD.
(HCL Career Development Centre)
4th Floor, J.B.Wing,
Mangalwari Complex, Sadar, Nagpur-440010.
Phone: 6630306, 6630307, 6630308.
E-mail: nagpur@hchcdc.in website: www.hclcdc.in

ACKNOWLEDGEMENT

A project work is a test of many qualities in a person, the ability to form goals, stick to a tight schedule, crisis management and overall ability to be a team player. As young students of engineering, who have not yet faced the grind of the industry, we have learnt many things, both the easy and the hard way from a host of people.

We are thankful to **Ms. Radha Deshpande** who gave us an opportunity to explore the industrial experience and help us develop our project with constant help from HCL Career Development Centre.

Special heartfelt thanks to **Mrs. Asmita Mulay (HCL CDC)** who has always been a rock of support in times of distress and a philosopher and guide whom one could follow blindly.

We are indeed thankful to **Mrs. Manali Kshirsagar (HOD C.Tech, Y.C.C.E)** for her guidance and for that very special place he provides to us students for our various problems.

Amit Prakash **Gaurangi Wanjari**
Kawaljot Singh Bagga **Kirti**

INDEX

INTRODUCTION

ABSTRACT

This project is aimed at developing an Institution administration system which can streamline the administration of Institution. It’s often felt that the Institution management is chaotic in nature. It involves many administrative work and co-ordination. It centralizes the mountains of data to learning and automates routing administrative functions. This package has education's most flexible and interactive scheduling function, thus meeting the communication and information needs of the entire Institution community in real time. And, it would be utterly simple for everyone to use. This project is to develop and deploy a web based application so that the different aspects of an Institution administration become easy to handle.

PROBLEM DEFINITION

Develop an Institution Automation System for a school having following feature:-

1. System should have different interface for the different user and provide facilities according to user.
2. System should provide an interface to administrator to administrate the whole system like maintaining the information regarding school’s infrastructure.
3. Accountant should be able to do all possible accounts related works like collecting fees, paying salary to staff, etc.
4. Teachers would be able to maintain student information like their attendance, result etc.
5. Students would be able to see information related to them like their attendance, result, profile etc.

LITERATURE REVIEW

JAVA HISTORY

Java was conceived by **James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Nike Sheridan** at '**Sun Microsystems Inc.' in 1991.** It took 18 months to develop the first working version. This language was initially called 'Oak' but was renamed "JAVA" in 1995. Between the implementation of Oak in the fall of 1992 and the Public announcement of Java in the spring of 1995, many more people contributed to the design and the evolution of the language.

The original impetus of Java was not the Internet. Instead, the primary, motivation was the need for platform independent, machine architecture neutral, language that could be used to create software to be embedded in various consumer electronic devices, such as PDAs, Mobile, embedded systems like oven, remote controlled devices etc. As we can probably guess, many different types of CPUs are used as controllers. The problem with c and c++ languages is that they are designed to be compiled for a specific target. Although it is possible to compile a c++ program targeted for the CPU. The problem with these compilers is that they are expensive and time consuming. An easier and more cost-effective solution was needed. In an attempt to find such a solution, Gosling and others began work on a portable, platform independent language that could be used to produce code that would run on a variety of CPUs under different environment. This effort ultimately led to the creation of Java.

Java and Internet

Java expands the universe of objects that can move about freely in cyberspace. In a network, two very broad categories of objects are transmitted between the server and client computer; passive information and dynamic, active programs. For example when we read our e-mail we are viewing passive data. Even when we download the programs code is a still only passive data until we execute it. A second type of object can be transmitted to agent to client system, yet is initiated by the server. For example, a program might be provided by the server is sending.

As, desirable as dynamic, network programs are, they also present serious problems in the area of security and the portability. Prior to Java, cyberspace was effectively closed to half the entities that now live are there. As we will see, Java addresses concerns and, by doing so, has opened the doors to an exciting new form of the program, the APPLETs.

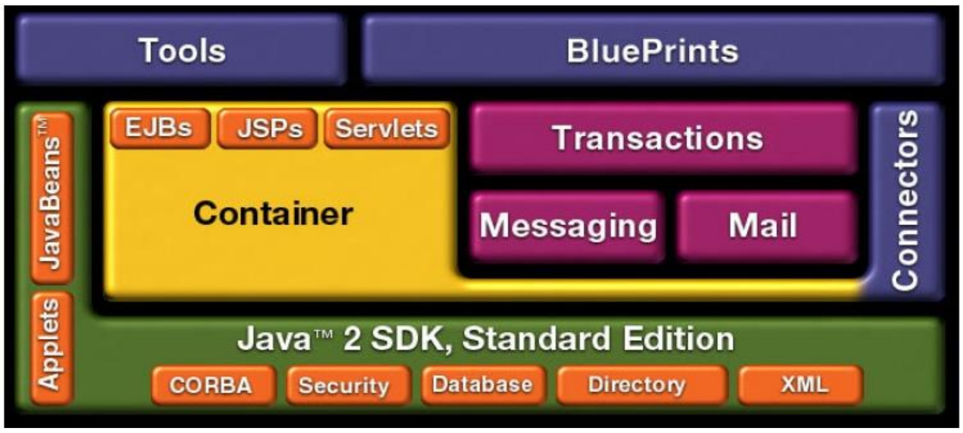
J2EE

Java 2 Enterprise Edition (J2EE) is a platform and design philosophy for large enterprise system. J2EE is marketing spin wrapping up a suite of toolkits. It comprises of three components.

- A conceptual definition for enterprise systems architecture. This definition provides a rigorous designphilosophy for building large, scalable, web-enabled systems.
- A collection of API extensions that relate to enterprise systems. These APIs range from e-mail access to database connectivity, bridging concepts as different as distributed computing and web-based applications.
- A new deployment specification for packaging Java components into a single enterprise solution. Basic Java uses a simple "Java Archive" standard for packaging a set of common class objects into a single deployable file. J2EE extends this concept to Web Archive (WAR) and Enterprise Archive (EAR) formats

J2EE ARCHITECTURE

The J2EE architecture is based on four key concepts: components, archives, containers, and Connectors.



1. Component
    - A component is an application level software unit.
    - THE J2EE platform supports the following types of components :
        - ➢ Applets
        - ➢ Application client
        - ➢ Web components
        - ➢ Enterprise Java Beans (EJBs)
2. Container
    - All J2EE components depend on runtime support of a system-level entity called a Container.
    - Containers provide components with services such as
        - ➢ Connection and object pooling
        - ➢ Security
        - ➢ Transaction
        - ➢ Deployment
3. Archives
    - An archive is a package of java code that contains one or more specific descriptors and manifests.
    - Descriptors provide configuration information environment settings, role-based security and vendor-specific information.
    - Manifests are a packing slip that is automatically generated by the archive process.
    - J2EE defines three types of archives:
        - ➢ Java Archives (JAR)
        - ➢ Web Archives (WAR)
        - ➢ Enterprise Archives(EAR)
4. Connectors
    - The connector is where the abstract really meets the concrete.
    - A connector is a translator between an enterprise information system and the J2EE interfaces.
    - Types of connectors :
        - ➢ JDBC driver.
        - ➢ JNDI service provider interface.

J2EE SUITE

J2EE (Java 2 Enterprise Edition) offers a suite of software specification to design, develop, assemble and deploy enterprise application. It provides a distributed , component-based , loosely coupled , reliable and secure ,platform independent and responsive application environment.

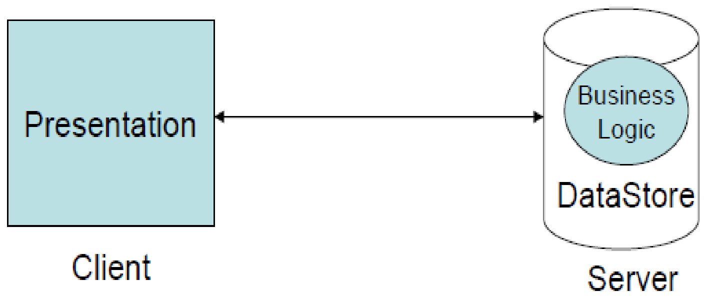
Web Technology

- Java servlets
- Java server pages
- Java server pages standard tag library

XML Technology

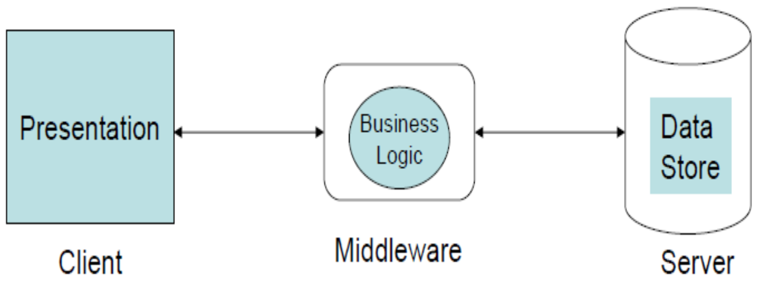
- The java API for XML processing (JAXP)
- The java API for XML-based RPC (JAX_RPC)
- The java API for XML registries (JAXR)
- SOAP with Attachments API for Java(SAAJ)

Client/Server Technology



- Business logic is maintained on the server in the form of stored procedures.
- Maintenance of business logic becomes easy.
- Since stored procedure language is still.
- Since stored procedure language is still proprietary , portability is the concern

Distributed Technology



- Business logic is maintained on the middleware.
- Client is not accessible to data directly.
- Different Distributed Technology
  - RMI, CORBA or DCOM
  - Frameworks like J2EE.

Java Servlet Technology

A servlet is a Java programming language class that is used to extend the capabilities of servers that host applications access via a request-response programming model. Although servlets can respond to any type of request, they are commonly used to extend the applications hosted by web servers. For such application, Java Servlet technology defines HTTP-specific servlets classes.

The **javax.servlet** and **javax.servlet.http** packages provide interfaces and classes for writing servlets. All servlets must implement the servlets interface, which defines life-cycle interface.

Servlet Life Cycle:

The life cycle of a servlet is controlled by the container in which the servlet has been deployed. When a request is mapped to a servlet, the container performs the following steps.

1. If an instance of the servlet does not exist, the web container
   a. Loads the servlet class.
   b. Creates an instance of the servlet class.
   c. Initializes the servlet instance by calling the init method. Initialization is covered in initializing a Servlet.

2. Invokes the service method, passing request and response objects. Service methods are discussed in Writing Service Methods.

3. If the container needs to remove the servlet, it finalizes the servlet by calling the servlet’s destroy method.

Java Server Pages (JSP)

Java Server Pages (JSP) technology allows you to easily create web content that has both static and dynamic components. JSP technology makes available all the dynamic capabilities of Java Servlet technology but provides a more natural approach to creating static content. The main features of JSP technology are as follows:

• A language for developing JSP pages, which are text-based documents that describe how to process a request and construct a response
• An expression language for accessing server-side objects
• Mechanisms for defining extensions to the JSP language JSP technology also contains an API that is used by developers of web containers, but this API is not covered in this tutorial.

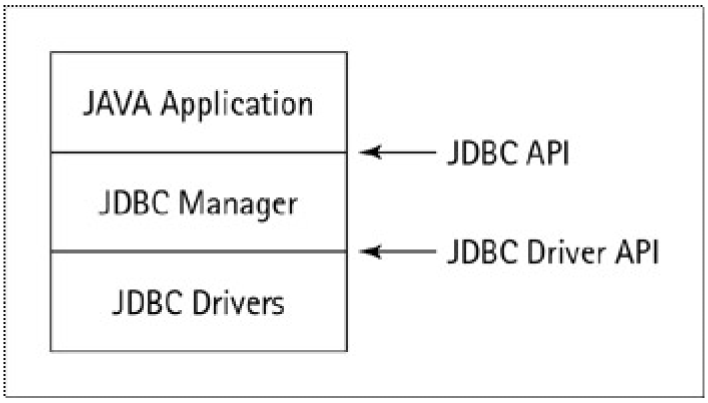
A *JSP page* is a text document that contains two types of text: static data, which can be expressed in any text-based format and JSP elements, which construct dynamic content.

Life Cycle of a JSP Page:

A JSP page services requests as a servlet. Thus, the life cycle and many of the capabilities of JSP pages are determined by Java Servlet technology. You will notice that many sections in this chapter refer to classes and methods described. When a request is mapped to a JSP page, the web container first checks whether the JSP page's servlet is older than the JSP page. If the servlet is older, the web container translates the JSP page into a servlet class and compiles the class. During development, one of the advantages of JSP pages over servlets is that the build process is performed automatically.

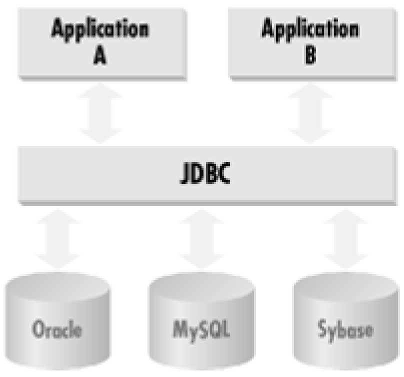
Java Database Connectivity (JDBC)

JDBC is a set of programming interface. The two major components of JDBC are the JDBC API and the JDBC Driver API. The JDBC API is a programming interface for database applications developers, where as the JDBC Driver API is a lower-level programming, interface for developers of specific drivers.



Structure of JDBC

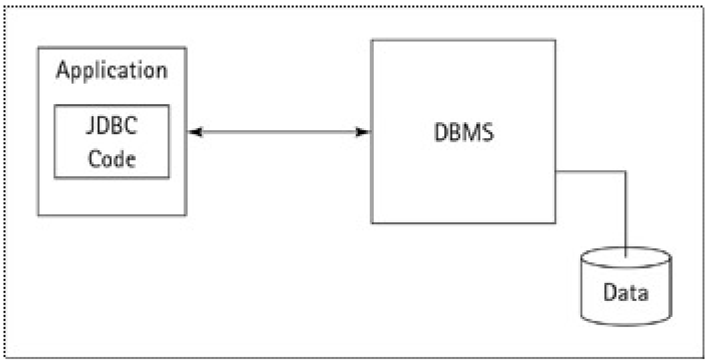
JDBC accomplishes its goals through a set of Java interfaces, each implemented differently by individual vendors. The set of classes that implement the JDBC interfaces for a particular database engine is called a JDBC driver. In building a database application, you do not have to think about the implementation of these underlying classes at all; the whole point of JDBC is to hide the specifics of each database and let you worry about just your application.



JDBC Characteristics

The JDBC characteristics are as follows:

- JDBC is a call level SQL interface for Java. This interface is totally independent of the available database management systems. It is a low-level Application Programming Interface (API) that allows a Java program to issue SQL statements and retrieve their results. JDBC also provides methods for error and warning messages management.



- Entry Level conformance is widely supported today and guarantees a wide level of portability.
- JDBC may be implemented on top of common SQL level APIs, in particular on top of ODBC.
- JDBC provides a Java interface that stays consistent with the rest of the Java system. There are no conflicts because of opposed philosophies expressed by the impedance mismatch between the object-oriented world (Java) and the tabular world (SQL).
- The JDBC mechanisms are simple to understand and use. This simplicity doesn't mean that functionality suffers.

PROJECT DESIGN

TECHNICAL SPECIFICATION

HARDWARE REQUIREMENT

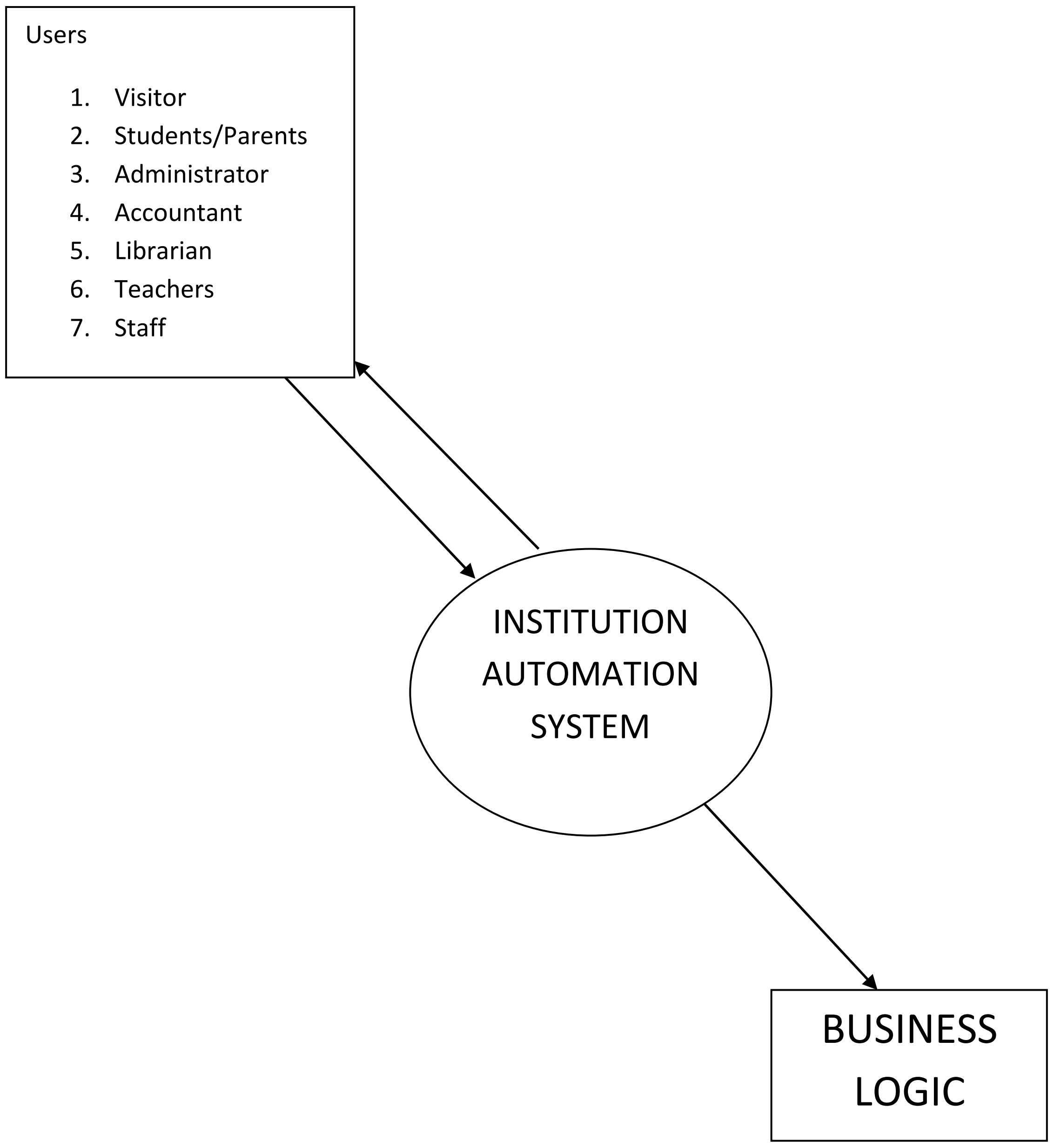
1. PIV processor
2. 128 MB RAM
3. 10 GB – HARDISK

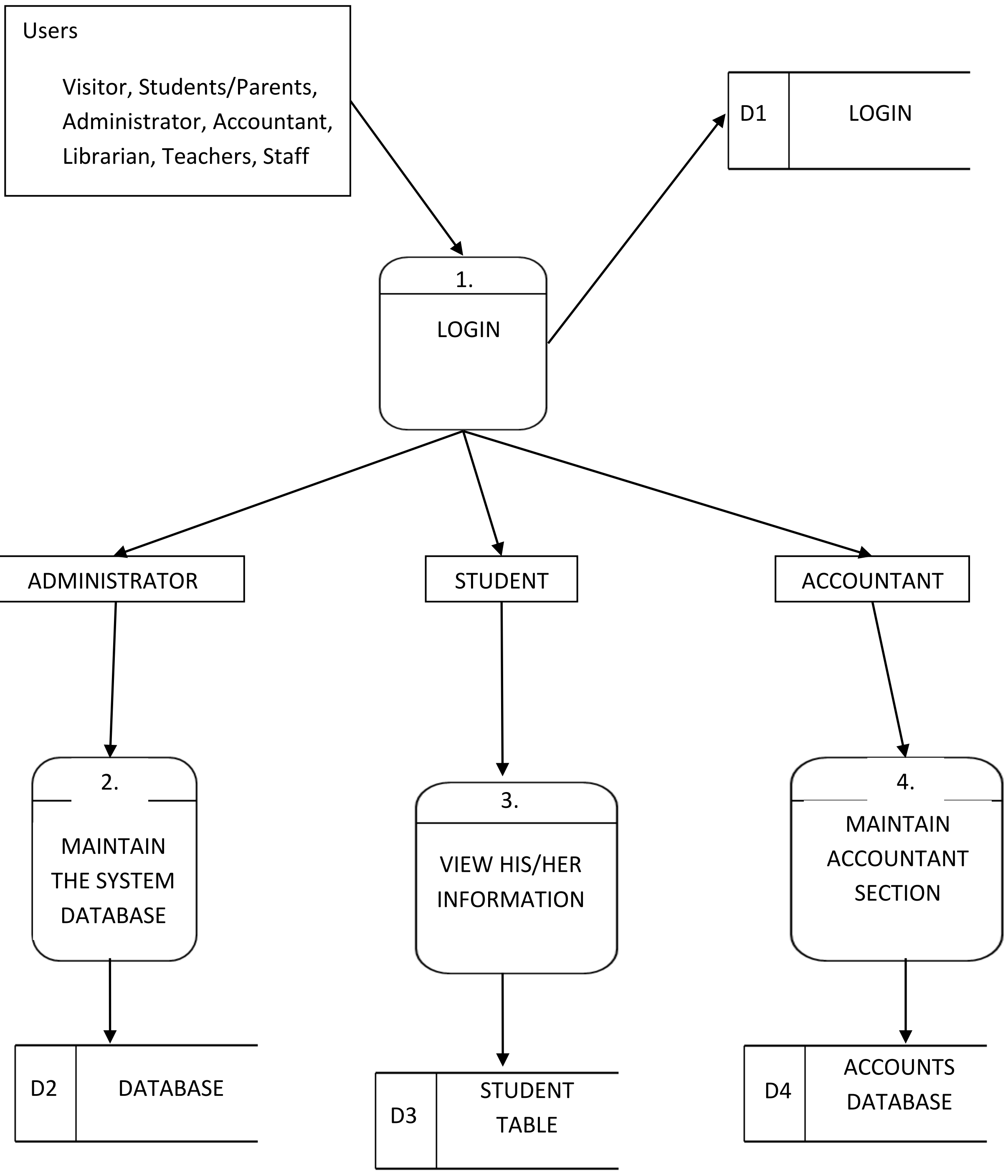
SOFTWARE REQUIREMENT

1. LINUX/WINDOWS 98 OR ABOVE
2. JAVA 6.0
3. APACHE TOMCAT 6.0
4. ORACLE 10g
5. WEB BROWSER- IE 6.0 OR ABOVE/MOZILLA FIREFOX

INSTITUTION AUTOMATION SYSTEM

- CONTEXT DIAGRAM




Users
Visitor, Students/Parents, Administrator, Accountant, Librarian, Teachers, Staff
D1
LOGIN
1.
LOGIN
ADMINISTRATOR
STUDENT
ACCOUNTANT
2.
MAINTAIN THE SYSTEM DATABASE
3.
VIEW HIS/HER INFORMATION
4.
MAINTAIN ACCOUNTANT SECTION
D2
DATABASE
D3
STUDENT TABLE
D4
ACCOUNTS DATABASE

WEB-FILES

1. about us.bmp
2. about_us.jsp
3. academics1.bmp
4. academics.jsp
5. accountant.jsp
6. activities.bmp
7. activities.jsp
8. administrator.jsp
9. ask.jsp
10. book_issue.jsp
11. book_required.jsp
12. book_return.jsp
13. book_search.jsp
14. button.JPG
15. class_teacher.jsp
16. contact.bmp
17. contact_us.jsp
18. employee_attendance.jsp
19. employee_info.jsp
20. employee_update.jsp
21. enter_class_attendance.jsp
22. enter_emp_attend.jsp
23. faculties.bmp
24. faculties.jsp
25. fee.jsp
26. formvalidation.js
27. home.bmp
28. home.jsp
29. infrastructure.bmp
30. infrastructure_home.jsp
31. librarian.jsp
32. library_fine.jsp

33. library.jsp
34. library_retrievals.jsp
35. login.jsp
36. logo.bmp
37. new_admin_req.jsp
38. new_book_entry.jsp
39. new_book_required.jsp
40. new_class.jsp
41. new_department.jsp
42. new_employee_info.jsp
43. new_employee.jsp
44. new_event.jsp
45. new_fee.jsp
46. new_item.jsp
47. new_job.jsp
48. new_lab_item.jsp
49. new_lab.jsp
50. new_room.jsp
51. new_sports.jsp
52. new_student_info.jsp
53. new_subject.jsp
54. not_found.jsp
55. registration.jsp
56. result.jsp
57. salary.jsp
58. staff.jsp
59. stud_attendance.jsp
60. student_fee.jsp
61. student_info.jsp
62. student.jsp
63. stud_result_entry.jsp
64. submit.jsp
65. teacher.jsp
66. try1.jsp

CLASS FILES

1. adminreq.class
2. adminrequirement.class
3. ask.class
4. attendance.class
5. book_issue.class
6. Book.class
7. bookrequired.class
8. booksearch.class
9. check.class
10. checksid.class
11. ClassTO.class
12. confirm.class
13. date.class
14. DepartmentTO.class
15. EmployeeTO.class
16. Event.class
17. feesubmit.class
18. FeeTO.class
19. Items.class
20. jdbcconnection.class
21. job.class
22. JobTO.class
23. Labitem.class
24. Lab.class
25. Libretrievals.class
26. Login.class
27. LoginTO.class
28. logout.class
29. newbook.class
30. Newbooks.class

31. newclass.class
32. newdepartment.class
33. newevent.class
34. newfee.class
35. newitem.class
36. newjob.class
37. newlabitem.class
38. newlab.class
39. new.properties
40. newroom.class
41. newsports.class
42. newsubject.class
43. recruitment.class
44. result.class
45. return_book.class
46. Room.class
47. salary.class
48. seeattendance.class
49. seeresult.class
50. Sports.class
51. studentattendanceentry.class
52. StudentTO.class
53. Subject.class
54. submit.class
55. test.class
56. try1.class
57. url1.class

LIBRARIES

1. classes12.jar
2. classes12.zip

DATABASE TABLES

1. ROOM
2. DEPARTMENT
3. FEES
4. JOB
5. EMPLOYEE
6. CLASS
7. STUDENT
8. LIBRARY_BOOK
9. LIBRARY_RETRIEVAL
10. LIBRARY_REQ
11. FEES_COLLECTION
12. SALARY
13. ADMIN_REQ
14. SUBJECT
15. ITEMS
16. MAINTAINANCE
17. EVENT
18. STUDENT_EVENT
19. ENQUIRY
20. ITEM_REQ
21. SPORTS
22. SPORTS_ITEM
23. STUDENT_SPORTS
24. LAB
25. LAB_ITEM
26. STUD_ATTEND
27. EMP_ATTEND
28. RESULT
29. PHOTO
30. LOGIN

VIEWS

1. ACTIVITIES
2. ADMIN_REQUIREMENT
3. CLASS_ATTENDANCE
4. CLASS_VIEW
5. DEPT_VIEW
6. DEPARTMENT_ITEMS
7. EMPLOYEE_INFO
8. EVENT_VIEW
9. FEE
10. FEE_COLLECT
11. ITEM_REQUIREMENT
12. JOB_VIEW
13. LAB_ITEM_VIEW
14. LAB_REQ
15. LABORATORY
16. LIBRARY
17. MAINTAIN
18. SPORTS_VIEW
19. SUBJECTS

CONCLUSION AND FUTURE SCOPE

It's often felt that the Institution management is chaotic in nature. It involves many administrative work and co-ordination. It centralizes the mountains of data to learning and automates routing administrative functions.

Institution Automation System fulfills these requirements to some extent. One can foresee including some of the following features which could meet the basic requirement and some additional features of any educational institution:

- Canteen management

  In this module one can include a system which would be able to manage canteen of any institution like
  - Maintaining price list of food items
  - Maintain the accounts
  - Maintaining the store
- Transport management

  This module can have some of the following sub-modules
  - Scheduling of buses arrival and departure
  - Maintain the accounts
  - Maintaining of student information using the facility
- e-book facilities
- e-notice board
- Hostel management

Bibliography

REFERENCES

1. Head First Servlets and JSP
2. Java Server Pages – Hans Bergsten
3. JDBC 3.0 – Bernard Van Haecke
4. JavaScript Bible
5. Beginning Database Design – Gavin Powell
6. Oracle – The complete reference

WEB REFERENCES

1. www.4shared.com
2. www.pdfchm.com
3. www.sun.org
4. www.javaworld.org
5. www.oracle.com