Chapter 1

Introduction

1.1 Higher Education in India

Higher Education sector has witnessed a tremendous increase in its institutional capacity in the years since Independence. The number of Universities/University-level institutions have increased 18 times from 27 in 1950 to 504 in 2009. The sector boasts of 42 Central universities, 243 State universities, 53 State Private universities, 130 Deemed universities, 33 Institutions of National Importance (established under Acts of Parliament) and five Institutions (established under various State legislations). The number of colleges have also registered manifold increase with just 578 in 1950 growing to be more than 30,000 in 2011.

The quantum of growth in the HE (Higher Education) sector is spear-headed by the Universities, which are the highest seat of learning. University word is derived from the Latin word "Universities," which means 'specialized associations between students and teachers'. This Latin word referred to institutions of learning, which granted degrees to its students. The present day Universities are no different from the ancient institutions except for the fact that Universities today are much bigger in terms of both the subjects taught and the number of students.

In India, "University" means a University established or incorporated by or under a Central Act, a Provincial Act or a State Act and includes any such institution as may, in consultation with the University concerned, be recognized by the University Grants Commission (UGC) in accordance with the regulations made in this regard under this Act. Every year, millions of students from within the country and abroad, enter these portals mainly for their post graduate studies while millions leave these portals for the world outside.

Higher Education is the shared responsibility of both the Centre and the States. The coordination and determination of standards in institutions is the constitutional obligation of the Central Government.

The Central Government provides grants to UGC and establishes Central Universities in the country. The Central Government is also responsible for declaring educational institutions as "deemed-to-be University" on the recommendation of the UGC.

1.2 Technical Educational Management Bodies

Indian Government has setup various Educational management bodies to assess technical institutions in India. AICTE is one of them.

1.2.1 All India Council for Technical Education

The All India Council for Technical Education (AICTE) was set up in 1945 as an advisory body and later on in 1987 given the statutory status by an Act of Parliament. The AICTE grants approval for starting new technical institutions, for introduction of new courses and for variation in intake capacity in technical institutions. The AICTE has delegated to the concerned state governments powers to process and grant approval of new institutions, starting new courses and variations in the intake capacity for diploma level technical institutions. It also lays down norms and standards for such institutions. It also ensures quality development of technical education through accreditation of technical institutions or programmes. In additional to its regulatory role, the AICTE also has a promotional role which it implements through schemes for promoting technical education for women, handicapped and weaker section of the society promoting innovations, faculty, research and development, giving grants to technical institutions.

The technical institutions under the AICTE include post-graduate, under-graduate and diploma in the whole spectrum of technical education covering engineering/technology, pharmacy, architecture, hotel management and catering technology, management studies computer applications and applied arts and crafts.

The AICTE has its headquarters in New Delhi and seven regional offices located at Kolkata, Chennai, Kanpur, Mumbai, Chandigarh, Bhopal and Bangalore. A new regional office at Hyderabad has been set up and is to be operational soon. The Council discharges its functions through an Executive Committee.

1.2.2 National Board of Accreditation

The National Board of Accreditation (NBA) was set up by the All India Council for Technical Education (AICTE) in September 1994 in order to assess the qualitative competence of educational institutions from the Diploma level to the Post-graduate level in Engineering and Technology, Architecture, Pharmacy, Town Planning and Management. The NBA is also concerned with assessing and assuring the quality of the various constituent elements of the educational institutive infrastructure, physical resources, human resources, supporting systems like library resources, computational resources, and avenues for moulding and developing the students personality and learning characteristics. The NBA does not accord accreditation, to the institutions as a whole, but at the programme level, e.g. the four-year under-graduate engineering degree course (after 10+2) in Mechanical Engineering (or any other branch). For NBA, accreditation is a process of quality assurance, whereby a programme (e.g. B.E./B.Tech.) in an approved Institution is critically appraised at intervals not exceeding six years to verify that the Institution Programmes meet the norms and standards prescribed by AICTE, established by an Act of the parliament in 1987. Accreditation provides quality assurance that the academic aims and objectives of the Institution are known to be honestly pursued and effectively achieved by the resources currently available, and that the institution has demonstrated capabilities to ensure continued effectiveness of the educational programme, over the period of accreditation.

1.2.3 Washington Accord and New System of Accreditation

The Washington Accord, signed in 1989, recognizes substantial equivalence in the accreditation of qualifications in professional engineering degrees normally of four years duration. It is an agreement between the bodies responsible for accrediting professional engineering degree programmes in each of the signatory countries. It recommends that graduates of accredited programs in any of the signatory countries be recognized by the other countries as having met the academic requirements for entry to the practice of engineering.

Signing the accord by India will enhance global mobility of the engineering graduates from India who can pursue further studies or take up jobs in the signatory countries. At present full

members of the accord are; Australia, Canada, Taipei, Hong Kong, Ireland, Japan, Korea, Malaysia, New Zealand, Singapore, South Africa, United Kingdom and United States . India became a provisional member of Washington Accord countries in 2007. With NBA changing its accreditation norms to bring them at par with global standards, India's chances to become a full-fledged member of the Washington Accord has brightened.

NBA has completely changed its guideline documents so that accreditation process is in tune with the Washington Accord. The new documents made available are :

1. Evaluation Guidelines: This gives various criteria of evaluation and the maximum points for each criterion. It also gives guidelines for evaluation for each of the sub-criterion. Appendix I of this document is a very critical input related to Program Educational Objectives (PEOs) and Outcomes. These are new concepts for degree awarding institutions, though these concepts in some form or other existed in polytechnic system in India. Also some sample Assessment Plans have been included.

2. Evaluation Report: It gives in a very comprehensive manner criteria for which evaluators have to award points.

3. Self Assessment Report: It has two sections; one related to the Institution and other related to the program. These proformas have to be filled and sent to the NBA before the visit. Proformas have been developed in such a way that all the information is collected objectively, so that giving points by evaluators becomes easy. List of documents to be made available to the accreditation team has also been specified clearly.

   Since, NBA accreditation process is exhaustive & involves enormous amount of data storage & processing, an automated process supported by specialized computer software is desirable. This software enables automation of this process thereby reducing the workload on department staff.

1.3 Cloud Computing

"Cloud computing is the delivery of computing as a service rather than a product." Before introduction of Cloud computing IT infrastructure had been a product. One has to buy IT infrastructure first to be able to use it. To be able to deploy an application on 100 servers, one has to first buy them, rack them, install software on them, and then be able to use them. This takes considerable time, upfront money, and effort. Cloud computing tries to eliminate the above mentioned drawbacks by providing this IT infrastructure as a service.

1.3.1 Layers in Cloud Computing:

- **Physical Layer**

  This layer includes your data center, machines and networking equipment. The data center is the backbone of the hosted application. It should be established in a secure location. Rackspace Cloud, Amazon EC2, Google Compute Engine come under the application layer.

- **Low-Level Hardware Resources**

  The low-level hardware resources are computation, storage, bandwidth, etc. Most people in the company like to think in terms these low-level hardware resources and not how you achieve them. For example one might be concerned with how much storage capacity one has and not concerned how many physical hard-drives or physical machines that translates to. Thus the physical layer in turn translates to some amount of low-level hardware resources. Amazon S3, Azure Blob Storage, Amazon ELB come under Low level Hardware Resources.

- **Low-Level Software Resources**

  This layer includes things like your web server, application server, database, etc. Most developers/testers work at this layer or above. As long as their database is functioning and their servers are up and running they are content, its only when a problem creeps up when they have to dig deeper into the lower layers and find the cause. Windows SQL Azure, Amazon RDBS, Google App Engine come under Low Level Software Resources.

- **Application Layer**

  Developers spend most of their time at this layer. As long as the lower layers are functioning it is acceptable. Again only when a problem creeps up that they have to dig deeper into lower layers and find the culprit. Google Apps comes under the application layer.

- **Users**

  Support should be provided to users in different geographic locations using a variety of devices to access the application. If it is browser-based application the users should be able to access it using their cell-phones, tablets, MACs or Windows systems.

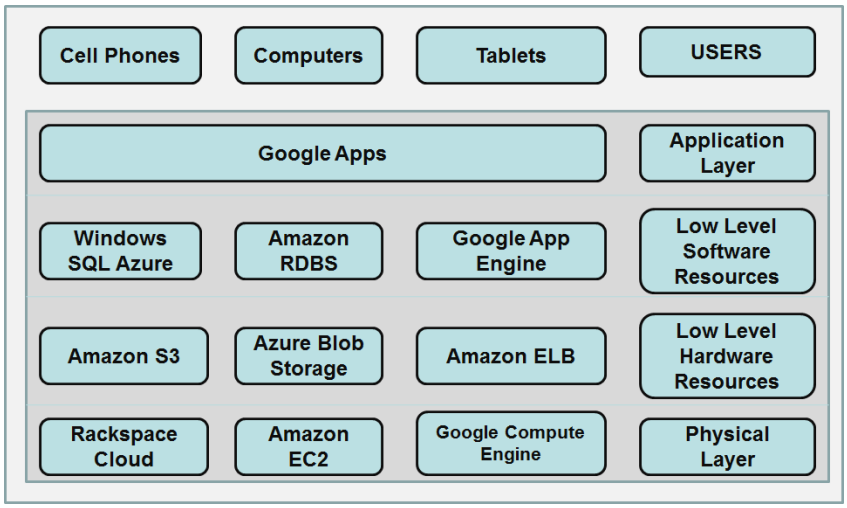


Fig1.1 : Cloud Infrastructure

1.3.2 Cloud Segments

Cloud computing services are divided into three segments

1. Infrastructure as a Service

2. Platform as a Service

3. Software as a Service

- **Infrastructure as a Service**

Enables computing, storage and/or network resources to be provisioned without owning physical devices. Provides only one of the key infrastructure components. In the most basic cloud-service model, providers of IaaS offer computers - physical or (more often) virtual machines - and other resources.

- **Platform as a Service**

Platform as a Service provide a software execution environment that application services can run on. In the PaaS model, cloud providers deliver a computing platform typically including operating system, programming language execution environment, database, and web server. Application developers can develop and run their software solutions on a cloud platform without the cost and complexity of buying and managing the underlying hardware and software layers.

- **Software as a Service**

Delivers applications and end-user tools. In the SaaS model, cloud providers install and operate application software in the cloud and cloud users access the software from cloud clients. Cloud users do not manage the cloud infrastructure and platform where the application runs. This eliminates the need to install and run the application on the cloud user's own computers, which simplifies maintenance and support.

1.3.3 Cloud Deployment Models

The primary cloud deployment models are

1. Public Cloud

2. Private Cloud

3. Community Cloud

4. Hybrid Cloud

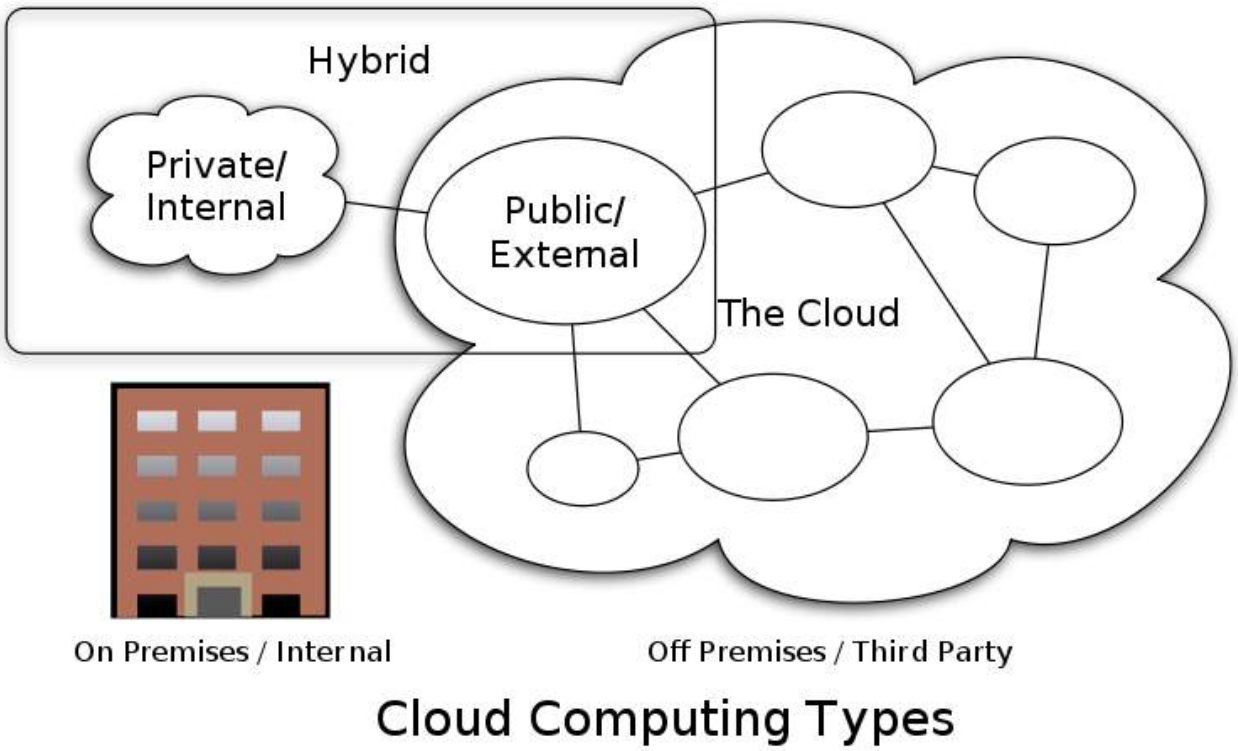


Fig 1.2: Cloud Deployment Models

- **Public Cloud**

The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services. A public cloud is one based on the standard cloud computing model, in which a service provider makes resources, such as applications and storage, available to the general public over the Internet. Public cloud services may be free or offered on a pay-per-usage model.

- **Private Cloud**

The cloud infrastructure is operated solely for an organization. Private cloud is cloud infrastructure operated solely for a single organization, whether managed internally or by a third-party and hosted internally or externally. Undertaking a private cloud project requires a significant level and degree of engagement to virtualize the business environment, and requires the organization to reevaluate decisions about existing resources. They have

attracted criticism because users "still have to buy, build, and manage them" and thus do not benefit from less hands-on management, essentially "[lacking] the economic model that makes cloud computing such an intriguing concept"

- **Community Cloud**

  The cloud infrastructure is shared by several organizations and supports a specific community that has shared goals.

- **Hybrid Cloud**

  The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability. Hybrid cloud is a composition of two or more clouds (private, community or public) that remain unique entities but are bound together, offering the benefits of multiple deployment models.

In our software, we are using public cloud. The most prevalent benefit in a public cloud is cost savings. Organizations do not buy, install, operate or maintain servers or other equipment as they would with private clouds. Organizations may develop their own applications, but they are hosted by a public cloud provider.

Public clouds are also designed as true multi-tenant environments. This allows a huge number of users to share a provider computing resources, which also makes public cloud services highly cost-efficient for users. Experts agree that the cost of a public cloud is typically far lower than a traditional data center and private cloud because there is no capital expense. There is also greater flexibility for users because organizations pay only for the computing resources that they actually use.

Chapter 2

Literature Survey

The survey undertaken by us included searching at numerous websites, technical magazines, consulting staff members etc. We arrived at following details which helped in arriving at our problem and proposing a solution for the same.

2.1 Present Application Scenarios

Currently, there are various applications models for the benefit of professors and also students. Moodle is an example for a free open-source PHP web application for producing modular internet-based courses that support a modern social constructionist pedagogy. Websites like Classle provide an online forum for students to interact with each other and take part in numerous competitions. Classle helps a student to find course related documents, execute projects, watch video lectures, attempt quizzes, find a job etc. In a way, it helps a student to build his/her knowledge and also provides a platform to exhibit it.

SlideShare is another website popular among students. It is Web 2.0 based slide hosting service. Students can upload files privately or publicly in the following file formats: PowerPoint, PDF, Keynote or OpenOffice presentations. Launched on October 4, 2006, the website is considered to be similar to YouTube, but for slideshows. The website was originally meant to be used for businesses to share slides among employees more easily, but it has since expanded to also become a host of a large number of slides which are uploaded merely to entertain. Although the website is primarily a slide hosting service, it also supports documents, PDFs, videos and webinars. SlideShare also provides users the ability to rate, comment on, and share the uploaded content. It has turned as a platform for exchanging educational content between students and experts.

These two websites along with numerous others are available to solve the problems of students and benefit them but there are very few websites to solve problems of departments. The few applications available, work to solve the conventional problems. Even while talking with our

faculty members, we found out that there is no application for storing/processing documents. Moreover there is no application to solve the problem of generating reports for the NBA accreditation process.

2.2 Similar Products

There are many applications which are similar to our application yet distinguishable. A company named robosoft provides an ERP for educational institutes called RoboVidya. RoboVidya is a complete ERP which manages all the transactions of the department. It has various modules for different aspects of the college as well for each department.

Our department uses software created by alumnus of our department. This softare title "Automation of Outcome Based Education (OBE)" handles all the aspects of the OBE suitably. It serves as a tool for uploading and downloading OBE documents and also lets students take proficiency test through it.

Though these applications proved helpful in serving their purpose, there is no application to solve the problem of generating reports for the NBA accreditation process. And thus the NBA Information Management System comes in to picture.

2.3 About NBA Document Requirement

The Self Assessment document procured by us proved a very useful source of knowledge for our project. The document clearly states all the reports which are required by the NBA assessment team for evaluation of every department. The document helped us to fine tune our application requirements as we removed some aspects which were already being taken care of by existing applications and add those which were not considered by other applications.

2.4 Problem Formulation

The next chapter defines the problem, the objectives to be fulfilled, functional and non-functional requirements and the motivation for carrying out the work in this project. It also briefs about the proposed solution.

2.5 Comparison of Cloud Service Providers

Following is a comparison between the popular cloud service providers. We chose Windows Azure over the other service providers because of its compatibility with .Net framework and SQL Server Relational Database Management System (RDBMS).

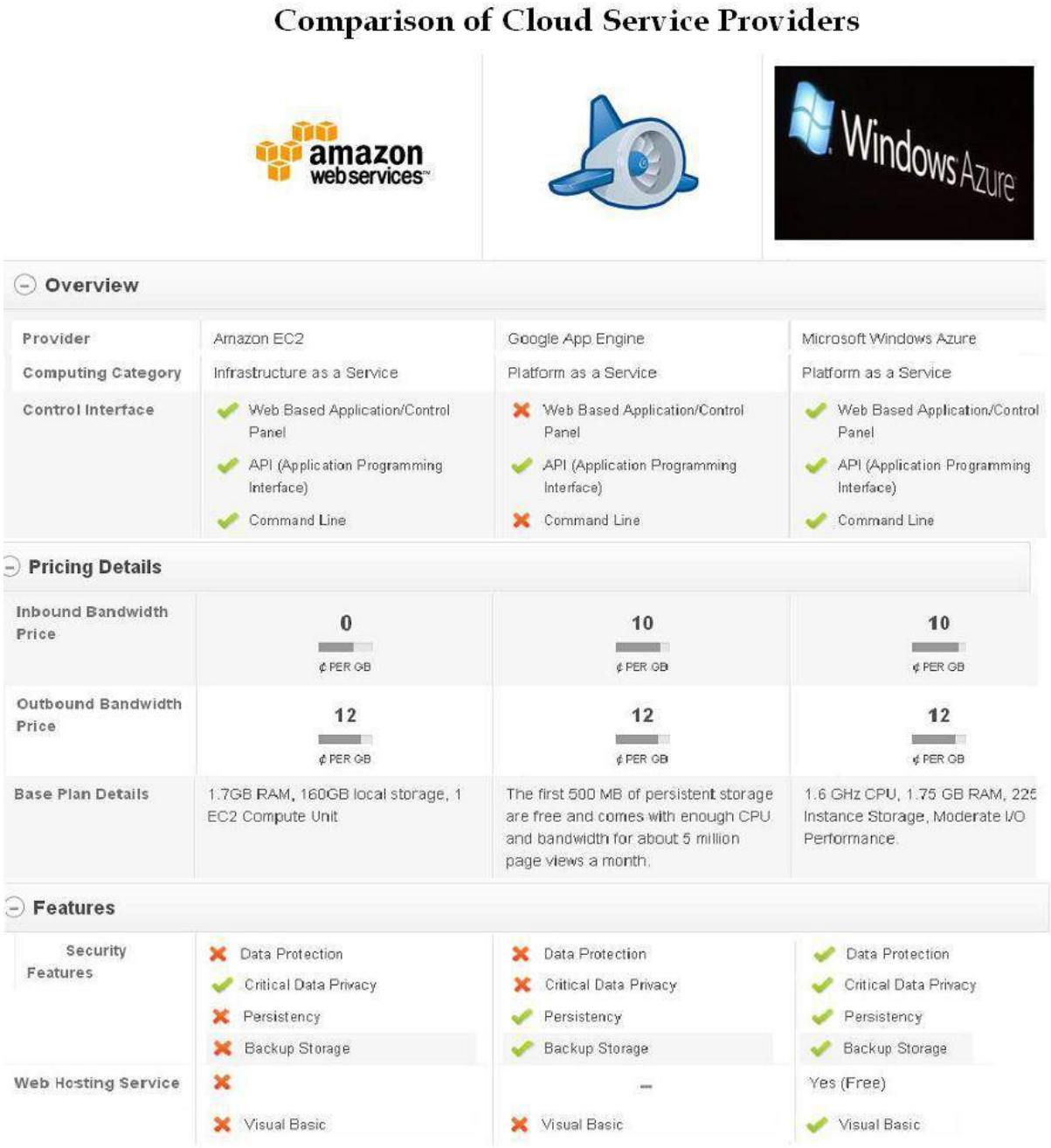


Fig 2.1: Comparison of Cloud Service Providers.

Chapter 3

Problem Definition

There is a need to provide software solution for the departments of an institution, by developing an information management system which stores multimedia information in a refined manner, enabling quick retrieval. The repository should include information and all the necessary documentation required for National Board of Accreditation (NBA).

3.1 Objective and Scope

The objective of our project is to develop an application which will reduce all the paperwork related to the assessment of a department undertaken by the National Board of Accreditation (NBA) every four years. We aim to create an application which will be used by everyone within the department. The main user or the administrator of the application will be a member of the non-teaching faculty. The job of the administrator will be to manage all the data and generate reports which will be beneficial to department during the assessment by NBA.As the application will reside on cloud, it will be accessible through a public domain. All the security related aspects will be taken care of by the cloud service provider. As everything will reside on the cloud, there won't be any maintenance issues to be taken care of.

3.2 Motivation

The main motivation behind taking up this project was to create something which will be used by our college. We wanted to create an application which will be beneficial to our college. On top of it, if the beneficiary turned out to be our own department along with other departments, our joy was doubled. The motivation behind putting our application on cloud rather than on a regular server was to learn a new and upcoming technology. Cloud computing is one of the latest technology in the market which is touted as the next best thing in the world of technology. Our aim was also to familiarize ourselves with it and thus learn a new thing.

3.3 Functional Requirements:

In software engineering, a functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define *what* a system is supposed to accomplish. Managing the Staff details like their Qualification, Designation, Teaching load, Achievements etc.

- Also the details of Non-Teaching staff like their technical skills, Qualifications and responsibilities are taken care of.
- Various documents (Quizzes, Assignments etc) as required by OBE are managed.
- Managing the Student Information related to OBE projects and seminars and various workshops attended.
- Keeping track of student's achievements in various events organized by department and other colleges.
- Details of the department like the budget allocation, admission details, facilities provided and the labs conducted.
- Providing Placement Information about the companies visited and the students recruited.

3.4 Non-Functional Requirements:

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. Non-functional requirements are often called qualities of a system.

- User Friendly Interactive interface
- Efficient storage and access.
- Portability of the software.
- Reduces the department workload.
- Storing documents in an organized manner.

3.5 Proposed Solution

The software solution provided for the departments of an institution, helps generating the necessary documents as required by the National Board of Accreditation. The Application being hosted in cloud makes it portable and easier to maintain. We provide web interface for user friendly interaction with the repository.

Chapter 4

Feasibility Study

Feasibility study is a test of a system proposal according to its workability impact on organization ability to meet user needs and effective use of resources.

The key considerations in Feasibility Study are:

1. Technical Feasibility
2. Economical Feasibility
3. Behavioral Feasibility

4.1 Technical Feasibility

Technical feasibility centers on the existing computer system and to what extent it can support the proposed addition. The proposed system can easily be implemented with the existing technology with very few changes. As there is a provision of getting a cloud on trial, the proposed system can be easily implemented on a cloud.

4.2 Economical Feasibility

Economical feasibility means expenditure incurred for developing the new system is cost effective. In our project there is no investment on any new infrastructure. As we will be able to procure cloud on a trial basis for free, there is no cost regarding cloud. All the softwares used for making our project are present within the cloud. Hence the cost of developing was nothing.

4.3 Behavioral Feasibility

A behavioral Feasibility centers on what the reaction of users is likely to be towards development of proposed system. There is no need of any special training for the non-teaching staff as it is easy to use. Also it is easy to use for staff and students as all the things are self explanatory.

This project has a well built user-interface which is user friendly as well as consistent, using which operator / administrator can manipulate the database and provide all the required functionalities.

CHAPTER 5

System Architecture

System architecture is the conceptual design that defines the structure and/or Behavior of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures of the system, which comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them, and provides a plan from which products can be procured, and systems developed, that will work together to implement the overall system.

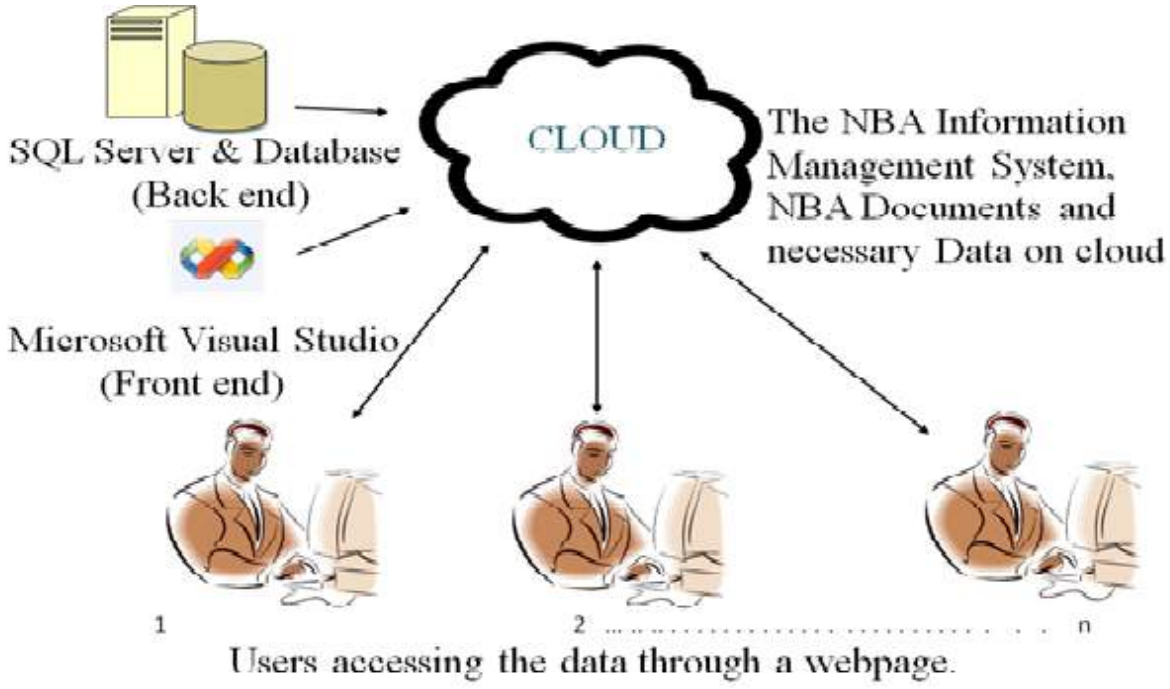


Fig: 5.1 System Architecture

This system includes a central cloud which includes Microsoft Visual Studio used for front end and SQL server RDBMS as back end. The website for this application is hosted on this cloud and various users can access this application through the website provided.

Website Link for our application:

http://gitnba.cloudapp.net

5.1 System Requirements Specification

A System Requirements Specification (abbreviated SRS) is a structured collection of information that embodies the requirements of a system.

5.1.1 Software Requirements

Software Requirements is a field within Software Engineering that deals with establishing the needs of stakeholders that are to be solved by software. Following are the software requirements for this application:

- Operating System- Windows 2003 server/XP/7
- IDE Used- Microsoft Visual Studio .NET 2010
- RDBMS - Microsoft SQL Server 2012

Tools

Following are the tools that we made use of, during the development of the project:

- Programming language used- ASP.NET
- Technology used- .NET

Cloud

Our Project requires a cloud service whose description is as follows:

- Windows Azure IAAS cloud service
- Pay as you use Subscription

5.1.2 Hardware Requirements

The hardware requirements as required by this application are:

- HDD (minimum 40GB)
- RAM (minimum 2GB)
- Processor (Intel core 2 Duo with 2.8 GHz or above)

Chapter 6

Design Artifacts

6.1 Class Diagram

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

The following is a class diagram for this application:

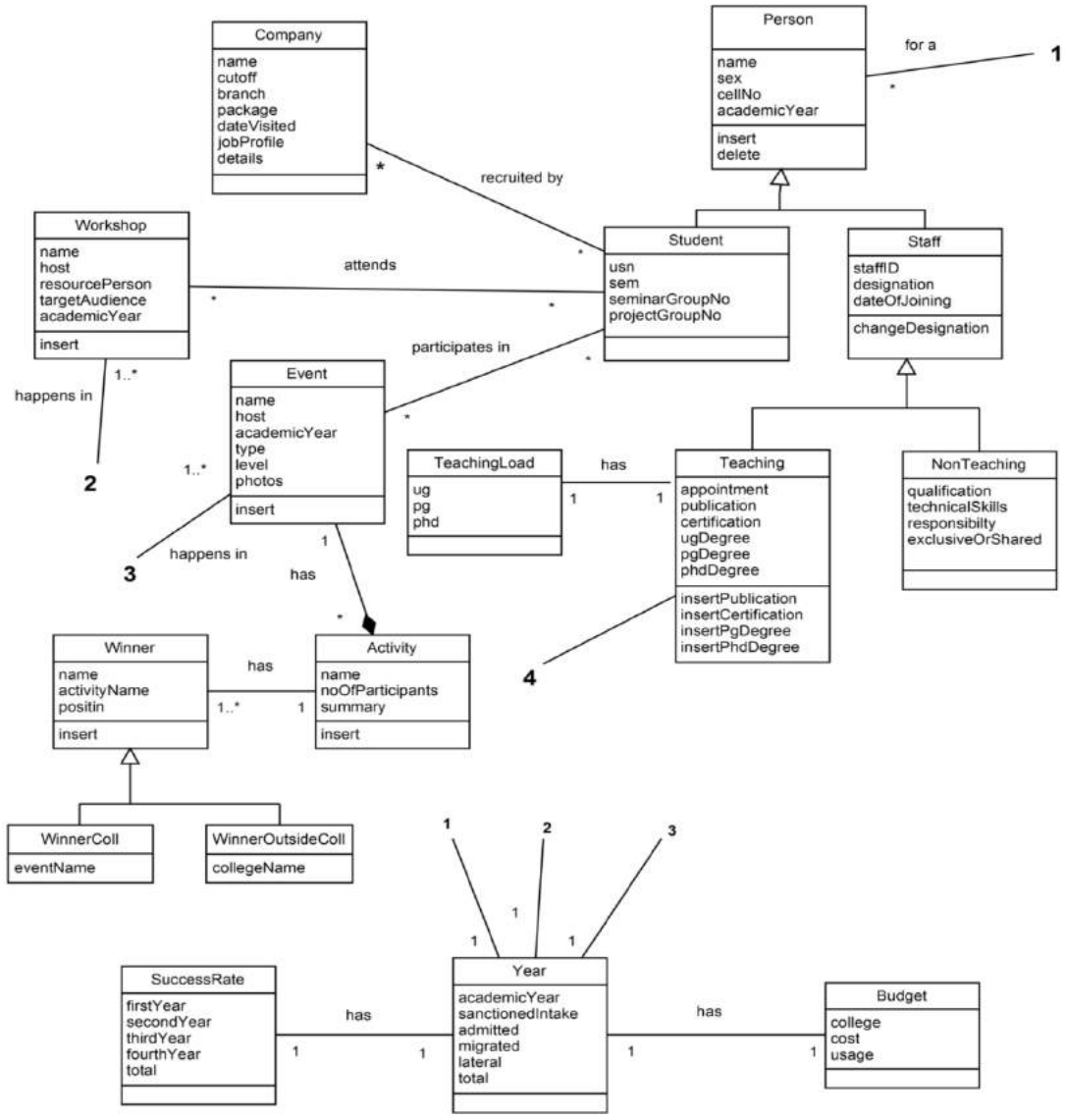


Fig 7.2: Class Diagram

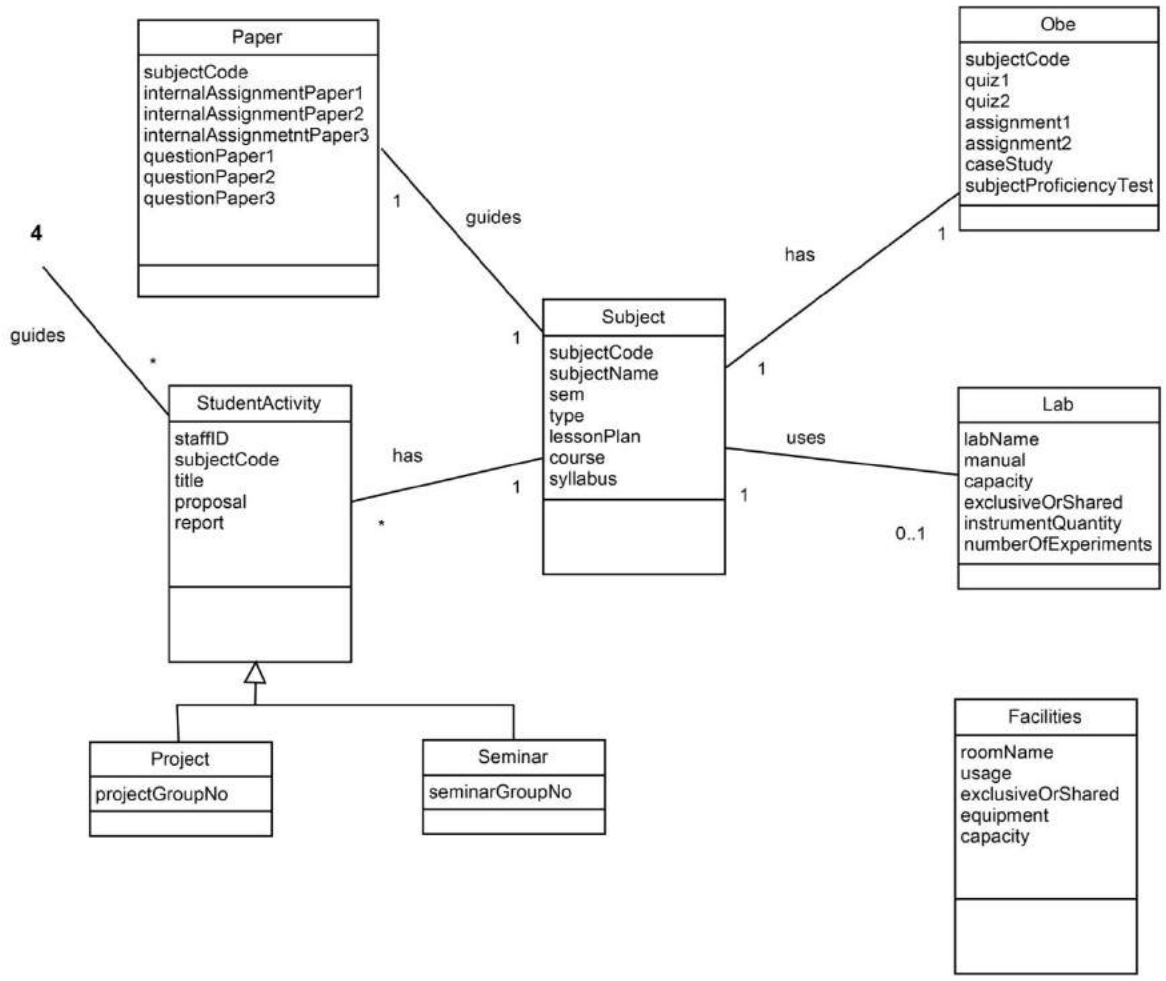


Fig 6.1: Class Diagram of this application (continued)

Here, Staff and Student are the classes which are inherited from the parent class Person. Each class has its own set of attributes and operations. The attributes and operations of Person class are derived to both of these classes. For example we can insert a new student, update student data or delete a student from the database.

Company, Workshop and Event are some of the other classes. A Student gets recruited in a Company, attends Workshops and participates in Events.

6.2 Use Case Diagrams

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

The application requires three users interacting with the system, admin, staff and student.

We shall now go through the use case diagrams for each of them:

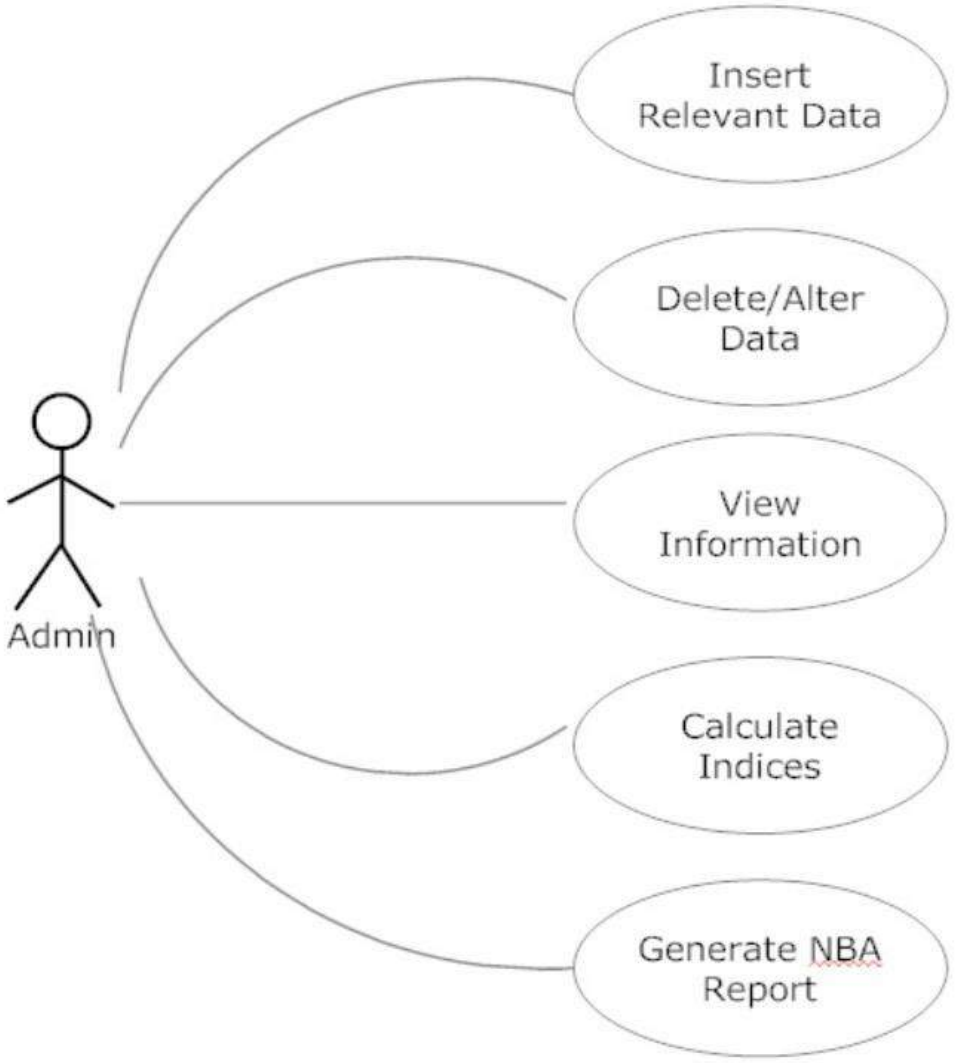


Fig 6.2: Use case diagram for admin

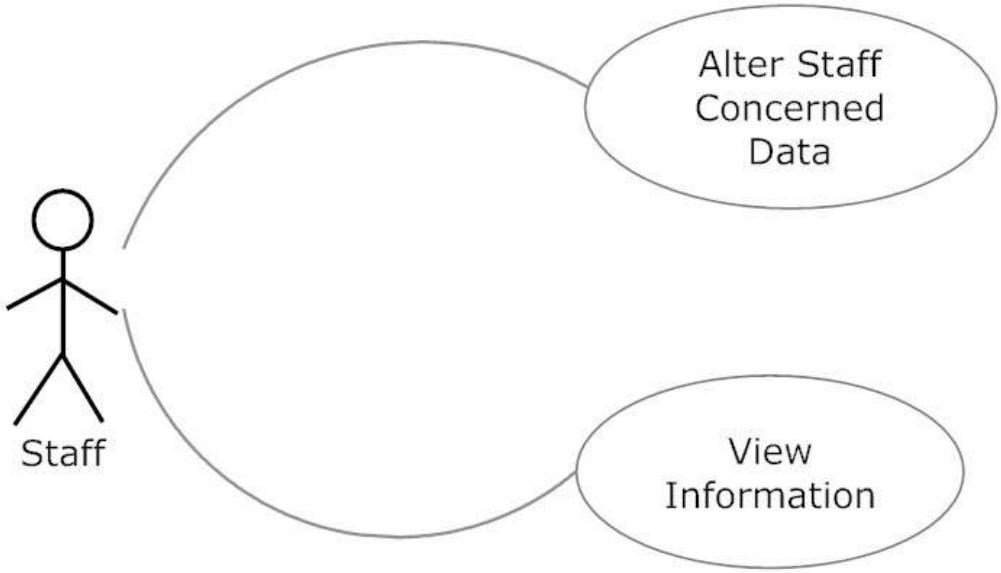


Fig 6.3: Use Case diagram for Teaching Staff

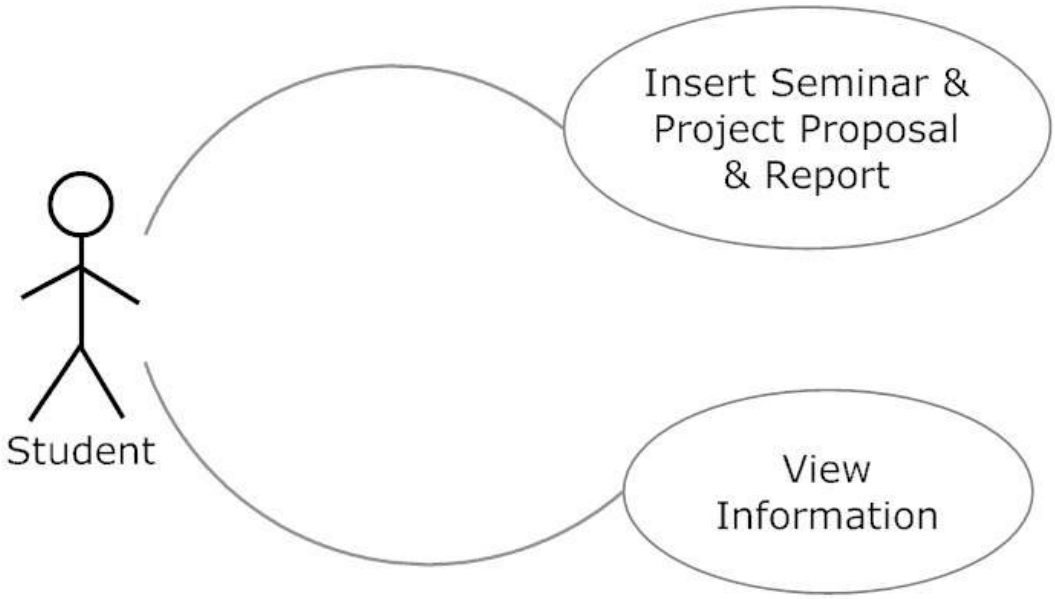


Fig 6.4: Use Case diagram for Student

6.3 Sequence Diagrams

A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

A sequence diagram shows, as parallel vertical lines (*lifelines*), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

We shall now look at the sequence diagrams for some of the interactions between the users and the system.

In this section we have one sequence diagram for administrator inserting teaching staff data, updating teaching staff data and deleting staff data each respectively.

Also the sequence diagrams for teaching staff uploading and viewing Internal Assessment (IA) papers and the students uploading and viewing seminar proposals and reports.

- SEQUENCE DIAGRAMS FOR ADMINISTRATOR

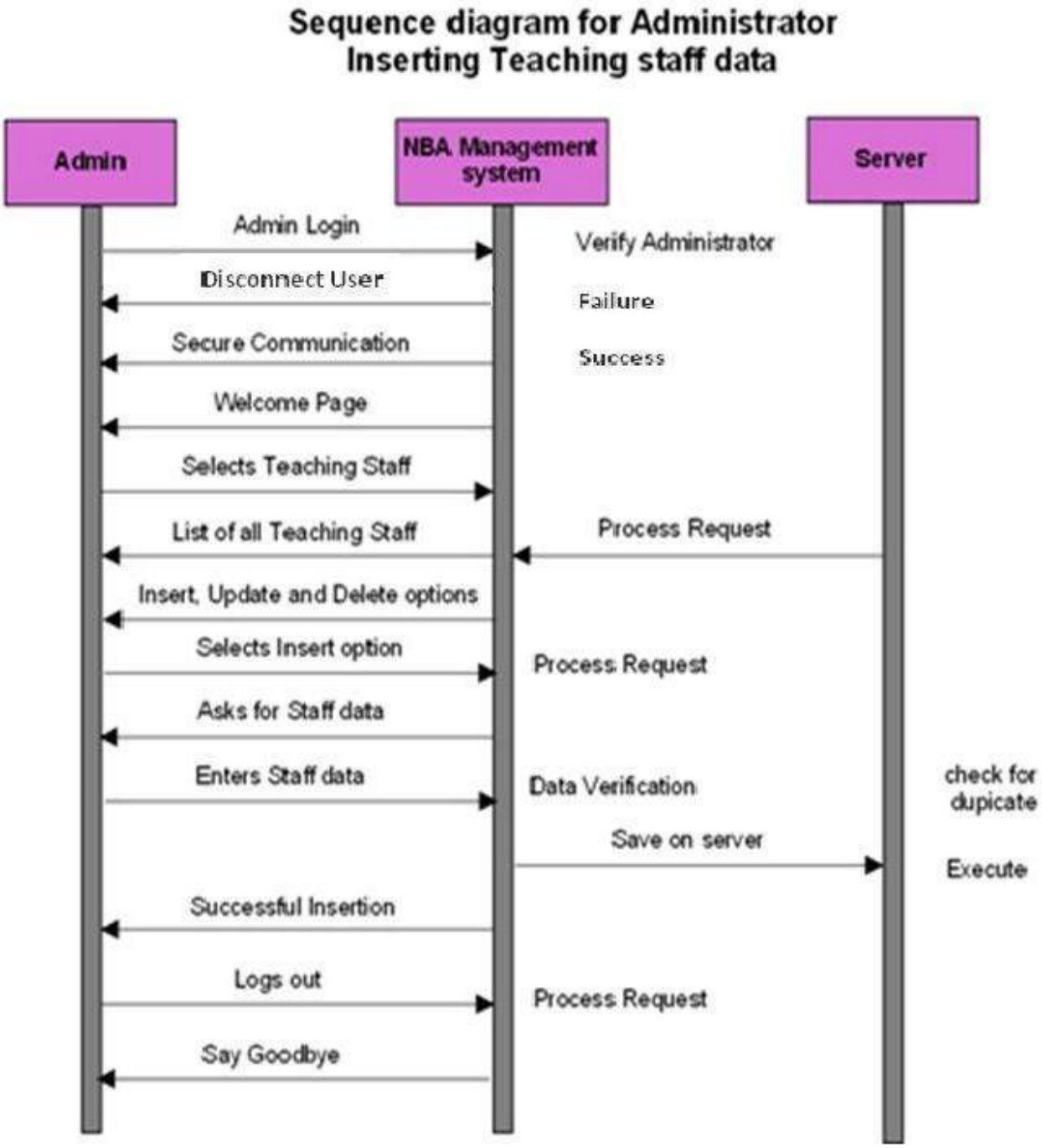


Fig 6.5: Sequence diagram for administrator inserting teaching staff data

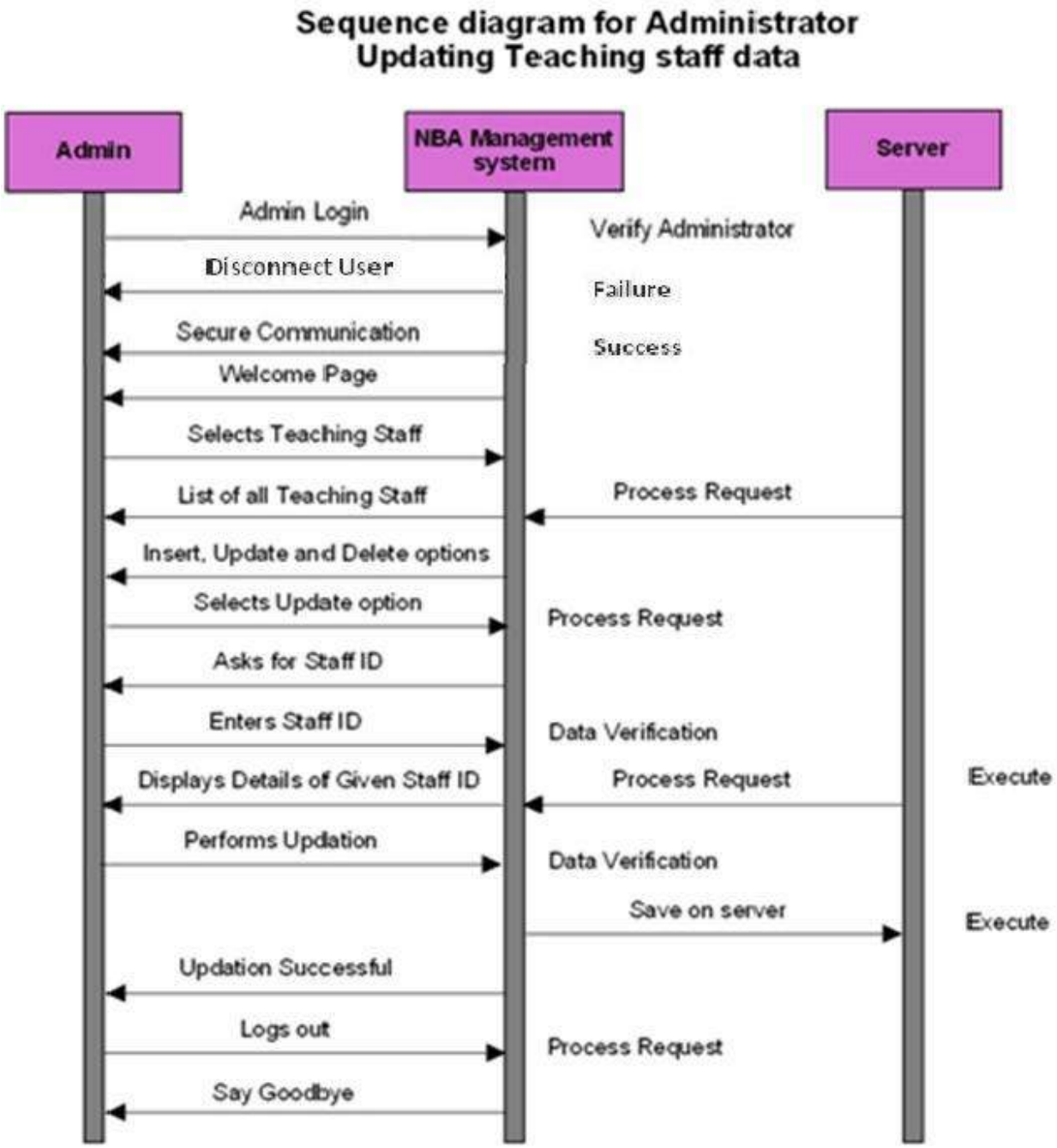


Fig 6.6: Sequence diagram for administrator updating teaching staff data

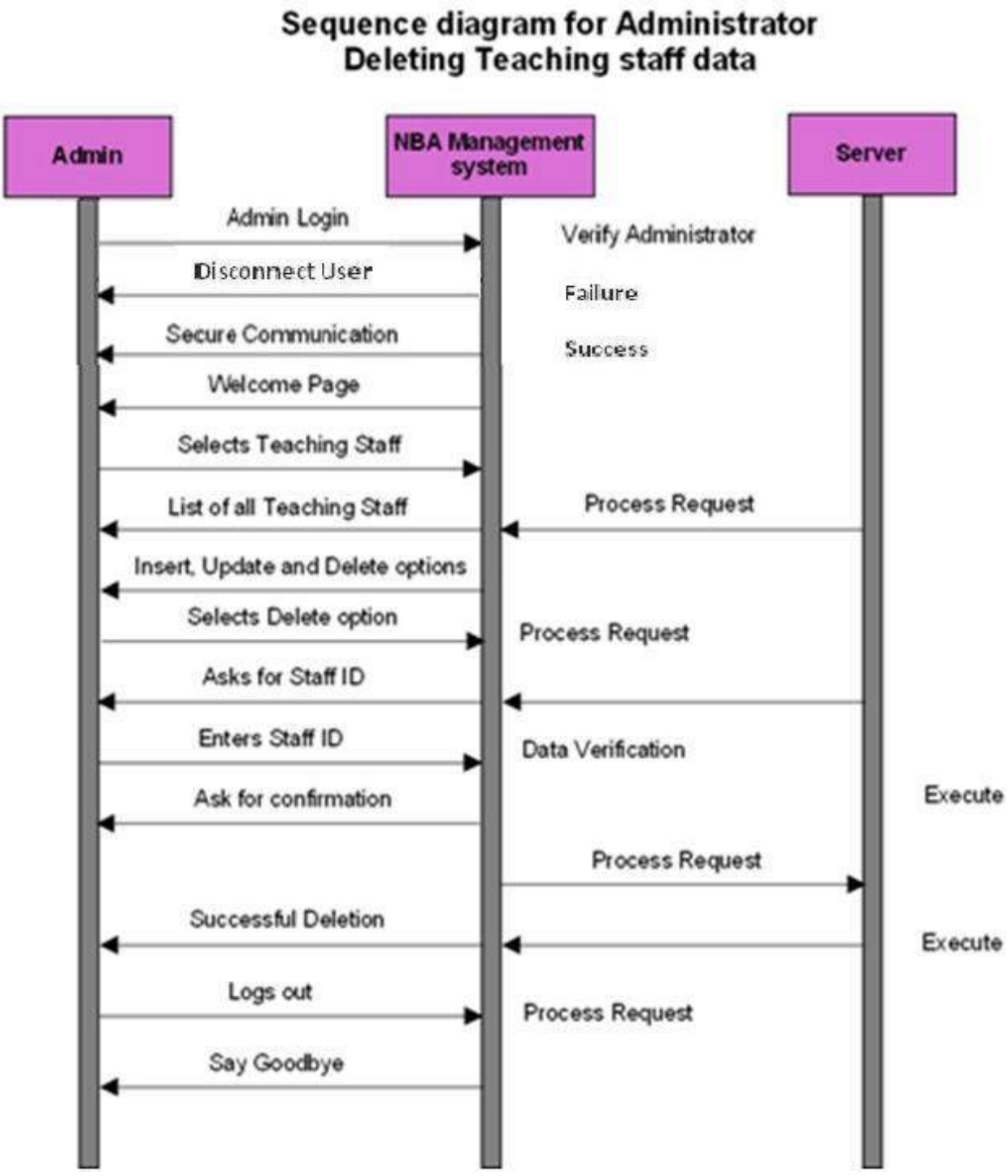


Fig 6.7: Sequence diagram for administrator deleting teaching staff data

- SEQUENCE DIAGRAMS FOR TEACHING STAFF

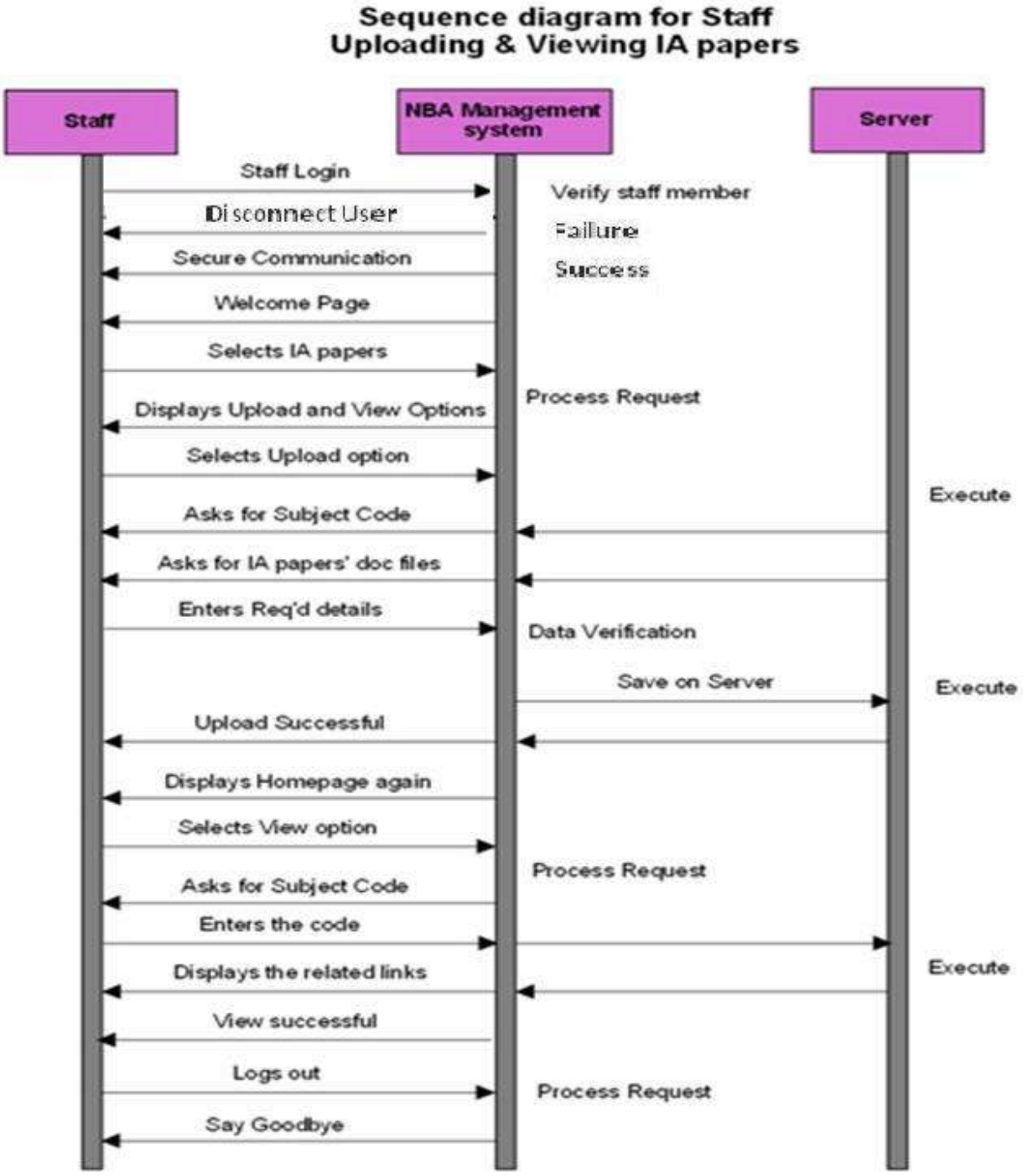


Fig 6.8: Sequence diagrams for teaching staff uploading and viewing IA papers

- SEQUENCE DIAGRAMS FOR STUDENT

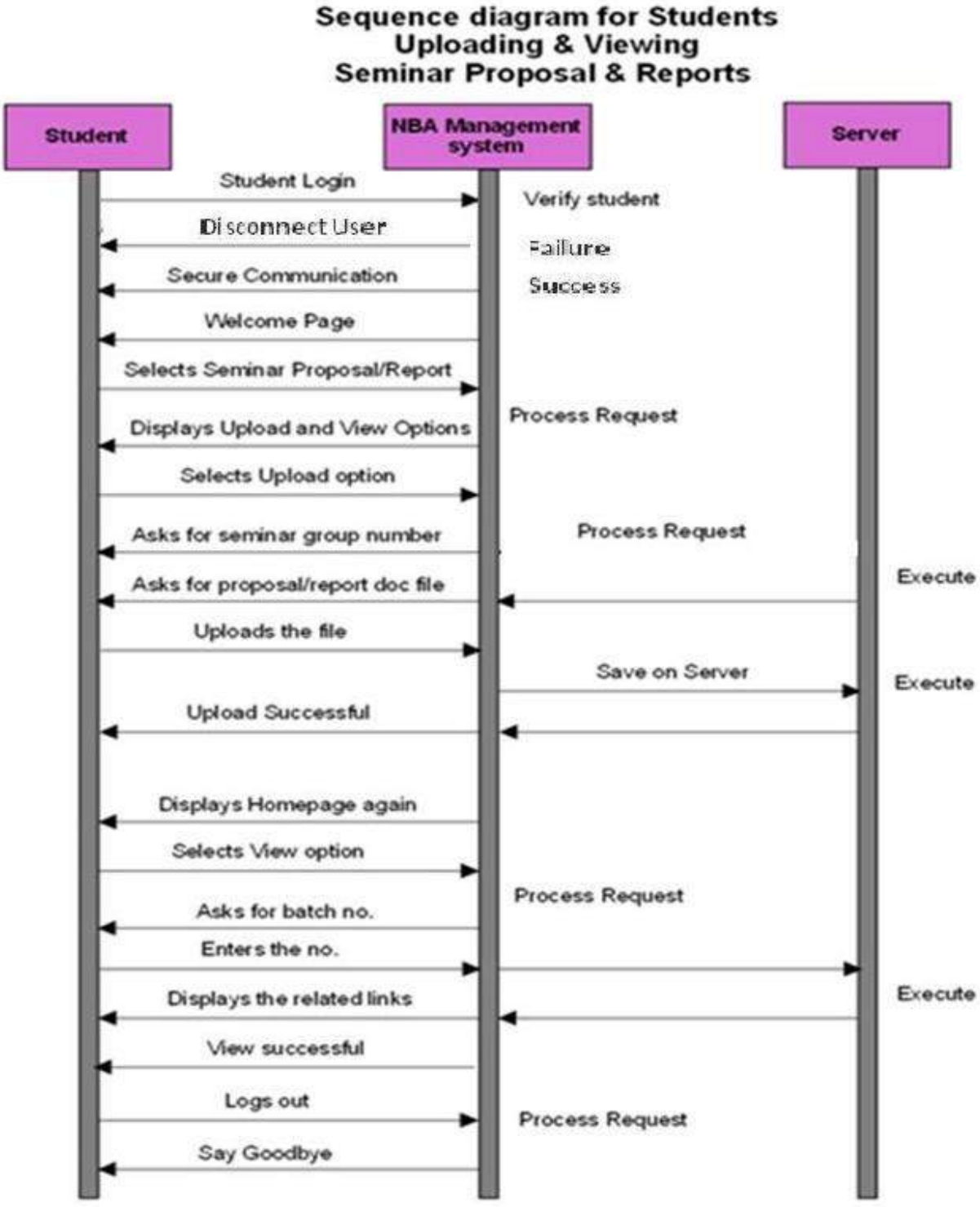


Fig 6.9: Sequence diagram for student uploading and viewing Seminar proposal and reports.

Chapter 7

Database Design

7.1 ER Diagram

An entity-relationship diagram is a data modeling technique that creates a graphical representation of the entities, and the relationships between entities, within an information system.

The three main components of an ERD are:

- The *entity* is a person, object, place or event for which data is collected. The entity is represented by a rectangle and labeled with a singular noun.
- The *relationship* is the interaction between the entities. A relationship may be represented by a diamond shape, or more simply, by the line connecting the entities. In either case, verbs are used to label the relationships.
- The *cardinality* defines the relationship between the entities in terms of numbers. The three main cardinal relationships are: one-to-one, expressed as 1:1; one-to-many, expressed as 1:M; and many-to-many, expressed as M:N.

For Example in our application, Staff, Student, Workshop, Company, Event and Subject are some of the entities and these entities have their own set of attributes.

The Student entity has attributes like USN, name, sem, gender, cellno etc, where USN is the primary key. The Company entity has attributes like C_name, cutoff, branches, package, date_visited, job_profile, details etc. Here, C_name is the primary key. Event has attributes like e_name, level, host, type, photos etc with e_name as primary key.

The Student entity is related to Company, through a relationship called Recruited. i.e Students get recruited in a company. The cardinality is N:N. The Student entity is also related to Event, through a relationship called Participates i.e Students participate in events. The cardinality in this case also is N:N.

Similarly, we have various other entities related to each other through the relationships.

The ER diagram of our application is shown in next page.

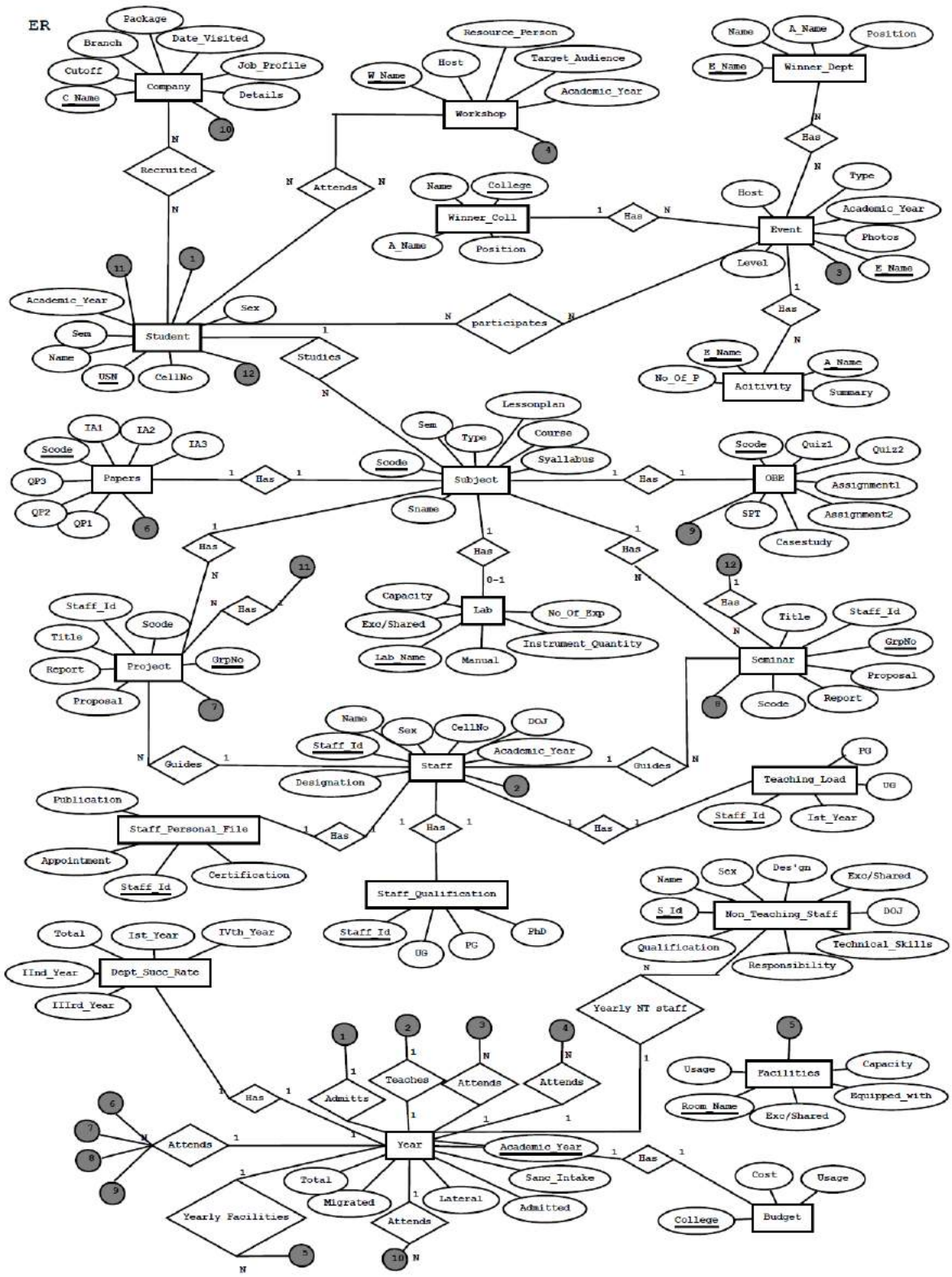


Fig 7.1: ER diagram

7.2 Relational Schema

A database schema of a database system is its structure described in a formal language supported by the database management system (DBMS) and refers to the organization of data to create a blueprint of how a database will be constructed (divided into database tables). The formal definition of database schema is a set of formulas (sentences) called integrity constraints imposed on a database. These integrity constraints ensure compatibility between parts of the schema. All constraints are expressible in the same language. A database can be considered a structure in realization of the database language.[1] The states of a created conceptual schema are transformed into an explicit mapping, the database schema. This describes how real world entities are modeled in the database.

7.2.1 ER-to-Relational Mapping

Step 1: Mapping of Regular Entity Types.

For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E. Include only the simple component attributes of a composite attribute. Choose one of the key attributes of E as the primary key for R. If the chosen key of E is a composite, then the set of simple attributes that form it will together form the primary key of R. If multiple keys were identified for E during the conceptual design, the information describing the attributes that form each additional key is kept in order to specify secondary (unique) keys of relation R. Knowledge about keys is also kept for indexing purposes and other types of analyses.

Step 2: Mapping of Weak Entity Types.

For each weak entity type W in the ER schema with owner entity type E, create a relation R and include all simple attributes (or simple components of composite attributes) of W as attributes of R. In addition, include as foreign key attributes of R, the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s); this takes care of mapping the identifying relationship type of W. The primary key of R is the combination of the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any. If there is a weak entity type E2 whose owner is also a weak entity type E1 , then E1 should be mapped before E2 to determine its primary key first.

Step 3: Mapping of Binary 1:1 Relationship Types.

For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. There are three possible approaches: (1) the foreign key approach, (2) the merged relationship approach, and (3) the cross reference or relationship relation approach. The first approach is the most useful and should be followed unless special conditions exist, as we discuss below.

1. Foreign key approach: Choose one of the relations—S, say—and include as a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S. Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type R as attributes of S.

2. Merged relation approach: An alternative mapping of a 1:1 relationship type is to merge the two entity types and the relationship into a single relation. This is possible when both participations are total, as this would indicate that the two tables will have the exact same number of tuples at all times.

3. Cross-reference or relationship relation approach: The third option is to set up a third relation R for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types. As we will see, this approach is required for binary M:N relationships. The relation R is called a relationship relation (or sometimes a lookup table), because each tuple in R represents a relationship instance that relates one tuple from S with one tuple from T. The relation R will include the primary key attributes of S and T as foreign keys to S and T. The primary key of R will be one of the two foreign keys, and the other foreign key will be a unique key of R. The drawback is having an extra relation, and requiring an extra join operation when combining related tuples from the tables.

Step 4: Mapping of Binary 1 : N Relationship Types.

For each regular binary 1 : N relationship type R, identify the relation S that represents the participating entity type at the N-side of the relationship type. Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R; we do this because each entity instance on the N-side is related to at most one entity instance on the 1-side of the relationship type. Include any simple attributes (or simple components of composite attributes) of the 1:N relationship type as attributes of S.

Step 5: Mapping of Binary M : N Relationship Types.

For each binary M:N relationship type R, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S. Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S. Notice that we cannot represent an M:N relationship type by a single foreign key attribute in one of the participating relations (as we did for 1:1 or 1:N relationship types) because of the M:N cardinality ratio; we must create a separate relationship relation S.

Step 6: Mapping of Multivalued Attributes.

For each multivalued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K—as a foreign key in R—of the relation that represents the entity type or relationship type that has A as a multivalued attribute. The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.

Step 7: Mapping of N-ary Relationship Types.

For each n-ary relationship type R, where n > 2, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types. Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S. The primary key of S is usually a combination of all the foreign keys that reference the relations representing the participating entity types. However, if the cardinality constraints on any of the entity types E participating in R are 1, then the primary key of S should not include the foreign key attribute that references the relation E corresponding to E.

We have gone through these ER to relational mapping techniques and guidelines to get the final relational schema.

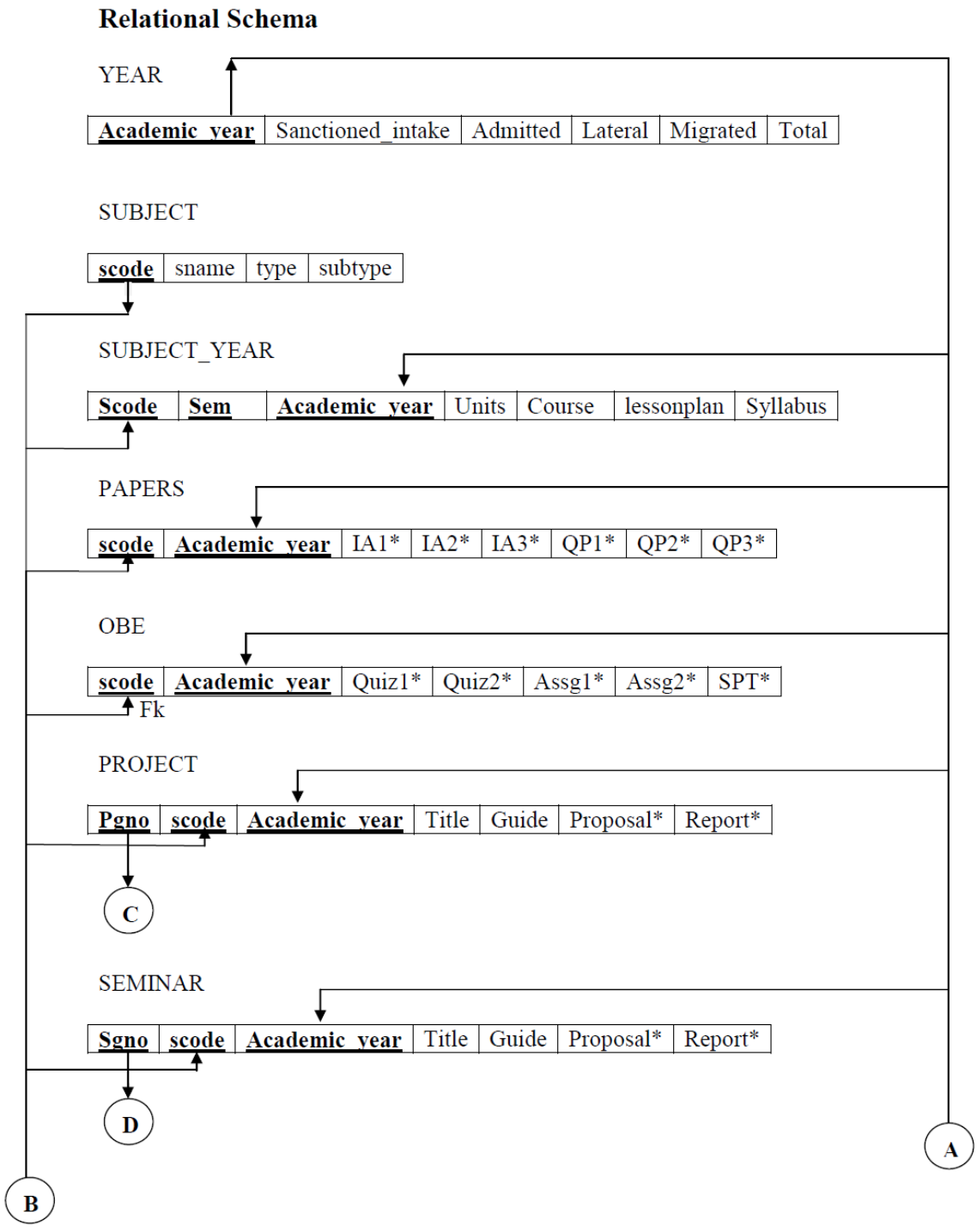


Fig 7.2: Relational Schema

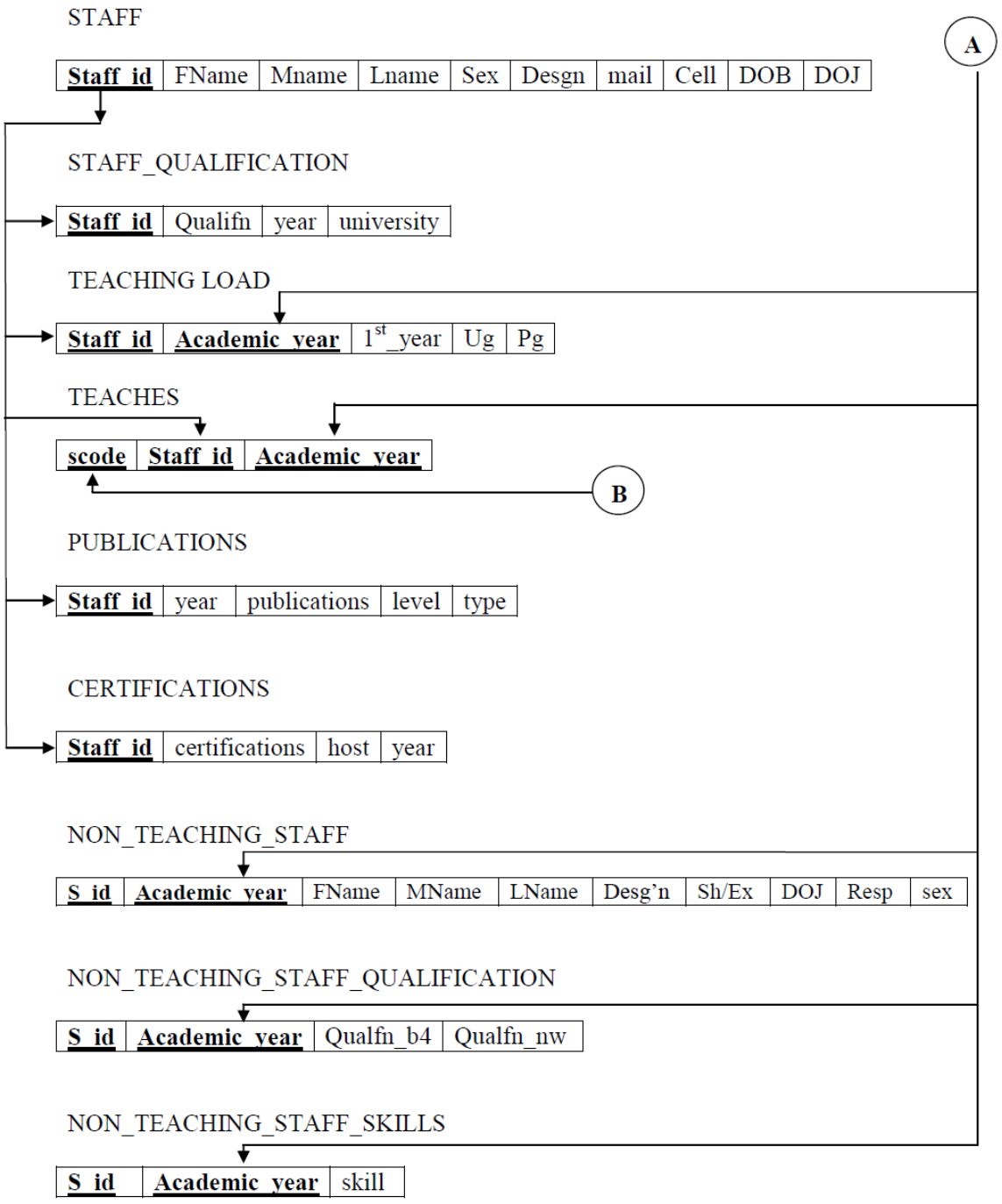


Fig 7.2: Relational Schema (continued)

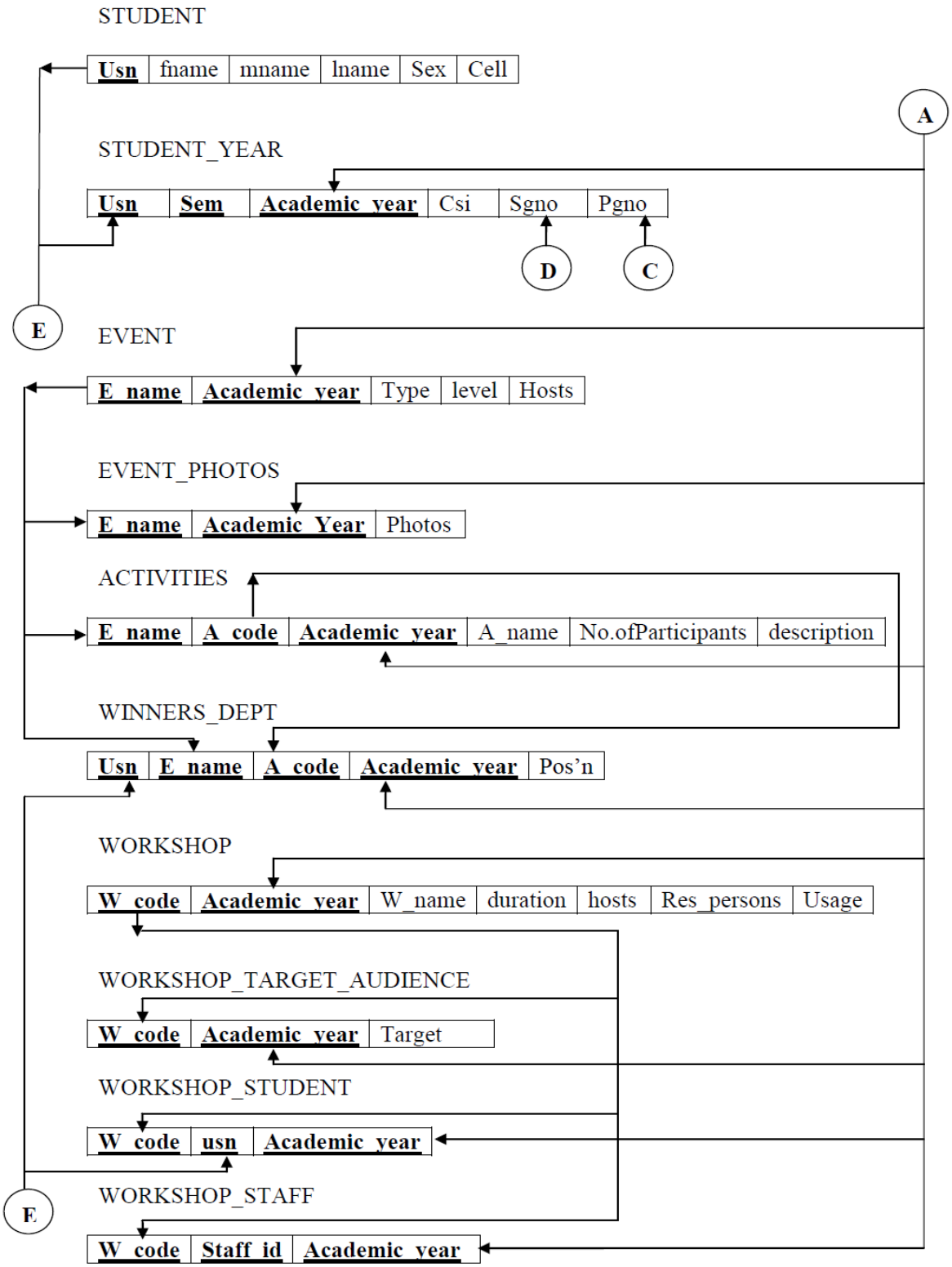


Fig 7.2: Relational Schema (continued)

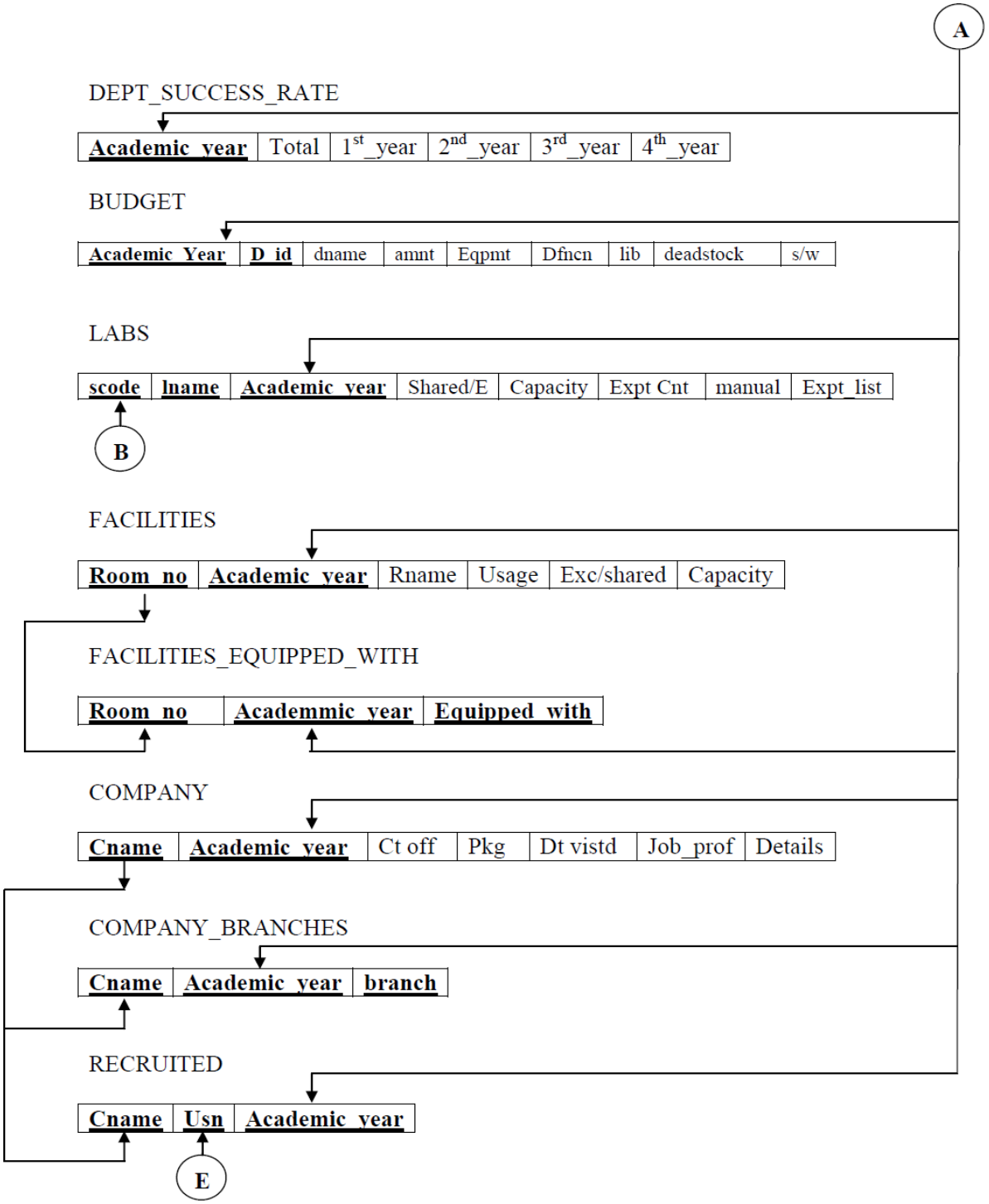


Fig 7.2: Relational Schema (continued)

7.2.3 Normalization

Normalization of data can be considered as a process of analyzing the given relation schemas based on their Functional Dependencies and primary keys to achieve the desirable properties of

- Minimizing redundancy
- Minimizing the insertion, deletion and update anomalies.

First Normal Form

It states that the domain of an attribute must include only atomic values and the value of any attribute in a tuple must be single value from the domain of that attribute.
The following table Student was as follows.

Usn	Name	Sem	Academic_year	Sex	Cell	Csi	P_grp.no	S_grpno

As one can see, it's not in First normal form. So we divided the composite attributes to form individual attributes.

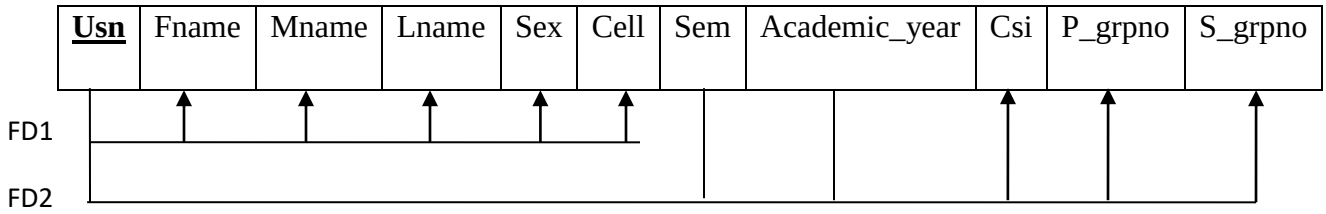
Usn	Fname	Mname	Lname	Sex	Cell	Sem	Academic_year	Csi	P_grpno	S_grpno

As can be seen, all the attributes are atomic and hence it's in First Normal Form.

Second Normal Form

A relation schema R is in second normal form (2NF) if every non-prime attribute A in R is fully functionally dependent on the primary key.
Where, Full functional dependency states that in a FD Y -> Z where removal of any attribute from Y means the FD does not hold any more.

After applying the first normal form, we had all the attributes as atomic values. Next, we checked the table for functional dependencies. We found that the attributes “Fname, Mname, Lname, Sex and Cell” were dependent solely on the primary key “Usn” while the attributes “Csi, P_grpno, S_grpno” were dependent not only on “Usn”, but also on “Academic_year” and “Sem”.



Hence we had to split the table in two. The first table remained as “Student” with “Usn” as the primary key while the second table was named as “Student_year” with “Usn, Sem and Academic_year” as the primary key.

Usn	Fname	Mname	Lname	Sex	Cell

Usn	**Sem**	**Academic_year**	Csi	SGrp_No	PGrp_No

Third Normal Form

A relation schema R is in third normal form (3NF) if it is in 2NF *and* no non-prime attribute A in R is transitively dependent on the primary key.

Where, Transitive functional dependency states that in a FD X -> Z that can be derived from two FDs X -> Y and Y -> Z

For Example, These two tables above don’t possess any transitive dependencies. Hence they are in Third Normal Form.

7.3 Database Table Diagram

The Database Designer is a visual tool that allows you to design and visualize a database to which you are connected. When designing a database, you can use Database Designer to create, edit, or delete tables, columns, keys, indexes, relationships, and constraints. To visualize a database, you can create one or more diagrams illustrating some or all of the tables, columns, keys, and relationships in it.

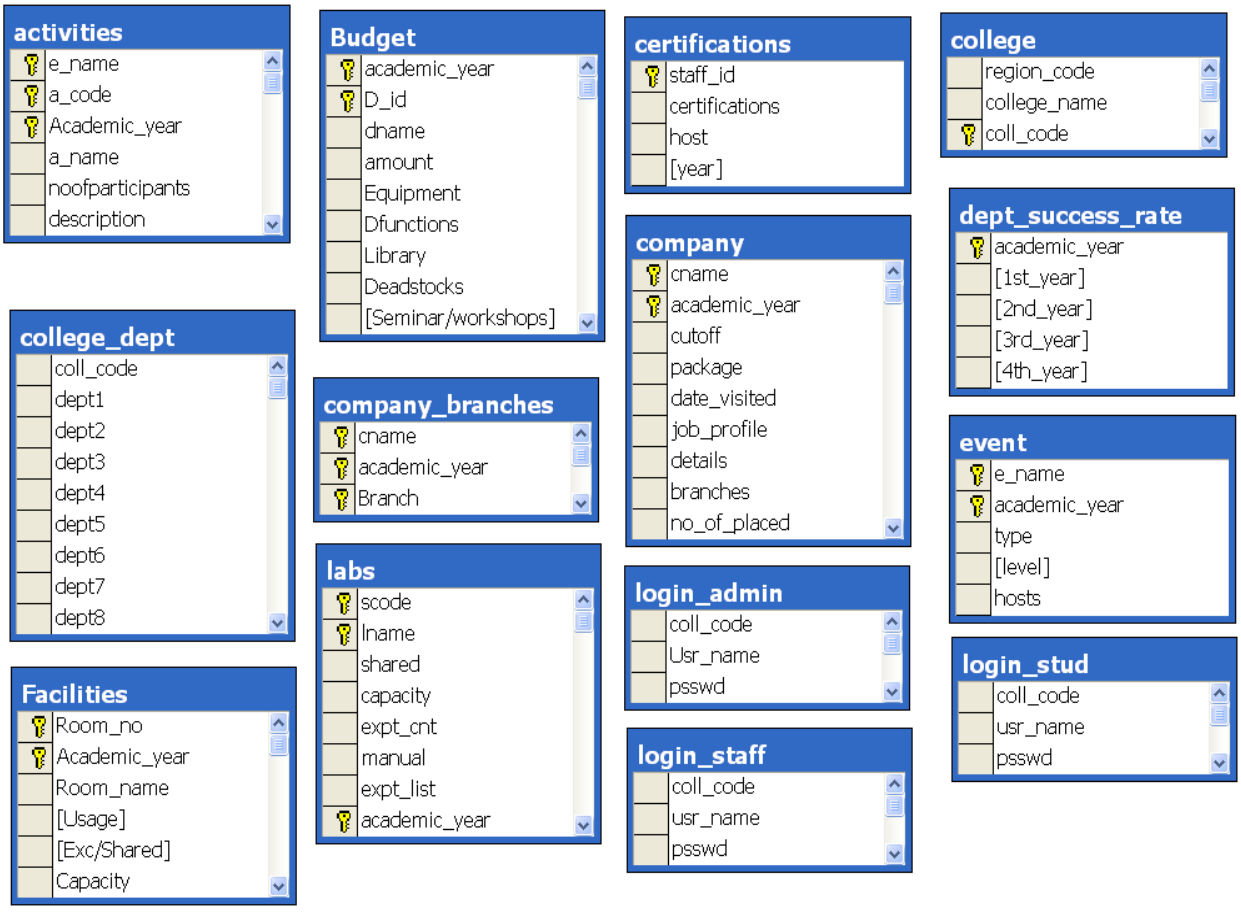


Fig 7.3: Database Table Diagram

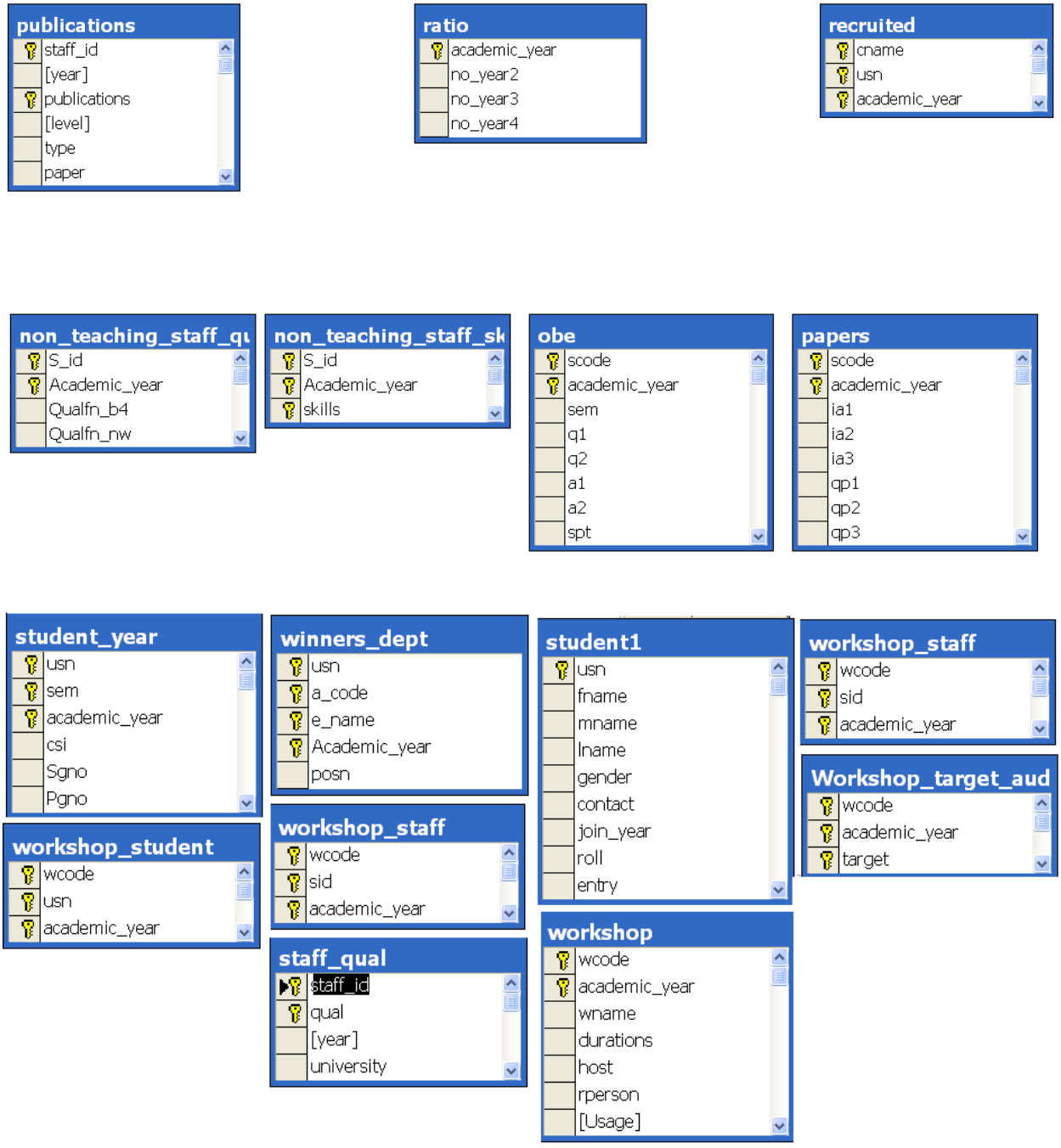


Fig 7.3: Database Table Diagram (continued)

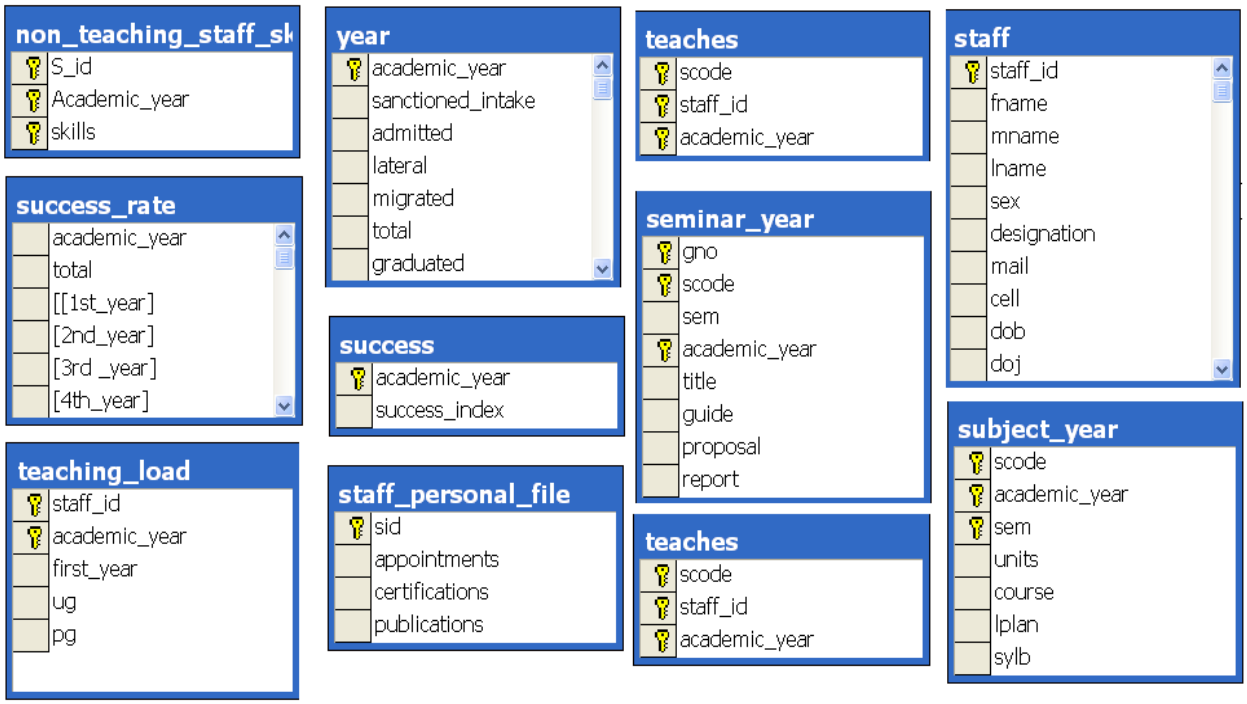


Fig 7.3: Database Table Diagram (continued)

7.4 GUI Design

Home page :

This is the home page of our website which acts as a login page. Here, a user can login as an Admin or Staff or Student, each user having his own password.

Fig 7.4: Login Page

ADMIN Home Page:

This is Admin home page. Here he can Add, Delete and Update the information of student, staff, subjects, department logistics etc and also generate necessary NBA reports.
By clicking on the links provided in sidetab,respective form will be displayed.

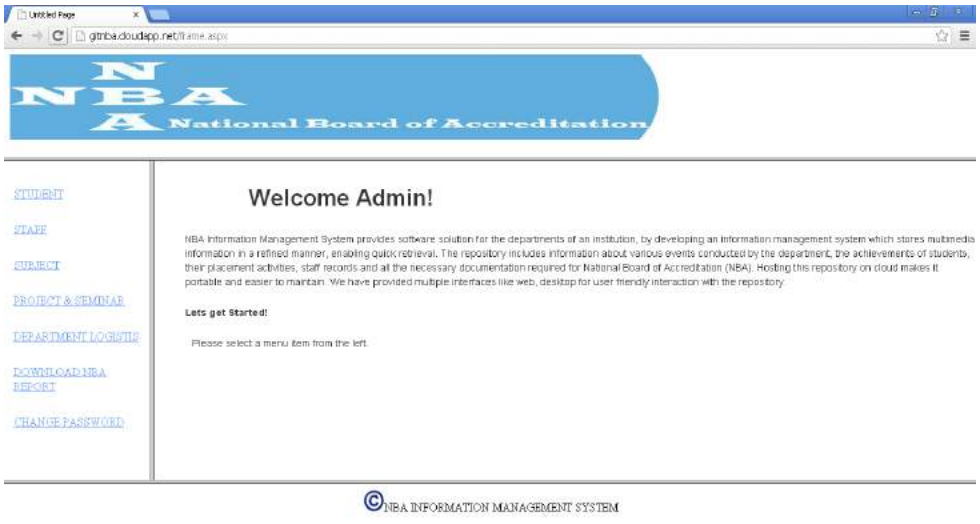


Fig 7.5: Admin Home Page

STAFF Home Page:

This is Staff Home page. The staff can view their personal file. They can insert update and delete IA papers, University question papers and obe related documents. They can also view the projects and seminar proposals and reports uploaded bythe students.
By clicking on the links provided in sidetab,respective form will be displayed.

Fig 7.6: Staff Home Page

STUDENT Home Page:

This is the Student home page. Students can upload and view seminar and project proposals and reports, for their respective batches.

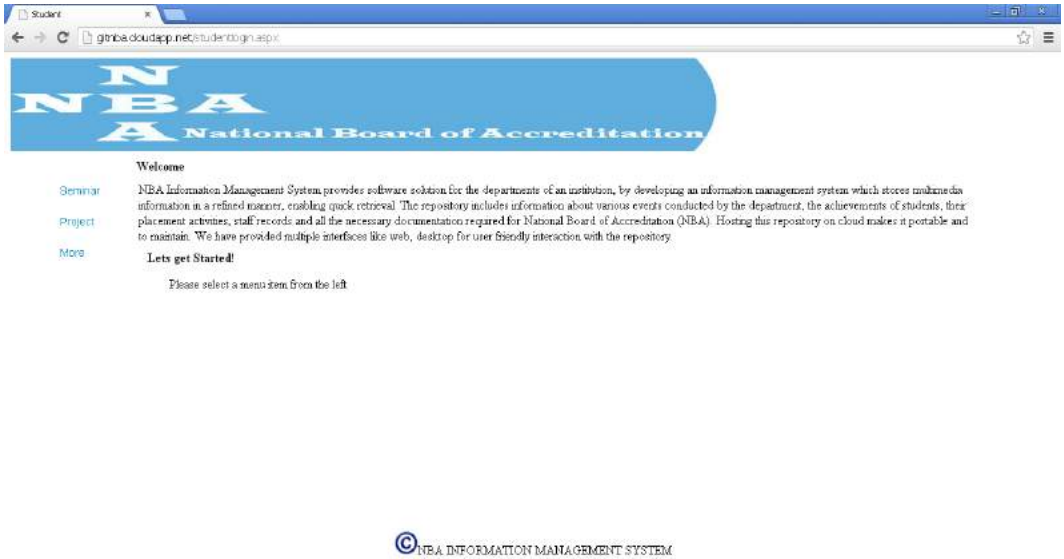


Fig 7.7: Student Home page

NBA Report Generation:

Numerous Reports are generated such as Success Index, Success Rate, Student-Staff qualification etc.

By clicking on the link, the selected report will be generated.

Fig 7.8: Reports for NBA

Staff Personal Profile:

This form displays the Certifications,Publicatons of the Staff based on staff id.

Fig 7.9: Staff Personal File

Uploading Papers :

This form is used to upload IA papers, Question Paper, OBE documents.

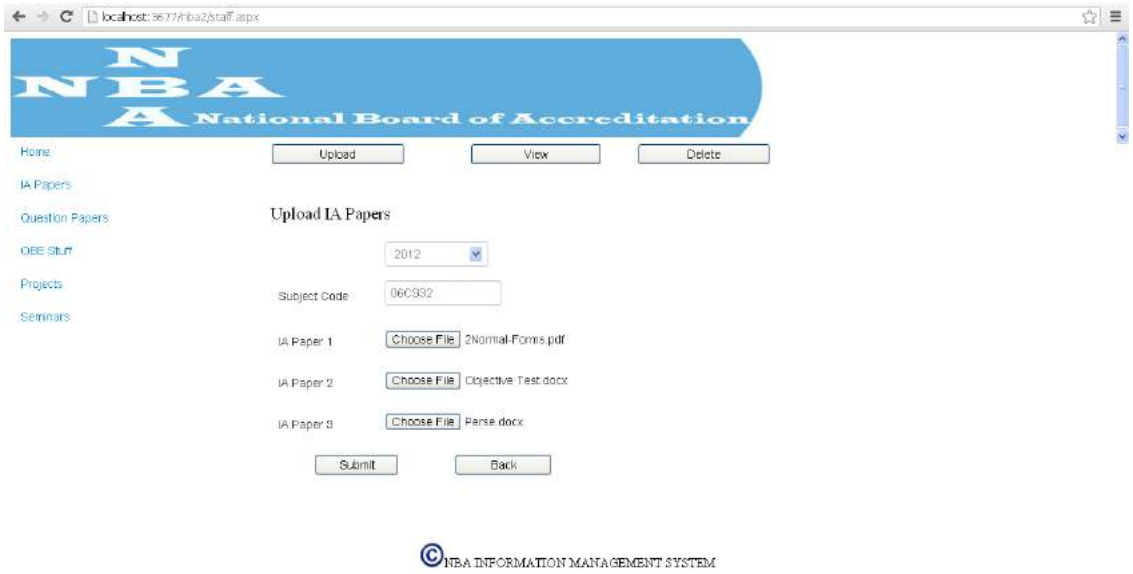


Fig 7.10: Staff uploading IA papers

Subject's Documents :

Contains Course Plan,Lesson Plan,Syllabus for a particular subject.

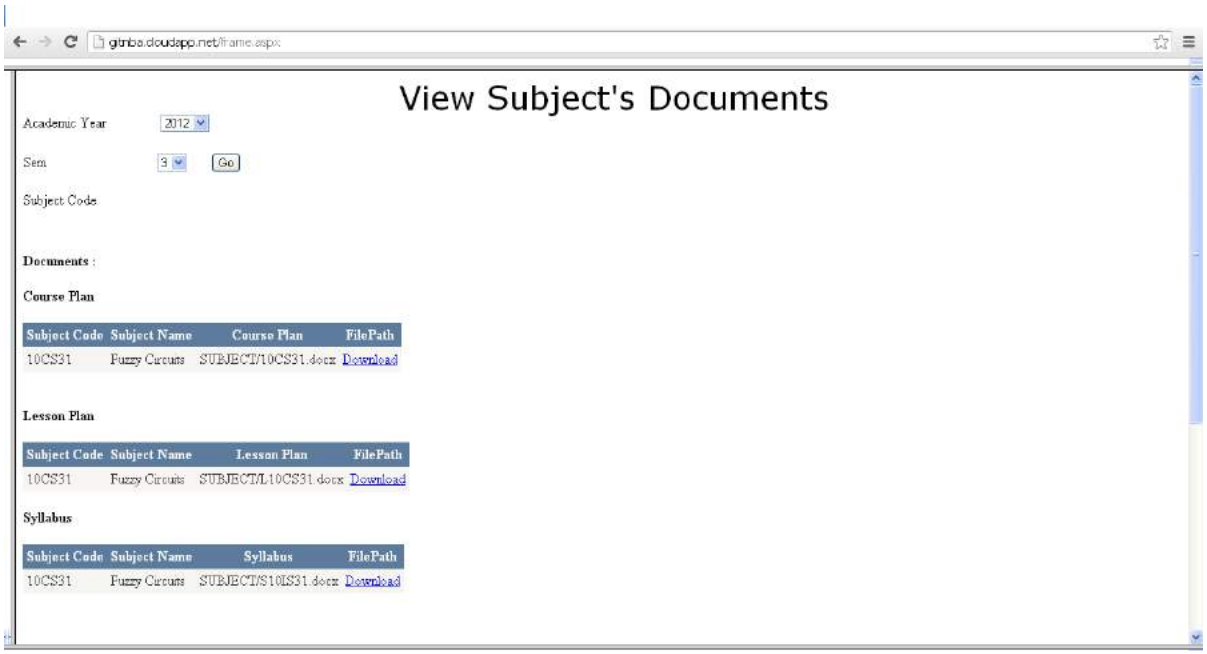


Fig 7.11: Downloading Subject's documents ie syllabus, course plan and lesson plan.

Display Student :

Displays the student information based on the academic year and semester.

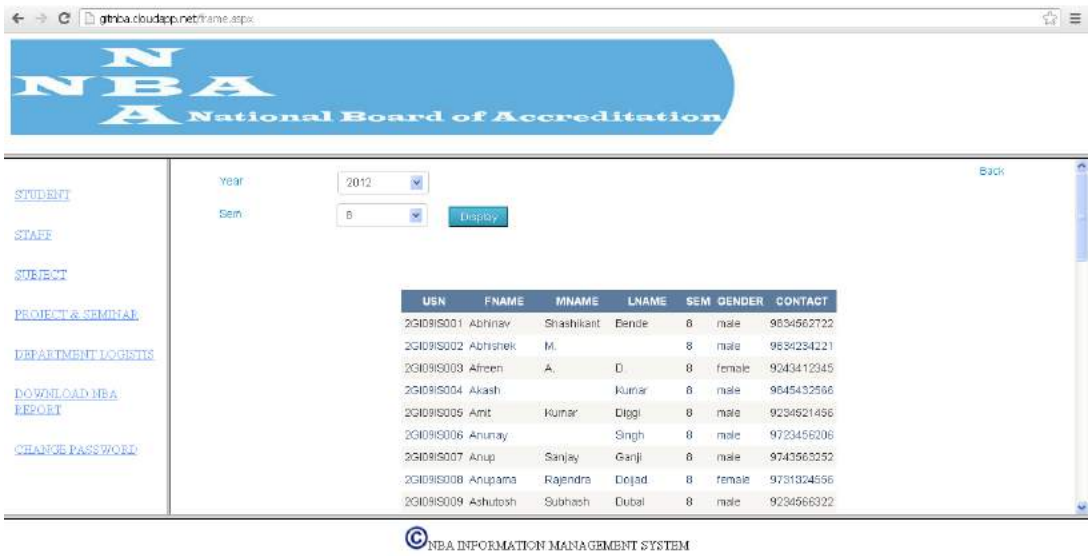


Fig 7.12: Displaying Students year and sem wise.

Staff Details :

This form is used to to generate staff informatiion such as Staff Id,Name,Designation,Staff Qualification,Teaching Load.

Fig 7.13: Displaying Staff Information

Chapter 8

Working and Implementation

The Working and Implementation of our project is described as below:

8.1 User Interface Navigation

Shown below is the home page of our website http:/gitnba.cloudapp.net. The website provides the option of logging as the Administrator or Student or Staff.

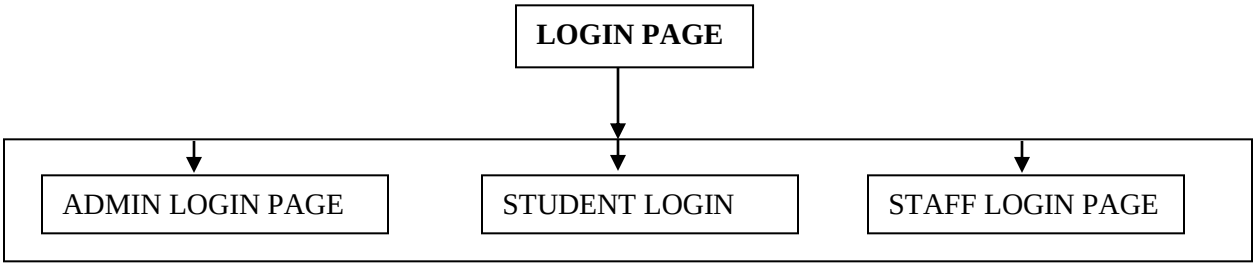


Fig: 8.1 Login Page

This is the page showing the options for the Admin after the Admin logs in. Admin can choose which page to open. The admin can go into Student module where he/she can enter student details or update them. Similar functionalities exist for the Staff Module. The admin can change various details related to Subject module. Admin can also view the other modules shown below. Another privilege residing with the Admin is of viewing the reports generated for the NBA Accreditation process.

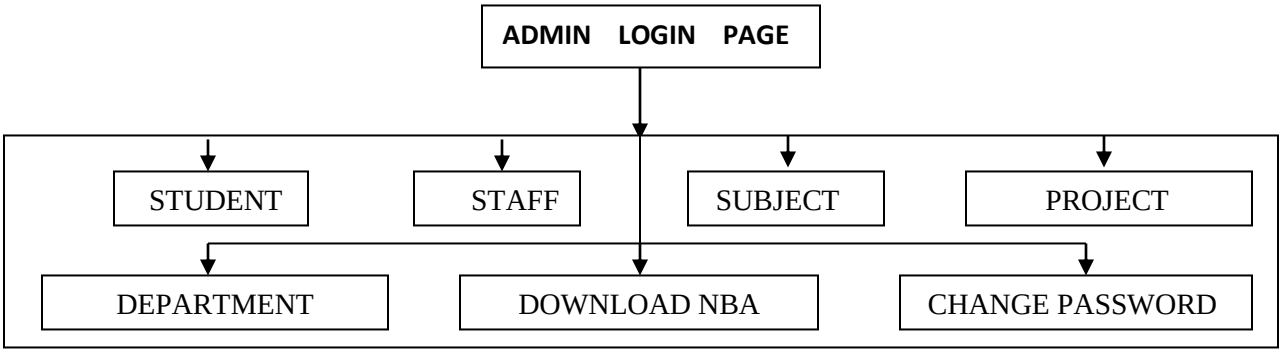


Fig: 8.2 Admin Login Page

This diagram shows the student module for the Admin in-depth. The options available in the student module are elaborated here. To display a student, the admin needs to select the academic year of the student followed by the semester and finally click on the display button. Also, to update an existing student, first the admin needs to select the academic year if the student followed by the semester. Secondly, the admin needs to select the USN of the student to select a particular student. Then the values for the student can be updated. Similar steps exist for adding a new student and deleting an existing one.

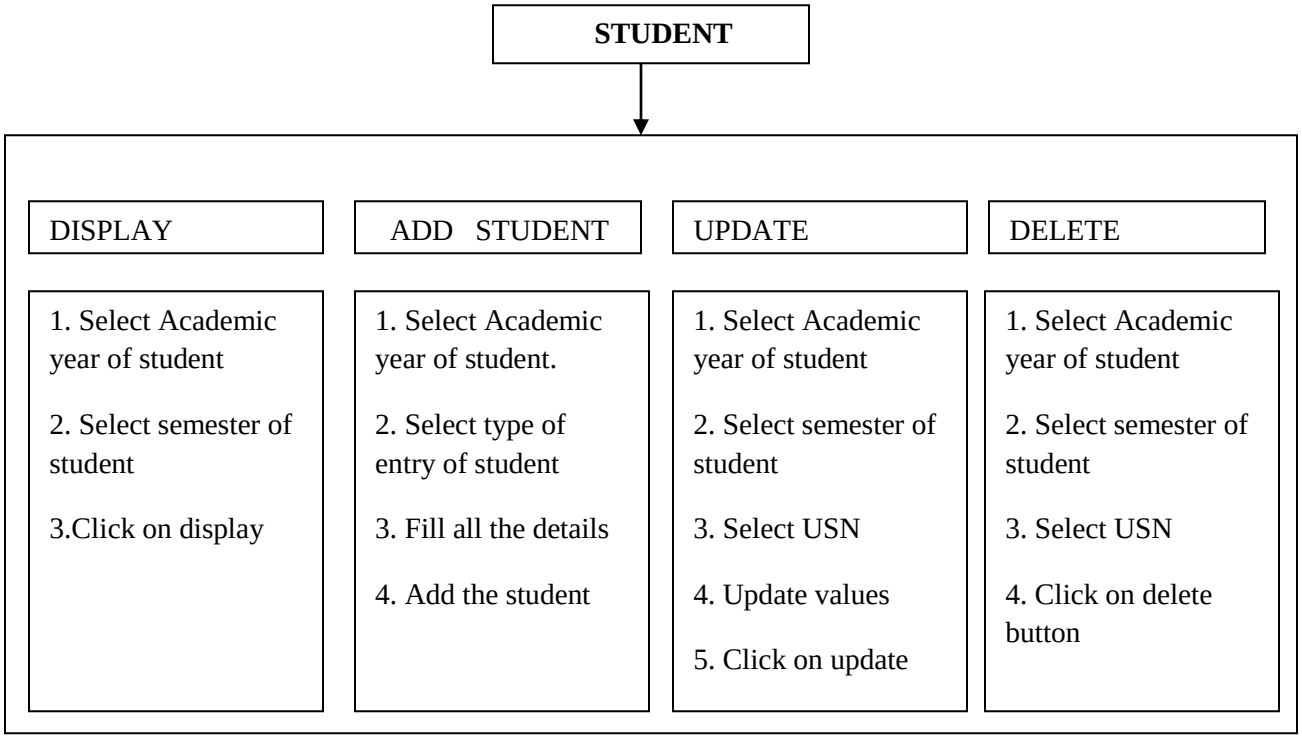


Fig: 8.3 Student Page

The staff module is divided further into teaching staff and non teaching staff modules. On choosing the staff module, the admin can then choose between these two modules

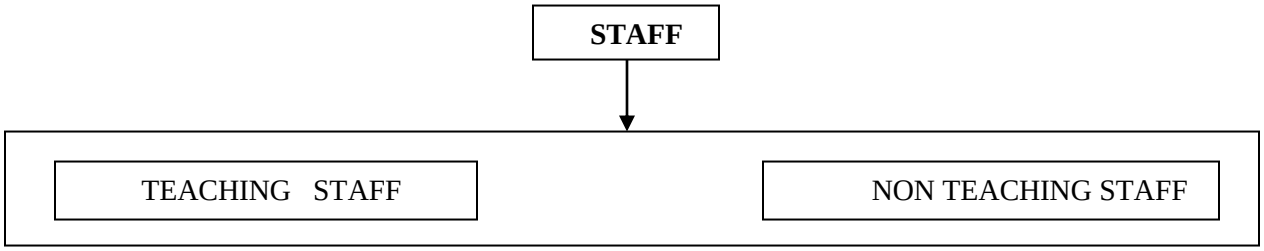


Fig: 8.4 Staff Page

Shown below are the in-depth details of the teaching staff module. Here also, the admin can insert, update and delete the details of teaching staff members. Further, the teaching staff module has qualification and teaching load modules where the admin can update the details of the qualification & teaching load for a particular teaching staff member for a particular academic year.

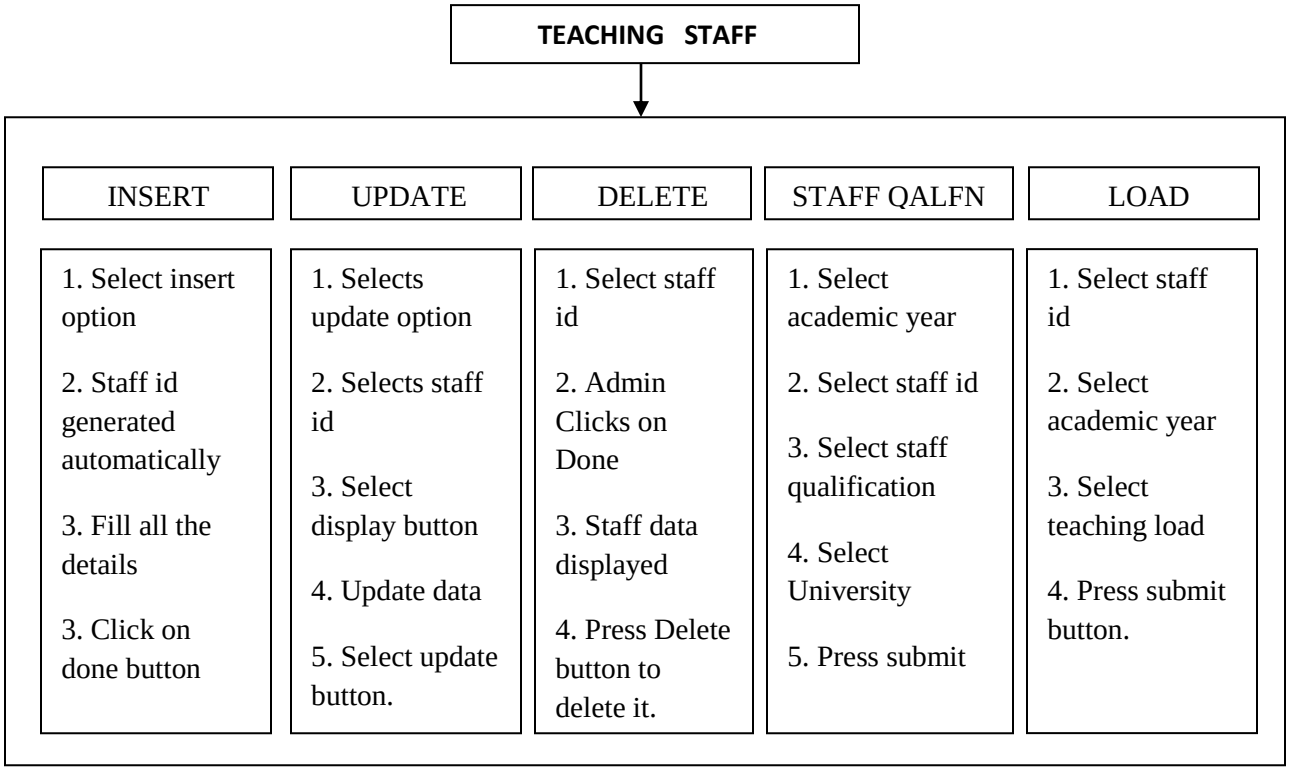
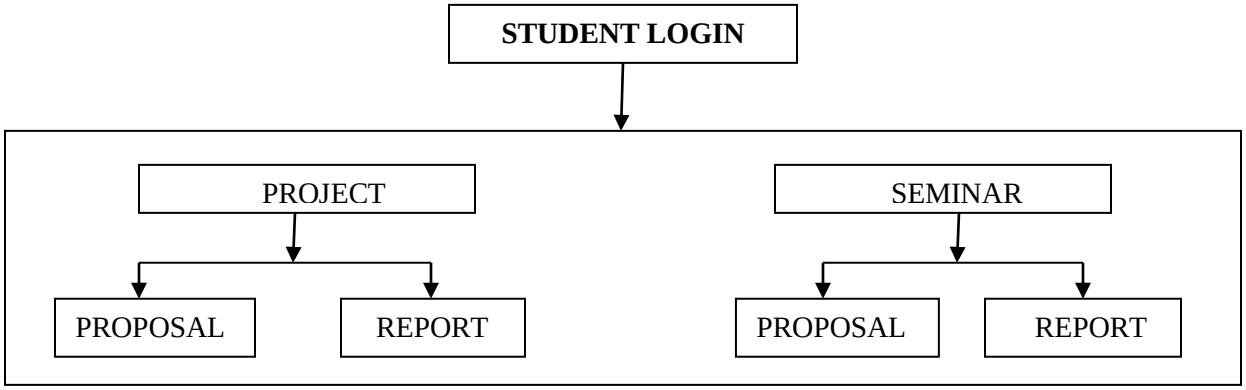


Fig: 8.5 Teaching Staff Page

Next, we have the home page for those users who are logged in as students. Students have the option to choose between the Project and the Seminar modules. Within these two modules, the students have the option of choosing proposal or report.



Shown below is the in-depth details of the Proposal and Report options which follow the same pattern. Within them, a student can upload, view or delete the selected document. To upload the document, the student first need the specify the academic year, project group number, title of the project/seminar, the guide and semester. Then the student needs to choose a document to upload. A student can view and download the documents uploaded by his/her peers. To delete the uploaded file, the student needs to specify a password so as to avoid a student deleting someone else's files.

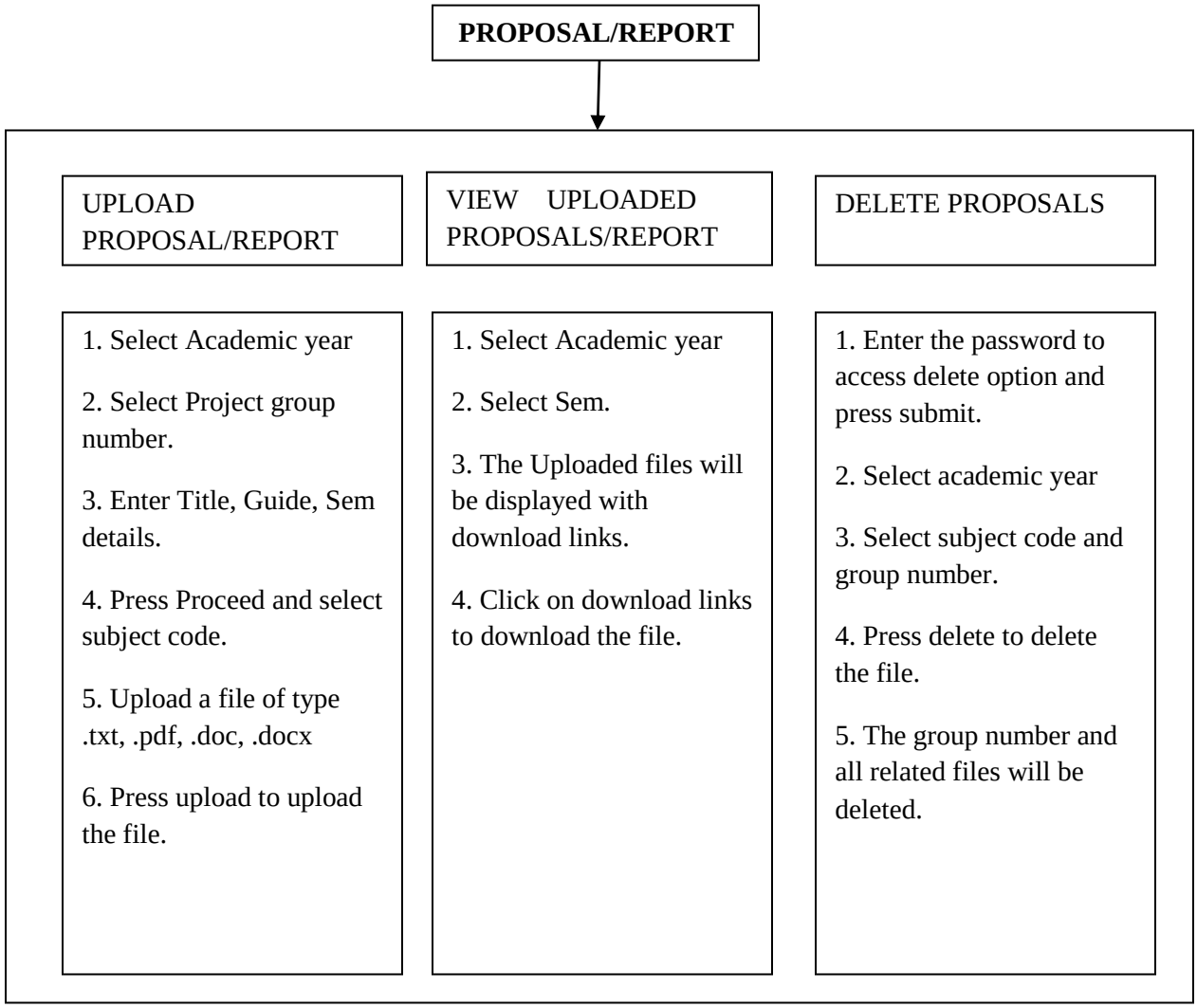


Fig: 8.6 Proposal/Report Page

Going further, we have the home page for those users who are logged in as teaching staff members. They can choose between the modules for OBE documents, IA papers, Question

Papers etc. They can also view the uploaded proposals and reports for both Project and the Seminar by the students.

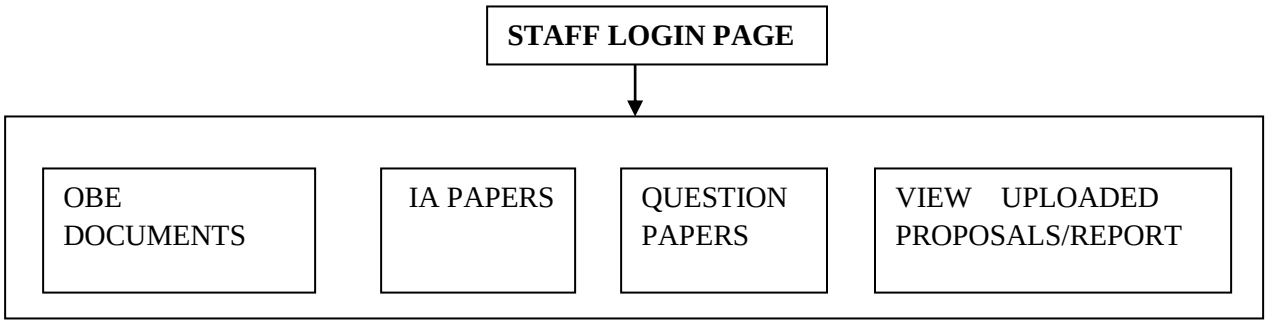


Fig: 8.7 Staff Login Page

OBE documents module is shown below. The teaching staff members can upload, view or delete the OBE documents. The basic process for all the three involves specifying the academic year, semester and the subject code. This is a must as it helps pinpoint the document and where it belongs.

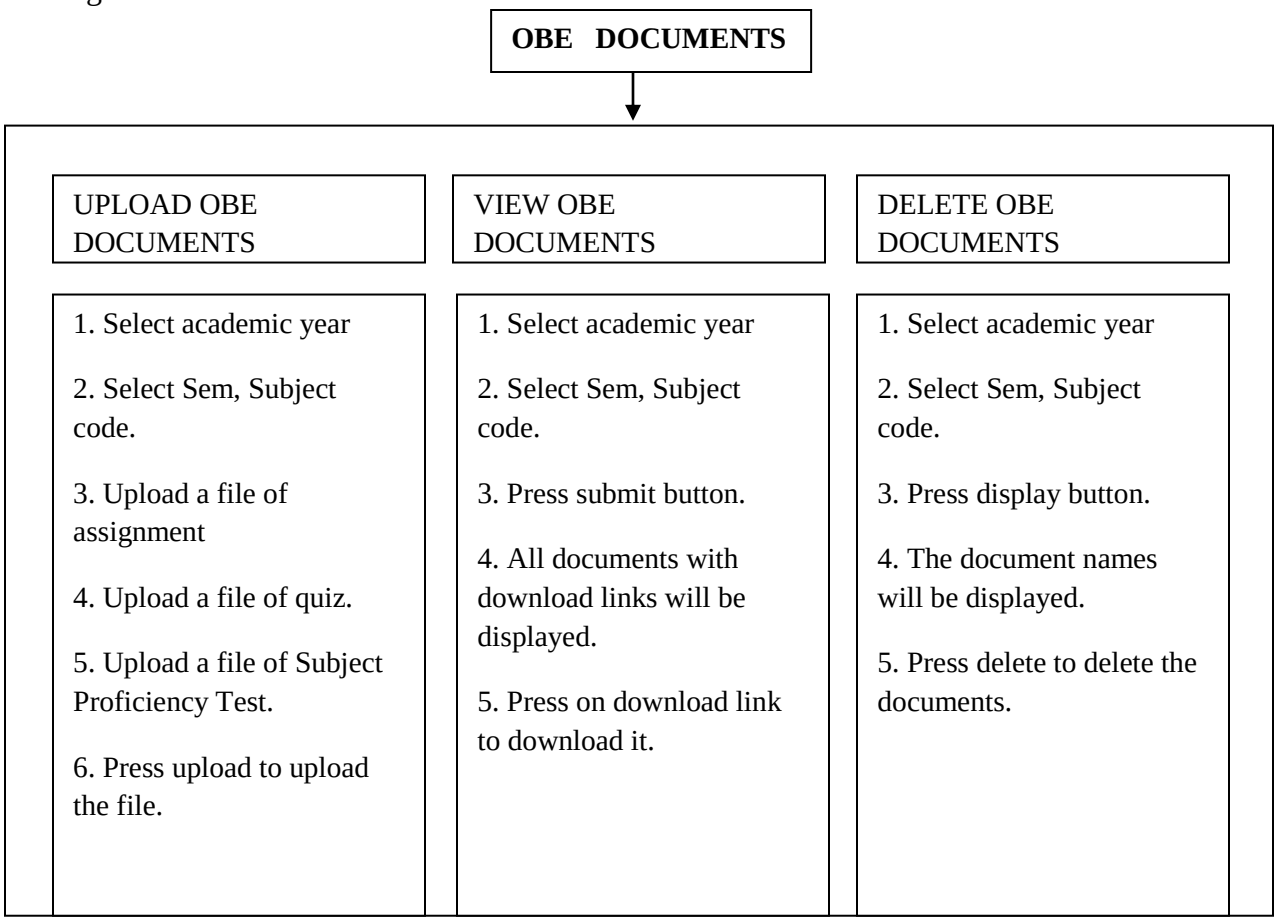


Fig: 8.8 OBE Document Page

Next detailed module is the one for IA/University question papers. Similar to OBE documents, one can upload, view and delete these documents. To pinpoint a document, the teaching staff members need to specify academic year, subject code and the semester one by one. In case of upload, one needs to choose a file while viewing the papers lets us download them.

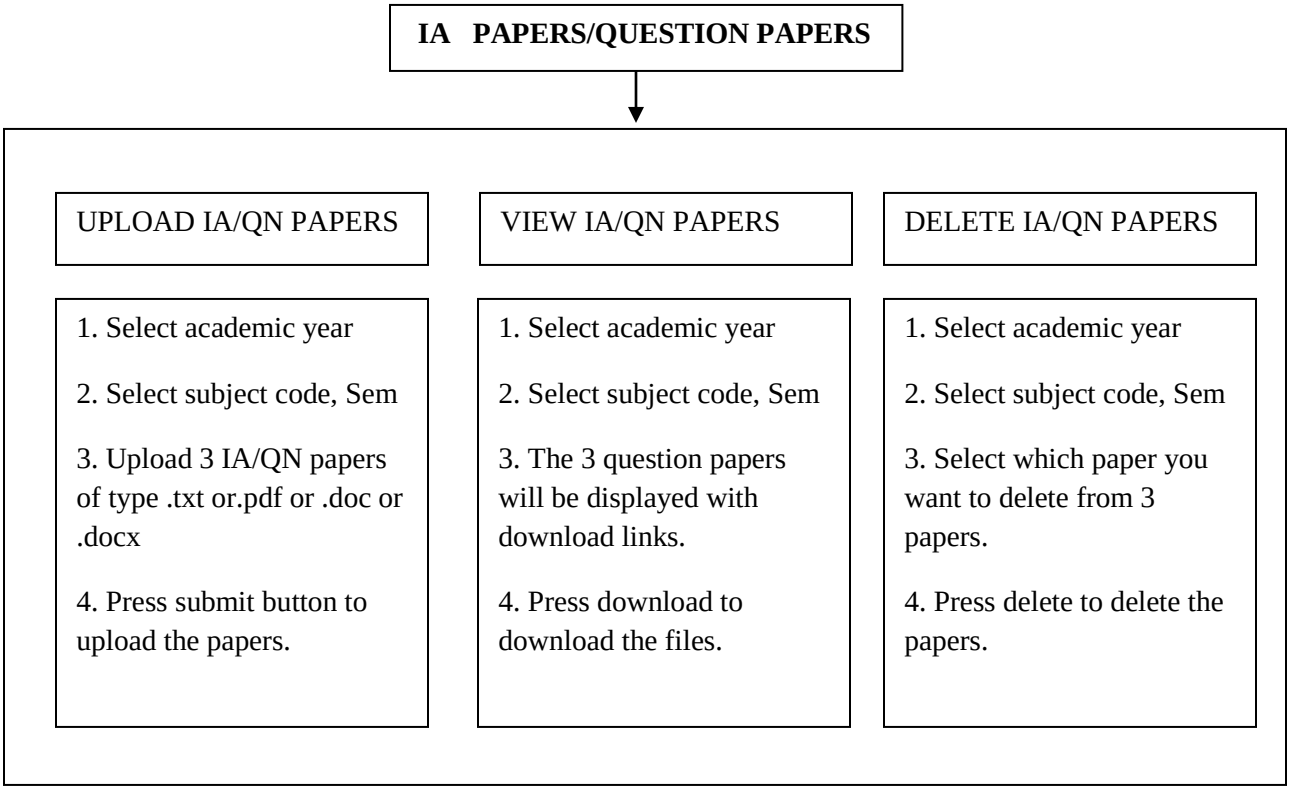


Fig: 8.9 IA / Question Papers Page

Chapter 9

Testing

9.1 Verification and Validation

Verification and Validation (V & V) is the name given to the checking and analysis process that ensures that software conforms to its specification and meets the needs of the customers who are paying for that software.

- 'Validation: Are we building the right product?'
- 'Verification: Are we building the product right?'

Verification involves checking that the software conforms to its specification. We should check that the system meets its specified functional and non-functional requirements.

Validation is a more general process. We should ensure that the software meets the expectations of the customer.

Within the V & V process, two techniques of system checking and analysis may be used:

1. Software inspections analyze and check system representations such as the requirements document, design diagrams and the program source code.
2. Software testing involves executing an implementation of the software with test data and examining the outputs of the software and its operational behavior to check that it is performing as required. Testing is a dynamic technique of verification and validation because it works with an executable representation of the system.

The testing phase our project included the following tests.

9.1.1 Defect testing

The goal of defect testing is to expose latent defects in a software system before the system is delivered. This contrasts with validation testing which is intended to demonstrate that system meets its specification. Validation testing requires the system to perform correctly using given acceptance test cases. A successful defect test is a test which causes the system to perform incorrectly and hence exposes a defect. This emphasizes an important fact about testing. It demonstrates the presence, not the absence, of program faults.

This software was exhaustively tested for defects and all the defects have been successfully countered.

9.1.2 Black-box testing

Functional or black-box testing is an approach to testing where the tests are derived from the program or component specification. The system is a 'black-box' whose behavior can only be determined by studying its inputs and the related outputs. Another name for this is functional testing because the tester is only concerned with the functionality and not the implementation of the software.

This software was tested repeatedly by supplying many inputs and observing the output. In each case it has performed up to the mark.

9.1.3 Structural testing

Structural testing is an approach to testing where the tests are derived from knowledge of the software's structure and implementation. This approach is sometimes called 'white-box' testing or 'clear-box' testing to distinguish it from black-box testing. Structural testing is usually applied to relatively small programs units such as sub-routines or the operations associated with an object. As the name implies, the tester can analyze the code and use knowledge about the structure of a component to derive test data. The analysis of the code can be used to find how many test cases needed to guarantee that all the statements in the program or component are executed to least once during the testing process.

Each of the small modules in the software was tested independently and satisfactory results were obtained. The different modules that were tested were subject master, student master, marks details, notifications and accessing module.

9.1.4 Integration testing

Once individual program components have been tested, they must be integrated to create a partial or complete system. This integration process involves building the system and testing the resultant system for problems that arise from component interactions. Integration tests should be developed from the system specification and integration testing should begin as soon as usable versions of the system components are available.

After all the modules had been tested independently, the whole system was integrated and tested again. The entire system included different modules that had to be tested exhaustively to get the desired results.

9.1.5 Interface testing

Interface testing takes place when modules or sub-systems are integrated to create larger systems. Each module or sub-system has a defined interface which is called by other program components. The objective of interface testing is to detect faults which may have been introduced into the system because of interface errors or invalid assumptions about the interfaces.

After all the modules had been tested independently, the whole system was integrated and tested again. The entire system included different modules that had to be tested exhaustively to get the desired results.

9.2 Test Cases

A test case in software engineering is a set of conditions or variables under which a tester will determine whether an application or software system is working correctly.

9.2.1 Link Test :

Login as

- Admin
- Staff
- Student

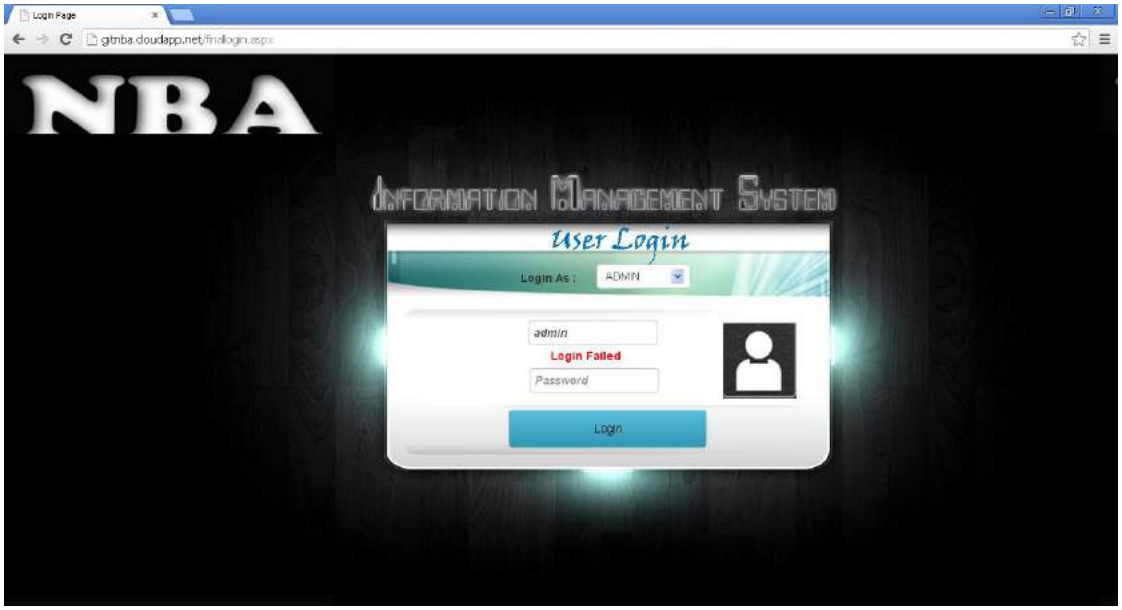


Fig 9.1: User login Link Test

9.2.2 Authentication Test

- User should enter Username and Password.
- If entered Username and Password does not match,Error Message is displayed .

9.2.3 Field Validation Tests

Following are the some of the field validation tests that are covered:

- Admin should enter all the details of the student. If any of the field is empty, a Message Box is displayed with appropriate message.
- Names should be Alphabetic.
- Contact Numbers should be Numeric and should be of 10 digits.
- Emails should be in proper format.
- Date of Birth should be greater than Date of Joining.
- Passwords should not be only 1 character.

For Example:

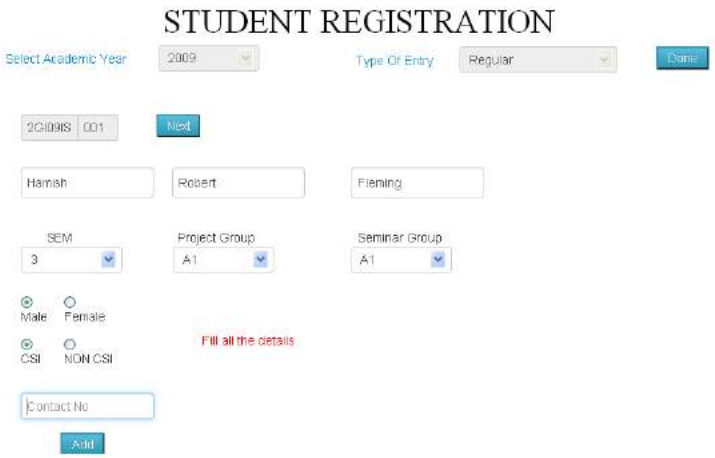


Fig 9.2: Student Registration form with appropriate field validations

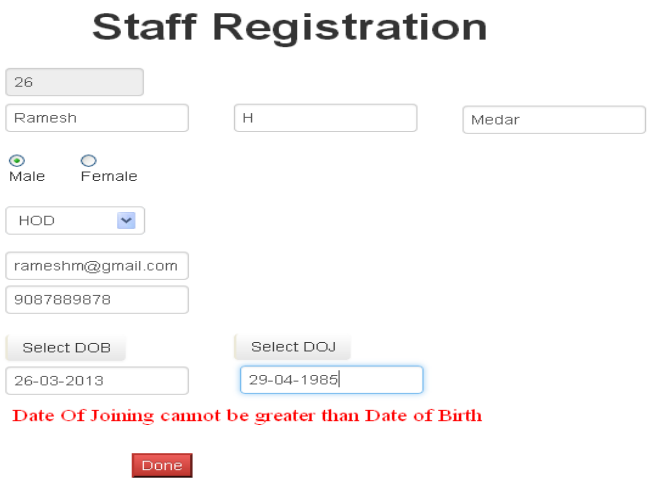


Fig 9.3: Staff Registration form with appropriate field validations

9.2.4 Database Validations

- Each Seminar Group can have max of 2 students.
- Each Project Group can have max of 3 students (from IIIrd to VIIth sem) and max of 4 students (VIIIth sem)
- Student should upload documents for an existing group.
- Each Department can have a single HOD.
- Uploaded files should be in .DOC,.DOCX or .PDF format.

For Example:

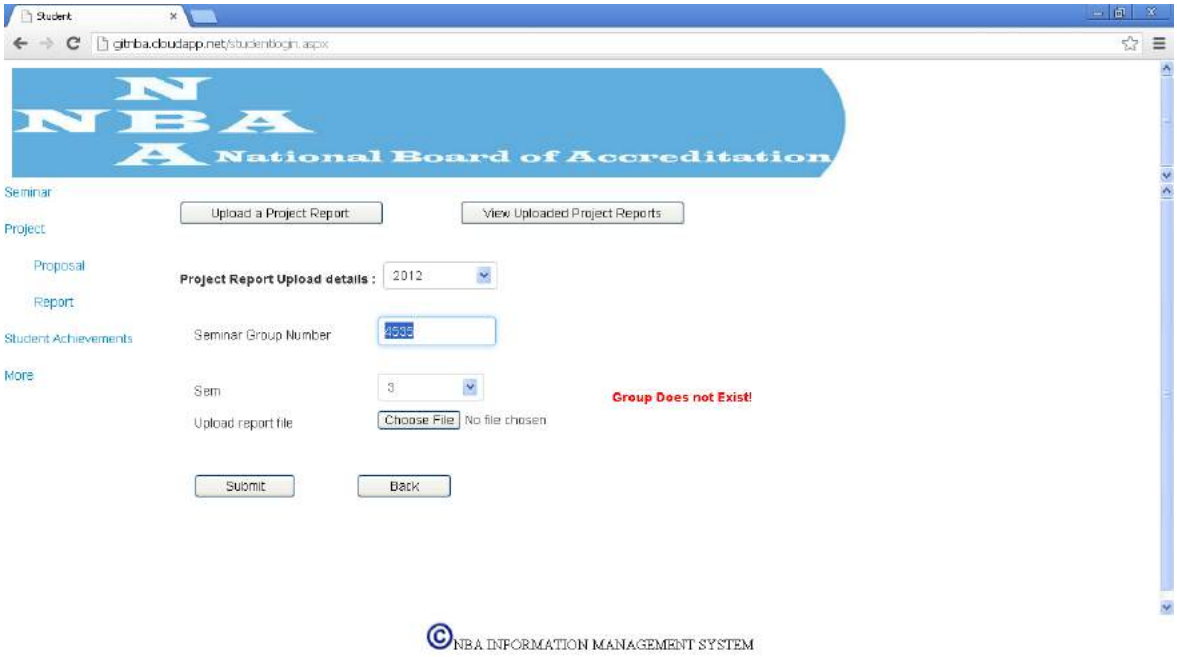


Fig 9.4: Student should upload project documents to an existing group.

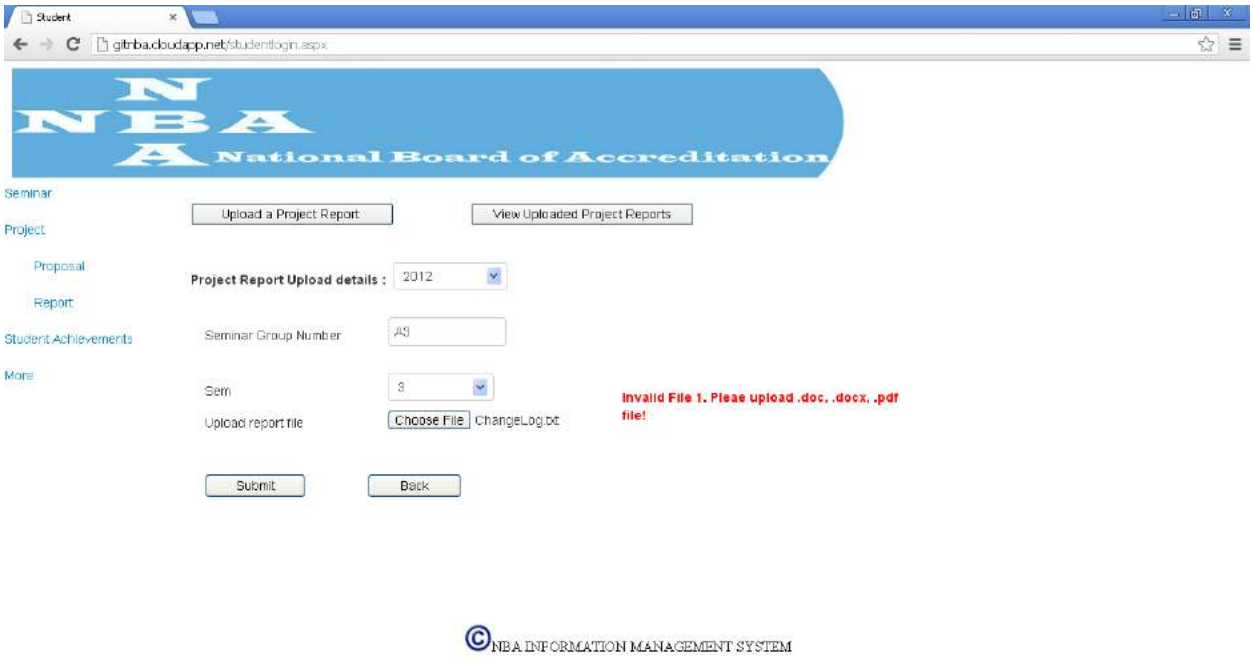


Fig 9.5: Only .doc,.docx,.pdf files are valid to be uploaded.

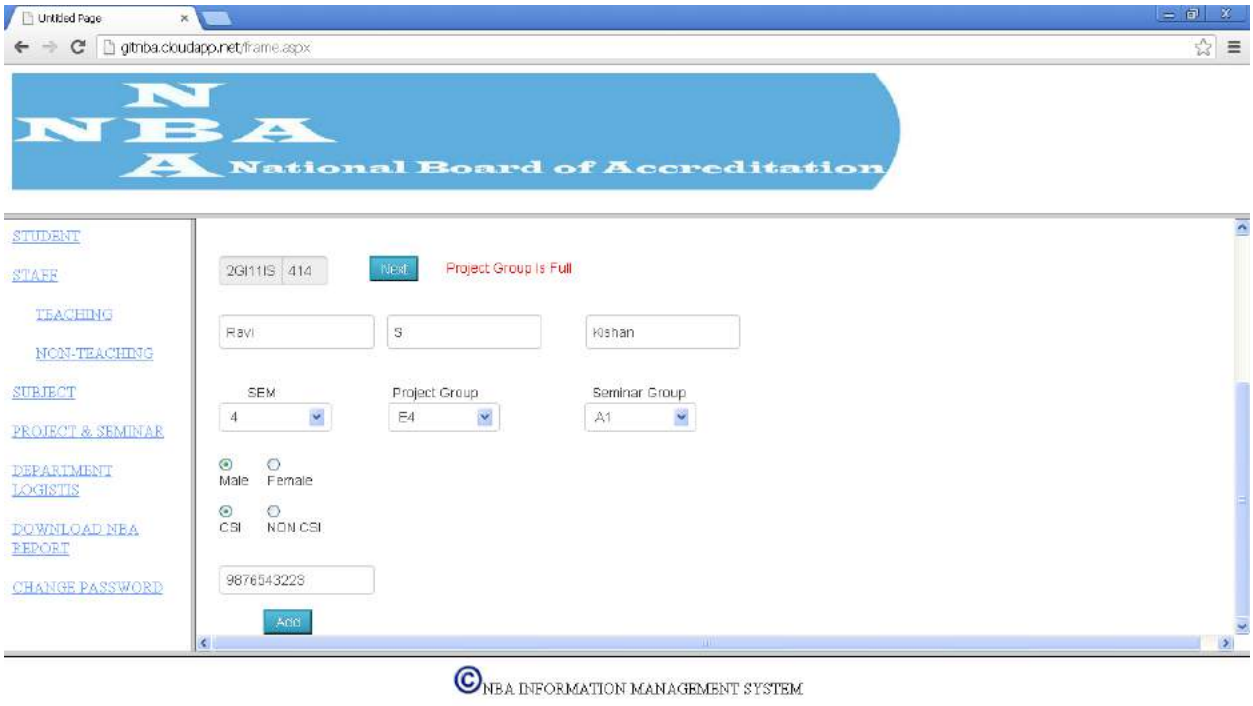


Fig 9.6: Project groups can have max of 4 students (IVth sem in this case)

Chapter 10

Results

The results of our application are the various reports that are required by the NBA. These results are in the form of reports which are created using crystal reports application.

Following are the some of the reports that can be generated.

Admission Intake:

Shown below is the report for the admission intake for the department. The total number of students is calculated based on sanctioned intake, actual number and lateral entry.

Admission Intake in the Programme

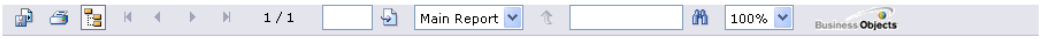
22-05-2013

Where,
Actual No = No. of students Admitted - No of students Migrated

CAY	Sanctioned Intake	Actual no	Lateral	Total
2006	60	58	14	72
2007	60	58	10	68
2008	60	60	8	68
2009	60	57	9	66
2010	60	60	15	75
2011	60	54	12	69
2012	60	59	9	69

Success Rate:

This report denotes the success rate of the department for all years in reverse chronological order for the past seven batches. The Success rate is found using parameters like students who progressed to 2nd year, those who progressed to 3rd year etc.



SUCCESS RATE

22-05-2013

Data For past 7 batches :
(Successfully completed implies Zero Backlogs)

Year of Entry (In reverse Chronological Order)	No of Students (Admitted in 1st year + admitted lateral)	No of Students successfully completed 1st Year	No of Students successfully completed 2nd Year	No of Students successfully completed 3rd Year	No of Students successfully completed 4th Year
2012	69				
2011	69	69	47		
2010	75	40	37	30	
2009	66	37	36	31	26
2008	68	37	36	31	30
2007	68	45	35	32	32
2006	72	51	43	39	35

Faculty Qualification Index :

Shown below is the report for faculty qualification index (FQI). It shows the assessment based on the qualification possessed by the faculty of the department. As shown below, more the faculty having M.Tech and PhD degrees, higher is the FQI.

22-05-2013

FACULTY QUALIFICATION INDEX

Assessment = 3 * FQI
Where FQI = Faculty Qualification Index
= (10 * x + 6 * y + 4 * z) / N
Where x = Number of Faculty Members with Ph. D.
y = Number of Faculty Members with M. E / M. Tech

Academic Year	X	Y	Z	N	FQI	Assessment
2012	8	9	0	19	7.05	21.16

FQI = 7.05

Assessment = 21.16

HOD SIGN

Laboratories Used:

This report lists all the laboratories used by the department and the labs are grouped by academic year. The report lists the subject code and the lab name and the experiment count for that subject. The lab can be shared or exclusive and its capacity is mentioned too.

Laboratories Used

22-05-2013

Academic Year	Subject Code	Lab Name	Shared	Capacity	Experiment Count
2010	10CSL37	DATA STRUCTURES WITH C/C++	Shared	40	14
	10CSL47	DESIGN AND ANALYSIS OF ALGORITHMS	Shared	40	12
	10CSL48	MICROPROCESSORS LABORATORY	Shared	40	12
	10CSL57	DATABASE APPLICATIONS LABORATORY	Shared	40	5
	10CSL58	SYSTEM SOFTWARE & OPERATING	Shared	40	12
	10CSL77	Networks Laboratory	Shared	40	12
	10CSL78	Web Programming Laboratory	Shared	40	14
	10ISL67	FILE STRUCTURES LABORATORY	Shared	40	12
2011	10ISL67	FILE STRUCTURES LABORATORY	Shared	40	12

Qualifications of Teaching Staff:

This report shows the qualifications of the teaching staff. It also shows the year and university for each qualification possessed by the teaching staff member.

24-05-2013

Qualifications of Teaching Staff

Staff ID	Staff Name	Qualification	Year	University
003	K K TANGOD	B.E.	2000	VTU
		M.Tech	2004	VTU
		UG	2009	vtu
004	A H Kulkarni	B.E.	1996	KUD
		M.Tech	2002	VTU
005	S K Madi	B.E.	1994	KUD
		M.Tech	2002	VTU
006	R J Kadkol	B.E.	1993	KUD
		M.Tech	2006	VTU
007	P S Upparmani	B.E.	1991	KUD
		M.Tech	2006	VTU
		PG	2009	KUD
008	S R Mangalwede	B.E.	1994	KUD
		M.Tech	2002	VTU
		Ph.D	2012	VTU

Facilities:

This report lists all the facilities used by the department and the labs are grouped by academic year. The report lists the room number, the room name and its use. The particular facility can be shared or exclusive and its capacity is specified too.

Facilities

22-05-2013

Academic Year	Room Number	Room Name	Usage	Exclusive/ Shared	Capacity
2011	A-S1	Classroom	Class room for 2nd Year	Exclusive	70
	A-S2	Classroom	Class room for 2nd Year	Exclusive	70
	A-T13	Classroom	Class room for 2nd Year	Exclusive	70
	A-T15	classroom	Elective/Seminar	Exclusive	70
	Faculty rooms(n)	Staff_Cabins	Staff Room	Exclusive	...
	Meeting room	HOD Room	HOD Room	Exclusive	15
	Seminar Room	classroom	A-T15	Exclusive	100
	Tutorial rooms	classroom	A-T15	Exclusive	100
2010	Tutorial rooms	classroom	A-T15	Exclusive	100
	Seminar Room	classroom	A-T15	Exclusive	100
	Meeting room	HOD Room	HOD Room	Exclusive	15
	Faculty rooms(n)	Staff_Cabins	Staff Room	Exclusive	...

Conclusion

The NBA software management system has been successfully implemented and deployed as cloud platform and tested for its correctness. All the modules have been extensively tested for their correct operation and results have been noted. The software meets all the functional and non functional requirements as listed in chapter 3. The end users have expressed their satisfaction over its ease of use, convenience and its use in overriding repeated work. However further improvements can be done to this project as described in the future scope.

Future Scope

Following are some of the future changes and improvements that can be made to this project:

- Current application is using 90 days trail version of the cloud service. On buying a pay per use plan, the application can make use of more resources as required.
- The application can be further extended to include reports based on marks based data as well.
- The website can be improved to avoid facing compatibility issues with old browsers like IE5 as well as for viewing it in mobile phones.

APPENDIX I

Technologies Used:

1. Crystal Reports 2010

Crystal Reports has been a part of Visual Basic since 1993, and a part of Visual Studio since its first release in 2002. Crystal Reports has been a very successful component of these products. With the release of Visual Studio 2010, SAP and Microsoft mutually decided to change how they deliver this important component to the .NET developer community going forward.

Crystal Reports for Visual Studio 2010 contains many new features compared to Crystal Reports Basic for Visual Studio 2008. SAP Developer Network goes into more detail on the new features and how they benefit report designers, .NET developers, and report consumers.

Crystal Reports for Visual Studio 2010 makes a .NET application more powerful by delivering more insight with fewer reports. It is completely free, embeds into the familiar Visual Studio environment, and benefits all users - Report Designers, .NET Developers, and Report Consumers.

2. JavaScript

JavaScript (JS) is an interpreted computer programming language. It was originally implemented as part of web browsers so that client-side scripts could interact with the user, control the browser, communicate asynchronously, and alter the document content that was displayed. Now though it has many uses involving popular game development and the creation of applications.

JavaScript is a prototype-based scripting language that is dynamic, weakly typed, and has first-class functions. Its syntax was influenced by the language C. JavaScript copies many names and naming conventions from Java, but the two languages are otherwise unrelated and have very different semantics. The key design principles within JavaScript are taken from the self and Scheme programming languages. It is a multi-paradigm language, supporting object-oriented, imperative, and functional programming styles.

3. Microsoft SQL Server 2012

Microsoft's SQL Server 2012, the latest version of its SQL Server platform, is in many ways a response to the demands of the cloud and "big data" applications. SQL Server has evolved from its humble beginnings as a relatively easy-to-use and less expensive database (compared with high-end commercial enterprise databases) for small-scale transactions and analysis into an organizational workhorse, especially for agencies looking to leverage cloud architecture.

SQL Server 2012 comes in three license configurations: Standard edition, Enterprise edition and the new Business Intelligence edition. The Standard version, which is oriented toward smaller organizations, lacks some of the new enterprise features but still has many features that make it well suited to a number of applications for organizations of many sizes. The Business Intelligence edition delivers many of the high-end data analysis features of the Enterprise edition, minus a few enhanced features.

4. Active Server Pages (ASP)

Web pages with the *.asp* file extension use ASP, although some web sites disguise their choice of scripting language for security purposes (e.g. still using the more common *.htm* or *.html* extension). Pages with the *.aspx* extension use compiled ASP.NET (based on Microsoft's .NET Framework), which makes them faster and more robust than server-side scripting in ASP, which is interpreted at run-time; however, ASP.NET pages may still include some ASP scripting. The introduction of ASP.NET led to use of the term *Classic ASP* for the original technology.

Programmers write most ASP pages using VBScript, but any other Active Scripting engine can be selected instead with the `@Language` directive or the <script language="manu" runat="server"> syntax. JScript (Microsoft's implementation of ECMAScript) is the other language that is usually available. PerlScript (a derivative of Perl) and others are available as third-party installable Active Scripting engines.

5. .NET Platform 3.5

NET Framework 3.5 builds incrementally on the new features added in .NET Framework 3.0. For example, feature sets in Windows Workflow Foundation (WF), Windows Communication Foundation (WCF), Windows Presentation Foundation (WPF) and Windows CardSpace. In addition, .NET Framework 3.5 contains a number of new features in several technology areas which have been added as new assemblies to avoid breaking changes.

They include the following:

- Deep integration of Language Integrated Query (LINQ) and data awareness. This new feature will let you write code written in LINQ-enabled languages to filter, enumerate, and create projections of several types of SQL data, collections, XML, and DataSets by using the same syntax.
- ASP.NET AJAX lets you create more efficient, more interactive and highly personalized Web experiences that work across all the most popular browsers.
- New Web protocol support for building WCF services including AJAX, JSON, REST, POX, RSS, ATOM, and several new WS-* standards.
- Full tooling support in Visual Studio 2008 for WF, WCF, and WPF, including the new workflow-enabled services technology.
- New classes in .NET Framework 3.5 base class library (BCL) that address many common customer requests.

6. Visual Basic

Visual Basic (VB) is a programming environment and language created by Microsoft, which based the language on an existing version for beginning programmers, BASIC. Microsoft developed VB to make it easy for developers to write programs for the Windows operating system. Prior to VB, programmers wrote programs in the C or C++ programming language, which had no built-in support for accessing Windows functions. VB does have such support as part of its object-oriented programming approach. Object-oriented programming involves the creation and application of virtual blueprints called classes, and objects made from those classes.

APPENDIX II

SQL SERVER CONNECTIVITY

This is the code used to connect the front end of our project to Microsoft SQL Server 2012. It is stored in an .vb file. We declare variables for the connection string, the data reader as well as the SQL command. The connection string is stored in a String variable in which we specify the data source, type of integrated security as well as the initial catalog. SqlConnection and SqlCommand fucntions are used to connect to the data source and execute the command respectively.

```
Imports System.Data.SqlClient

Public Module Module1

    Public con As SqlConnection
    Public reader As SqlDataReader
    Public com As SqlCommand
    Public str_con As String = "Data Source='" & My.Computer.Name &
    "';Integrated security=SSPI;Initial catalog=nba"
    Public sql As String
    Public count As Integer
    Public username, yesno As String


    Public Sub connect()
        con = New SqlConnection(str_con)
    End Sub



    Public Sub executequery()
        com = New SqlCommand(sql, con)
        reader = com.ExecuteReader
    End Sub

End Module
```

REFERENCES

[1] Fundamentals of *Database Systems* – By Elmasri and Navathe, 5th Edition, Addison-Wesley, 2007

[2] *.NET* The *Complete Reference* - By Jeffrey R. Shapiro

[3] *Beginning ASP.NET 4 In C# And VB 1st Edition* - By Imar Spaanjaars Published by Wiley India Pvt Ltd (2010).

[4] *Web Technologies Black Book : HTML, JavaScript, PHP, Java, JSP, XML and AJAX* - Published by Dreamtech Press (2012).

[5] *To the Cloud: Cloud Powering an Enterprise 1st* Edition - By Pankaj Arora Published by Tata Mcgraw Hill Education Pvt Ltd (2012)

[6] *Cloud Computing: Principles and Paradigms* - By James Broberg, Andrzej Goscinski, Rajkumar Buyya Published by Wiley (2013)

[7] Paper on Cloud Computing Learning by Panagiotis Kalagiakos, Panagiotis Karampelas Department of Information Technology Hellenic American

[8] Paper on Analysis and Research of Cloud Computing System Instance – By Shufen Zhang Shuai Zhang Xuebin Chen Shangzhuo Wu, Hebei Polytechnic University

[9] *www.msdn.microsoft.com/* - The Microsoft Developer Network

[10] *www.windowsazure.com/* - Azure Cloud Services

[11] *www.stackoverflow.com/asp.net*

[12] *www.codeproject.com/KB/aspnet/*

[13] *www.nbaind.org/* - National Board of Accreditation

[14] *http://en.wikipedia.org/wiki/Asp.net*

[15] *http://en.wikipedia.org/wiki/Internet_Information_Services*

[16] *http://www.w3schools.com/css/css_syntax.asp*