

PROFESSIONAL TRAINING REPORT

at

Sathyabama Institute of Science and Technology

(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award
of Bachelor of Technology Degree in

Information Technology

By

Santhosh.y
(Reg.No.37120086)



DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR,
RAJIV GANDHI SALAI,
CHENNAI - 600119

DECEMBER 2020



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with 'A' grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119.

Phone: 044 - 2450 3150 / 3151 / 3152 / 3154 / 3155 Fax: 044 - 2450 2344
www.sathyabama.ac.in



DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of **SANTHOSH.Y (REG NO.37120086)** who underwent the professional training in “**MATERIAL FEEDBACK SYSTEM**” under our supervision from AUGUST 2020 to JANUARY 2021.

Internal Guide

Mr. P.SARAVANAN, M.E.,(Ph.D).,

Head of the Department

Dr. SUBHASHINI, M.E.,Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **SANTHOSH.Y (REG NO.37120086)** hereby declare that the Professional Training Report on “**Material Feedback System**” done by me under the guidance **Mr. P. SARAVANAN,.M.E.,(Ph.D)** at Sathyabama Institute of Science and Technology is submitted in partial fulfilment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE:

PLACE:

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

The satisfaction and elation that accompany the successful completion of any task would be incomplete without the mention of the people who have made it possible. It is our great privilege to express our gratitude and respect to all those who have guided me during the course of my Professional Training.

I convey my thanks to **Dr.T.SASIKALA,M.E.,Ph.D.**, Dean, School of computing and **Dr. R. SUBHASHINI, M.E., Ph.D.**, Head of the Department, Department of Information Technology for providing necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my project guide **Mr.P.SARAVANAN.,M.E.,(Ph.D)** for his valuable guidance, suggestions and constant Encouragement which paved my way to successful completion of my project.

I convey my thanks to the management and all the teaching staffs and non teaching staffs of the Computer science and engineering department who were helpful in many ways for the completion of the project.

Also, I thank my **Parents** for supporting me in the completion of the Professional Training.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	vii
	LIST OF FIGURES	ix
	LIST OF TABLES	x
	LIST OF ABBREVIATIONS	x
1	INTRODUCTION	5
	1.1. ABOUT THE PROJECT	
	1.2. EXISTING SYSTEM	
	1.3. PROPOSED SYSTEM	
2	SYSTEM REQUIREMENTS	7
	2.2 SOFTWARE REQUIREMENTS	
	2.3 HARDWARE REQUIREMENTS	
3	REQUIREMENT ANALYSIS	8
4	UML DIAGRAMS	12
5	TESTING	34
6	IMPLEMENTATION	40
7	SCREEN SHOTS	46
8	SOURCE CODE	61
9	CONCLUSION AND FUTURE WORK	64
10	REFERENCES	65

LIST OF TABLES

TABLE NO.	TABLE NAME	PAGE NO.
2.1	HARDWARE REQUIREMENTS	3

ABSTRACT

This project titled **material feed back system** a web based application for the online implementation which can be used for a campus or organization for proving it for usage by all the people means to use it at any moment and at any time from any place. The project server administrator views the details of faculty and students and maintain their database in an efficient and effective manner so that their maintenance will be easy. The services providing for administrator are approving the faculty and students, maintain the personal details of faculty and students, add/delete the courses, consign a announcement regarding the course, view the feedbacks. The services provided to the faculty are the faculty can view and update their personal details, upload the content for a course, conduct a test for the students, and places key and results of a test, view the discussion board and feedbacks and reply to the students. The services provided to the students are view and update their details, view the content of a course, add/view the discussion, gives feedback to the faculty. Whenever anything is modified by the users, then the modified data, about the person who updated is maintained in the database. By this we are providing security to the users.

1. INTRODUCTION

1.1 ABOUTTHEPROJECT

This project titled **Online Course Portal for a Campus is** a web based application for the online implementation which can be used for a campus or organization for proving it for usage by all the people means to use it at any moment and at any time from any place.

The project server administrator views the details of faculty and students and maintain their database in an efficient and effective manner so that their maintenance will be easy.

The services providing for administrator are approving the faculty and students, maintain the personal details of faculty and students, add/delete the courses, consign a announcement regarding the course, view the feedbacks.

The services provided to the faculty are the faculty can view and update their personal details, upload the content for a course, conduct a test for the students, and places key and results of a test, view the discussion board and feedbacks and reply to the students.

The services provided to the students are view and update their details, view the content of a course, take a test and view the key and result of a exam, add/view the discussion, gives feedback to the faculty.

Whenever anything is modified by the users, then the modified data, about the person who updated is maintained in the database. By this we are providing security to the users.

Functional components:

There will be a Course Portal home page where there will be a registration link as well as a login screen is available. There are three types of users in the system:

- ❖ Administrator
- ❖ Faculty (for a course)
- ❖ Student (for a course)

Modules:

There will be a Course Portal home page where there will be a registration link as well as a login screen is available.

There are three modules in the system:

❖ Administrator

- ✓ Add faculty
- ✓ view registered students
- ✓ view announcements
- ✓ compose mail
- ✓ view mails
- ✓ add course and delete course.

❖ Faculty (for a course)

- ✓ Register
- ✓ Login
- ✓ Upload material
- ✓ Give announcements
- ✓ Send mail to admin
- ✓ View mails
- ✓ Clarify student doubts through discussion board

❖ Student (for a course)

- ✓ Register
- ✓ Login
- ✓ View courses
- ✓ View faculty
- ✓ Give feedback
- ✓ Ask doubts in Discussion board
- ✓ Download material

1.2. Existing System:

In the present course portal pre-registered usernames and passwords will be given to the faculty members. There is no on spot registration to the faculty members. Because of these pre-registrations the faculty members were constrained to certain number. So, only that pre-registered faculty member can upload the materials.

There is no proper self assessment test in the current course portal. Only sample test papers were uploaded in the portal and those were downloaded by the students. There is no time constraint in writing sample test in the present system.

Discussion board plays a vital role in clarifying the doubts for students which was lacked in the current scenario. Direct interaction of faculty and student will be helpful in clarifying the doubts through discussion board. A student can post a query at any time and reply can be given at any time.

Registered students or faculty will search about the new topics or new courses that were created. But they may not know latest updates done in the portal. The present existing system is lacking in publishing the latest announcements.

Faculty approval mechanism is lacking the current course portal. As there are no on spot registrations, administrator cannot approve the faculty in the current system.

Feedback also helps the faculty to know the proper usage of their materials that were uploaded. In the existing system there is no chance to give the feedback to any faculty and there is no chance of viewing the feedback by the administrator.

1.3. Proposed System:

In the proposed system faculty can register to the course portal at any time as there is no restriction in registration process. Here no pre-registered usernames and passwords will be given to the faculty as it was not constrained to certain number. Every registered faculty member can upload their own material. It makes the student to grab more content on a particular course.

In the developing system online self assessment test will be provided by the faculty. Timer indicates that the test should be completed in limited time span. The system will

automatically generates the random paper each time when the student takes the test and the faculty can view the test result.

The problem of discussion board was overcome in proposed system so that the students can easily clarify their doubts by direct interaction with faculty through discussions.

Faculty approval mechanism was the new idea that is going to implement in this proposed system to differentiate the faculty from students by comparing the registered faculty with faculty working in the organization.

The announcements make the portal active always. Latest updates in the portal help the users to know about the course management and the latest materials that were uploaded.

Feedback module helps the faculty to assess the usage of materials by the students. Administrator view the feedback given to all the faculty members and faculty can view their own feedback to any particular course.

2. SYSTEM REQUIREMENTS

2.1. Hardware Requirements:

- HDD : 500GB
- RAM : 2GB
- Processor : Dual core

2.2. Software Requirements:

- Operating System : Windows7
- Browsers : Mozilla Firefox/ Internet Explorer/Google Chrome
- Data Base server : MY-SQL
- Web Server : Tomcat 7.0
- Language : Java, servlet, jsp, html, CSS

3. REQUIREMENT ANALYSIS

Requirement analysis is a software engineering task that bridges the gap between system software allocation and software design. It provides system engineer to specify software function and performance indicate software's interface with other system elements and establish constraints that software must need. The basic is to obtain a clear picture of the needs and requirements of the end user and also the organization.

3.1 Purpose:

Analysis involves interaction between analysts and clients. The analyst research the problem from the question ask and reading existing documents.

During Analysis it is essential that a complete and consistent set of specification emerge for the system. The essential final consistent specifications are divided into five parts.

- a. Problem recognition.
- b. Evaluating and Synthesis.
- c. Modeling.
- d. Specification.
- e. Review.

3.1.1 Project Scope:

Overview:

The proposed Online Course Portal for a Campus is expected to automate the process of registration to the portal, publishing the announcements and communication between the faculty and students of a campus. The process of Test Creation will also be automated. The registration and login records management will be automated and the process of learning courses is made easier using the system.

3.1.1.1: Assumptions:

- Only one Administrator for the portal
- Faculty can even register as student to learn courses.
- All the maintenance of the portal is done only by the administrator.
- Other than faculty of the campus are not allowed to access the portal.

3.2. Requirement Specification:

The requirement specification gives the detailed requirements for each individual working in the development of the new system & the functionalities required fulfilling the objective of the new system.

Developer responsibilities:

- i) Developing the system, this meets the SRS and solves all the requirements.
- ii) Demonstrating the system and installing the system at clients' location after the acceptance testing is successful.
- iii) Submitting the required user manual describing the system interfaces to work on it and also documents of system.
- iv) Conducting any user training that might be needed for using the system interfaces to work on it.
- v) Maintaining the system for a period of one year.

3.3 Functional Requirements:

The **Administrator** should be able to:

- Create a Course, by providing the course title and description.
- Approve Faculty members for a Course
- Delete the members of a Course
- Publish announcements
- A page to view all the feedbacks received.

The **faculty** should be able to:

- Login to the system through the first page of the application.
- Register for a particular course
- Upload and download the required information.
- View the courses and accepted faculty
- View the Announcements
- View the feedback given by the students to the faculty
- Create a Test of multiple choice questions
- Create a course Specific announcements
- Create a discussion

- View the discussion board

The **Student** should be able to do:

- Login to the system through the first page of the application.
- Register for a particular course
- Download the required information.
- View the course faculty
- View the Announcements
- Give feedback to the faculty
- Take a test
- View course Specific announcements
- Create a discussion
- View the discussion board

3.3. Data Validation:

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is possibility for the user to commit errors.

The system will not accept any invalid data. Whenever an invalid data is keyed in the system immediately prompts the user and the user has to again key the data and the system will accept if only the data is correct. Validations have been included where ever necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user.

3.3.1 Performance requirements:

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit in to required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has

been designed and on the other hand designing a system, which doesn't cater to the requirements of the user, is of accurate

- a) The system should be better than the existing system
- b) The existing system is completely dependent on the user to perform all the duties.

3.4 Non Functional Requirements:

- The application should support the mechanism of avoiding interruptions while sending and receiving mails (Supportability).
- The system should be available 24 X 7. (Reliability/ availability)
- Must run on any Windows Operating System.

4. UML DIAGRAMS

4.1 Use case diagram:

Use case Diagrams represent the functionality of the system from a user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

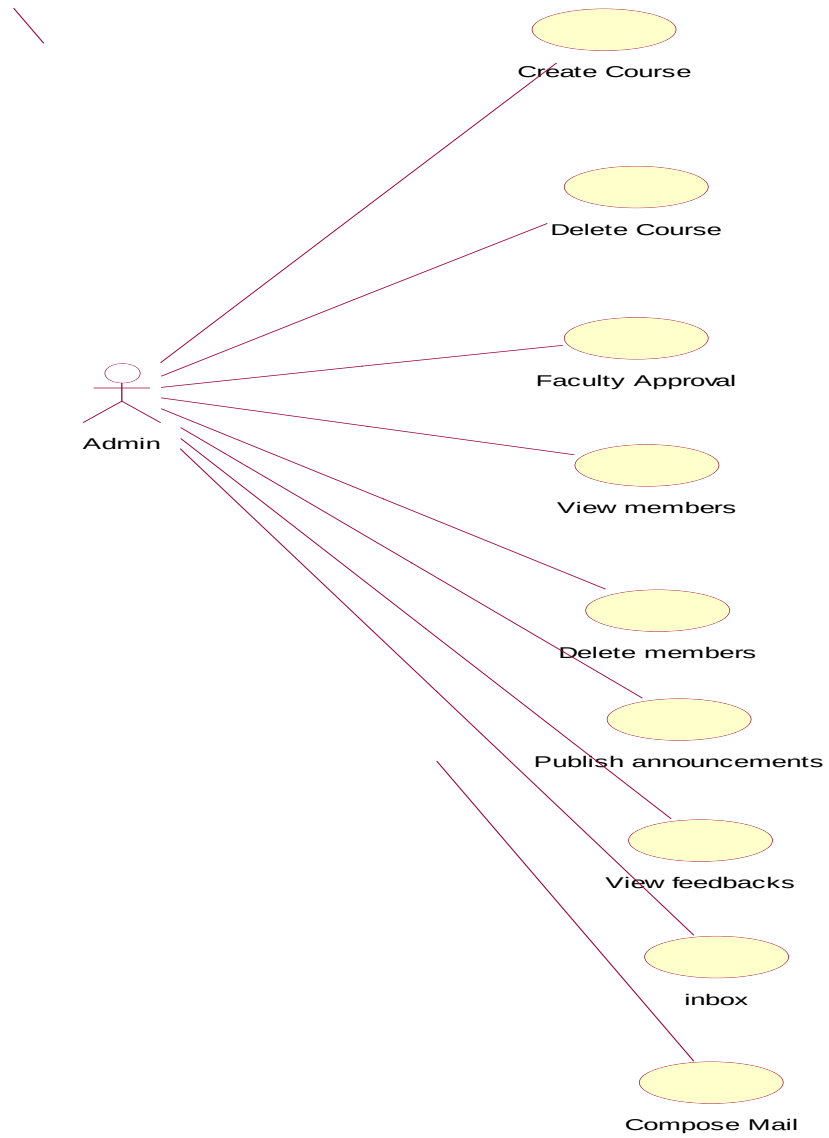
Use case diagram show actors and use cases together with their relationships. The use cases represents functionality of a system or a file classifier, like a subsystem or a class, as manifested to external integrators with the system or the classifier. A use case diagram is a graph of actors, a set of use cases, possibly some interfaces and the relationships between these elements. Use case diagram also describes the interaction between user and system.

Actor:

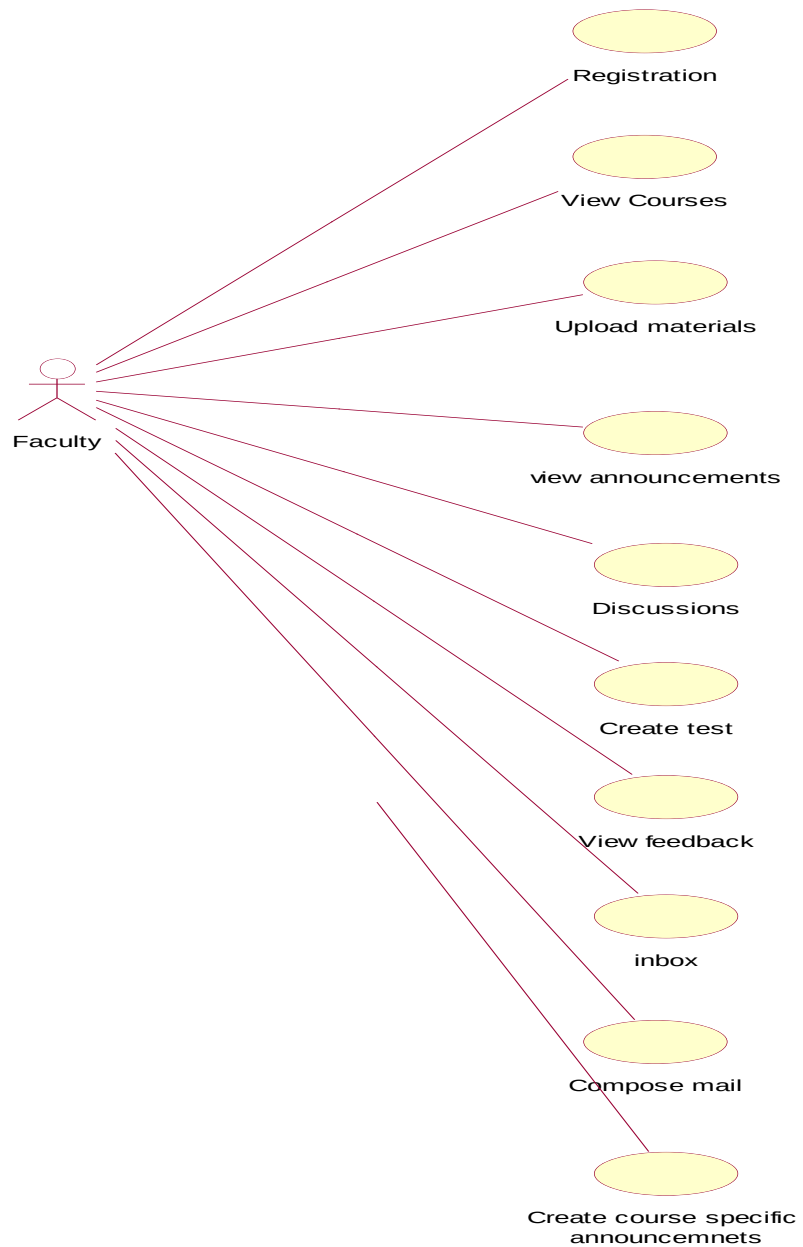
An actor defines a coherent set of roles that users of an entity can play when interacting with the entity. An actor may be considered to play with regard to each use case with which it communicates.

Actor represents external entity that interacts with their system. An actor can be human or an external system. During this activity developers identify the different type of users that future system will support. Examples of actors include users like administrator, bank customer ...etc., or another system like central database.

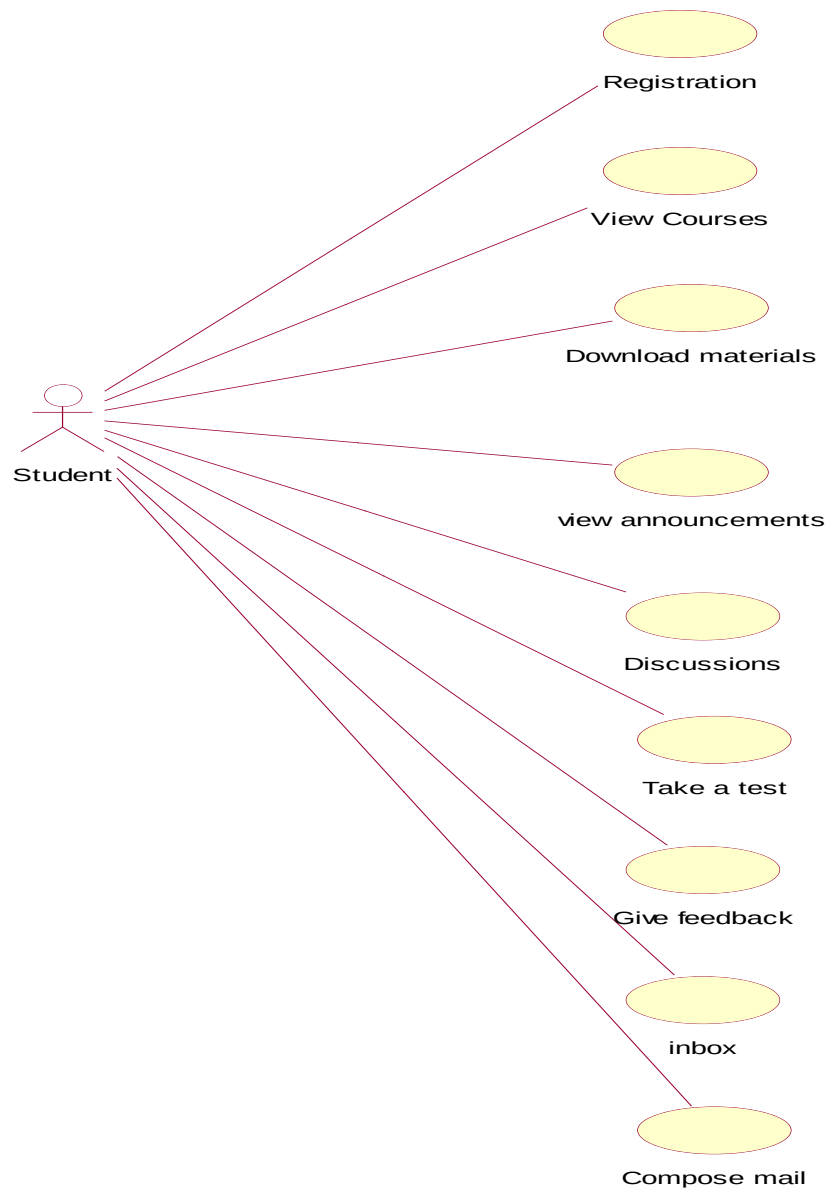
4.1.1 Use case Diagram for Admin :



4.1.2 Use case Diagram For Faculty:



4.1.3 Use case Diagram For Student:



4.2 SEQUENCE DIAGRAMS:

Sequence diagram describes the pattern of communication among set of interacting objects. An object interacts with other objects by sending messages. The reception of a message by an object triggers the execution of an operation, which in turn may send messages to other objects. Arguments may be passed along with a message and are bound to the parameters of the executing operation in the receiving object. Sequence diagram represents the behavior of the system from the perspective of single use case.

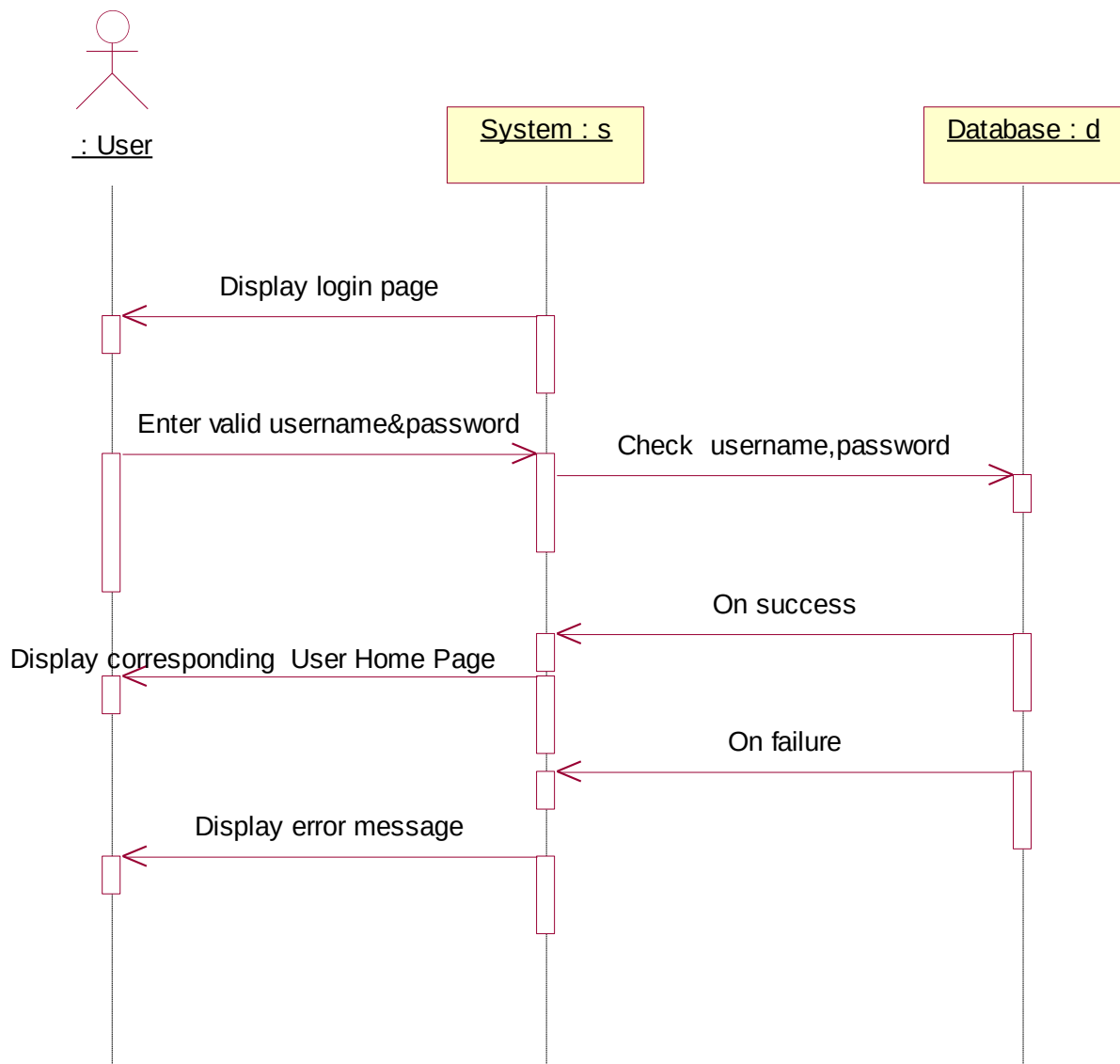


FIG:4.2.1 Sequence Diagram For Login

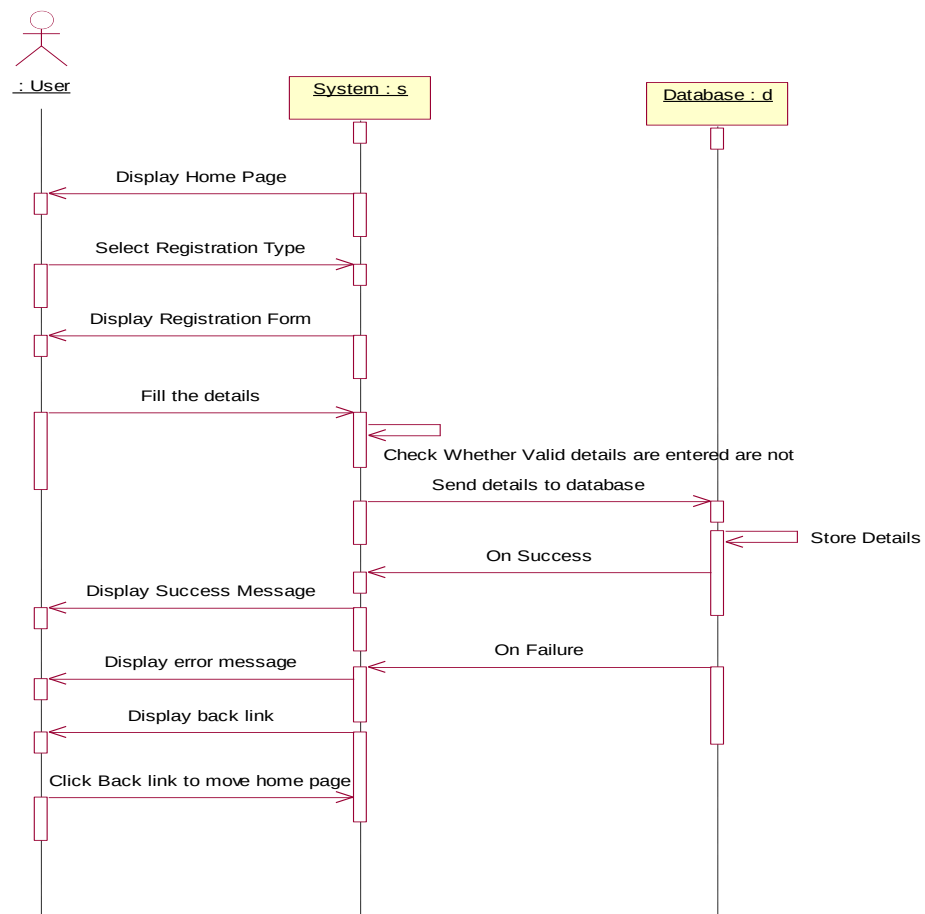


FIG:4.2.2 Sequence Diagram For Registration

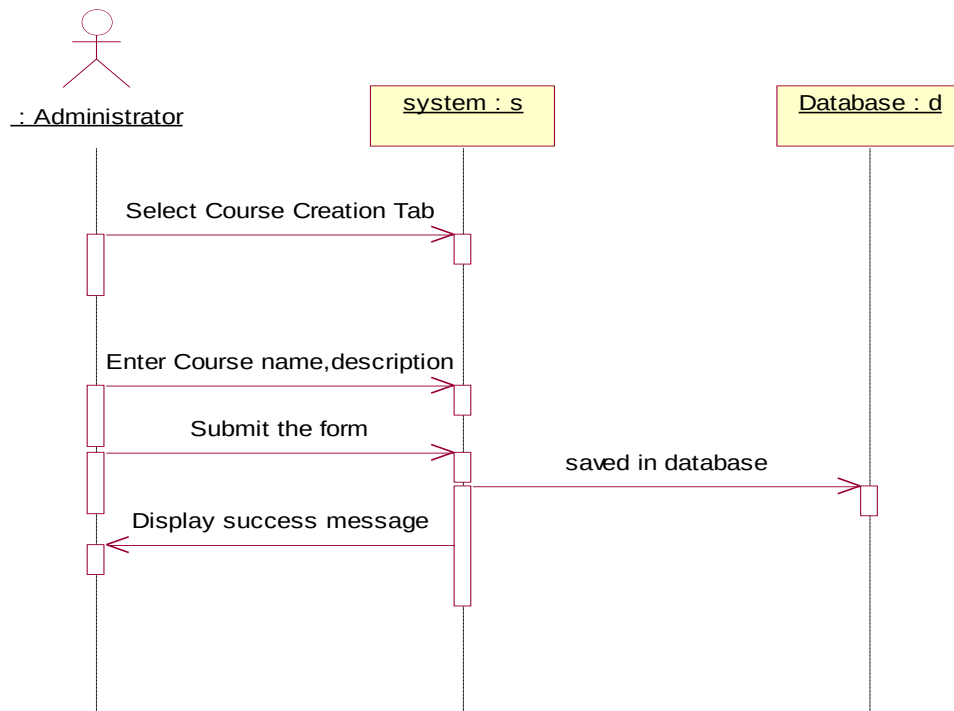


FIG:4.2.3 Sequence Diagram For Course Creation

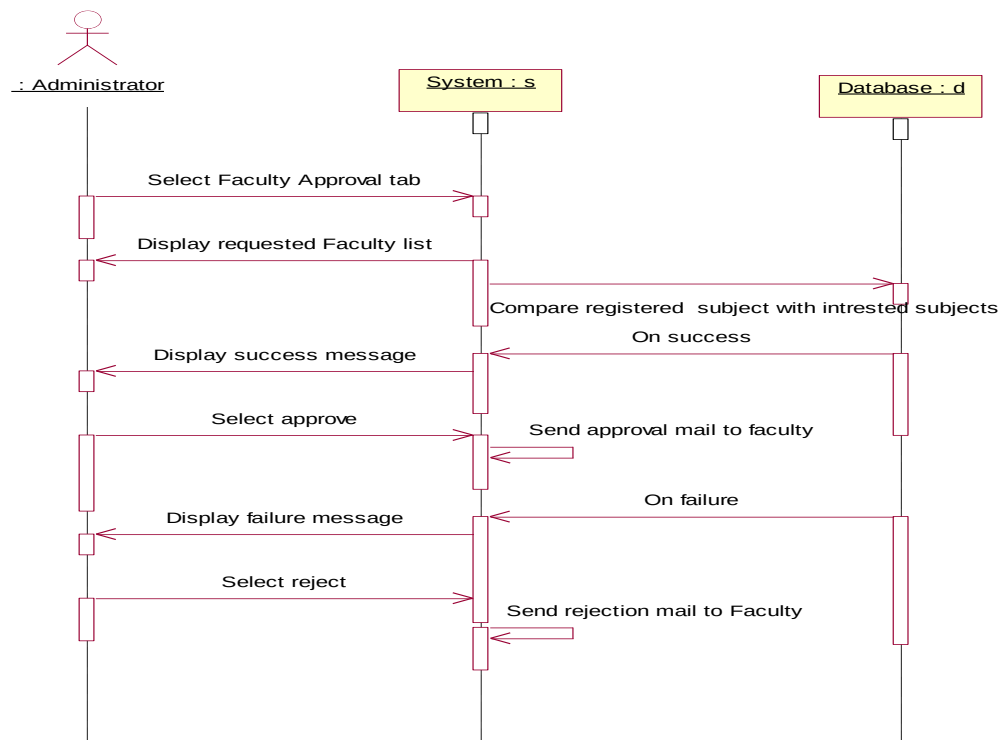
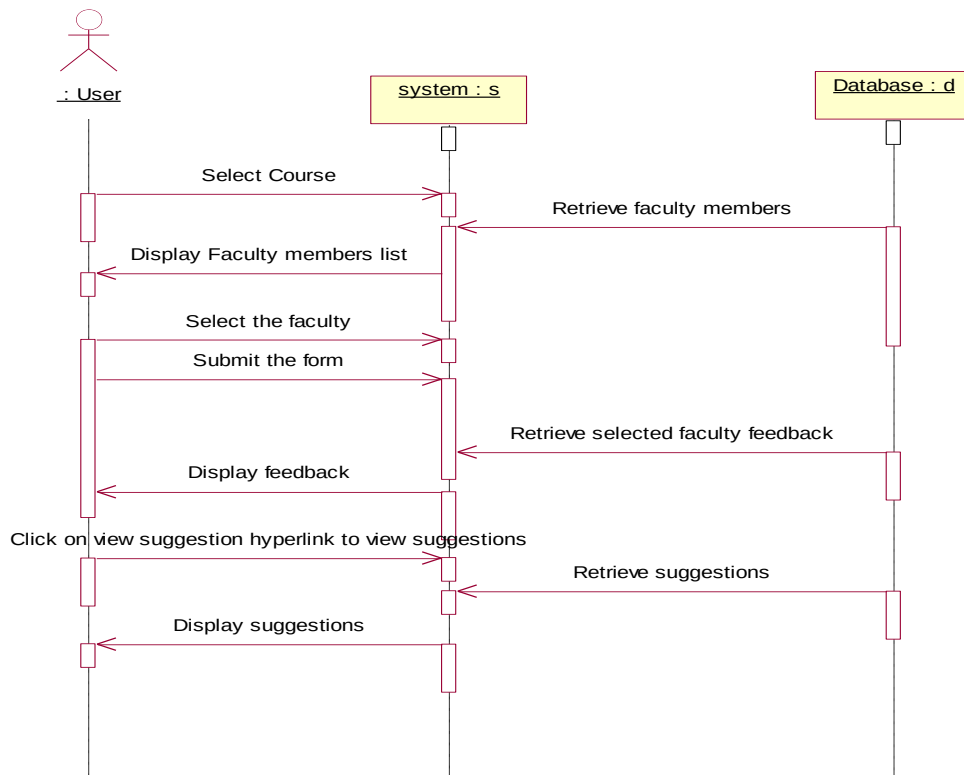


FIG:4.2.4 Sequence Diagram For Faculty Approval



4.3 Collaboration Diagrams

Collaboration is used for describing the realization of an operation or a classifier. A collaboration that describes a classifier like a use case but use case describes references classifier and associations in general, while a collaboration describing an operation includes the arguments and local variables of the operation, as well as ordinary associations attached to the classifier owning the operation.

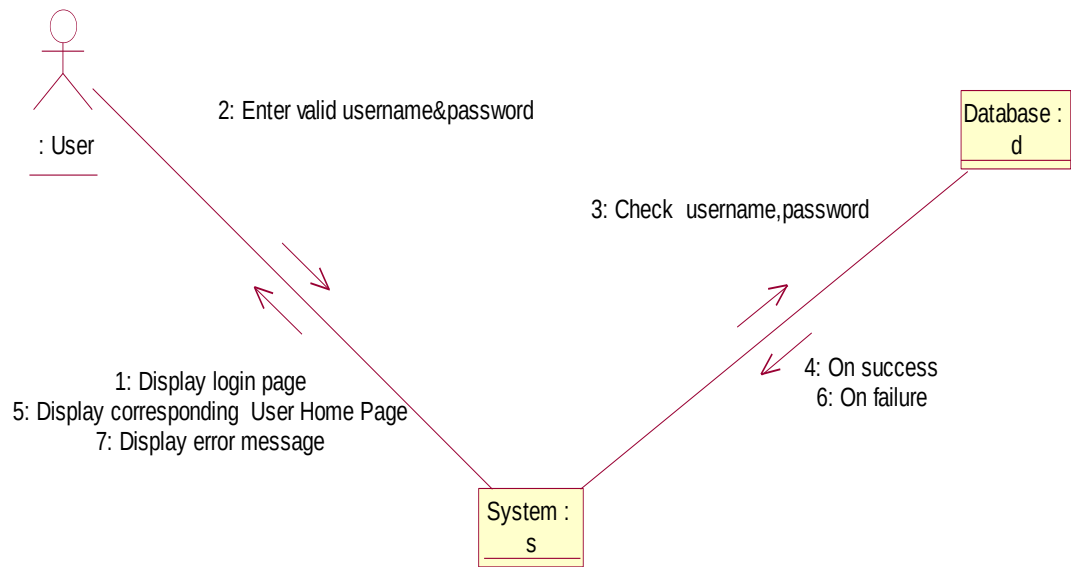


Fig: 4.3.1 Collaboration Diagram for Login

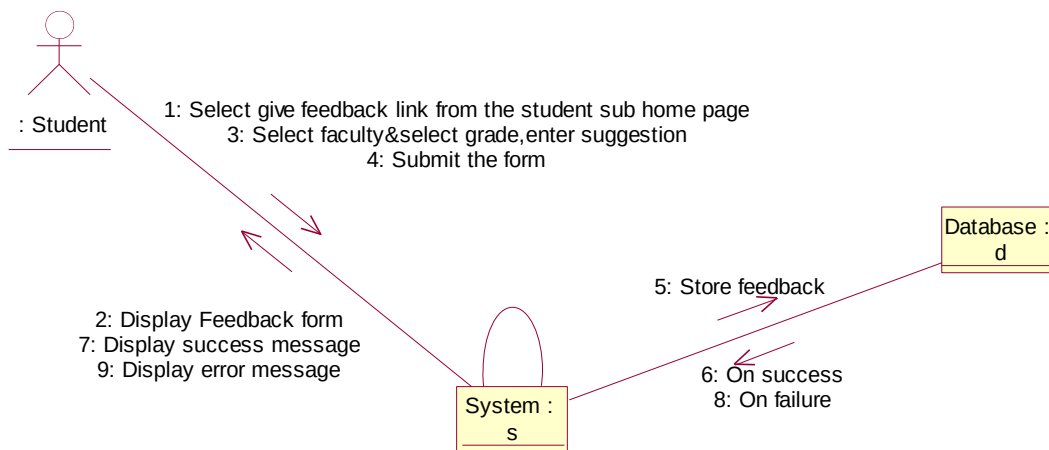


Fig: 4.3.2 Collaboration Diagram for Give feedback

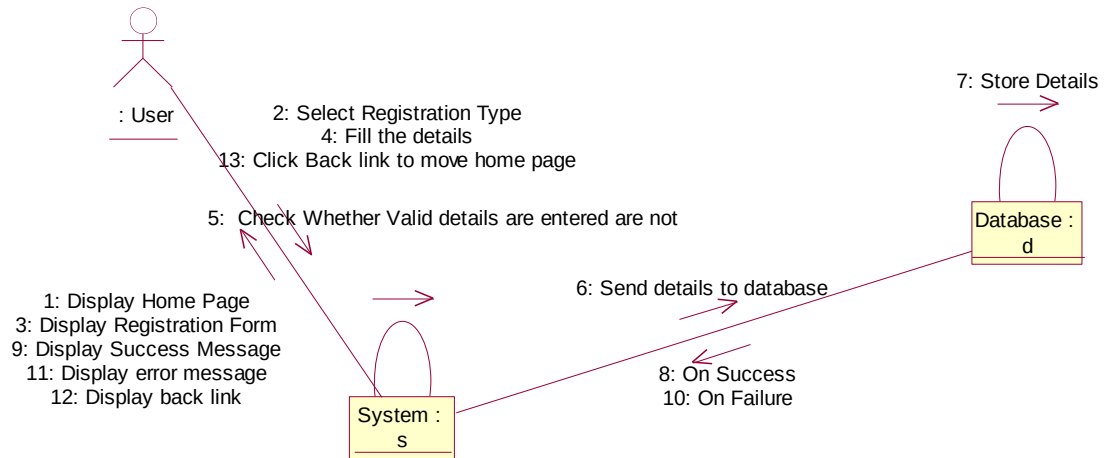


Fig: 4.3.3 Collaboration Diagram for Registration

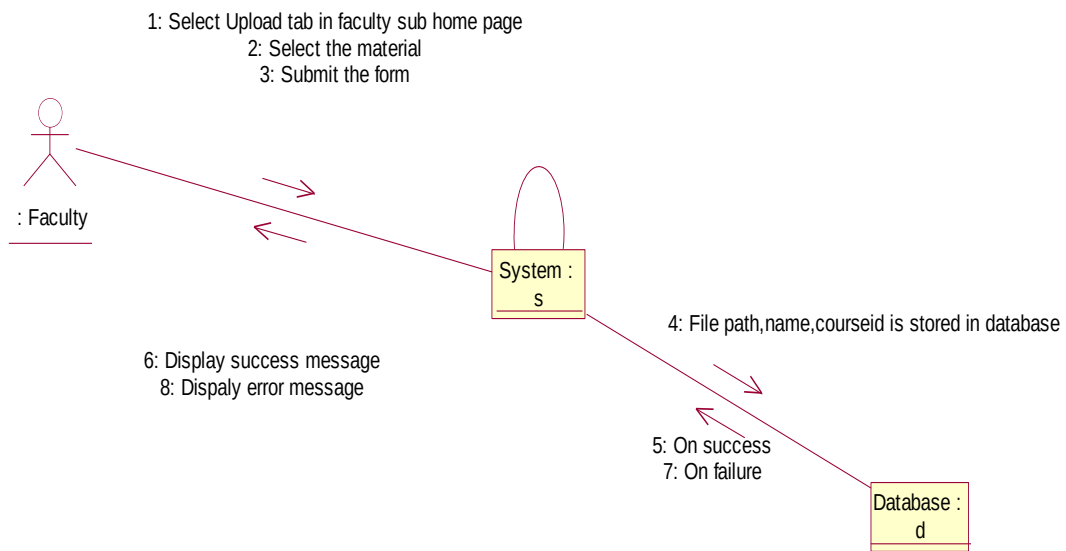


Fig: 4.3.4 Collaboration Diagram for Upload

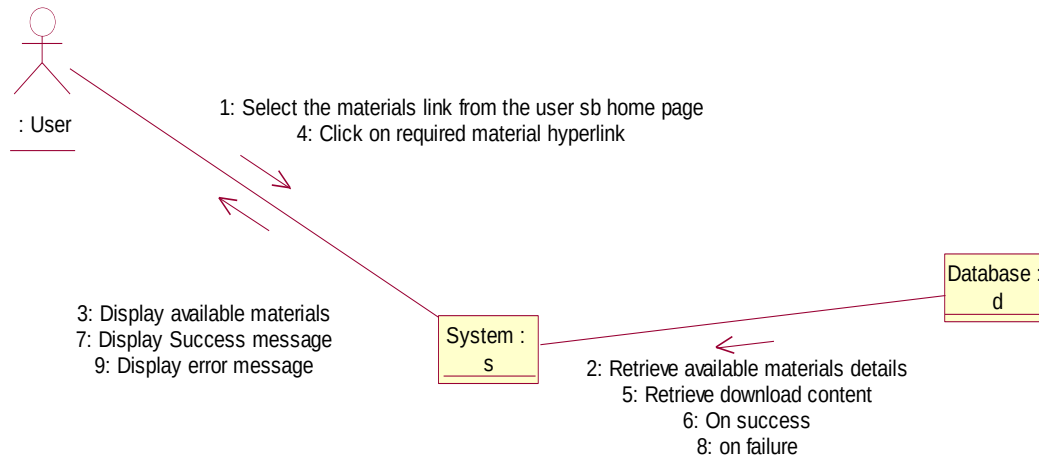


Fig: 4.3.5 Collaboration Diagram for Download

4.4 CLASS DIAGRAMS:

A class diagram is similar to a family tree. A class diagram consists of a group of classes and interfaces reflecting important entities of the business domain of the system being modeled, and the relationships between these classes and interface. The classes and in the diagram represents the members of a family tree and the relationships between the classes are analogous to relationships between the members in a family tree. Classes in class diagram are interconnected in a hierarchical fashion, like a set of parent classes and related child classes under the parent classes.

By definition, a class diagram is a diagram showing a collection of classes and interfaces, along with collaborations and relationships among classes and interfaces. Class diagrams are widely used to describe the types of objects in a system and their relationships. Class diagrams model class structure and contents using design elements such as classes, packages, and objects. Class diagrams describe here different perspectives when designing a system, conceptual, specification, and implementation. Classes are composed of three things: a name, attribute and operations. Class diagrams also display relationships such as containment, inheritance, associations and others.

In class diagrams, multiplicity is only specified to association. There are three types of multiplicities. They are **One to One**, **One to Many**, **Many to Many**. The access specifies used

in the UML diagrams are **public** '+', **private** '-', **protected** '#'. In association relationships, we represent the involvement through multiplicity, which can be of "0, 1...n". In class diagram, advanced classes consists of three types. They are **Concrete Class**, **Abstract Class** and **Final Class**.

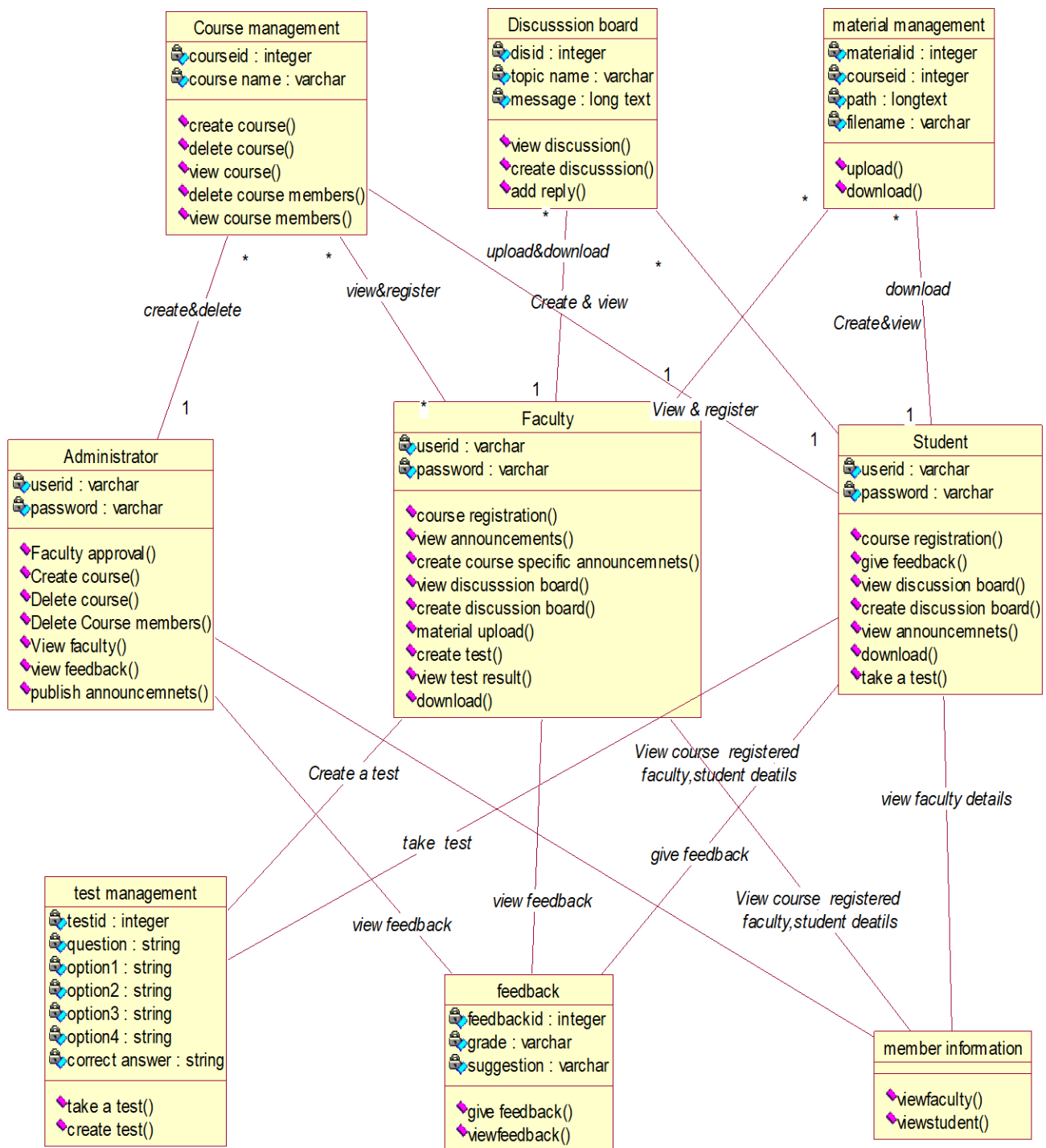
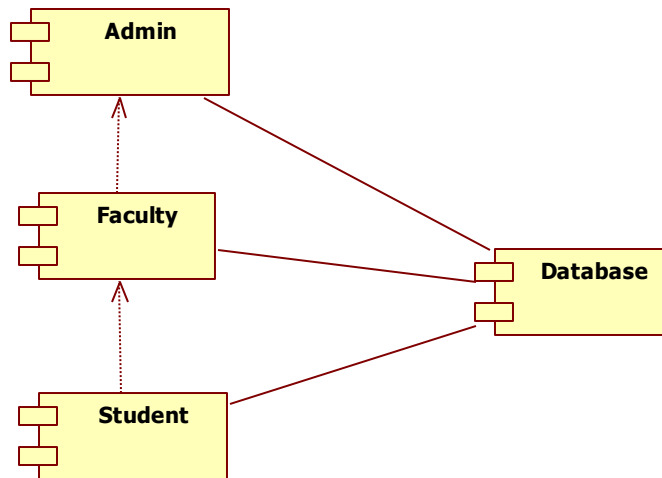


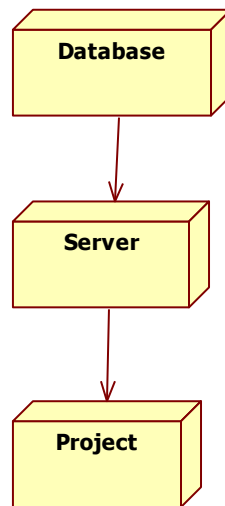
FIG: 4.4.1. Class Diagram

4.5 ACTIVITY DIAGRAMS:

Activity diagrams are one of the five diagrams in the UML for modeling the dynamic aspects of systems. An activity diagram is essentially a flow chart, showing flow of control from activity to activity. We use activity diagrams to model the dynamic aspects of a system. For the most part, this involves modeling the sequential (and possibly concurrent) steps in a computational process. With an activity diagram, you can also model the flow of an object as it moves from state to state at different points in the flow of control. Activity diagrams may stand alone to visualize, specify, construct and document the dynamics of a society of objects, or they may be used to model the flow of control of an operation. Whereas interaction diagrams emphasize the flow of control from object to object, activity diagrams emphasize the flow of control from activity to activity. An activity is an ongoing non-atomic execution within a state machine. Activities ultimately result in a change in state of the system or the return of the value. Activity diagrams are not only important for modeling the dynamic aspects of the system, but also for constructing executables systems through forward and reverse engineering.



Deployment Diagram:



4.6. Database Design:

4.6.1 Normalization

Database normalization is a design technique by which relational database tables are structured in such a way as to make them invulnerable to certain types of logical inconsistencies and anomalies. Tables can be normalized to varying degrees: relational database theory defines "normal forms" of successively higher degrees of stringency, so, for example, a table in third normal form is less open to logical inconsistencies and anomalies than a table that is only in second normal form. Although the normal forms are often defined (informally) in terms of the characteristics of tables, rigorous definitions of the normal forms are concerned with the characteristics of mathematical constructs known as relations. Whenever information is represented relationally—that is, roughly speaking, as values within rows beneath fixed column

headings—it makes sense to ask to what extent the representation is normalized.

4.6.2 Normal forms

The normal forms (abbrev. NF) of relational database theory provide criteria for determining a table's degree of vulnerability to logical inconsistencies and anomalies. The higher the normal form applicable to a table, the less vulnerable it is to such inconsistencies and anomalies. Each table has a "highest normal form" (HNF): by definition, a table always meets the requirements of its HNF and of all normal forms lower than its HNF; also by definition, a table fails to meet the requirements of any normal form higher than its HNF.

The normal forms are applicable to individual tables; to say that an entire database is in normal form n is to say that all of its tables are in normal form n .

Newcomers to database design sometimes suppose that normalization proceeds in an iterative fashion, i.e. a 1NF design is first normalized to 2NF, then to 3NF, and so on. This is not an accurate description of how normalization typically works. A sensibly designed table is likely to be in 3NF on the first attempt; furthermore, if it is 3NF, it is overwhelmingly likely to have an HNF of 5NF.

4.6.2.1. First normal form:

The criteria for first normal form (1NF) are:

A table must be guaranteed not to have any duplicate records; therefore it must have at least one candidate key.

There must be no repeating groups, i.e. no attributes which occur a different number of times on different records. For example, suppose that an employee can have multiple skills: a possible representation of employees' skills is {Employee ID, Skill1, Skill2, Skill3 ...}, where {Employee ID} is the unique identifier for a record. This representation would not be in 1NF.

Note that all relations are in 1NF. The question of whether a given representation is in 1NF is equivalent to the question of whether it is a relation.

4.6.2.2. Second normal form:

The criteria for second normal form (2NF) are:

The table must be in 1NF.

None of the non-prime attributes of the table are functionally dependent on a part (proper subset) of a candidate key; in other words, all functional dependencies of non-prime attributes on candidate keys are full functional dependencies. For example, consider a "Department Members" table whose attributes are Department ID, Employee ID, and Employee Date of Birth; and suppose that an employee works in one or more departments. The combination of Department ID and Employee ID uniquely identifies records within the table. Given that Employee Date of Birth depends on only one of those attributes – namely, Employee

ID – the table is not in 2NF.

Note that if none of a 1NF table's candidate keys are composite – i.e. every candidate key consists of just one attribute – then we can say immediately that the table is in 2NF.

4.6.2.3. Third normal form:

The criteria for third normal form (3NF) are:

The table must be in 2NF.

There are no non-trivial functional dependencies between non-prime attributes. A violation of 3NF would mean that at least one non-prime attribute is only indirectly dependent (transitively dependent) on a candidate key, by virtue of being functionally dependent on another non-prime attribute. For example, consider a "Departments" table whose attributes are Department ID, Department Name, Manager ID, and Manager Hire Date; and suppose that each manager can manage one or more departments. {Department ID} is a candidate key. Although Manager Hire Date is functionally dependent on {Department ID}, it is also functionally dependent on the non-prime attribute Manager ID. This means the table is not in 3NF.

4.6.2.4. Boyce-Codd normal form:

- The criteria for Boyce-Codd normal form (BCNF) are:
- The table must be in 3NF.
- Every non-trivial functional dependency must be a dependency on a superkey.

After carefully understanding the requirements of the candidate the entire data storage requirements are divided into tables.

4.6.3. Tables:

Column name	Data type	Size	Nulls (y/n)	Constraints	Description
Name	Varchar	25	N	-----	-----
Username	Varchar	25	N	Primary Key	-----
Password	Varchar	25	N	-----	-----
Confirm_pwd	Varchar	25	N	-----	-----
MobileNumber	Varchar	15	Y	-----	-----
DOB	Varchar	10	N	-----	-----
College	Varchar	40	N	-----	-----
Department	Varchar	50	N	-----	-----
Qualification	Varchar	50	Y	-----	-----
Semistr	Varchar	10	N	-----	-----
Address	Varchar	50	N	-----	-----
Designation	Varchar	20	N	-----	-----
Reg-id	Int	10	N	Unique-key	-----
Role	Varchar	45	N	-----	-----
Fac-id	Varchar	45	N	Unique-key	-----

Table: 4.6.3.1.Registration Table

```

CREATE TABLE `registration` (
  `Name` varchar(45) NOT NULL,
  `Username` varchar(45) NOT NULL,
  `Password` varchar(45) NOT NULL,
  `Confirmpwd` varchar(45) NOT NULL,
  `Email` varchar(45) NOT NULL,
  `Mobileno` varchar(45) NOT NULL,
  `DOB` varchar(45) NOT NULL,
  `College` varchar(45) NOT NULL,
  `Department` varchar(45) NOT NULL,
  `Qualification` varchar(45) default NULL,
  `Semister` varchar(45) default NULL,
  `Address` varchar(100) NOT NULL,
  `Designation` varchar(45) default NULL,
  `Reg_id` int(11) NOT NULL,
  `Role` varchar(45) NOT NULL,
  `Fac_id` varchar(45) default NULL,
  PRIMARY KEY (`Username`),
  UNIQUE KEY `Reg_id_UNIQUE` (`Reg_id`),

```

```
UNIQUE KEY `Fac_id_UNIQUE` (`Fac_id`)
)
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Cid	Int	11	Y	<i>Primary key</i>	-----
Cname	Varchar	45	N	-----	-----
Username	Varchar	45	N	-----	-----
Status	Varchar	45	N	-----	-----

Table: 4.6.3.2.Course Registration Table

```
CREATE TABLE `coursereg`
(
  `Cid` int(11) NOT NULL auto_increment,
  `Cname` varchar(45) default NULL,
  `Username` varchar(45) default NULL,
  `Status` varchar(45) default NULL,
  PRIMARY KEY (`Cid`)
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Username	Varchar	45	N	<i>Primary key</i>	-----
Password	Varchar	45	N	-----	-----
Role	Varchar	45	N	-----	-----
Mail	Varchar	45	Y	----	-----

Table: 4.6.3.3.Login Table

```
CREATE TABLE `login` (
  `Username` varchar(45) NOT NULL default "",
  `Password` varchar(45) NOT NULL,
  `Role` varchar(45) NOT NULL,
  `Mail` varchar(45) default NULL,
  PRIMARY KEY (`Username`),
  KEY `Username` (`Username`),
  CONSTRAINT `Username` FOREIGN KEY (`Username`) REFERENCES `registration`
  (`Username`) ON DELETE CASCADE ON UPDATE CASCADE
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Course-id	Int	11	N	-----	-----
Faculty_name	Varchar	45	N	Primarykey	-----

Table: 4.6.3.4. Faculty_Courses Table

```
CREATE TABLE `faccourse` (
  `Username` varchar(45) NOT NULL default "",
  `Courseid` int(11) NOT NULL default '0',
  PRIMARY KEY (`Username`,`Courseid`)
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Course-id	int	11	N	Primarykey	-----
User_name	Varchar	45	N	Primarykey	-----

Table: 4.6.3.5.Student_Courses Table

```
CREATE TABLE `stdcourse` (
  `Username` varchar(45) NOT NULL,
  `Courseid` int(11) NOT NULL,
  PRIMARY KEY (`Username`,`Courseid`)
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Course id	Number	7	N	Primary key	-----
Cname	Varchar	45	N	-----	-----
Description	Varchar	500	N	-----	-----
Status	Varchar	45	N	-----	-----
Date	Date	10	N	-----	-----

Table: 4.6.3.6.Course Table

```
CREATE TABLE `course` (
  `Courseid` int(11) NOT NULL auto_increment,
  `Cname` varchar(45) NOT NULL,
  `Description` varchar(200) default NULL,
  `Status` varchar(45) default NULL,
```

```
`Date` date default NULL,  
PRIMARY KEY (`Courseid`)  
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
disid	Int	11	N	-----	-----
Course_id	Int	11	N	-----	-----
User name	Varchar	45	N	-----	-----
Topic	Varchar	100	N	-----	-----
Replymsg	Varchar	1000	N	-----	-----
Id	Int	11	Y	<i>Primary key</i>	-----

Table: 4.6.3.7.Reply Table

```
CREATE TABLE `reply` (  
  `disid` int(11) NOT NULL,  
  `Courseid` int(11) NOT NULL,  
  `Username` varchar(45) default NULL,  
  `Topic` varchar(100) default NULL,  
  `Replymsg` varchar(1000) default NULL,  
  `id` int(11) NOT NULL auto_increment,  
  PRIMARY KEY (`id`)  
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Rec	varchar	45	Y	-----	-----
Send	Varchar	45	Y	-----	-----
Sub	Varchar	45	N	-----	-----
Message	Varchar	250	N	-----	-----
Id	Int	11	N	<i>Primary key</i>	-----
Date	DATE		N	-----	-----

Table: 4.6.3.8.Mail Table

```
CREATE TABLE `mail` (  
  `Rec` varchar(45) NOT NULL,  
  `Send` varchar(45) NOT NULL,  
  `Sub` varchar(45) default NULL,  
  `Message` varchar(250) default NULL,  
  `id` int(11) NOT NULL auto_increment,
```

```
`Date` date default NULL,  
PRIMARY KEY (`id`)  
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Announcement id	Int	11	Y	Primary key	-----
Course_id	Int	11	N	Foreign key	-----
Date	Date		N	-----	-----
Description	Varchar	45	N	---	-----

Table: 4.6.3.9.Announcements Table

```
CREATE TABLE `announcement` (  
  `Announcementid` int(11) NOT NULL auto_increment,  
  `Courseid` int(11) default NULL,  
  `date` date default NULL,  
  `Description` varchar(100) default NULL,  
  PRIMARY KEY (`Announcementid`),  
  KEY `Courseid` (`Courseid`),  
  CONSTRAINT `Courseid` FOREIGN KEY (`Courseid`) REFERENCES `course` (`Courseid`)  
  ON DELETE CASCADE ON UPDATE CASCADE  
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Feedback id	Int	11	Y	Primary key	-----
Username	Varchar	45	N	-----	-----
Course id	Int	11	N	-----	-----
Grade	Varchar	45	N	----	-----
Suggestion	Varchar	100	N	-----	-----

Table: 4.6.3.10.Feedback Table

CREATE TABLE `feedback` (

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Disid	Int	11	Y	<i>Primary key</i>	-----
Course id	Int	11	N	-----	-----
Username	Varchar	45	N	----	-----
Topic name	Varchar	100	N	-----	-----
Query	Varchar	300	N	-----	-----

```
`feedbackid` int(11) NOT NULL,
`Username` varchar(45) default NULL,
`Courseid` int(11) default NULL,
`Grade` varchar(45) NOT NULL,
`Suggestion` varchar(100) default NULL,
PRIMARY KEY (`feedbackid`);
```

Table: 4.6.3.11.Discussion Table

```
CREATE TABLE `discussion` (
`disid` int(11) NOT NULL auto_increment,
`Courseid` int(11) default NULL,
`Username` varchar(45) default NULL,
`Topicname` varchar(100) default NULL,
`Query` varchar(300) default NULL,
PRIMARY KEY (`disid`)
);
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Idtest	Int	11	N	<i>Primary key</i>	-----
Question	Varchar	100	N	-----	-----
Option1	Varchar	100	N	-----	-----
Option2	Varchar	100	N	-----	-----
Option3	Varchar	100	N	-----	-----
Option4	Varchar	100	N	-----	-----
Answer	Varchar	100	N	-----	-----
Course id	Int	11	Y	----	-----

Table: 4.6.3.12.Test Table

```
CREATE TABLE `test` (
`idtest` int(11) NOT NULL auto_increment,
```

```

`Question` varchar(100) NOT NULL,
`option1` varchar(100) NOT NULL,
`option2` varchar(100) NOT NULL,
`option3` varchar(100) NOT NULL,
`option4` varchar(100) NOT NULL,
`answer` varchar(100) NOT NULL,
`Courseid` int(11) default NULL,
PRIMARY KEY (`idtest`)
);

```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Fac-id	Varchar	45	N	<i>Primary key</i>	-----
Fac_name	varchar	25	N	-----	-----
Colz	Varchar	50	N	-----	-----
Dept	Varchar	45	N	-----	-----
Qual	Varchar	45	N	-----	-----
Designation	Varchar	45	N	-----	-----

Table: 4.6.3.13.Campus Table

```

CREATE TABLE `campus` (
  `Fac_id` varchar(45) NOT NULL,
  `Facname` varchar(45) NOT NULL,
  `Colz` varchar(50) NOT NULL,
  `Dept` varchar(45) NOT NULL,
  `Qual` varchar(45) NOT NULL,
  `Designation` varchar(45) NOT NULL,
  PRIMARY KEY (`Fac_id`)
);

```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
File-id	Int	11	N	<i>Primary key</i>	-----
Cname	varchar	45	N	-----	-----
Filepath	Mediumtext	45	N	-----	-----
Filename	Longtext	45	N	-----	-----
Date	Dater	45	N	-----	-----

Table: 4.6.3.14.File Table

```
CREATE TABLE `file` (
  `Fileid` int(11) NOT NULL auto_increment,
  `Cname` varchar(45) NOT NULL,
  `Filepath` mediumtext NOT NULL,
  `Filename` longtext NOT NULL,
  `Date` date default NULL,
  PRIMARY KEY (`Fileid`));
```

<i>Column name</i>	<i>Data type</i>	<i>Size</i>	<i>Nulls (y/n)</i>	<i>Constraints</i>	<i>Description</i>
Id	integer	11	N	<i>Primary key</i>	-----
Username	varchar	45	N	-----	-----
Course	varchar	45		-----	-----
Score	integer	11	N	-----	----
Date	Date	----	N	-----	-----

Table: 4.6.3.15.Test Result Table

```
CREATE TABLE `testresult` (
  `id` int(11) NOT NULL auto_increment,
  `Username` varchar(45) NOT NULL,
  `Course` varchar(45) NOT NULL,
  `Score` int(11) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=latin1;
```

5. TESTING

Software Testing is the process used to help identify the correctness, completeness, security, and quality of developed computer software. Testing is a process of technical investigation, performed on behalf of stakeholders, that is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. This includes, but is not limited to, the process of executing a program or application with the intent of finding errors. Quality is not an absolute; it is value to some person. With that in mind, testing can never completely establish the correctness of arbitrary computer software; testing furnishes a criticism or comparison that compares the state and behaviour of the product against a specification. An important point is that software testing should be distinguished from the separate discipline of Software Quality Assurance (SQA), which encompasses all business process areas, not just testing.

There are many approaches to software testing, but effective testing of complex products is essentially a process of investigation, not merely a matter of creating and following routine procedure. One definition of testing is "the process of questioning a product in order to evaluate it", where the "questions" are operations the tester attempts to execute with the product, and the product answers with its behavior in reaction to the probing of the tester[citation needed]. Although most of the intellectual processes of testing are nearly identical to that of review or inspection, the word testing is connoted to mean the dynamic analysis of the product—putting the product through its paces. Some of the common quality attributes include capability, reliability, efficiency, portability, maintainability, compatibility and usability. A good test is sometimes described as one which reveals an error; however, more recent thinking suggests that a good test is one which reveals information of interest to someone who matters within the project community.

In general, software engineers distinguish software faults from software failures. In case of a failure, the software does not do what the user expects. A fault is a programming error that may or may not actually manifest as a failure. A fault can also be described as an error in the correctness of the semantic of a computer program. A fault will become a failure if the exact computation conditions are met, one of them being that the faulty portion of computer software executes on the CPU. A fault can also turn into a failure when the software is ported to a different hardware platform or a different compiler, or when the software gets extended.

Software testing is the technical investigation of the product under test to provide stakeholders with quality related information.

5.1 Test Plan:

A problem with software testing is that the number of defects in a software product can be very large, and the number of configurations of the product larger still. Bugs that occur infrequently are difficult to find in testing. A rule of thumb is that a system that is expected to function without faults for a certain length of time must have already been tested for at least that length of time. This has severe consequences for projects to write long-lived reliable software.

5.1.1 User id and password:

Test case ID	Input	Description	Expected result
OCP_TC_01	login id blank, password blank	Blank loginid,password	login id and password fields are mandatory
OCP_TC_02	valid id and blank password	password is mandatory	password is mandatory
OCP_TC_03	password and blank id	login id is mandatory	login id is mandatory
OCP_TC_04	id<4 ,valid password	id should contain more than 4 characters	id should contain more than 4 characters
OCP_TC_05	valid id and password<6	password should contain minimum of 6 characters	password should contain minimum of 6 characters
OCP_TC_06	invalid id, invalid password	Invalid ID and Password	Invalid ID and Password
OCP_TC_07	valid id, password	Valid ID and valid password are given	Login Successful

Table: 5.1.1 User id and password

5.1.2 Phone Number:

Test case ID	Input	Description	Expected result
OCP_TC_01	Blank Ph number	A blank number is given by the User	Display that the ph number is mandatory
OCP_TC_02	Ph number with only numbers	Ph number with only numbers is given by the User	Successful storage
OCP_TC_03	Ph number with characters ,symbols	Ph number with characters is given by the User	Display Enter valid number
OCP_TC_04	Ph number >max length	PH number with >max length is given by the User	Display that Ph number is>max length

Table: 5.1.2 Phone Number

5.1.3 Mail id:

Test case ID	Input	Description	Expected result
OCP_TC_01	Blank mail id	A blank mail id is given by the User	Display Enter valid mail id
OCP_TC_02	Invalid mail id	Invalid Mail id given by the User	Display Enter valid mail id
OCP_TC_03	Valid mail id	Valid Mail id given by the User	Successful storage

Table: 5.1.3 Mail id

5.2 Test Report:

5.2.1 User id and password:

TEST CASE ID	INPUT	DESCRIPTION	EXPECTED RESULT	PASS/FAIL
OCP_TC_01	login id blank, password blank	login id and password fields are mandatory	login id and password fields are mandatory	Pass
OCP_TC_02	valid id and blank password	password is mandatory	password is mandatory	Pass
OCP_TC_03	password and blank id	login id is mandatory	login id is mandatory	Pass
OCP_TC_04	id<4 ,valid password	id should contain more than 4 characters	id should contain more than 4 characters	Pass
OCP_TC_05	valid id and password<6	password should contain minimum of 6 characters	password should contain minimum of 6 characters	Pass
OCP_TC_06	invalid id, invalid password	Invalid ID and Password	Invalid ID and Password	Pass
OCP_TC_07	valid id, password	Valid ID and valid password are given	Login Successful	Pass

Table: 5.2.1 User id and password

5.2.2 Phone Number:

TEST CASE ID	INPUT	DESCRIPTION	EXPECTED RESULT	PASS/FAIL
OCP_TC_01	Blank Ph number	A blank Ph number is given by the User	Display that Ph number is mandatory	Pass
OCP_TC_02	Ph number with only numbers	Ph number with only numbers is given by the User	Successful storage	Pass
OCP_TC_03	Ph number with characters, symbols	Ph number with characters, symbols is given by the User	Display that enter valid phone number	Pass
OCP_TC_04	Ph number >max length	Ph number with >max length is given by the User	Display that Ph number is >max length	Pass

Table: 5.2.2 Phone Number

5.2.3Mail id:

TEST CASE ID	INPUT	DESCRIPTION	EXPECTED RESULT	PASS/FAIL
OCP_TC_01	Blank mail id	A blank mail id is given by the User	Display Enter valid mail id	Pass
OCP_TC_02	Invalid mail id	Invalid Mail id given by the User	Display Enter valid mail id	Pass
OCP_TC_03	Valid mail id	Valid Mail id given by the User	Successful storage	Pass

Table: 5.2.3.Mail id

IMPLEMENTATION

6. SYSTEM IMPLEMENTATION

6.1.1 Introduction to JAVA Programming:

Java, whether you love it, or hate it, it's here to stay. Like everyone's favorite language C, Java has had a major impact on the computing scene. When the history of computers is written, its name will be up there with the stars.

Features of Java

- Compiled and Interpreted
- Platform-independent and Portable
- Robust and Secure
- Familiar, Simple and Small
- Multithreaded and interactive
- High Performance
- Dynamic and Extensible

These features have made Java the first application of the World Wide Web. Java will also become the premier language of general-purpose stand-alone applications.

6.1.2. JSP (Java Server Pages):

JSP is a technology invented by Sun Microsystems which allows the easy creation and maintenance of server side HTML pages, which can be used as both a kind of Dynamic HTML and CGI replacement.

A JSP page is ultimately compiled to a Servlet. Thus, a JSP page is a nothing but Servlet in certain sense. However, JSP s can be more easily deployed since they can live in any directory; also it is far easier to embed long HTML string in a JSP than Servlet.

There are three main types of JSP constructs that one can embed in a page

1. Scripts
2. Directives
3. Actions

Scripting elements are used to specify Java code that will become part of resultant Servlet, directives let us to control the overall structure of the Servlet and actions let us to specify existing components that should be used and otherwise control the behavior of the JSP engine.

JSP Elements

```
<%= expression %>
<% code %>
<%! Code declaration %>
<%-- comment --%>
<jsp:include page ="url" />      etc....
```

JSP Directives

A JSP directive start with <%@

Ex. 1) Page directive <%@ page import ="url" %>

2(10)) Include Directive <%@ include file="url" %>

Actions

Actions use constructs in XML syntax to control the behavior of the Servlet engine. You can dynamically insert a file, reuse JavaBeans components, etc.

Ex.jsp:include, jsp:useBean, jsp:setProperty etc....

JSP built-in Objects

- a) Request
- b) response
- c) session
- d) config etc.

Features of JSP:

- Portability
- Components
- Processing
- Access Models

Why use JSP?

JSP is easy to learn and allow developers to quickly produce web sites and applications in an open and standard way. JSP is based on Java, an object oriented language. JSP offers a robust platform for web development.

Main reasons to use JSP

- Multi-platform.
- Component reuse by using Java beans and EJB.
- Advantages of Java.

JSP files can be moved from one to another platform, web server or JSP Servlet engine.

Steps required for a JSP request

1. The user goes to a web site made using JSP. The user goes to a page (ending with .jsp). the web browser makes the request via the internet.
2. The JSP request gets sent to the Web server.
3. The Web server recognizes that the file required is special (.jsp), therefore passes the JSP file to the JSP Servlet engine.
4. If the JSP file has been called the first time, the JSP file is passed, otherwise go to step 7.
5. The next step is to generate a special Servlet from the JSP file. All the HTML required is converted to println statements.
6. The Servlet source code is compiled into a class.
7. The Servlet is initialized by calling **init** and **service** methods.
8. HTML from the Servlet output is sent via the Internet.
9. HTML results are displayed on the user's web browser.

JAVA INCLUDES A LIBRARY OF CLASS AND INTER FACES

The Java platform include an extensive class library so that programmers can use already existing classes as it is, create subclasses to modify existing classes, or implement interfaces to augment the capabilities of classes.

A package is a collection of related Java classes and interfaces. The following list, gives example of some Java packages.

Java.lang - The basic classes. This package is so basic that it automatically is included in any Java program. It includes classes dealing with numeric, strings, objects, runtime, security and threads.

Java.util – Miscellaneous utility classes, including generic data structures, bit sets, date, the string manipulation, random number generation, system properties, notification and enumeration of data structures.

Java.sql – The JDBC API, classes interfaces that access databases and send SQL

statements.

6.1.3 HTML (Hyper Text Markup Language):

HTML (hypertext markup language) is a language used to create hypertext documents that have hyperlinks embedded in them .it consists of tags embedded in the text of a document with HTML. We can build web pages or web documents. it is basically a formatting language and not a programming language. The browser reading the document interprets markup tags to help format the document for subsequent display to a reader. HTML is a language for describing structured documents. HTML is a platform independent. WWW (World Wide Web) pages are written using HTML. HTML tags control in part the representation of the WWW page when view with web browser. The browser interprets HTML tags in the web document and displays it. Different browsers show data differently. Examples of browsers used to be web pages include:

1. Netscape
2. Internet Explorer
3. Mosaic
4. Neo Planet etc.,

HTML is composed of *tags*. HTML tags are always enclosed in angle-brackets (<>) and are case-insensitive; that is, it doesn't matter whether you type them in upper or lower case. We almost always use upper case, because it makes the tags easier to pick out in a document.

Tags typically occur in begin-end pairs. These pairs are in the form

<tag> ... </tag>

where the<**tag**> indicates the beginning of a tag-pair, and the </**tag**> indicates the end. (The three dots indicate an arbitrary amount of content between the tags.) The usual way to refer to each tag is "**tag**" for the first and "**slash-tag**" for the second, where tag is the actual name of the tag.

These pairs define **containers**. Any content within a container has the rules of that container applied to it. For example, the text within a "boldface container" would be boldfaced. Similarly, paragraphs are defined using a "paragraph container."

Thinking of tag-sets as containers will help in another way: it will help you remember that tags should always be balanced. In other words, you should keep containers nested within each other.

"Document tags" means the tags which divide a Web page into its basic sections, such as the header information and the part of the page which contains the displayed text and graphics.

So, putting everything into one file, we have:

```
<HTML>
<HEAD>
<TITLE>Document Title</TITLE>
</HEAD>
<BODY>
</BODY>
</HTML>
```

6.1.4 Apache Tomcat Server:

Introduction:

For administrators and web developers alike, there are some important bits of information you should familiarize yourself with before starting out. This document serves as a brief introduction to some of the concepts and terminology behind the Tomcat container. As well, where to go when you need help.

Terminology

In the course of reading these documents, you'll run across a number of terms; some specific to Tomcat, and others defined by the [Servlet](#) or [JSP](#) specifications.

- a) Context - In a nutshell, a Context is a web application.
- b) Term2 (10) - This is it.
- c) Term3 - This is it!

Directories and Files

Throughout the docs, you'll notice there are numerous references to `$CATALINA_HOME`. This represents the root of your Tomcat installation. When we say, "This information can be found in your `$CATALINA_HOME/README.txt` file" we mean to look at the `README.txt` file at the root of your Tomcat install

These are some of the key tomcat directories, all relative to `$CATALINA_HOME`:

1. `/bin` - Startup, shutdown, and other scripts. The `*.sh` files (for UNIX systems) are functional duplicates of the `*.bat` files (for Windows systems). Since the Win32 (10) command-line

lacks certain functionality, there are some additional files in here.

2. /conf - Configuration files and related DTDs. The most important file in here is server.xml. It is the main configuration file for the container.
3. /logs - Log files are here by default.
4. /webapps - This is where your webapps go.

Configuring Tomcat

This section will acquaint you with the basic information used during the configuration of the container. All of the information in the configuration files is read at startup, meaning that any change to the files necessitates a restart of the container.

6.1.5 MYSQL

What is a Database?

A database is a structure that comes in two flavors: a flat database and a relational database. A relational database is much more oriented to the human mind and is often preferred over the gabble-de-gook flat databases that are just stored on hard drives like a text file. MySQL is a relational database.

MySql is a powerful database. It's very good and free of charge. Many developers in the world selected mysql and php for developing their website.

History of MySQL

We started out with the intention of using the MySQL database system to connect to our tables using our own fast low-level (ISAM) routines. However, after some testing, we came to the conclusion that mSQL was not fast enough or flexible enough for our needs. This resulted in a new SQL interface to our database but with almost the same API interface as MySQL. This API was designed to allow third-party code that was written for use with mSQL to be ported easily for use with MySQL.

MySQL is named after co-founder Monty Widenius's daughter, My. The name of the MySQL Dolphin (our logo) is "Sakila," which was chosen from a huge list of names suggested by users in our "Name the Dolphin" contest. The winning name was submitted by Ambrose Twebaze, an Open Source software developer from Swaziland, Africa. According to Ambrose, the feminine name Sakila has its roots in SiSwati, the local language of Swaziland. Sakila is also

the name of a town in Arusha, Tanzania, near Ambrose's country of origin, Uganda.

The history of MySQL dates back to 1995. For those of you that don't know what MySQL is, MySQL is a database program. Best MySQL Web Hosting has written this short article on the history of MySQL. We hope you learn from this article. As mentioned before, MySQL was started in 1995 under the name of MySQL AB. MySQL was founded by three guys, Michael Widenius, David Axmark, and Allan Larsson. The company was eventually sold in 2008 to Sun Microsystems at a cost of one billion dollars. MySQL has offices in Sweden and California.

Throughout MySQL's history, there have been numerous highlights that have made the computer industry better and what it is today. Those highlights are the following:

1998: In 1998, Windows decided to release their version for the Windows 95 and NT programs.

2001: MySQL released version 3.23.

2003: MySQL released version 4.0. Version 4.0 featured the union clause. The union clause allowed two data groups to be merged into one group.

2004: MySQL released version 4.1. Version 4.1 featured R-trees, B-trees, subqueries, and prepared statements. R-trees are data structures that are used for spatial access methods. B-trees are also data structures but their main purpose is to keep data sorted for easier access.

2005: MySQL released version 5.0. Version 5.0 contained cursors, triggers, views, stored procedures, and XA transactions. Cursors are important because they assist in the processing of rows within database queries. Triggers come in two different versions; row and statement. Basically, triggers are codes that are the end result of certain actions within the database.

2008: MySQL releases version 5.1. Version 5.1 includes partitioning, event scheduler, row based replications, server log tables, and plugin API.

MySQL is not stopping with version 5.1. Just like they have done since 1995, they are continuing to improve MySQL. MySQL 6.0 is already in the works and it is going to contain referential integrity, additional unicodes, and a new storage engine.

Not only is MySQL the world's most popular open source database, it's also become the database of choice for a new generation of applications built on the LAMP stack (Linux, Apache, MySQL, PHP / Perl / Python.) MySQL runs on more than 20 platforms including Linux, Windows, OS/X, HP-UX, AIX, Netware, giving you the kind of flexibility that puts you in control.

Whether you're new to database technology or an experienced developer or DBA, MySQL offers a comprehensive range of certified software, support, training and consulting to make you successful.

Advantages:

1. Scalability and Flexibility
2. High Performance
3. High Availability
4. Robust Transactional Support
5. Web and Data Warehouse Strengths
6. Strong Data Protection
7. Comprehensive Application Development
8. Management Ease
9. Open Source Freedom and 24 x 7 Supports
10. Lowest Total Cost of Ownership

7. SCREEN SHOTS

Index Page :



Fig:7.1 Index page

Faculty Registration Form :

This screenshot shows the first step of the faculty registration process. The browser window displays the URL 'localhost:8084/olcp/facreg1.html'. The page header includes a group photo of students and the title 'ONLINE COURSE PORTAL FOR A CAMPUS' with a home icon. The main heading is '!!!REGISTRATION!!!'. The form contains the following fields and values:

- Name: Srinivas
- Faculty id: sa0015
- Username: srinivas (with a red message 'Username Available to Use')
- Password: *****
- Confirm Password: *****
- Date of Birth: 1984, JAN, 1
- College: Sai Aditya
- Department: Computer Science
- Qualification: MTECH
- Interested Subjects: JAVA, SE, Network programming, OOAD

The Windows taskbar at the bottom shows the start button and several open applications including NetBeans IDE 6.5, Microsoft Office Word, and MySQL Workbench.

This screenshot shows the second step of the faculty registration process. The browser window displays the URL 'localhost:8084/olcp/facreg1.html'. The page header is identical to the first screenshot. The main heading is '!!!REGISTRATION!!!'. The form contains the following fields and values:

- Confirm Password: *****
- Date of Birth: 1984, JAN, 1
- College: Sai Aditya
- Department: Computer Science
- Qualification: MTECH
- Interested Subjects: JAVA, SE, Network programming, OOAD
- Address: kakinada
- Designation: AssociateProfessor

At the bottom of the form, there are 'Submit' and 'Reset' buttons. The Windows taskbar at the bottom shows the start button and several open applications including NetBeans IDE 6.5, Microsoft Office Word, and MySQL Workbench.

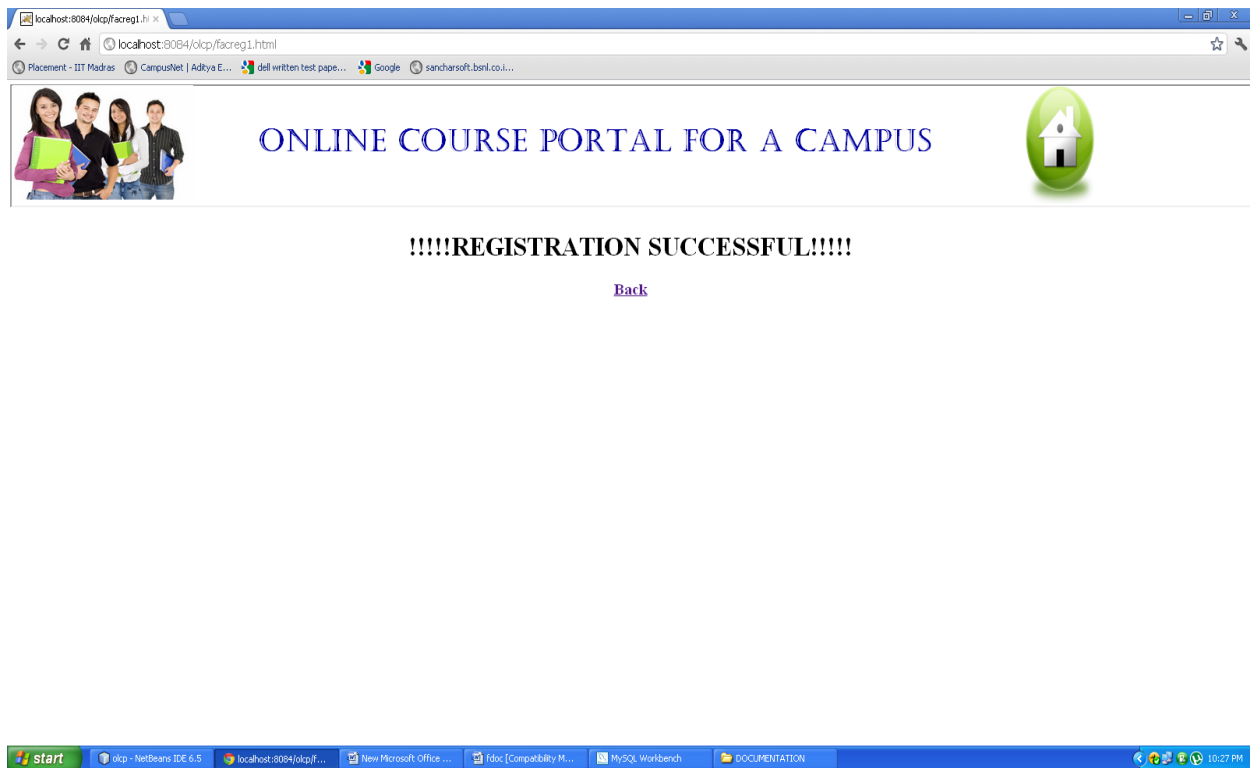


Fig:7.2.Faculty Registration Page

Admin Home Page:

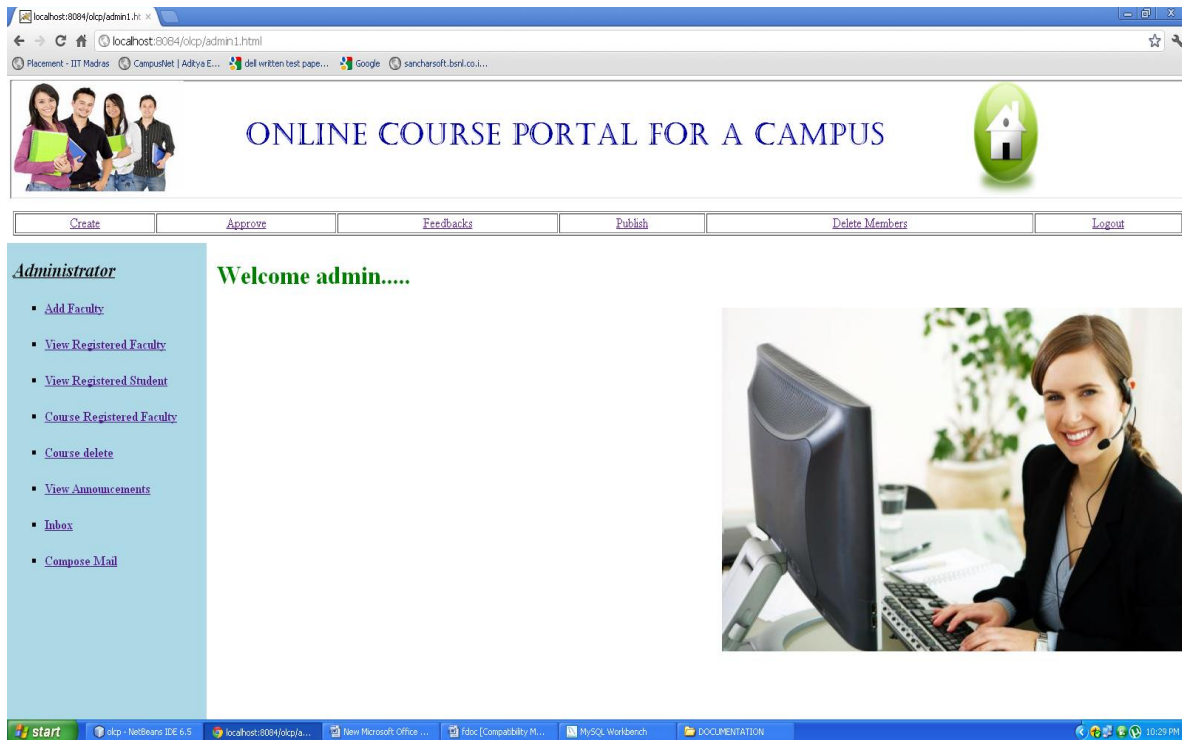


Fig:7.3.Admin Home Page

Course Creation By Admin :

The screenshot shows a web browser window with the URL `localhost:8084/olcp/admin1.html`. The page header features a navigation bar with links: [Create](#), [Approve](#), [Feedbacks](#), [Publish](#), [Delete Members](#), and [Logout](#). On the left, an 'Administrator' sidebar lists various management options. The main content area is titled 'COURSE CREATION' and contains a form with the following fields and buttons:

- CourseName :** A text input field containing the value 'HCI'.
- Upload Description File :** A 'Choose File' button followed by the text 'HUMAN CO...FACE.pdf'.
- Buttons:** 'create' and 'clear' buttons.

A large green 'create' logo is positioned on the right side of the form area. The browser's taskbar at the bottom shows several open applications, including NetBeans IDE 6.5, Microsoft Office Word, and MySQL Workbench.

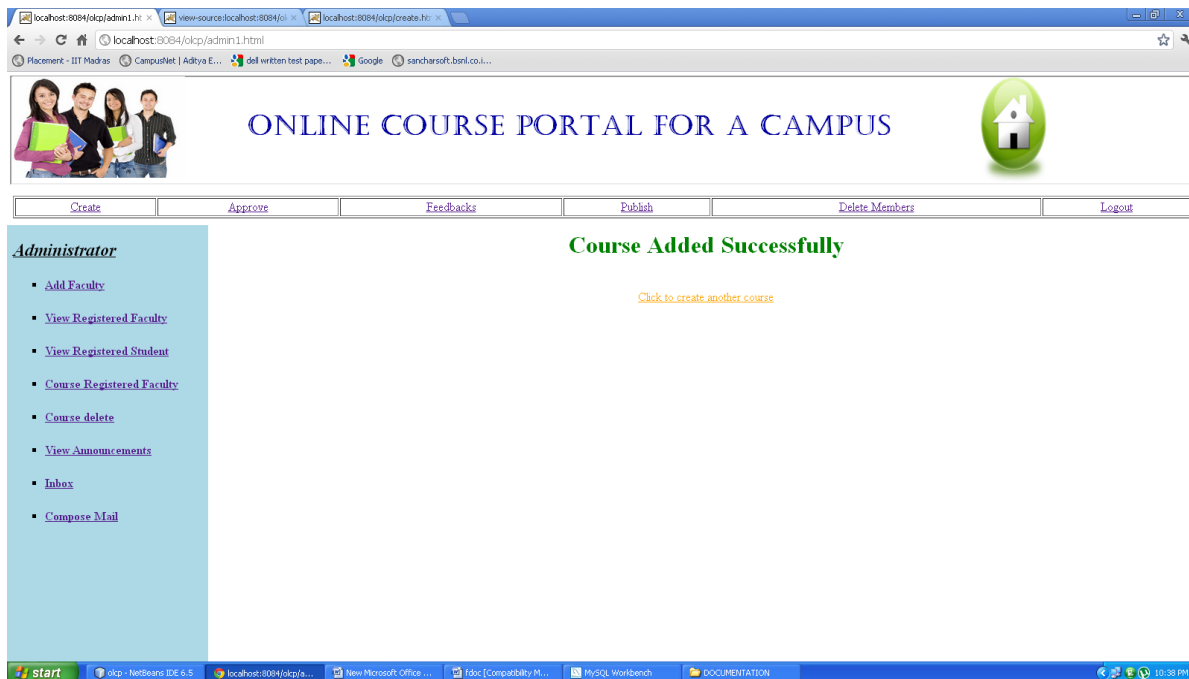


Fig:7.4.Course Creation Page

View Feedback Page:

The screenshot shows a web browser window displaying an online course portal. The browser's address bar shows the URL `localhost:8084/olcp/admin1.html`. The page header features a navigation bar with links: [Create](#), [Approve](#), [Feedbacks](#), [Publish](#), [Delete Members](#), and [Logout](#). Below the navigation bar, the page is titled "ONLINE COURSE PORTAL FOR A CAMPUS" with a green house icon on the right. On the left, there is a sidebar for the **Administrator** with links: [Add Faculty](#), [View Registered Faculty](#), [View Registered Student](#), [Course Registered Faculty](#), [Course delete](#), [View Announcements](#), [Inbox](#), and [Compose Mail](#). The main content area is titled "VIEW FEEDBACK..." and contains a form with a dropdown menu labeled "Select Course Name:" with "JAVA" selected, and a "submit" button. To the right of the form is a large image of a checklist with the following items: ☒ Excellent, ☐ Very good, ☐ Good, ☐ Average, and ☐ Poor. The Windows taskbar at the bottom shows the start button and several open applications: `olcp - NetBeans IDE 6.5`, `localhost:8084/olcp/a...`, `New Microsoft Office ...`, `doc [Compatibility M...`, `MySQL Workbench`, and `DOCUMENTATION`. The system clock shows 10:50 PM.

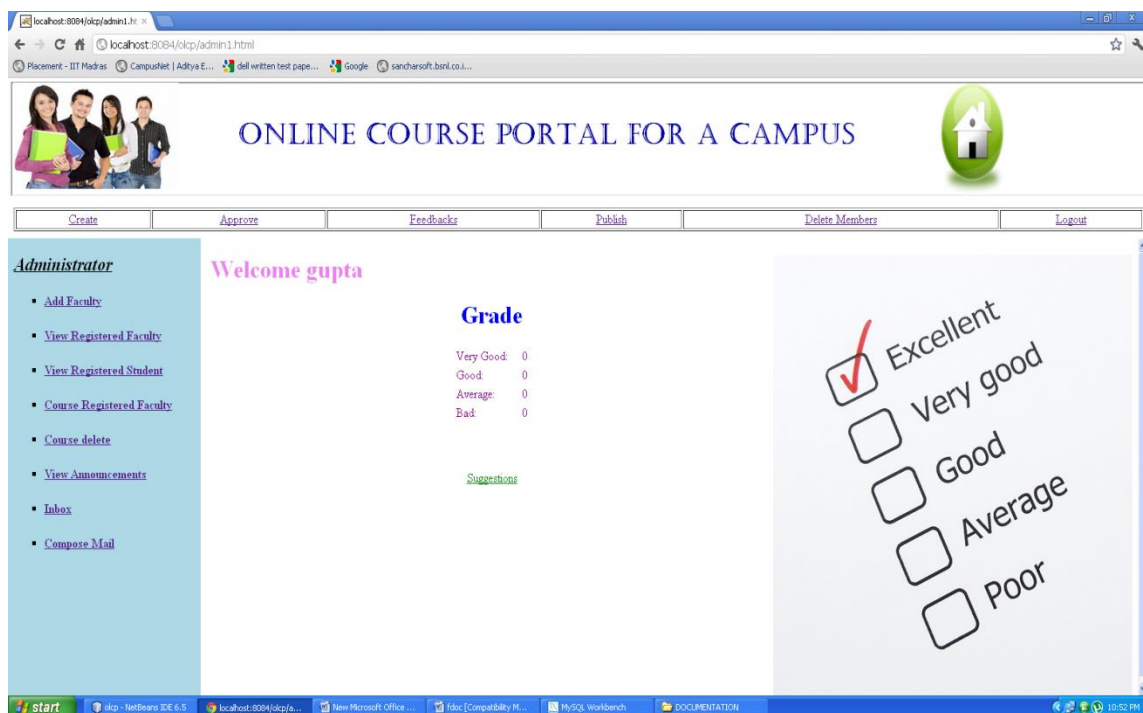
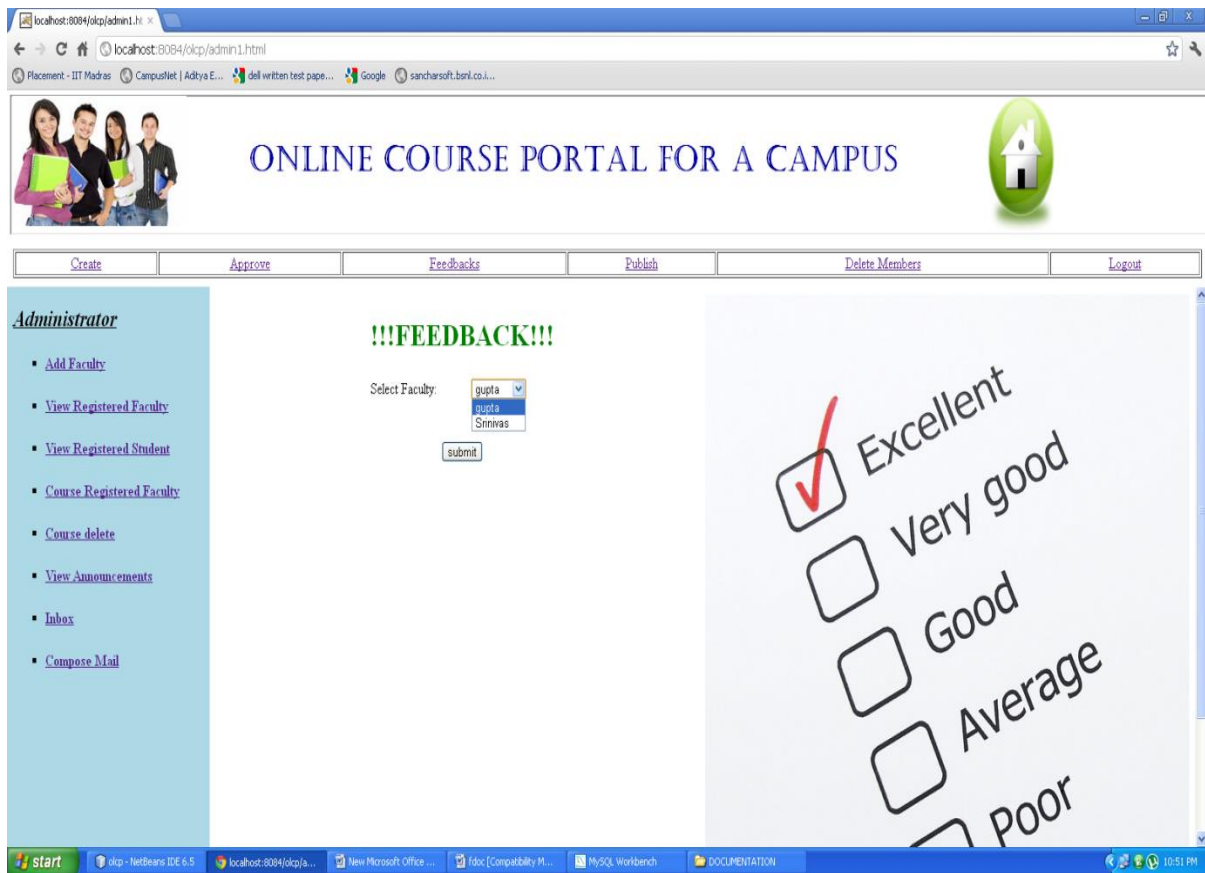
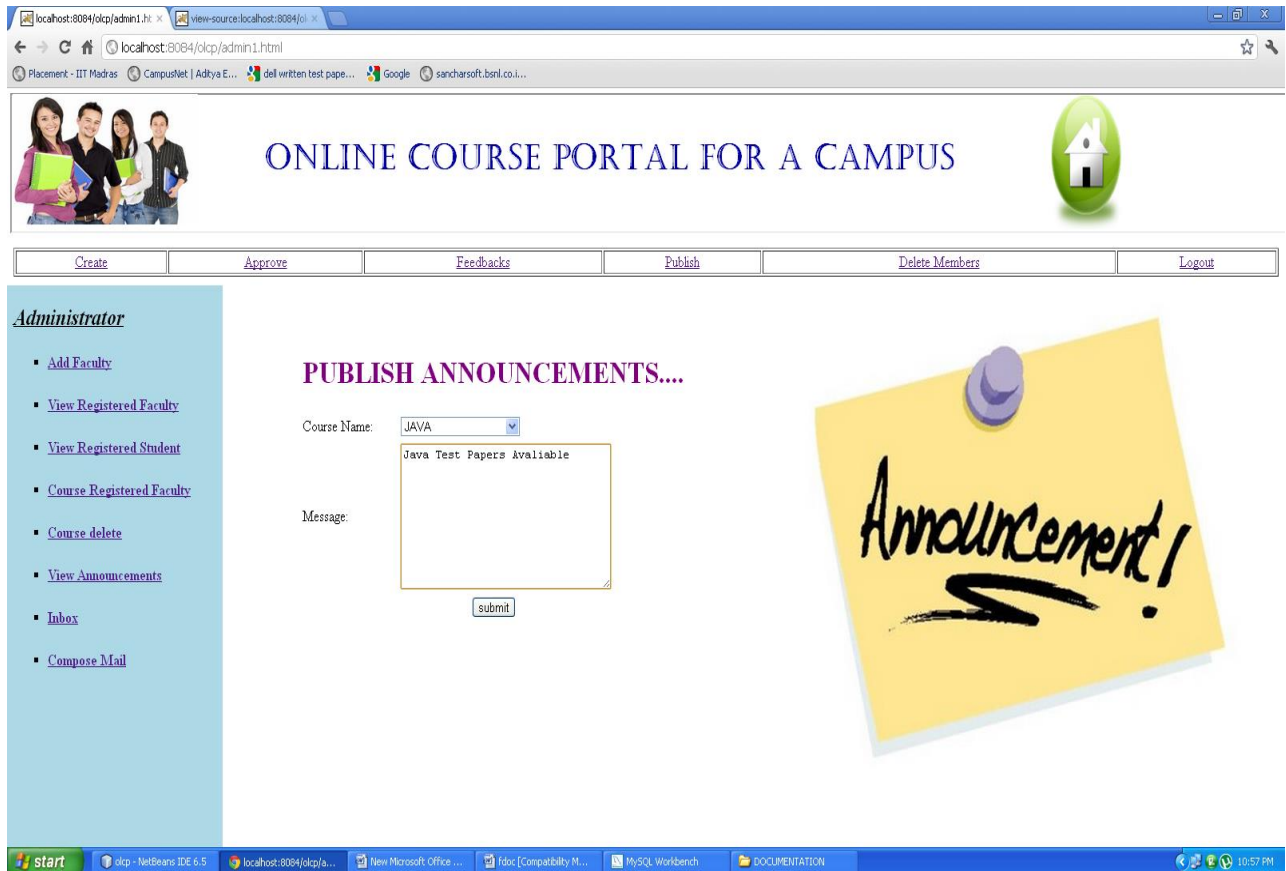


Fig:7.5 View Feedback Page

Publishing Announcements Page:



localhost:8084/olcp/admin1.ht x view-source:localhost:8084/olcp/admin1.html

Placement - IIT Madras CampusNet | Aditya E... del written test pape... Google sancharsoft.bsni.co.l...

 **ONLINE COURSE PORTAL FOR A CAMPUS** 

Create Approve Feedbacks Publish Delete Members Logout

Administrator

- [Add Faculty](#)
- [View Registered Faculty](#)
- [View Registered Student](#)
- [Course Registered Faculty](#)
- [Course delete](#)
- [View Announcements](#)
- [Inbox](#)
- [Compose Mail](#)

PUBLISH ANNOUNCEMENTS....

Course Name:

Message:



start NetBeans IDE 6.5 localhost:8084/olcp/a... New Microsoft Office ... Idoc [Compatibility M... MySQL Workbench DOCUMENTATION 10:57 PM

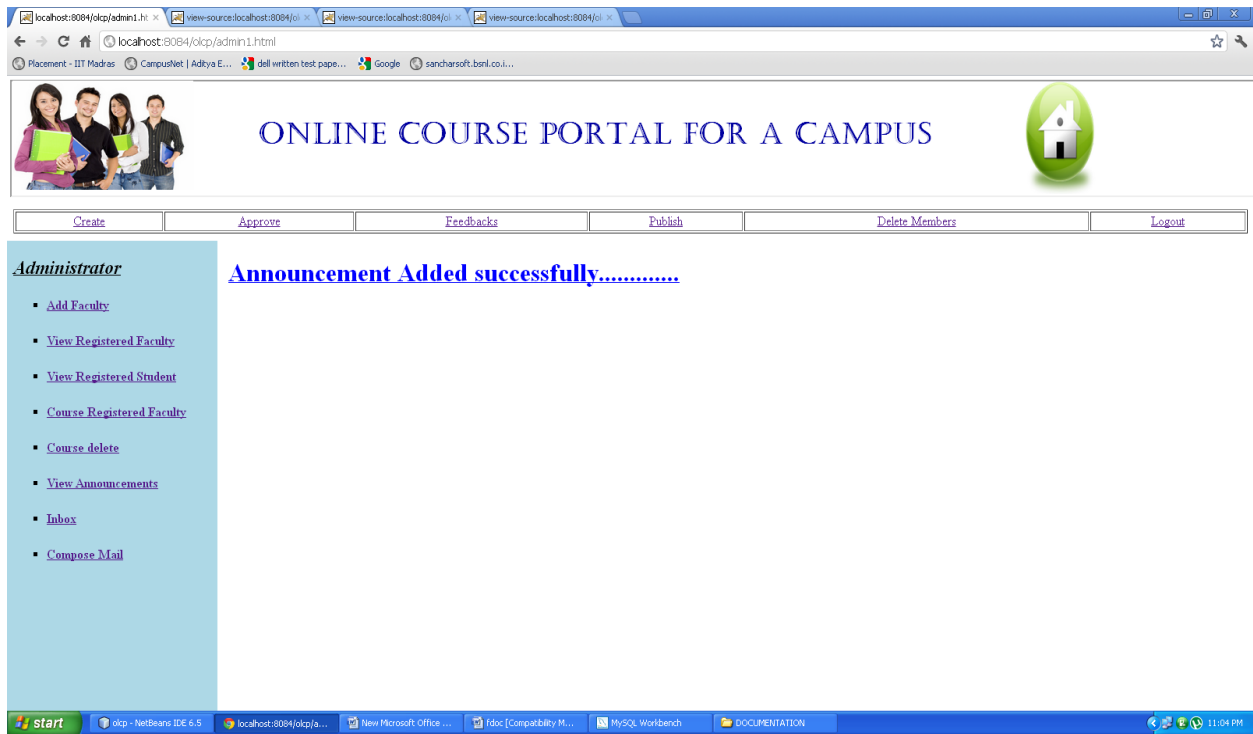


Fig:7.6.Publish announcements Page

Faculty Home Page:

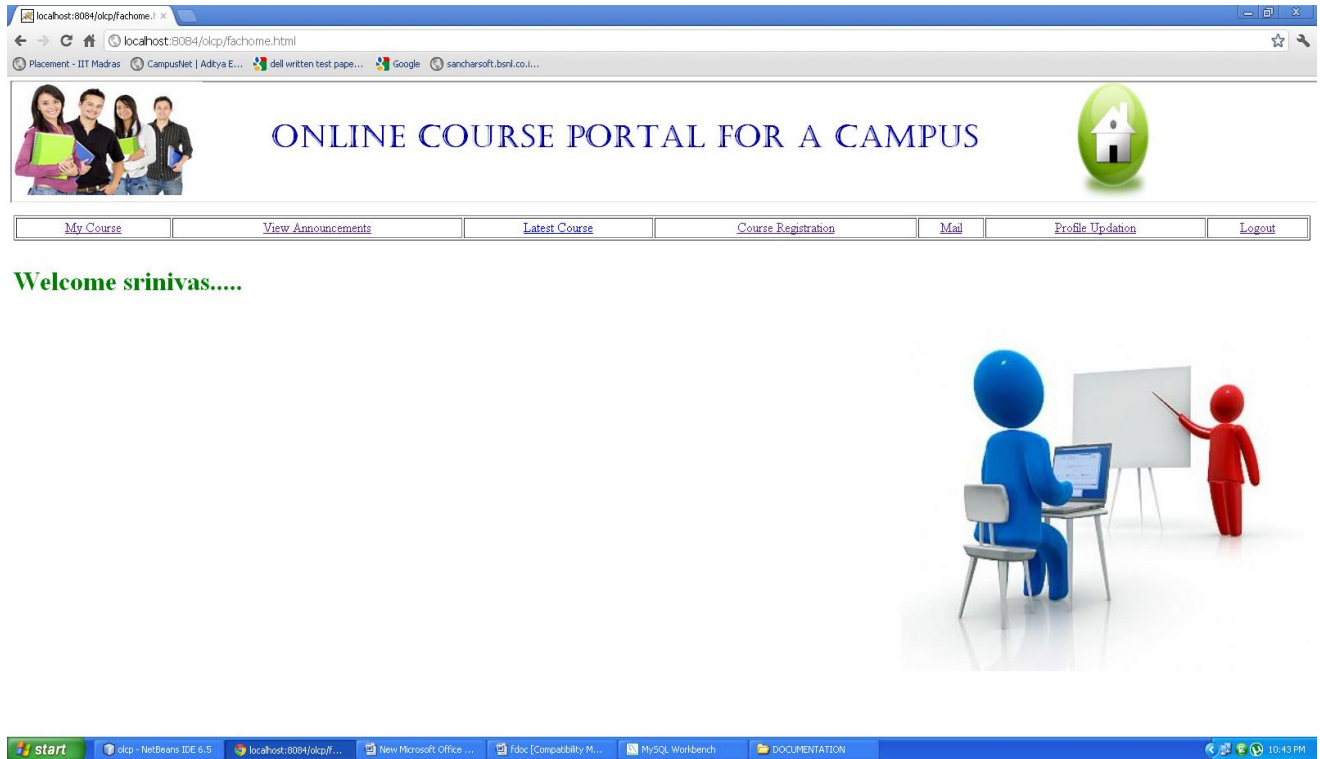
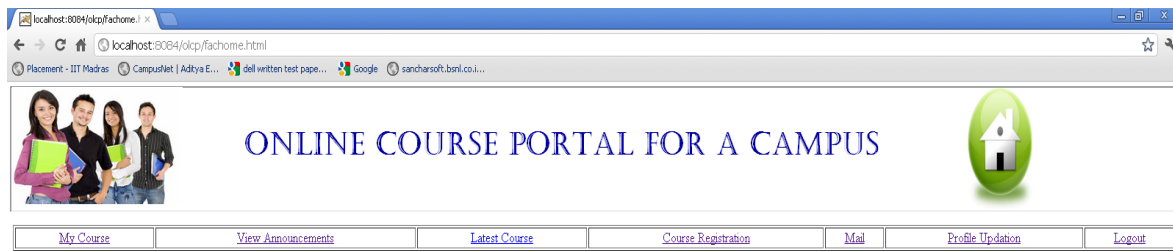


Fig:7.7.Faculty Home Page

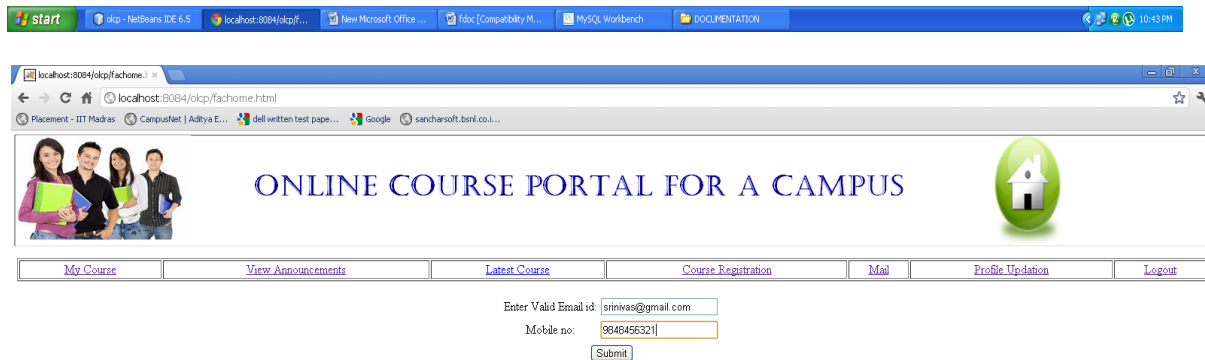
Course Registration Page :



The screenshot shows a web browser window with the URL `localhost:8084/olcp/fachome.html`. The page features a header with a group of students on the left, the title "ONLINE COURSE PORTAL FOR A CAMPUS" in the center, and a home icon on the right. Below the header is a navigation bar with links: [My Course](#), [View Announcements](#), [Latest Course](#), [Course Registration](#) (highlighted), [Mail](#), [Profile Updation](#), and [Logout](#).

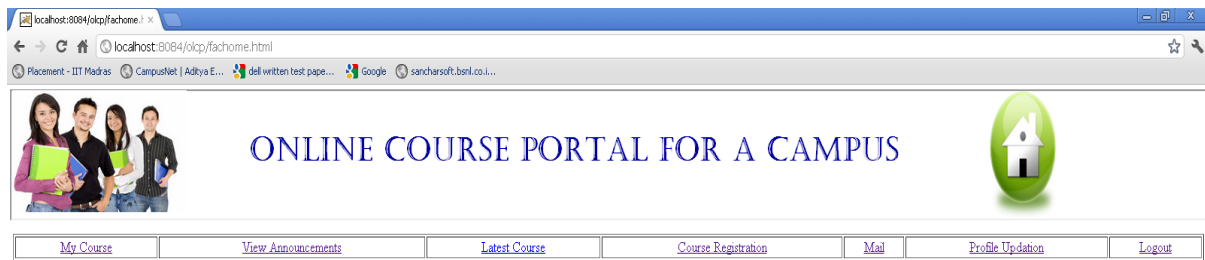
Courses

Course Name	Registration
JAVA	Register
SE	Register
Networkprogramming	Register
OOAD	Register
OS	Register
HCI	Register



This screenshot shows the same web page as above, but with a registration form displayed below the navigation bar. The form includes the following fields and elements:

- Enter Valid Email id:
- Mobile no:
-



Course Registration is done successfully.....

Fig:7.8.Course Registration Page

Courses Page:

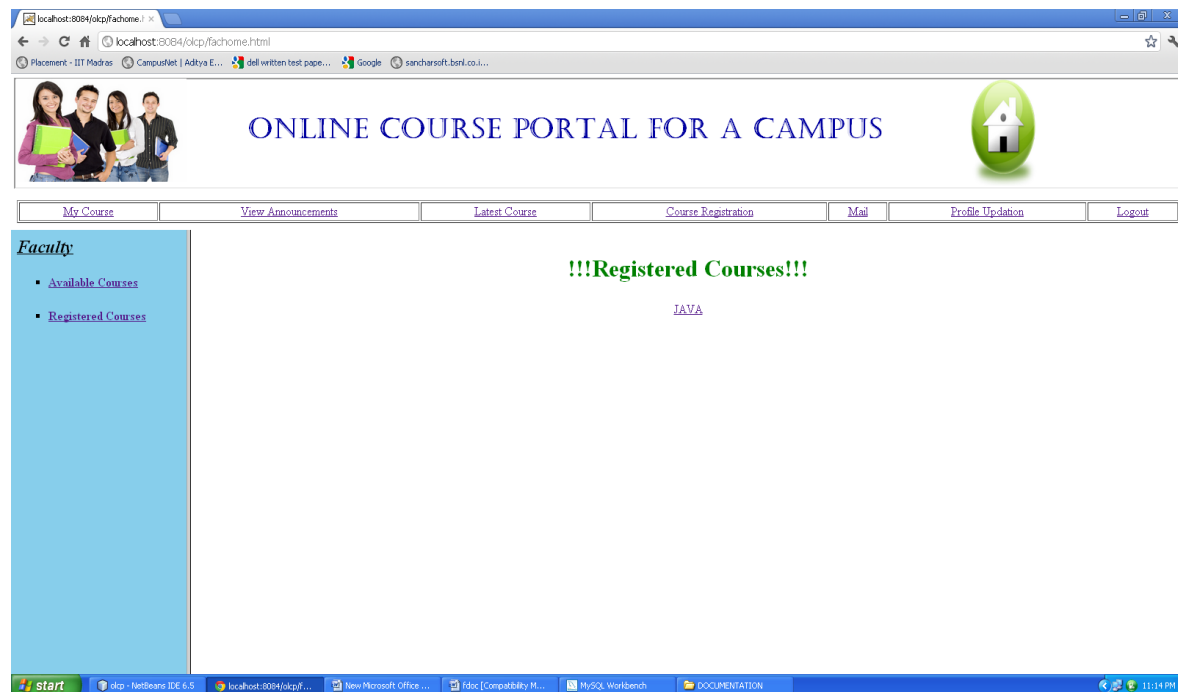
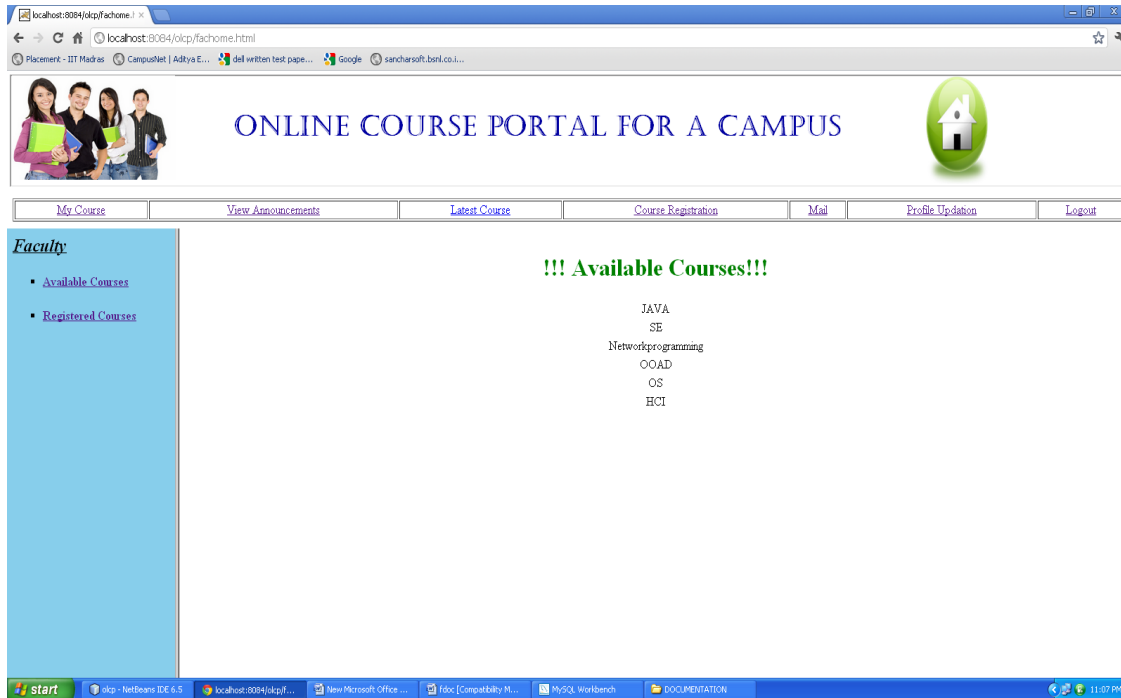


Fig:7.9.Course Page

Mailing Page :

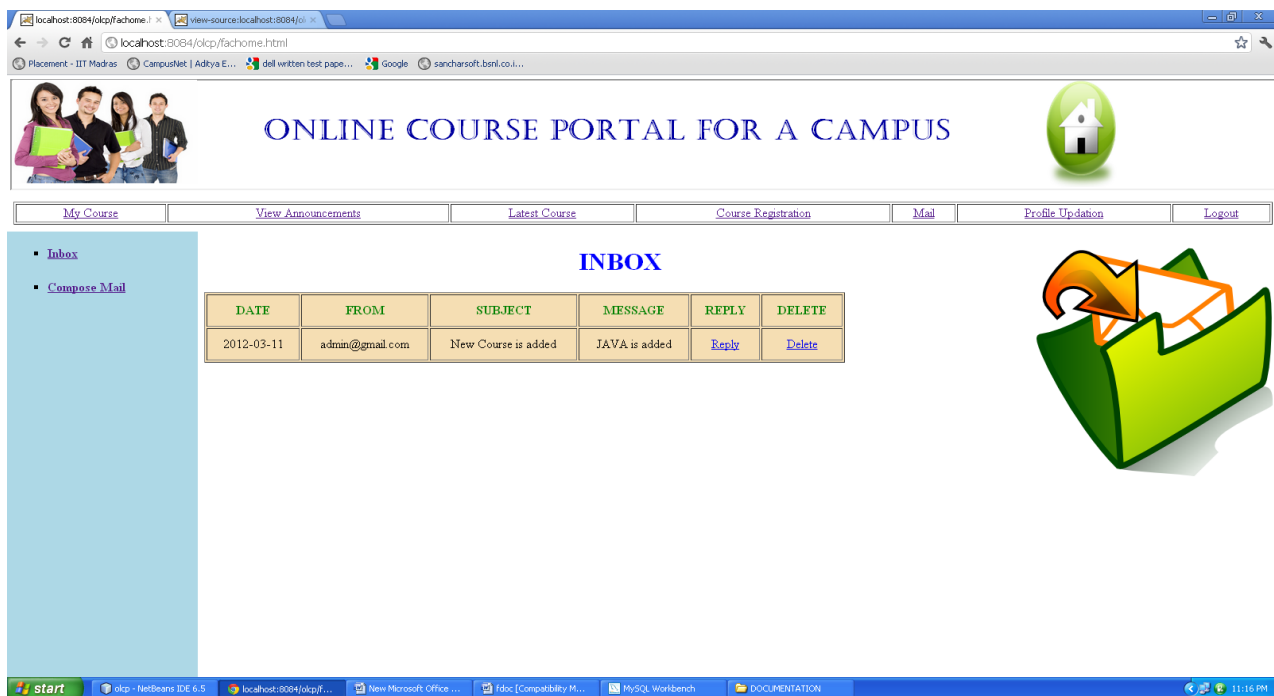


Fig:7.10.Inbox Page

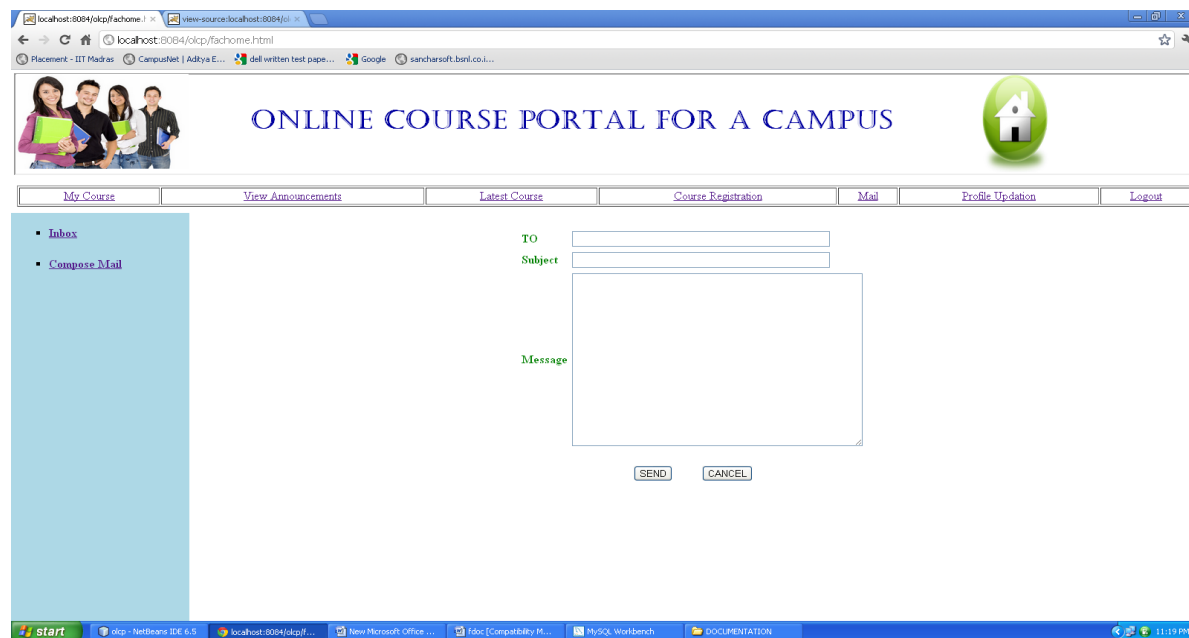
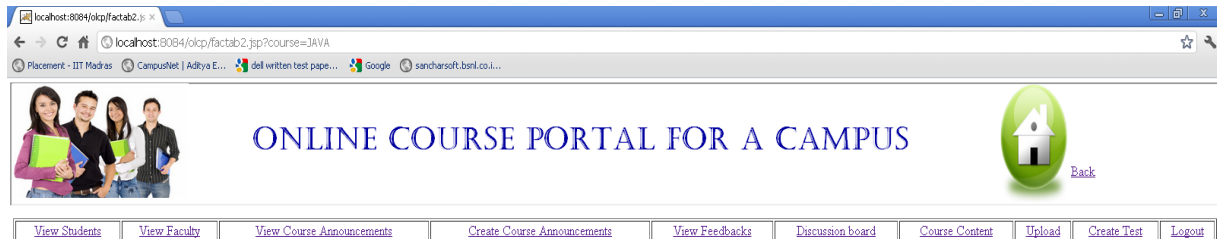


Fig:7.11.Compose Mail Page

Course(JAVA) Home Page For Faculty :



Java

Java is a programming language originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to bytecode (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is a general-purpose, concurrent, class-based, object-oriented language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere", meaning that code that runs on one platform does not need to be recompiled to run on another. Java is currently one of the most popular programming languages in use, particularly for client-server web applications, with a reported 10 million users. The original and reference implementation Java compilers, virtual machines, and class libraries were developed by Sun from 1995. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java and GNU Classpath.

Principles

There were five primary goals in the creation of the Java language:

- It should be "simple, object-oriented and familiar"
- It should be "robust and secure"
- It should be "architecture-neutral and portable"
- It should execute with "high performance"
- It should be "interpreted, threaded, and dynamic"

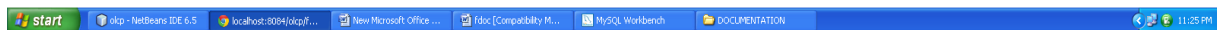
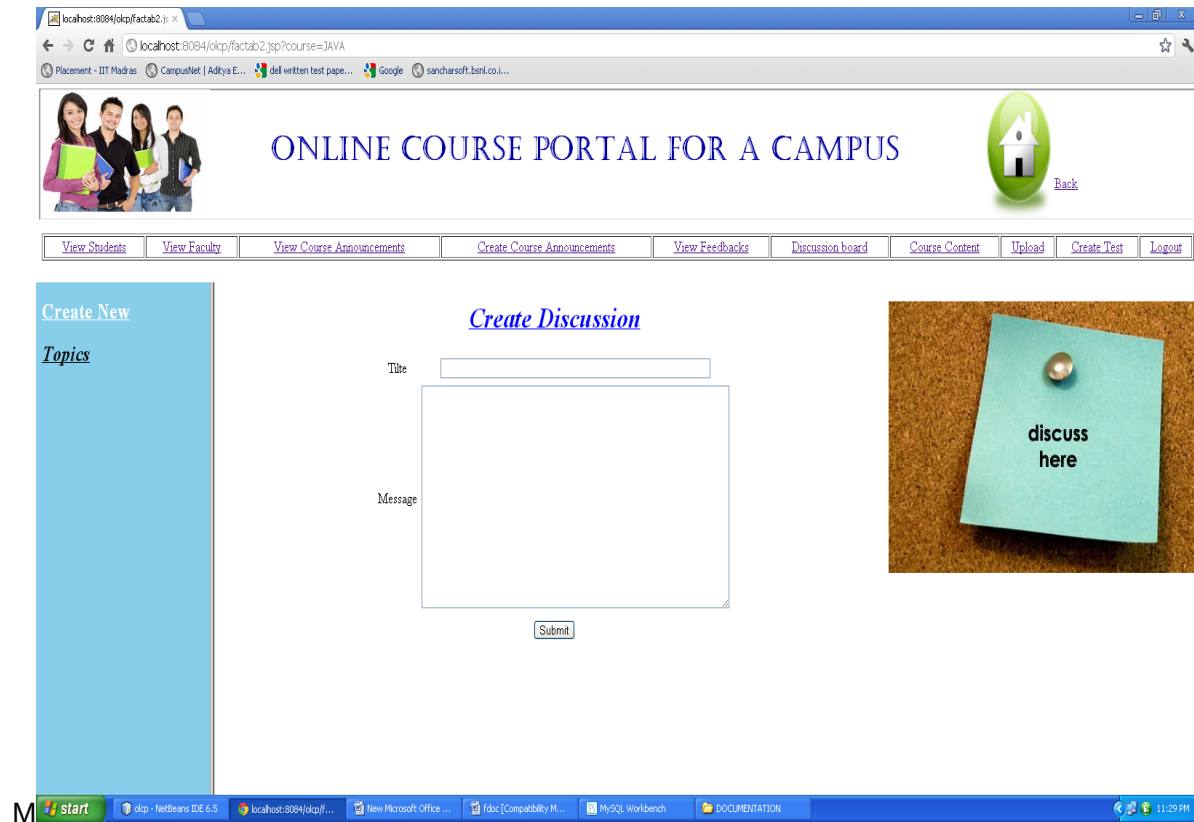


Fig:7.12.Course Home Page

Create Discussion Page:



The screenshot shows a web browser window displaying an online course portal. The browser's address bar shows the URL `localhost:8084/olcp/factab2.jsp?course=JAVA`. The page header features a logo of four students on the left, the title "ONLINE COURSE PORTAL FOR A CAMPUS" in the center, and a home icon with a "Back" link on the right. Below the header is a navigation menu with buttons for "View Students", "View Faculty", "View Course Announcements", "Create Course Announcements", "View Feedbacks", "Discussion board", "Course Content", "Upload", "Create Test", and "Logout".

The main content area is titled "Create New Topics" on the left. The central section is titled "Create Discussion" and contains a form with a "Title" label and a text input field, a "Message" label and a larger text area, and a "Submit" button at the bottom. To the right of the form is an image of a green sticky note pinned to a corkboard with the text "discuss here".

The Windows taskbar at the bottom shows the Start button and several open applications: NetBeans IDE 6.5, the current web browser window, New Microsoft Office Word, a file explorer window, MySQL Workbench, and a folder named DOCUMENTATION. The system clock in the bottom right corner shows the time as 11:29 PM.

Fig:7.13.Create discussion Page

File Uploading Page :

localhost:8084/olcp/factab2.jsp

localhost:8084/olcp/factab2.jsp?course=JAVA

Placement - IIT Madras | CampusNet | Aditya E... | del written test pape... | Google | sancharsoft.bsrl.co...

 **ONLINE COURSE PORTAL FOR A CAMPUS**  [Back](#)

[View Students](#) [View Faculty](#) [View Course Announcements](#) [Create Course Announcements](#) [View Feedbacks](#) [Discussion board](#) [Course Content](#) [Upload](#) [Create Test](#) [Logout](#)

MULTIPLE FILES UPLOAD

Select File: colz forwards.zip

Select File: No file chosen

Select File: No file chosen

start | olcp - NetBeans IDE 6.5 | localhost:8084/olcp/f... | New Microsoft Office ... | fdoc [Compatibility M... | MySQL Workbench | DOCUMENTATION | 11:31 PM

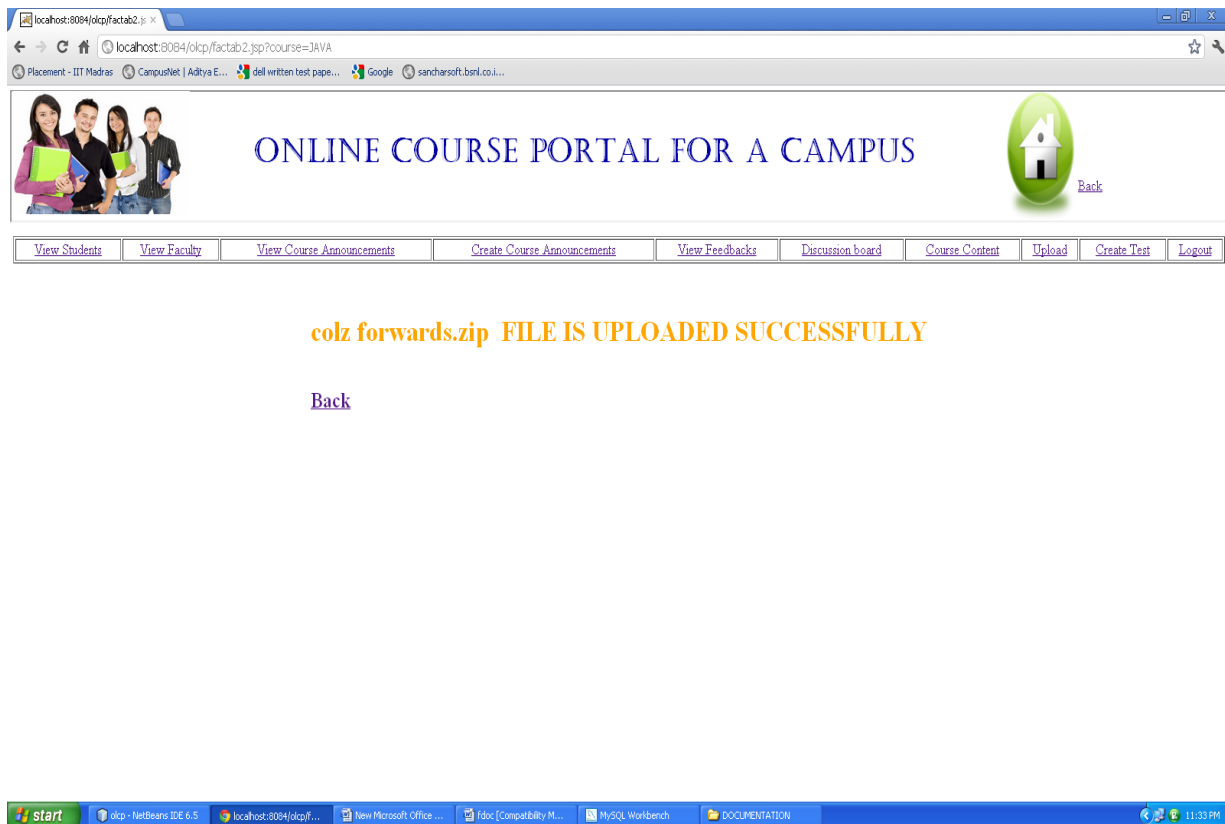


Fig:7.14.File Upload Page

Course Content (Downloading) Page:

The screenshot displays a web browser window at the URL `localhost:8084/olcp/factab2.jsp?course=JAVA`. The page header features a group of students and the title "ONLINE COURSE PORTAL FOR A CAMPUS". A navigation bar includes links such as "View Students", "View Faculty", "View Course Announcements", "Create Course Announcements", "View Feedbacks", "Discussion board", "Course Content", "Upload", "Create Test", and "Logout". The "Course Content" section is titled "Materials" and lists two items: "2012-03-11 [colt forwards.zip](#) Uploaded by minivas" and "2012-03-09 [fb.jpg](#) Uploaded by gupta". A "Save As" dialog box is open, showing the file "colt forwards" being saved as a "WinRAR ZIP archive" to the Desktop. The taskbar at the bottom shows various applications including NetBeans IDE 6.5, Microsoft Office, and MySQL Workbench.

Fig:7.15.Material Download Page

Test Creation Page :

The screenshot shows a web browser window displaying the 'Create Test' page of an 'ONLINE COURSE PORTAL FOR A CAMPUS'. The browser's address bar shows the URL `localhost:8084/olcp/factab2.jsp?course=JAVA`. The page header includes a logo of four students and a 'Back' button. Below the header is a navigation bar with links: [View Students](#), [View Faculty](#), [View Course Announcements](#), [Create Course Announcements](#), [View Feedbacks](#), [Discussion board](#), [Course Content](#), [Upload](#), [Create Test](#), and [Logout](#).

The 'Create Test' form is a pink box with the following fields:

- Enter Question:**
- Enter Answer(A):**
- Enter Answer(B):**
- Enter Answer(C):**
- Enter Answer(D):**
- Correct Answer:**

At the bottom of the form are two buttons: [Submit](#) and [Reset](#).

The Windows taskbar at the bottom shows the Start button and several open applications: olcp - NetBeans IDE 6.5, localhost:8084/olcp/f..., New Microsoft Office..., fdoc (Compatibility M..., MySQL Workbench, and DOCUMENTATION. The system clock shows 11:39 PM.

Fig:7.16.Create Test Page

Student Home Page:

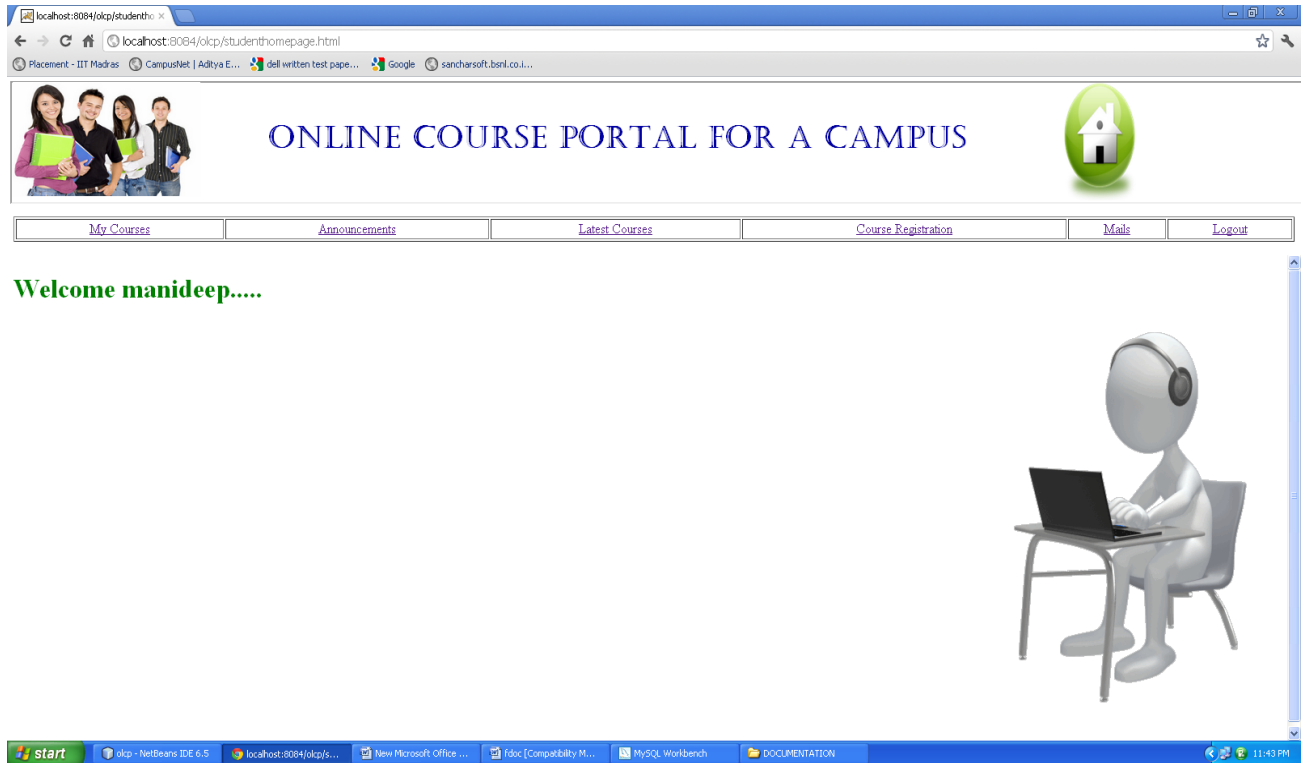


Fig:7.17.Student Home Page

Take aTest:

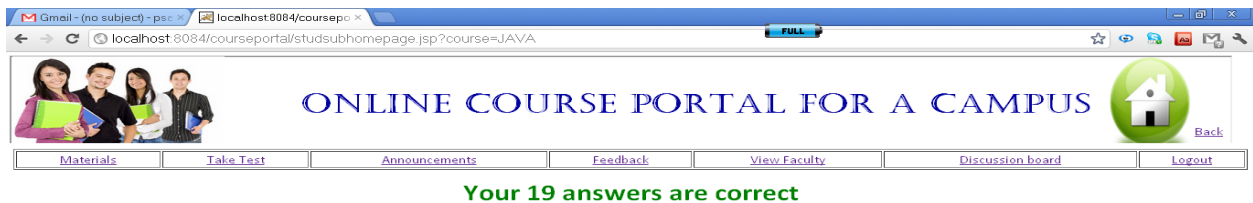
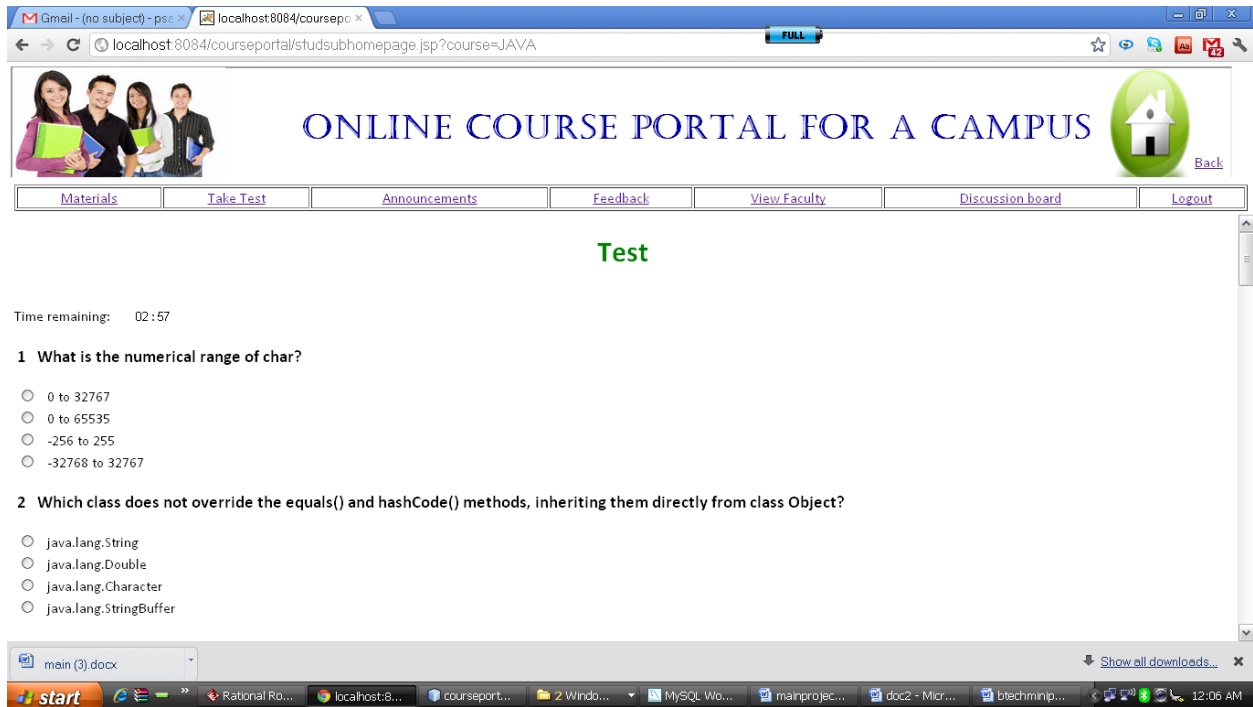


Fig:7.18.Take a Test Page

8.SAMPLE CODE

Test.jsp:

```
<%@page import="java.util.Random"%>
<%@page import="java.sql.*"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<html><head>
<%
String mins = request.getParameter( "mins" );
if( mins == null ) mins = "3";

String secs = request.getParameter( "secs" );
if( secs == null ) secs = "1";
%>
<script>
<!--
var mins = <%=mins%>; // write mins to javascript
var secs = <%=secs%>; // write secs to javascript
var t;
function timer()
{
// tic tac
if( --secs == -1 )
{
secs = 59;
--mins;
}

// leading zero? formatting
if( secs < 10 ) secs = "0" + secs;
if( mins < 10 ) mins = "0" + parseInt( mins, 10 );

// display
document.f1.mins.value = mins;
document.f1.secs.value = secs;

// continue?
if( secs == 0 && mins == 0 ) // time over
{
alert("Time Over");
document.f1.action="testresult.jsp";
document.f1.submit();
}
```

```

else // call timer() recursively every 1000 ms == 1 sec
{
t=window.setTimeout( "timer()", 1000 );
}
}
function stoper()      {
clearTimeout(t);
}

//-->
</script>
</head>
<body>
<formname="f1" action="testresult.jsp">
<center><h1><font color="green">Test</font></h1></center><br>
Time remaining: <input type="text" name="mins" size="1" style="border:0px solid black;text-align:right">:<input type="text" name="secs" size="1" style="border:0px solid black">
<table>
<script>
<!--
timer();
//-->
</script>
<%

ResultSet rs=null;
ResultSet rs1=null;
ResultSet rs2=null;
Connection con1=null;
Connection con2=null;
int count=0;

try{

String url="jdbc:mysql://localhost:3306/olcp?user=root&password=admin";
Class.forName("com.mysql.jdbc.Driver");
con1=DriverManager.getConnection(url);
Statement st=con1.createStatement();
String cname=(String)session.getAttribute("SessionCourse");
String q1="select Courseid from course where Cname='"+cname+"'";
rs1=st.executeQuery(q1);
int cid=0;
while(rs1.next())
{
cid=rs1.getInt(1);

```

```

}
int arr[]=new int[100];
int k=0;
rs2=st.executeQuery("select idtest from test where Courseid="+cid+"");
while(rs2.next()){
arr[k]=rs2.getInt(1);
k++;
}
if(k!=0){
Random r = new Random();
int a[] = new int[20];
for(int i=0;i<a.length;i++)
{
int t = r.nextInt(k);
a[i] = t;
int j;
for(j=0;t!=a[j]&&j<i;j++);
if(j!=i) i--;
}
int arr1[]=new int[20];
for(int i=0;i<a.length;i++)
{
rs=st.executeQuery("Select * from test where idtest = "+arr[a[i]]);
while(rs.next())
{
arr1[i]=arr[a[i]];
session.setAttribute("arr"+i,arr1[i]);
%>
<tr><td><h3><%= (i+1) %> &nbsp;&nbsp;&nbsp;<%=rs.getString("Question")%></h3></td><tr>
<td><input type="radio" value="<%=rs.getString("option1")%>"
name="radio<%=i%>" />&nbsp;&nbsp;&nbsp;<%=rs.getString("option1")%></td></tr><tr><td>
<input type="radio" value="<%=rs.getString("option2")%>"
name="radio<%=i%>" />&nbsp;&nbsp;&nbsp;<%=rs.getString("option2")%></td></tr><tr><td>
<input type="radio" value="<%=rs.getString("option3")%>"
name="radio<%=i%>" />&nbsp;&nbsp;&nbsp;<%=rs.getString("option3")%></td></tr><tr><td>
<input type="radio" value="<%=rs.getString("option4")%>"
name="radio<%=i%>" />&nbsp;&nbsp;&nbsp;<%=rs.getString("option4")%></td></tr>
<%
}
}
}
else
{
out.println("NO PAPER IS AVAILABLE");
}
}

```

```
} catch(Exception e)
{
out.println(e);
out.println("<h1>Error in your submission<br> So retry the exam</h1>");
}
%>
<tr><td><input type="submit" value="submit"></td></tr>
</table>
</form></body>
</html>
%>
```

9. CONCLUSION & FUTURE SCOPE

CONCLUSION:

The application, 'Online Course Portal for a Campus' has been developed as a web-based application so that students can get benefited as well as the organization as all the course materials can be available up to date for the students which also reduces the burden to the faculty in the organization. The Course Portal is a custom web-based application that integrates campus- specific information, course details, activities etc, so that students are aware of latest things like latest announcements, materials updated from time to time and utilize privileges like internal mailing system, feedback system, discussion board to interact with different users...etc

FUTURE ENHANCEMENT:

The scope for future enhancements of our project is good. The main extension is to make the mode of creating test in a more flexible manner and to make the portal include not only single cse course but also all the other courses.

10. REFERENCES

www.javasoft.com

www.sun.com

www.roseindia.com

www.wikipedia.com

www.w3schools.com

HTML, JSP books by O'Reilly.