DESCRIPTION OF THE SYSTEM

In today’s education system, question banks to help the students and the question papers to assess the students are created manually by the teachers/professors which requires a lot of time, research and efforts.

LIMITATIONS OF CURRENT SYSTEM

Lack of Security

1. Tampering of question papers

2. Physical destruction of question papers

3. Unauthorized access of question paper by another person

Overhead of Data management

1. Lots of paper work and research work

2. Lot of paper is required

Updating data is complicated

PROPOSED SYSTEM

The major objective of the project is to reduce the time taken to generate question banks and question papers with minimal efforts as the teachers/professors would just have to enter the questions and the pattern at one go and the software or the system generates the question paper and question banks for them.

PROPOSED SYSTEM REQUIREMENTS

1. To make provision for user friendly interface to enter all the required fields

2. Having a provision to keep a track of each question paper and also generate certain reports

3. Maintaining all the updated details.

ADVANTAGES OF PROPOSED SYSTEM

1. Data Security

- No tampering
- No physical destruction
- No unauthorized access

2. Faster processing

- Searching is easier hence the access time is reduced

3. User Friendly interface

4. Easy updating and modification.

5. Easy data interpretation

FEASIBILITY STUDY

Every project would be feasible, if provided with unlimited resources and unlimited time. But unfortunately, the development of computer-based system of product is more likely plagued by a security of resources and difficult time constraint. It is both necessary and prudent to evaluate the feasibility of the project at the earliest possible time. During the software engineering we concentrate our attention on four primary area of interest. The techniques used in conducting an initial investigation and feasibility study are very similar but the objectives are not so.

The major objectives of feasible study are to further define the problem and to determine the best way to solve it. The typical primary objectives are:

- A centralized database will be developed.
- Runtime operation decision-making will be determined.
- Unnecessary control procedure will be automated.
- The minimum of paper work should be produced.
- The output produced by the system should be in useable format.

Economic feasibility

The organization needs to provide a domain or sub-domain for this system.

The front end used is HTML-5, CSS, JavaScript and Bootstrap

The back end used is MySQL

The organization doesn’t require buying the product or renewing it once it has installed/deployed.

Technical Feasibility

Technology used in this project:

1. Front End: JSP / HTML

2. Back End: MySQL

Operational Feasibility

The management is in favor and providing full support for the project.

All the stakeholders are involved in the planning and development of the project thus reducing probability of any kind of resistance towards the proposal of the project.

Model Followed

Agile Development Model is followed for the project

STAKEHOLDERS

The system has identified three stake holders.

1. **Administrator**
   Here the administrator is referred to a person who is capable of making dynamic changes on the web-pages and also generating the reports as per the need. Also validate the new – user’s coming up. Administrator will have full control over the web-site and can also manipulate the data as and when needed
2. **User/Faculty(Including HOD)**
   Here the users are referred to people having a limited access over the web-site, the user here is capable of registering him to the web-site and hence a profile-page of the user would be created.
3. **Student**
   The web-site could also be used by common user who can simply view the web-page and check the current updates if any.
4. **Exam Team/Committee**
   The web-site could also be used by exam team who can simply view the web-page and check the current updates if any and print the question paper.
5. **Parents**
   The web-site could also be used by parents to view the students’ progress time to time.

QUESTIONNAIRE

1.Do you need a data driven website?

a. Yes b. No

2.Do you want ongoing code maintenance for security and updates?

a. Yes b. No

3.Who is your main target Audience ?

a. Government organisations
b. Companies
c. Educational institutions

4.What is your primary reason for having this type of website?

a. For having a questionnaire / get reviews from the ppl
b. For testing the knowledge of a person
c. Conducting examinations

5.Do you have brand guidelines, logos, and brand colors to incorporate into the site?

a.Yes
b.No

6.What is that phenomenon that best portrays your company's purpose for the software?

a.effective
b.creative
c.professional
d.all of the above

7.What platforms are you interested in using do you have experience with?

a.Joomla
b.Drupal
c.Magneto
d.None of the above

8.What sections of the site that will need regular updating after launch?

a. Main page of the site
b. Other pages inside the website
c. Specify if any ________

9.Maintenance and updates after launch would be?

a.Monthly
b.Quarterly
c.Yearly

10.What type of look are you trying to achieve?

a.high-tech
b.warm
C.flashy

11.Creating question papers manually is a tough task

a.Yes b.No

12.If,yes,why?

a.Takes a lot time b.Lot of resources are required
c.Repetition happens d.All of the above

13.If required,you would provide your credentials to the system?

a.yes b.No

14.Do you require updation features on any module?

a.Yes b.No

15. If there's a login system ,how would you like the updates given to the users?

a. via sms
b. via email
c. Specify if any ______

16.Do you require your site to be mobile friendly (responsive design)?

a. Yes b.No

17.Do you need any password protected areas?

a. Yes b.No

TECHNOLOGIES USED

Front-End:
HTML-5, CSS, JavaScript and Bootstrap

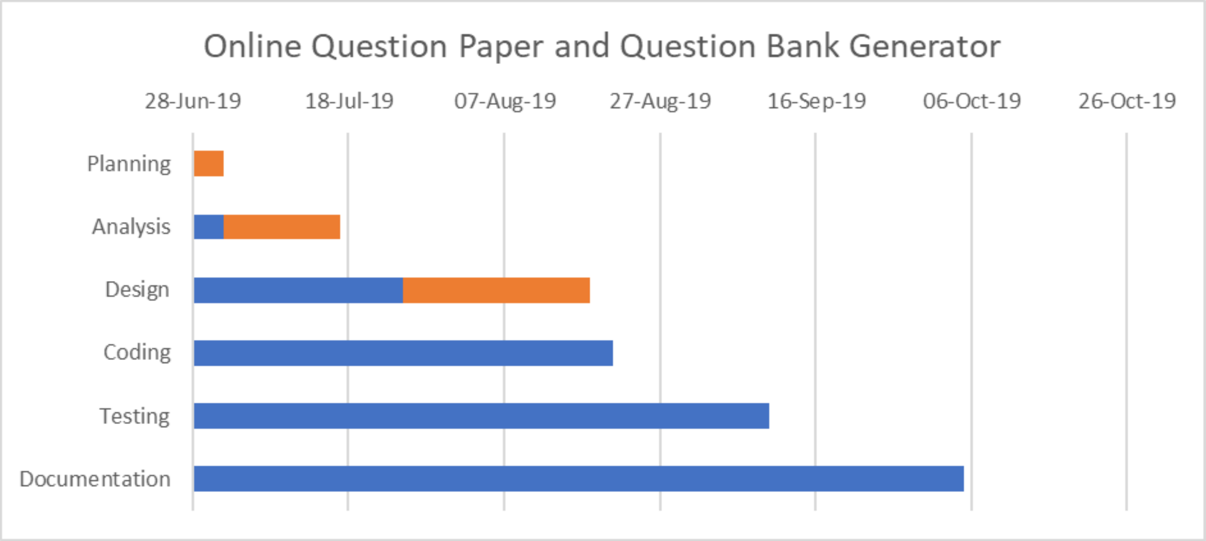
Back-End:
MySQL

STATUS CHART

Sr.no	**Phase**	**Duration days**	**Start date/** **Expected Start Date**	**End date**	**Signature**
1	**Project Planning Phase**	4	28/06/19	01/07/19	
2	**Analysis Phase**	15	02/07/19	17/07/19	
3	**Design Phase**	24	25/07/19	19/08/19	
4	**Implementation/** **Coding Phase**		21/08/19		
5	**Testing Phase**		10/09/19		
6	**Documentation Phase**		05/10/19		

GANTT CHART

A Gantt chart is constructed with a horizontal axis representing the total time span of the project, broken down into increments (for example, days, weeks, or months) and a vertical axis representing the tasks that make up the project (for example, if the project is outfitting your computer with new software, the major tasks involved might be: conduct research, choose software, install software). Horizontal bars of varying lengths represent the sequences, timing, and time span for each task. Using the same example, you would put "conduct research" at the top of the vertical axis and draw a bar on the graph that represents the amount of time you expect to spend on the research, and then enter the other tasks below the first one and representative bars at the points in time when you expect to undertake them. The bar spans may overlap, as, for example, you may conduct research and choose software during the same time span. As the project progresses, secondary bars, arrowheads, or darkened bars may be added to indicate completed tasks, or the portions of tasks that have been completed. A vertical line is used to represent the report date.



Gantt chart of progress till date

EVENT TABLE

Event	Source	Trigger	Activity	Response	Destination
Faculty wants to register	Faculty	Insert faculty details	Generate a new user id and password	Provide the user id and password to the faculty	Faculty, Admin, HOD
Faculty wants to sign in	Faculty	Fetch and compare user id and password	Validation	1.If credentials are correct: Grant access 2.If details are incorrect: Deny access	Admin, Faculty
Faculty selects a subject and adds number of units	Faculty	Insert number of units	Validation	1.If Units value is correct, accept 2. If unit value is incorrect, deny	Faculty, HOD, Admin
Faculty wants to enter theory questions (unit wise)	Faculty	Instantiate appropriate table	Validation	Take the faculty to appropriate interface	Faculty
Faculty selects a unit and type of questions (subjective, objective, both)	Faculty	Check whether the question already exists, if not insert the question	Validation	Question inserted or rejected	Faculty

Faculty wants to add Practical questions	Faculty	Check whether the question already exists, if not insert the question	Validation	Question inserted or rejected	Faculty
Faculty wants to delete an objective question	Faculty	Fetch the question from the table	Validation and Deletion	If question exists, delete	Faculty
Faculty wants to delete a subjective question	Faculty	Fetch the question from the table	Validation and Deletion	If question exists, delete	Faculty
Faculty wants to delete a practical question	Faculty	Fetch the question from the table	Validation and Deletion	If question exists, delete	Faculty
Faculty wants to update a subjective question	Faculty	Fetch the question from the table	Validation and Updating	If question exists, update	Faculty
Faculty wants to update an objective question	Faculty	Fetch the question from the table	Validation and Updating	If question exists, update	Faculty
Faculty enters number of questions pattern wise	Faculty	Instantiate the questions table	Validation	Ask the faculty for number of sets	Faculty, HOD
Faculty wants to create set of papers	Faculty	Insert sets value to the table	Validation	Ask the faculty for type of questions	Faculty, HOD
Faculty wants to update a	Faculty	Fetch the question	Validation and	If question	Faculty

practical question		from the table	Updating	exists, update	
Faculty wants to generate objective questions	Faculty	Fetch the question from the table	Validation and generation	Generate a pdf of objective questions	Faculty
Faculty wants to generate subjective questions	Faculty	Fetch the question from the table	Validation and generation	Generate a pdf of subjective questions	Faculty
Faculty wants to generate objective and subjective questions	Faculty	Fetch the question from the table	Validation	Generate a pdf of objective and subjective questions	Faculty
Faculty enters number of questions pattern wise	Faculty	Instantiate the questions table	Validation and generation	Ask the faculty for number of sets	Faculty, HOD
Faculty wants to generate practical questions	Faculty	Fetch the question from the table	Validation and generation	Generate a pdf of practical questions	Faculty, HOD
Faculty sends the paper for approval	Faculty	Fetch the question paper(s) from the table	Validation	Confirmation of paper sent	Faculty, HOD
Faculty wants to generate Question bank.	Faculty	Fetch question from the table	Validation and generation	Generate a pdf of all subjective questions.	Faculty

HOD wants to sign in	HOD	Fetch and compare user id and password	Validation	1.If credentials are correct: Grant access 2.If details are incorrect: Deny access	HOD
HOD wants to add course	HOD	Check whether the course already exists, if not insert the course	Validation and addition of course	course inserted or rejected	HOD
HOD wants to remove course	HOD	Check whether the course already exists, if exists then remove.	Validation and removal of course	course deleted	HOD
HOD wants to update course	HOD	Check whether the course already exists, if exists then update course	Validation and updating course	course updated	HOD
HOD wants to add subjects	HOD	Check whether the subject already exists, if not insert the subject	Validation and addition of subjects	Subject inserted or rejected	HOD
HOD wants to remove subjects	HOD	Check whether the course already exists, if exists then removes subject.	Validation and removal of subjects	Subject deleted	HOD
HOD wants to approve or disapprove Generated question paper	HOD	Fetch the question paper(s) from the table	Validation	Approval or Disapproval sent to Faculty	Faculty, HOD

HOD send the question paper for printing	HOD	Fetch the question paper(s) from the table	Validation	Confirmation of paper sent	HOD
HOD selects a subject and adds number of units	HOD	Insert number of units	Validation	1.If Units value is correct, accept 2. If unit value is incorrect, deny	HOD, Admin
HOD wants to enter theory questions (unit wise)	HOD	Instantiate appropriate table	Validation	Take the HOD to appropriate interface	HOD
HOD selects a unit and type of questions (subjective, objective, both)	HOD	Check whether the question already exists, if not insert the question	Validation	Question inserted or rejected	HOD
HOD wants to add Practical questions	HOD	Check whether the question already exists, if not insert the question	Validation	Question inserted or rejected	HOD
HOD wants to delete an objective question	HOD	Fetch the question from the table	Validation	If question exists, delete	HOD
HOD wants to delete a subjective question	HOD	Fetch the question from the table	Validation	If question exists, delete	HOD

HOD wants to delete a practical question	HOD	Fetch the question from the table	Validation	If question exists, delete	HOD
HOD wants to update a subjective question	HOD	Fetch the question from the table	Validation	If question exists, update	HOD
HOD wants to update an objective question	HOD	Fetch the question from the table	Validation	If question exists, update	HOD
HOD enters number of questions pattern wise	HOD	Instantiate the questions table	Validation	Ask the HOD for number of sets	HOD
HOD wants to create set of papers	HOD	Insert sets value to the table	Validation	Ask the HOD for type of questions	HOD
HOD wants to update a practical question	HOD	Fetch the question from the table	Validation	If question exists, update	HOD
HOD wants to generate objective questions	HOD	Fetch the question from the table	Validation	Generate a pdf of objective questions	HOD
HOD wants to generate subjective questions	HOD	Fetch the question from the table	Validation	Generate a pdf of subjective questions	HOD

HOD wants to generate objective and subjective questions	HOD	Fetch the question from the table	Validation	Generate a pdf of objective and subjective questions	HOD
HOD enters number of questions pattern wise	HOD	Instantiate the questions table	Validation	Ask the faculty for number of sets	HOD
HOD wants to generate practical questions	HOD	Fetch the question from the table	Validation	Generate a pdf of practical questions	HOD
HOD wants to generate Question Bank.	HOD	Fetch question from the table	Validation	Generate a pdf of all subjective questions.	HOD
Admin wants to add streams	Admin	Check whether the stream already exists, if not insert the stream	Validation	Stream inserted	Admin, HOD, Faculty
Admin wants to remove streams	Admin	Check whether the stream already exists, if exists remove stream	Validation	Stream removed	Admin, HOD, Faculty
Admin wants to Update stream	Admin	Check whether the stream already exists, if exists update stream	Validation	Stream updated	Admin, HOD, Faculty
Admin wants to add department	Admin	Check whether the department already exists, if not insert the stream	Validation	department inserted	Admin, HOD, Faculty

Admin wants to remove department	Admin	Check whether the department already exists, if exists remove department	Validation	department deleted	Admin, HOD, Faculty
Admin wants to Update department	Admin	Check whether the department already exists, if exists update stream	Validation	department updated	Admin, HOD, Faculty
Admin wants to add Faculty	Admin	Insert the value into the table	Validation	Faculty inserted and login id and password created	Admin, HOD, Faculty
Admin wants to remove Faculty	Admin	Check whether the Faculty already exists, If exists remove Faculty	Validation	Faculty deleted	Admin , HOD , Faculty
Admin wants to Update Faculty	Admin	Check whether the Faculty already exists, If exists remove Faculty	Validation	Faculty updated	Admin , HOD , Faculty
Admin wants to appoint a HOD	Admin	Insert the value into the table	Validation	HOD appointed	Admin, HOD, Faculty
Admin wants to remove a HOD	Admin	Check whether the HOD already exists, If exists remove HOD.	Validation	HOD deleted	Admin , HOD , Faculty
Admin wants to see reports, logs, error	Admin	Fetch the report from the table	Validation	Reports are presented	Admin

Examination Committee wants to access question paper	Examination committee	Fetch question paper from the table	Validation	Access the question paper for printing	Faculty, HOD
Examination Committee wants to confirm paper	Examination Committee	Change the value of flag (Flag Up)	Validation	Confirmation of paper sent	Faculty , HOD
HOD wants to see notification	HOD	Fetch the value of flag	Validation	Display the confirmation of the paper	Admin, HOD
Faculty wants to see notification	Faculty	Fetch the value of flag	Validation	Display the confirmation of the paper	Admin, HOD, Faculty
HOD Wants to view faculty dashboard	HOD	Fetch faculty details	Validation and display	Display the dashboard of the faculty	HOD, Admin
Reset Password	HOD, Admin, Examination Team, Faculty	Fetch and compare Old Password	Ask for new password	Reset the password	HOD, Admin, Examination Team, Faculty
Forgot Password	HOD, Admin, Examination Team, Faculty	Fetch and compare username/email id	Send a reset password mail	Reset and sign in again	HOD, Admin, Examination Team, Faculty
Admin wants to check notification	Admin	Fetch flag value	Display the notification	View the notification	Admin

HOD wants to set a pattern	HOD	Insert the pattern	Validation	Pattern Accepted/ Rejected	Admin, Faculty, HOD
HOD wants to enter a timetable	HOD	Insert timetable	Validation	Timetable accepted/ rejected	Admin, Faculty, HOD
HOD wants to delete a timetable	HOD	Delete timetable	Validation	Timetable deleted	Admin, Faculty, HOD
HOD wants to update a timetable	HOD	Update timetable	Validation	Timetable updated	Admin, Faculty, HOD
Student Registers	Student	Registration	Validation	Accept Student entry	Admin, Student
Student Logs in	Student	Check Username and password	Validation	Grant Access	Admin, Student
Student fills in profile	Student	Add details	Validation	Details Locked	Admin, Student
Student selects a scheme	Student	Select a scheme/package	Validation	Activate a package	Admin, Student
Student makes payment	Student	Initiate payment	Validation	Lock scheme and payment details	Admin, Student, Bank
Student updates a scheme	Student	Select a scheme/package	Validation	Activate a package	Admin, Student

Student withdraws from a scheme	Student	Remove a scheme/package	Validation	Deactivate a package	Admin, Student
Student submits a feedback	Student	Add feedback	Validation	Add feedback and reviews	Admin, Student

UML DIAGRAMS

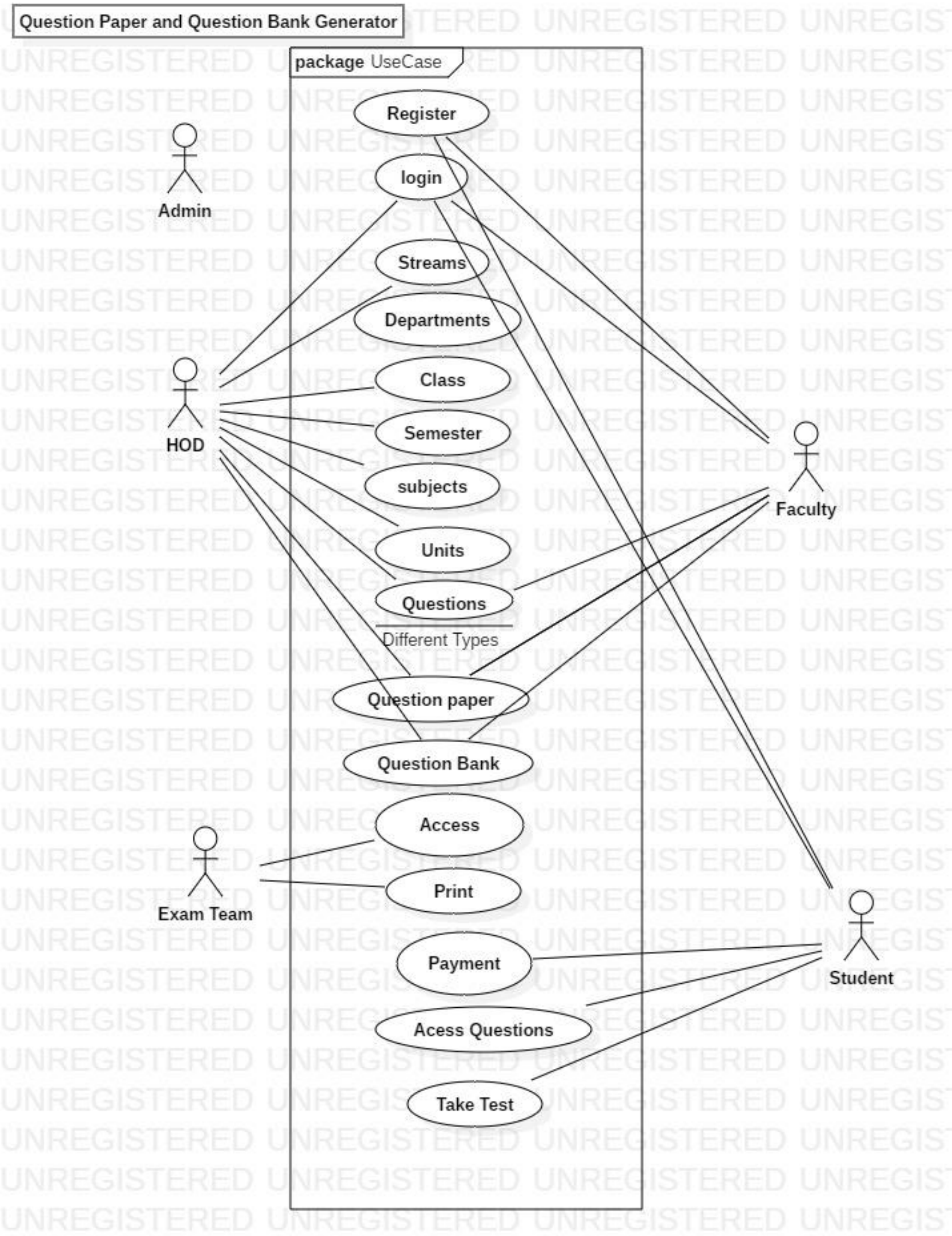
Use-Case Diagram

Use case diagrams are used during requirements elicitation and analysis as a graphical means of representing the functional requirements of the system. Use cases are developed during requirements elicitation and are further refined and corrected as they are reviewed (by stakeholders) during analysis. Use cases are also very helpful for writing acceptance test cases. The test planner can extract scenarios from the use cases for test cases.

An actor represents whoever or whatever (person, machine, or other) interacts with the system. The actor is not part of the system itself and represents anyone or anything that must interact with the system to:

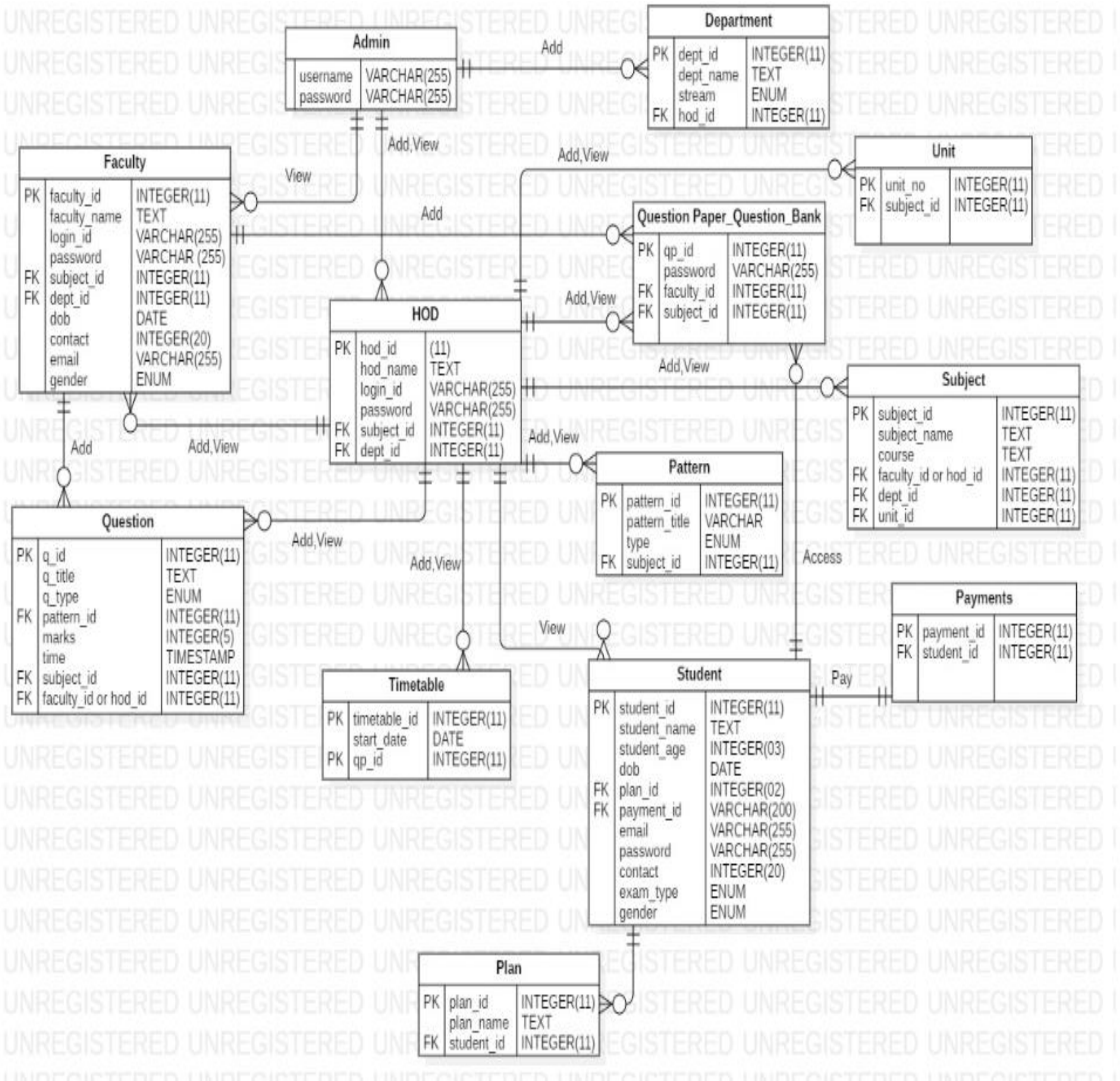
- Input information to the system;
- Receive information from the system; or

Both input information to and receive information from the system


Question Paper and Question Bank Generator
package UseCase
Register
login
Streams
Departments
Class
Semester
subjects
Units
Questions
Different Types
Question paper
Question Bank
Access
Print
Payment
Acess Questions
Take Test
Admin
HOD
Exam Team
Faculty
Student

Entity Relationship Diagram

A graphical model of the data needed by a system , including things about which information is stored and the relationship among them , produced in structured analysis and information engineering. ER Diagram represents entities or tables and their relationships with one another.



Admin

	username	VARCHAR(255)
	password	VARCHAR(255)

Department

PK	dept_id	INTEGER(11)
	dept_name	TEXT
	stream	ENUM
FK	hod_id	INTEGER(11)

Unit

PK	unit_no	INTEGER(11)
FK	subject_id	INTEGER(11)

Faculty

PK	faculty_id	INTEGER(11)
	faculty_name	TEXT
	login_id	VARCHAR(255)
	password	VARCHAR (255)
FK	subject_id	INTEGER(11)
FK	dept_id	INTEGER(11)
	dob	DATE
	contact	INTEGER(20)
	email	VARCHAR(255)
	gender	ENUM

Question Paper_Question_Bank

PK	qp_id	INTEGER(11)
	password	VARCHAR(255)
FK	faculty_id	INTEGER(11)
FK	subject_id	INTEGER(11)

HOD

PK	hod_id	(11)
	hod_name	TEXT
	login_id	VARCHAR(255)
	password	VARCHAR(255)
FK	subject_id	INTEGER(11)
FK	dept_id	INTEGER(11)

Subject

PK	subject_id	INTEGER(11)
	subject_name	TEXT
	course	TEXT
FK	faculty_id or hod_id	INTEGER(11)
FK	dept_id	INTEGER(11)
FK	unit_id	INTEGER(11)

Pattern

PK	pattern_id	INTEGER(11)
	pattern_title	VARCHAR
	type	ENUM
FK	subject_id	INTEGER(11)

Question

PK	q_id	INTEGER(11)
	q_title	TEXT
	q_type	ENUM
FK	pattern_id	INTEGER(11)
	marks	INTEGER(5)
	time	TIMESTAMP
FK	subject_id	INTEGER(11)
FK	faculty_id or hod_id	INTEGER(11)

Payments

PK	payment_id	INTEGER(11)
FK	student_id	INTEGER(11)

Timetable

PK	timetable_id	INTEGER(11)
	start_date	DATE
PK	qp_id	INTEGER(11)

Student

PK	student_id	INTEGER(11)
	student_name	TEXT
	student_age	INTEGER(03)
	dob	DATE
FK	plan_id	INTEGER(02)
FK	payment_id	VARCHAR(200)
	email	VARCHAR(255)
	password	VARCHAR(255)
	contact	INTEGER(20)
	exam_type	ENUM
	gender	ENUM

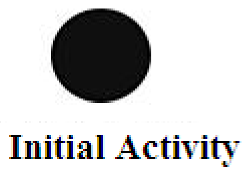
Plan

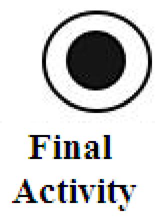
PK	plan_id	INTEGER(11)
	plan_name	TEXT
FK	student_id	INTEGER(11)

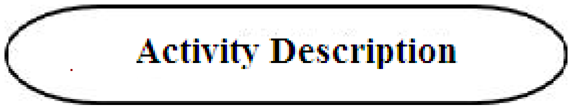
Activity Diagram

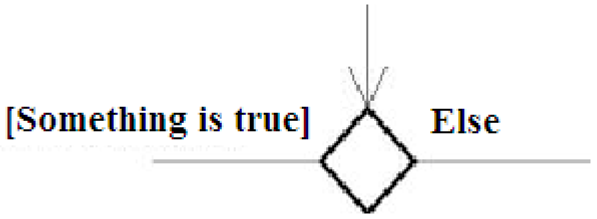
Activity diagram are useful for analyzing a use case by describing what actions needs to be taken place a complicated sequential algorithm modeling applications with parallel processes.

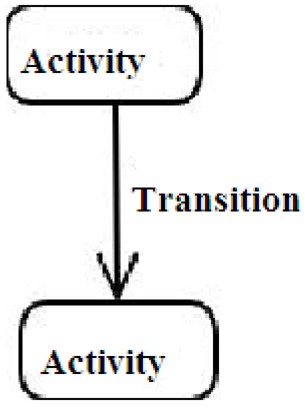
Elements of activity diagram:

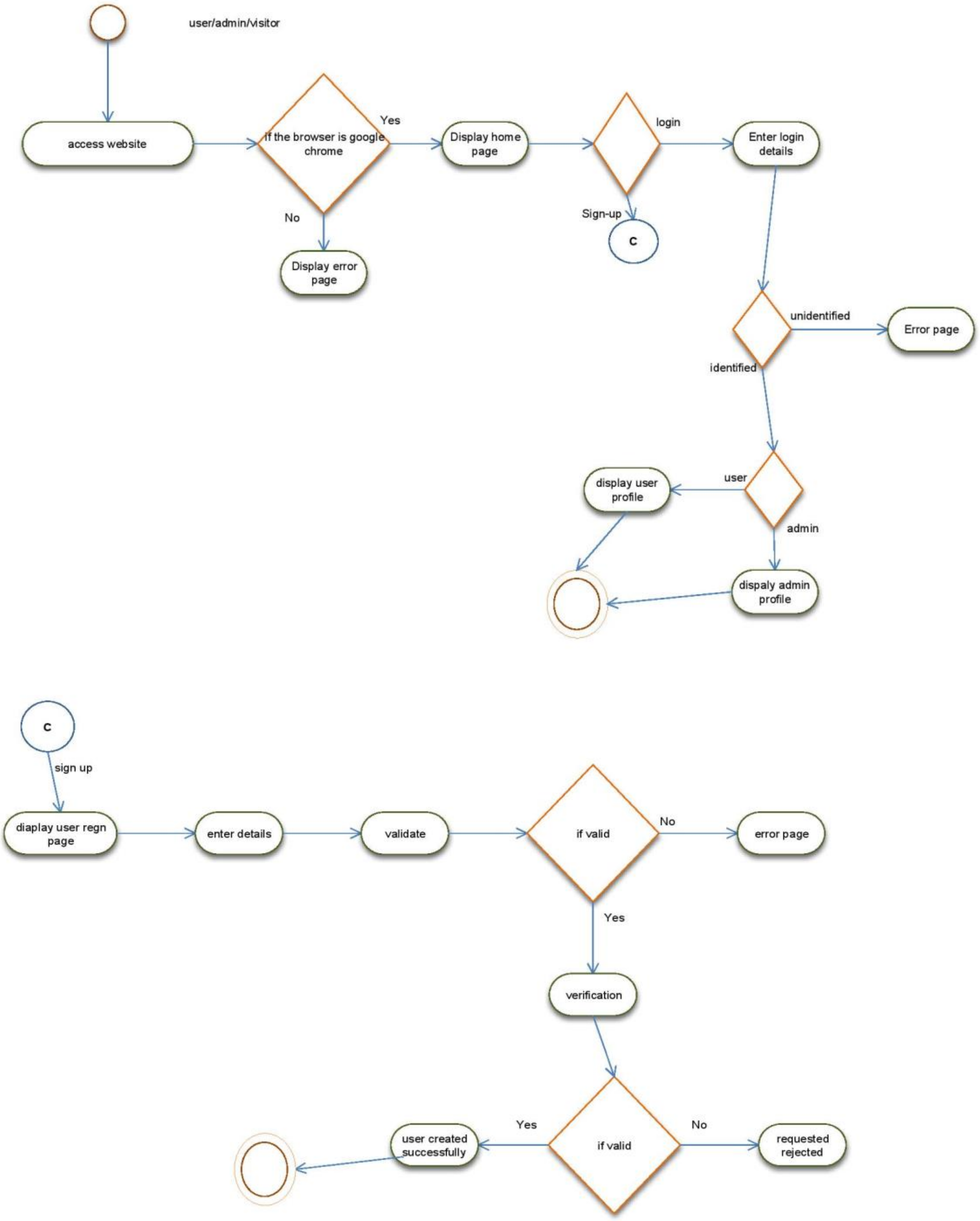


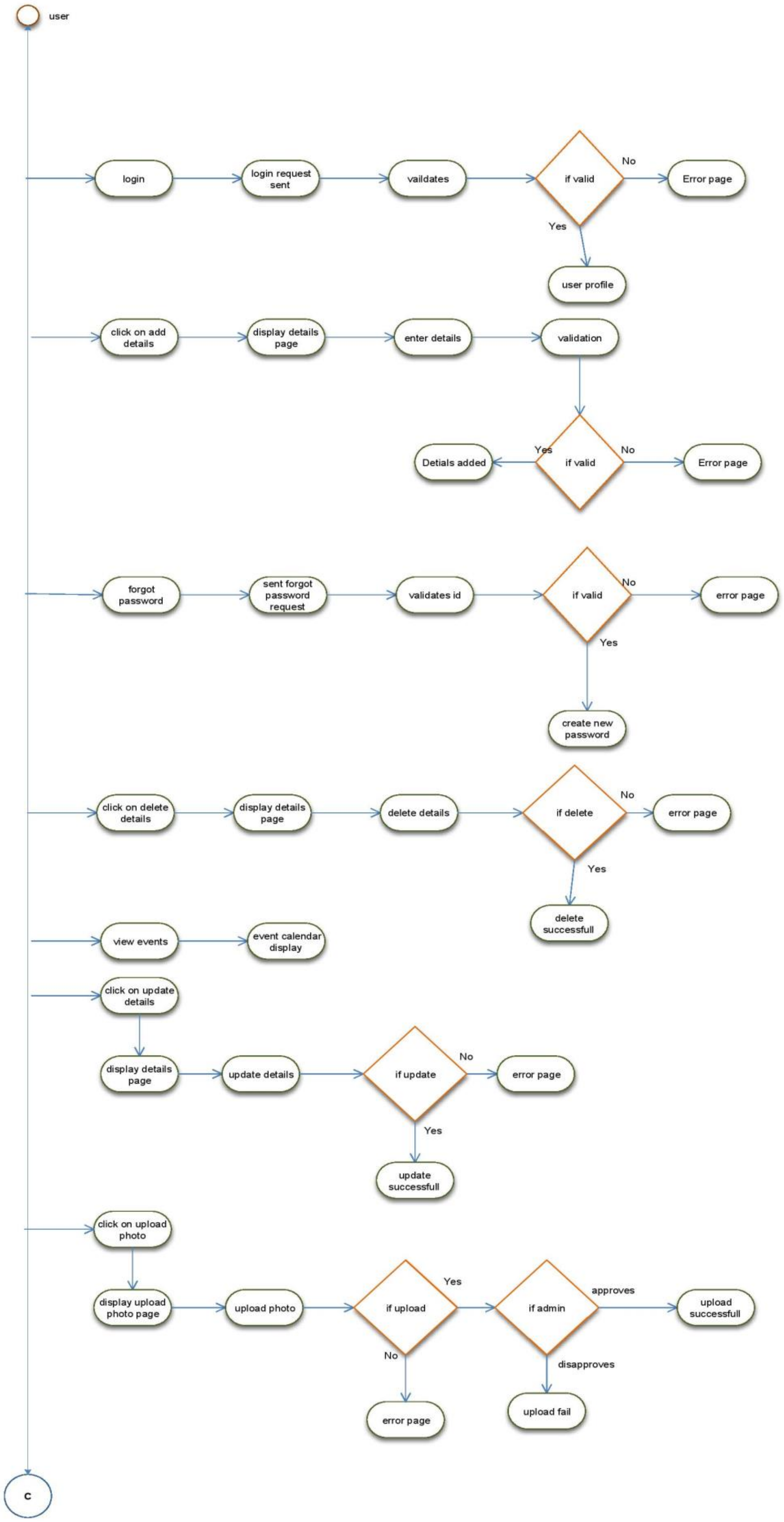


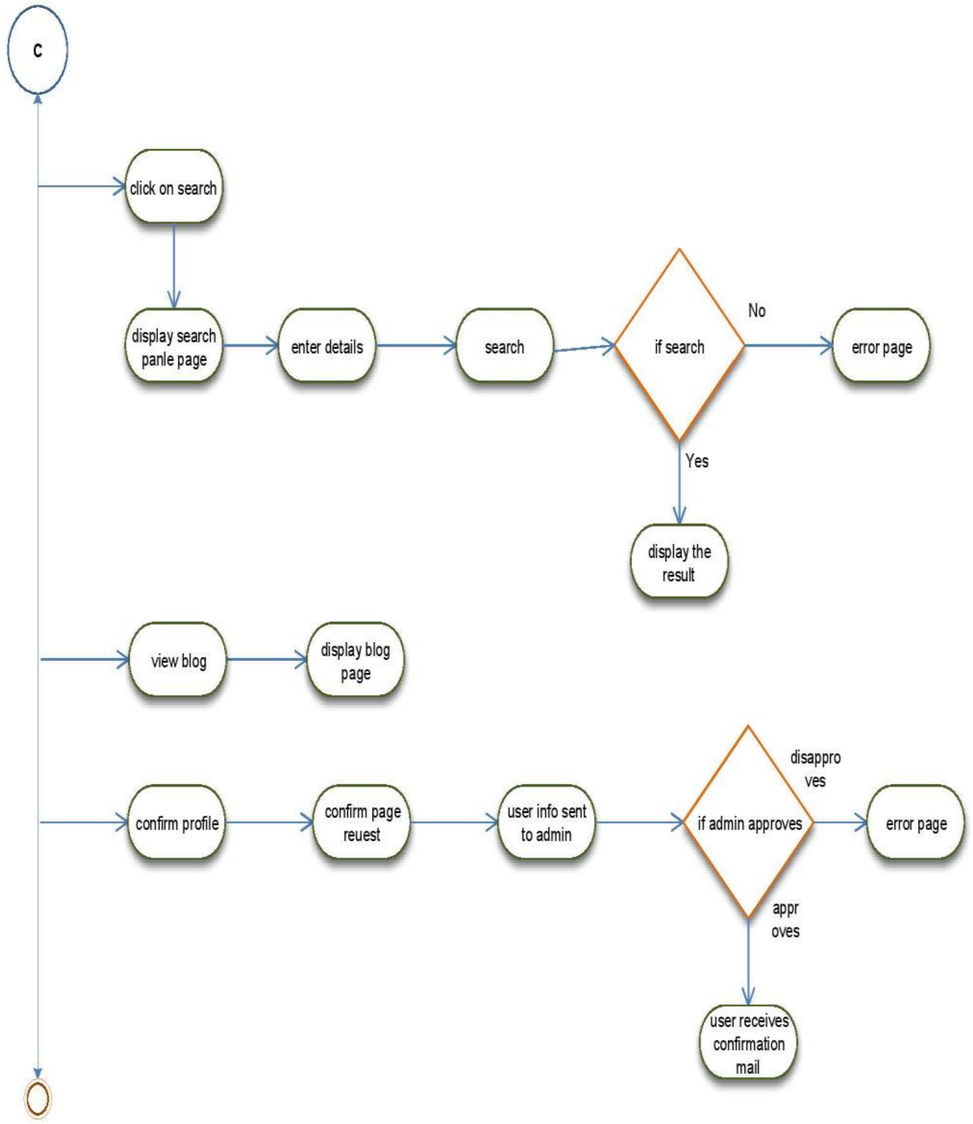


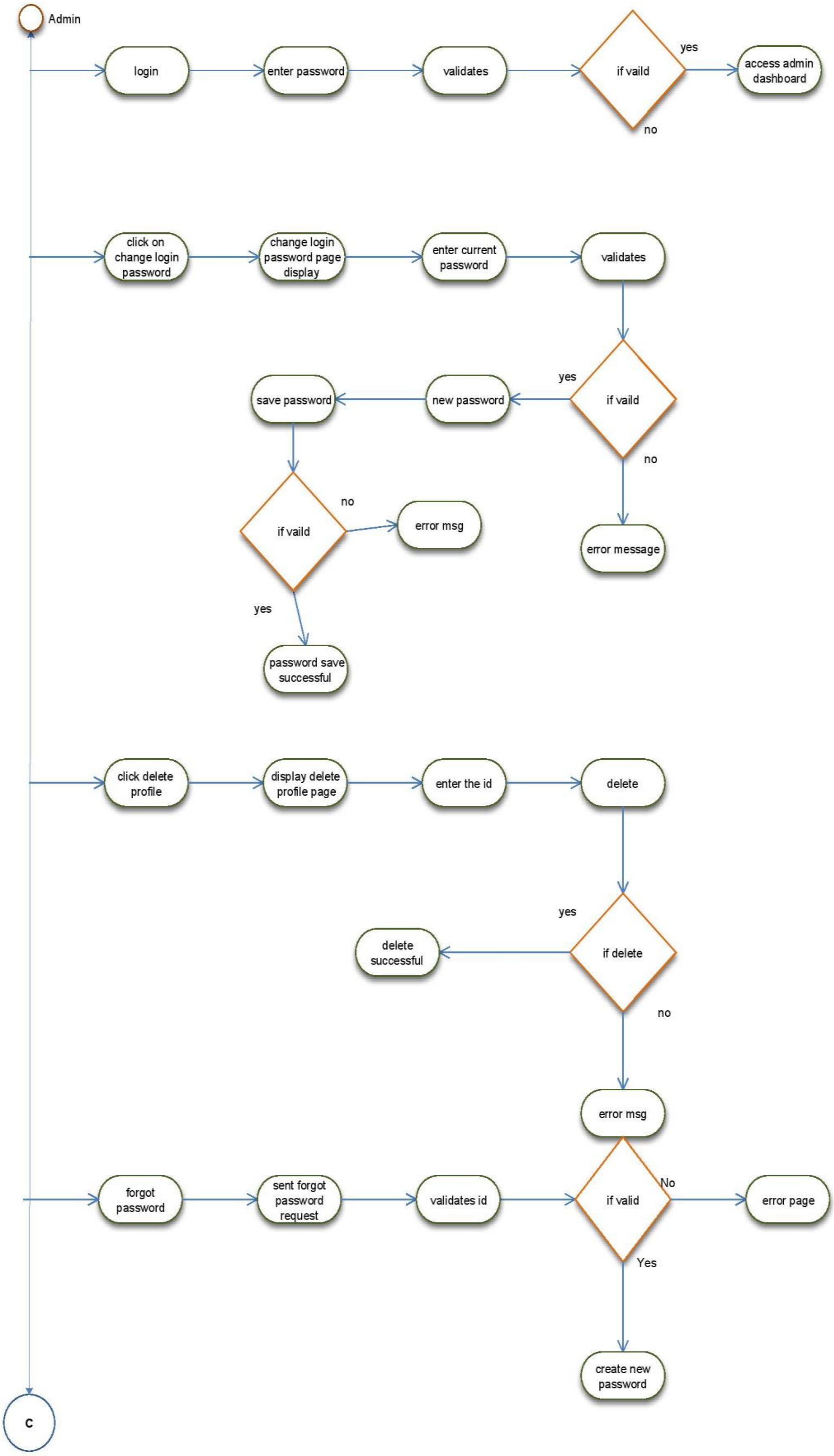





user/admin/visitor
access website
If the browser is google chrome
Yes
No
Display error page
Display home page
login
Sign-up
C
Enter login details
unidentified
Error page
identified
user
display user profile
admin
dispaly admin profile
C
sign up
diaplay user regn page
enter details
validate
if valid
No
error page
Yes
verification
Yes
if valid
No
user created successfully
requested rejected


user
login
login request sent
vaildates
if valid
No
Error page
Yes
user profile
click on add details
display details page
enter details
validation
Detials added
Yes
if valid
No
Error page
forgot password
sent forgot password request
validates id
if valid
No
error page
Yes
create new password
click on delete details
display details page
delete details
if delete
No
error page
Yes
delete successfull
view events
event calendar display
click on update details
display details page
update details
if update
No
error page
Yes
update successfull
click on upload photo
display upload photo page
upload photo
if upload
Yes
if admin
approves
upload successfull
No
error page
disapproves
upload fail
c

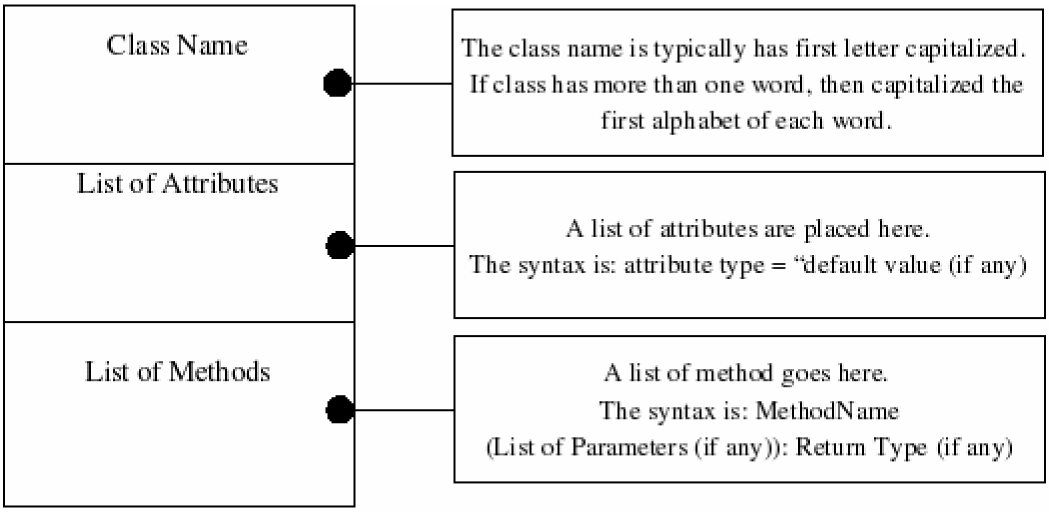

c
click on search
display search panle page
enter details
search
if search
No
error page
Yes
display the result
view blog
display blog page
confirm profile
confirm page reuest
user info sent to admin
if admin approves
disappro ves
error page
appr oves
user receives confirmation mail

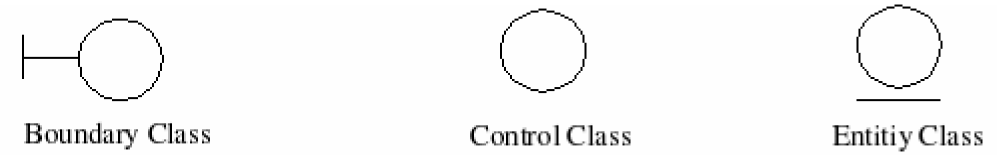

Admin
login
enter password
validates
if vaild
yes
no
access admin dashboard
click on change login password
change login password page display
enter current password
validates
yes
if vaild
new password
save password
no
error message
if vaild
no
error msg
yes
password save successful
click delete profile
display delete profile page
enter the id
delete
yes
if delete
delete successful
no
error msg
forgot password
sent forgot password request
validates id
if valid
No
error page
Yes
create new password
c

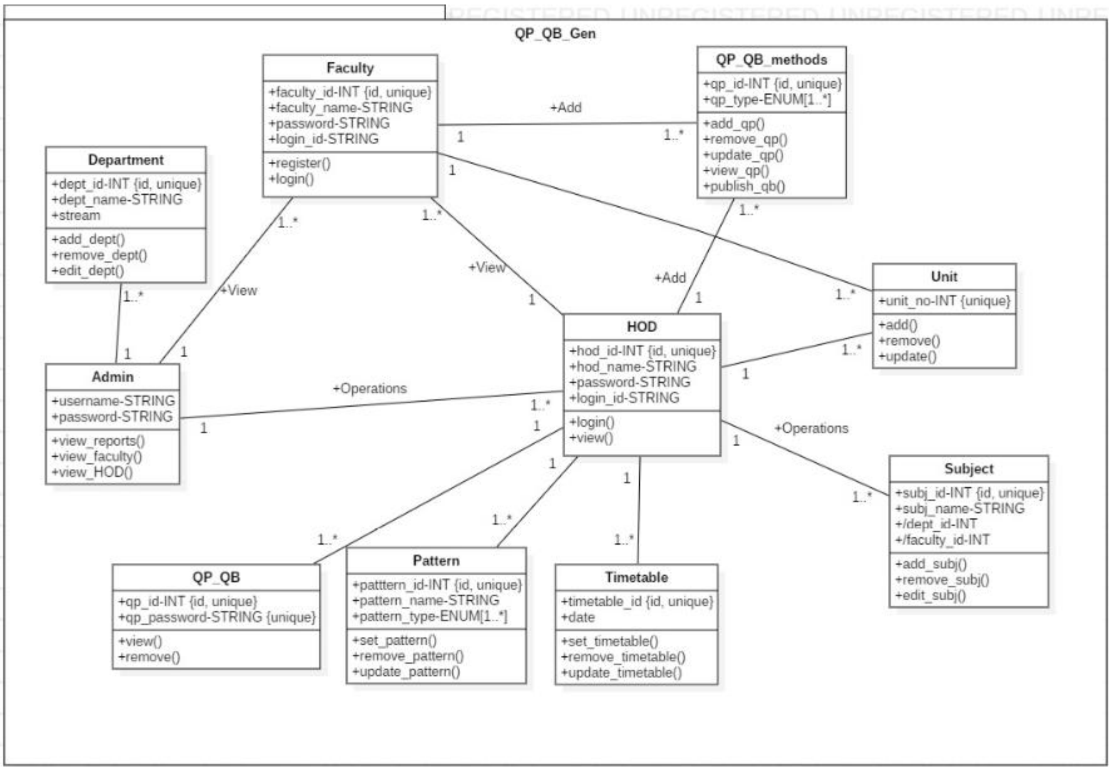
Class Diagram

Class diagrams are used in both the analysis and the design phases. During the analysis phase, a very high-level conceptual design is created. At this time, a class diagram might be created with only the class names shown or possibly some pseudo code-like phrases may be added to describe the responsibilities of the class. The class diagram created during the analysis phase is used to describe the classes and relationships in the problem domain, but it does not suggest how the system is implemented. By the end of the design phase, class diagrams that describe how the system to be implemented should be developed. The class diagram created after the design phase has detailed implementation information, including the class names, the methods and attributes of the classes, and the relationships among classes.

Elements of class diagram:


Class Name
List of Attributes
List of Methods
The class name is typically has first letter capitalized. If class has more than one word, then capitalized the first alphabet of each word.
A list of attributes are placed here.
The syntax is: attribute type = “default value (if any)
A list of method goes here.
The syntax is: MethodName
(List of Parameters (if any)): Return Type (if any)

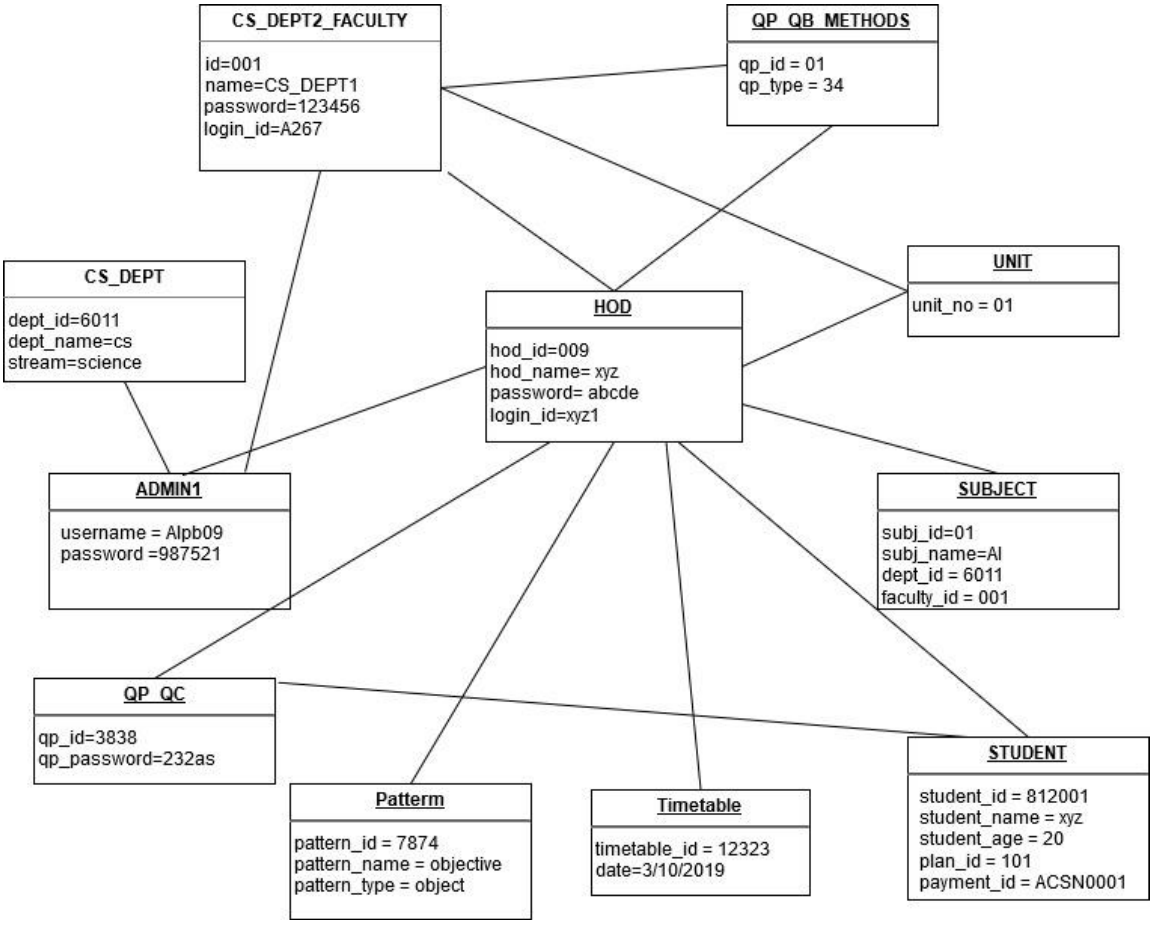



QP_QB_Gen
Faculty
+faculty_id-INT {id, unique}
+faculty_name-STRING
+password-STRING
+login_id-STRING
+register()
+login()
QP_QB_methods
+qp_id-INT {id, unique}
+qp_type-ENUM[1..*]
+add_qp()
+remove_qp()
+update_qp()
+view_qp()
+publish_qb()
+Add
1
1..*
1
Department
+dept_id-INT {id, unique}
+dept_name-STRING
+stream
+add_dept()
+remove_dept()
+edit_dept()
1..*
1..*
+View
1..*
+View
1
+Add
1
1..*
Unit
+unit_no-INT {unique}
+add()
+remove()
+update()
1..*
1
1
Admin
+username-STRING
+password-STRING
+view_reports()
+view_faculty()
+view_HOD()
HOD
+hod_id-INT {id, unique}
+hod_name-STRING
+password-STRING
+login_id-STRING
+login()
+view()
1..*
1
+Operations
1..*
1
1
+Operations
1
1..*
Subject
+subj_id-INT {id, unique}
+subj_name-STRING
+/dept_id-INT
+/faculty_id-INT
+add_subj()
+remove_subj()
+edit_subj()
1
1
1..*
1..*
1..*
QP_QB
+qp_id-INT {id, unique}
+qp_password-STRING {unique}
+view()
+remove()
Pattern
+patttern_id-INT {id, unique}
+pattern_name-STRING
+pattern_type-ENUM[1..*]
+set_pattern()
+remove_pattern()
+update_pattern()
Timetable
+timetable_id {id, unique}
+date
+set_timetable()
+remove_timetable()
+update_timetable()

Object Diagram

Object diagrams are derived from class diagrams so object diagrams are dependent upon class diagram.
Object diagram represent an instance of a class diagram. The basic concepts are similar for class diagrams and object diagrams.
Object diagrams also represent the static view of a system, but the static view is a snapshot of the system at a particular moment.
The difference is that a class diagram represents an abstract model consists of classes and their relationships. But an object diagram represents an instance at a particular moment which is concrete in nature.
So the purpose of the object diagram can be summarized as:

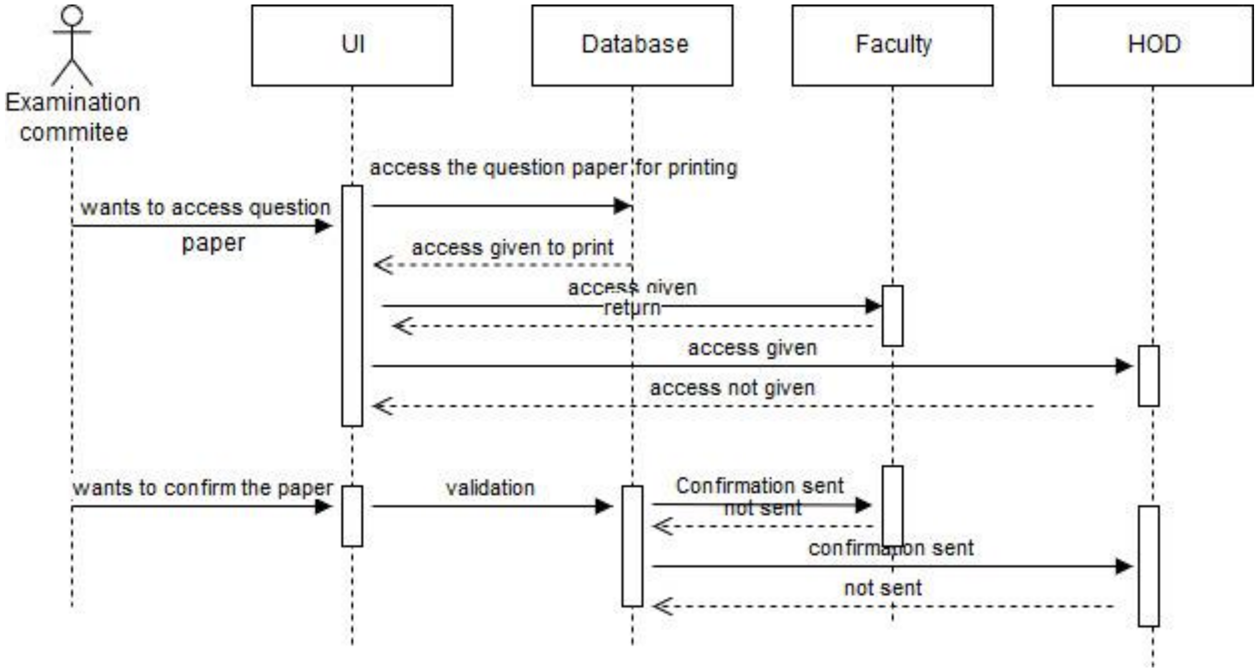
- Forward and reverse engineering.
- Object relationships of a system
- Static view of an interaction.
- Understand object behavior and their relationship from practical Perspective

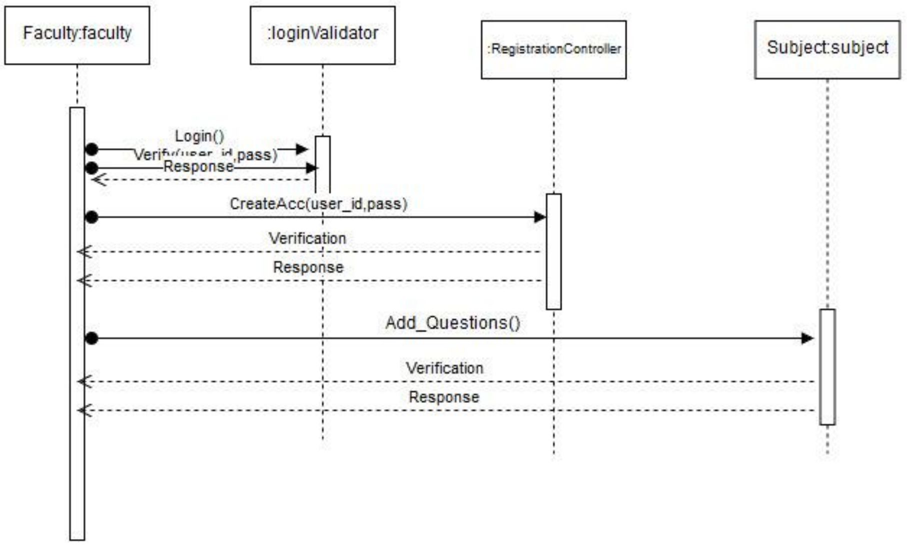


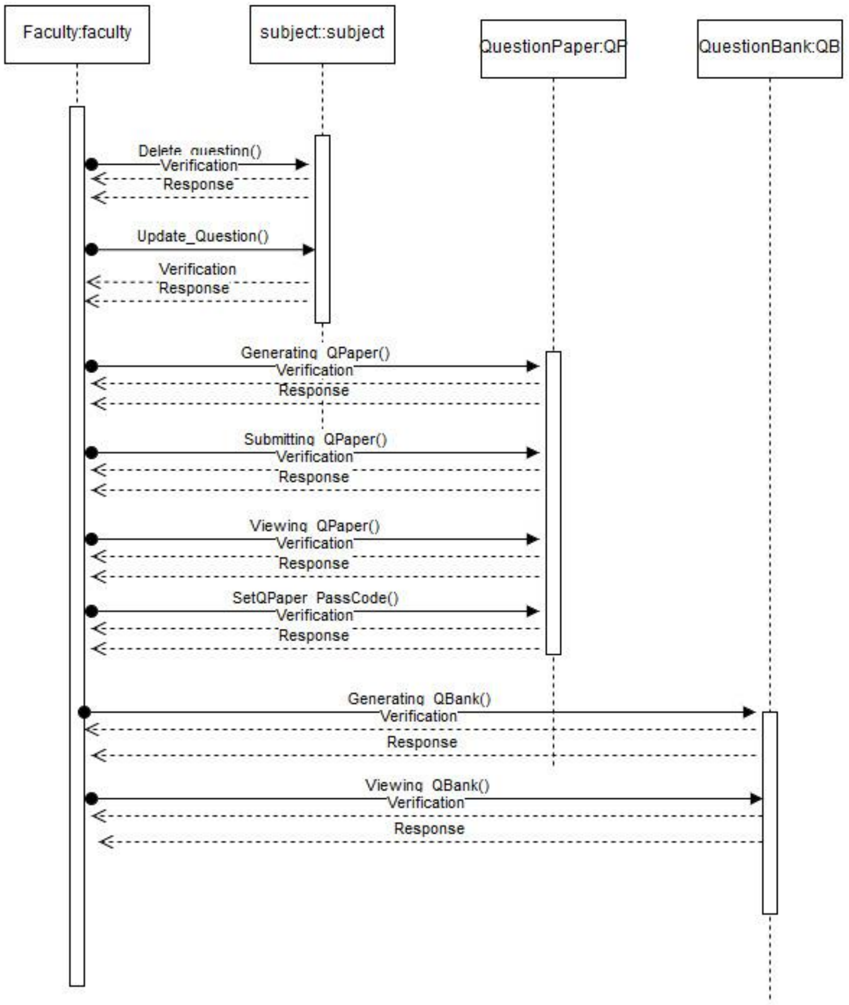
Sequence Diagram

Sequence diagrams are used in the analysis and design phases. Sequence diagrams are often used to depict the chronologically-structured event flow through a use case. By creating a sequence diagram, the objects that participate in the use case are identified. Additionally, pieces of the use case behavior are assigned to objects in the form of services. The process of creating a sequence diagram often results in the refinement of the use case, potentially identifying missing but desired behaviors.

Sequence diagrams represent a system behavior based upon the needed interactions among a set of objects in terms of the messages that exchange among them to produce the desired result. Sequence diagrams highlight the sequence of messages through time. However, they do not show how objects are linked and may send messages to each other.




Faculty:faculty
:loginValidator
:RegistrationController
Subject:subject
Login()
Verify(user_id,pass)
Response
CreateAcc(user_id,pass)
Verification
Response
Add_Questions()
Verification
Response


Faculty:faculty
subject::subject
QuestionPaper:QP
QuestionBank:QB
Delete_question()
Verification
Response
Update_Question()
Verification
Response
Generating_QPaper()
Verification
Response
Submitting_QPaper()
Verification
Response
Viewing_QPaper()
Verification
Response
SetQPaper_PassCode()
Verification
Response
Generating_QBank()
Verification
Response
Viewing_QBank()
Verification
Response

Deployment Diagram

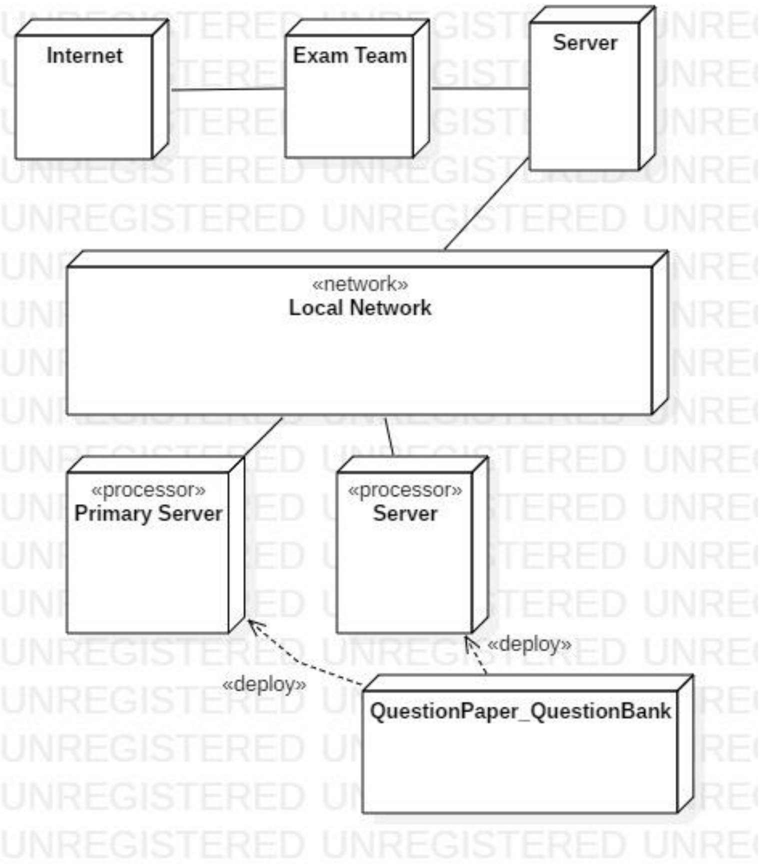
Deployment diagrams depict the physical resources in a system, including node, components, and connections.

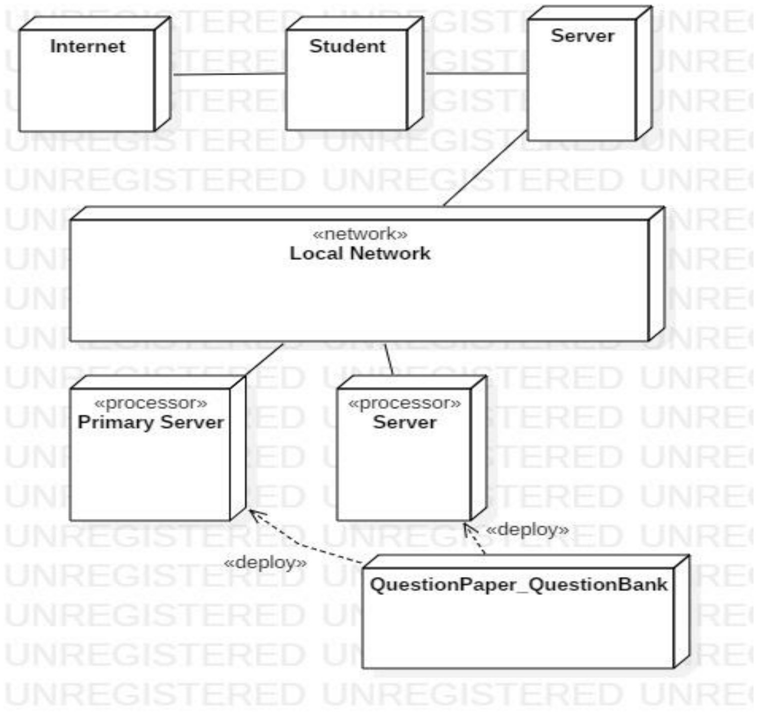
Node: A node is a physical resource that executes code components. A node is either hardware or a software element. It is shown as a three-dimensional box shape.

Node-Instance: A node instance can be shown on a diagram. An instance can be distinguished from a node by the fact that its name is underlined and has a colon before its base node type.

Node Stereotype: A number of standard stereotypes are provided for nodes, namely <<cd-rom>>, <<computer>> etc.

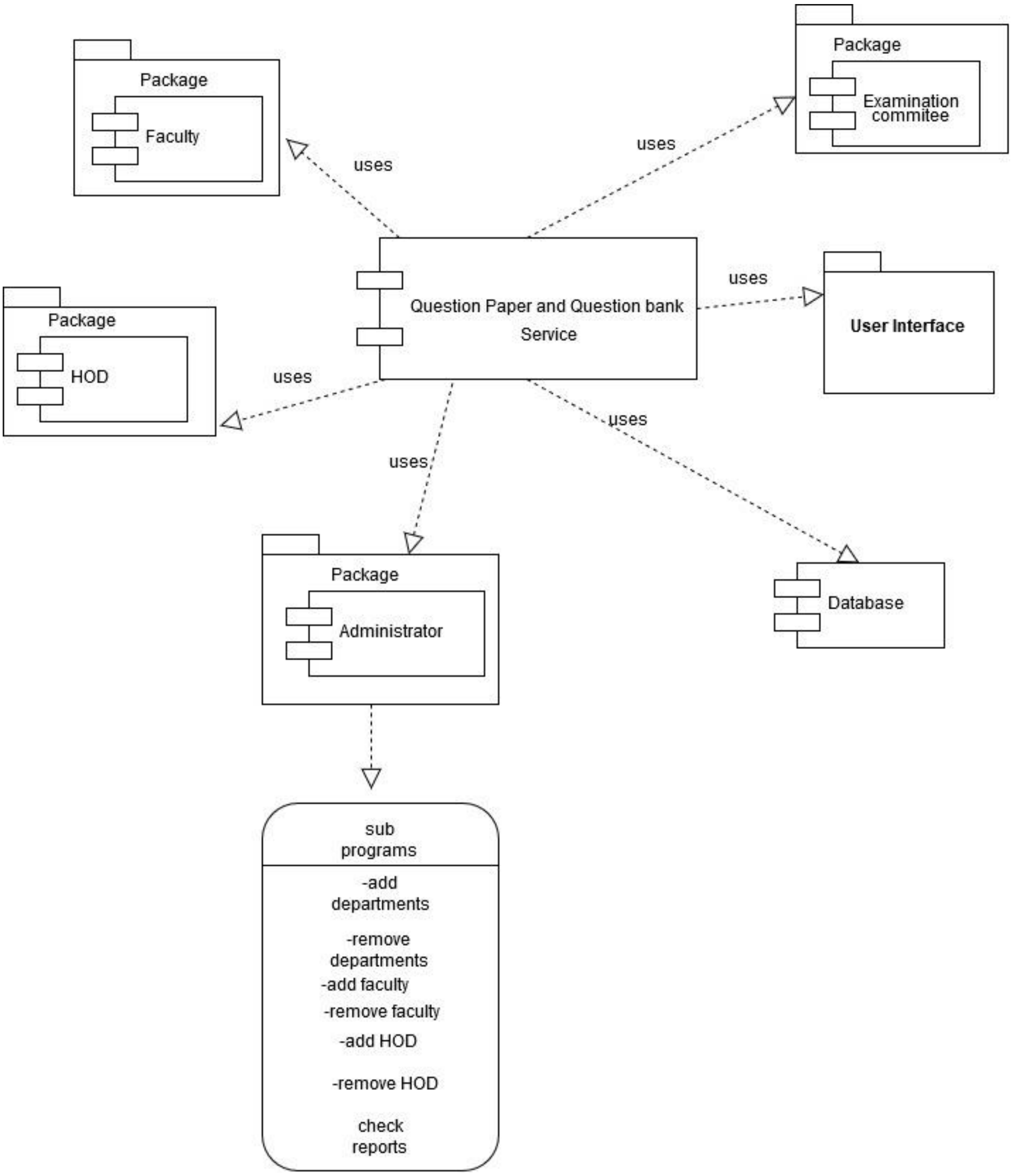
Association: Association refers to a physical connection between nodes, such as Ethernet. In the context of a deployment diagram, an association represents a communication path between nodes. The following diagram shows a deployment diagram for a network, depicting network protocols as stereotypes and multiplicities at the association ends.


Internet
Exam Team
Server
«network»
Local Network
«processor»
Primary Server
«processor»
Server
«deploy»
«deploy»
QuestionPaper_QuestionBank


Internet
Student
Server
«network»
Local Network
«processor»
Primary Server
«processor»
Server
«deploy»
«deploy»
QuestionPaper_QuestionBank

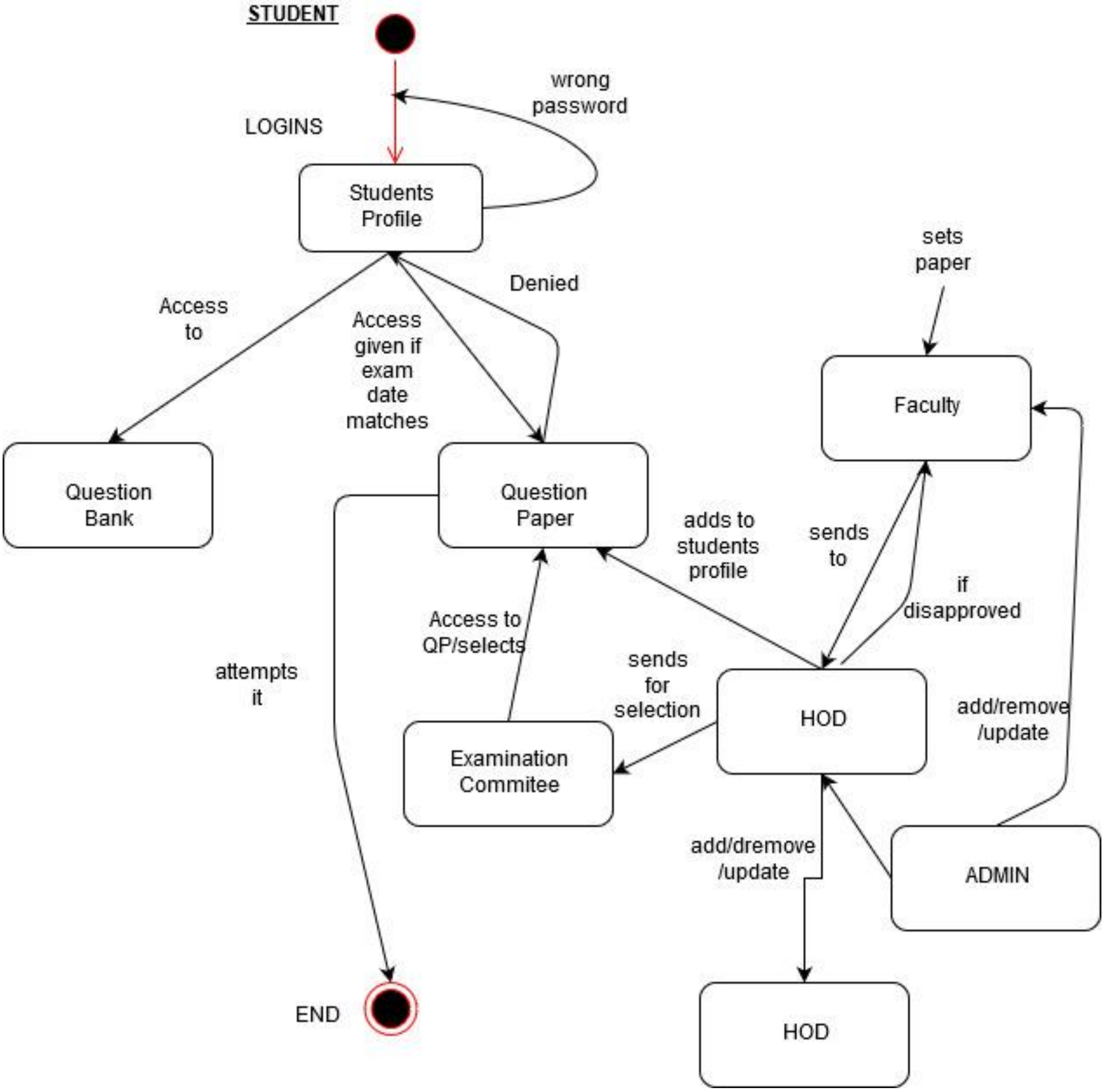
Component Diagram

Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.



State Diagram

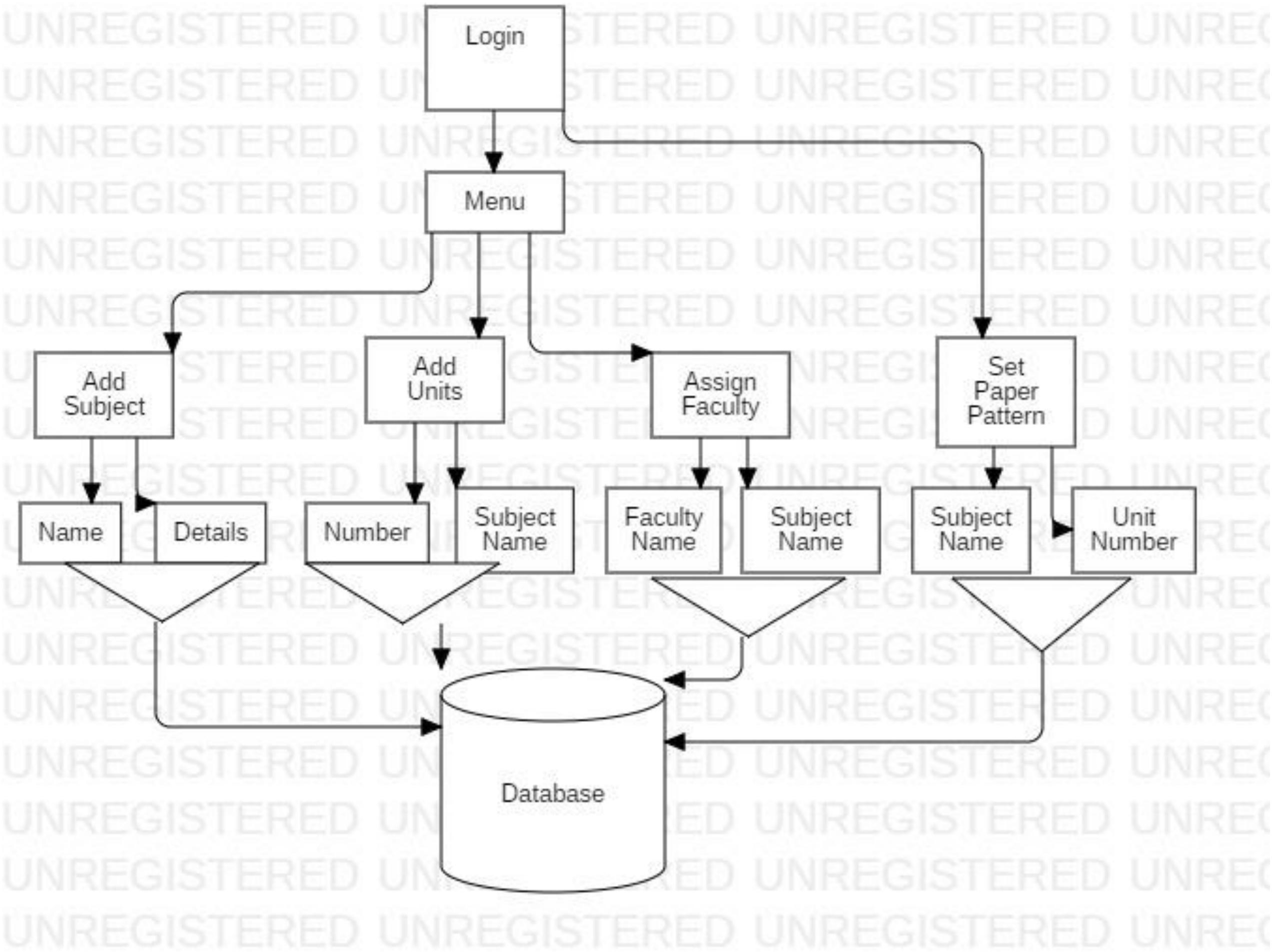
State diagrams are used to give an abstract description of the behavior of a system. This behavior is analyzed and represented as a series of events that can occur in one or more possible states. Hereby "each diagram usually represents objects of a single class and track the different states of its objects through the system".



System Design

System design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering

Here is system design for HOD



User Interface

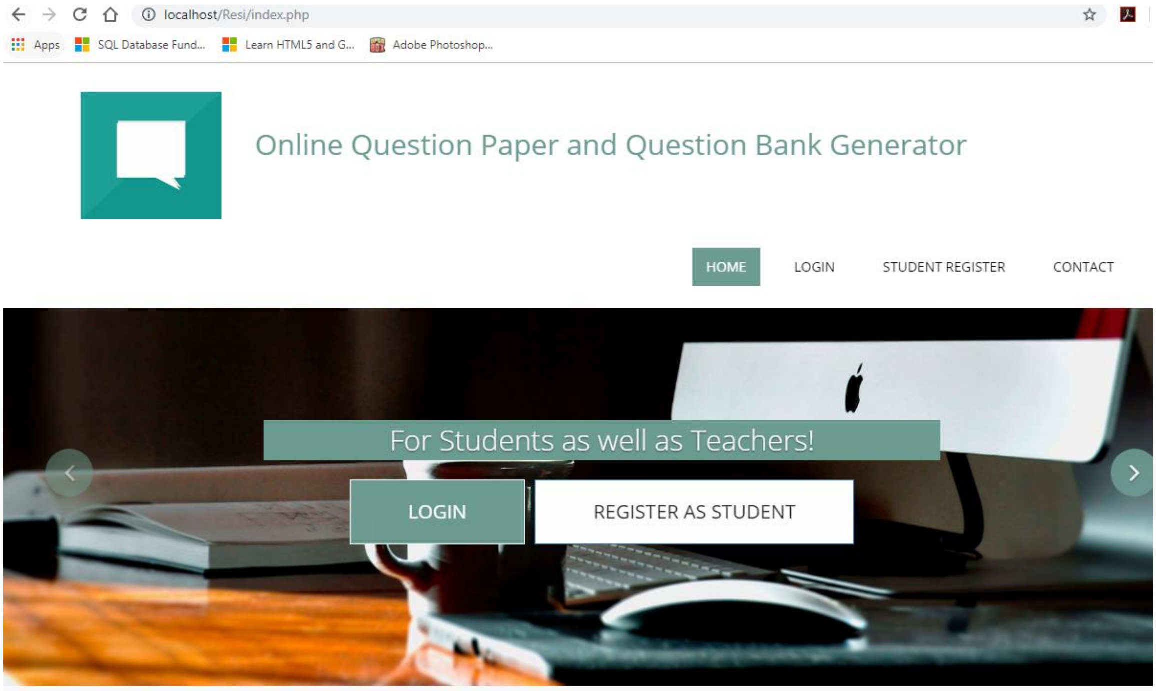
The user interface (UI) is the point of human-computer interaction and communication in a device. This can include display screens, keyboards, a mouse and the appearance of a desktop. It is also the way through which a user interacts with an application or a website. The growing dependence of many businesses on web applications and mobile applications has led many companies to place increased priority on UI in an effort to improve the user's overall experience.

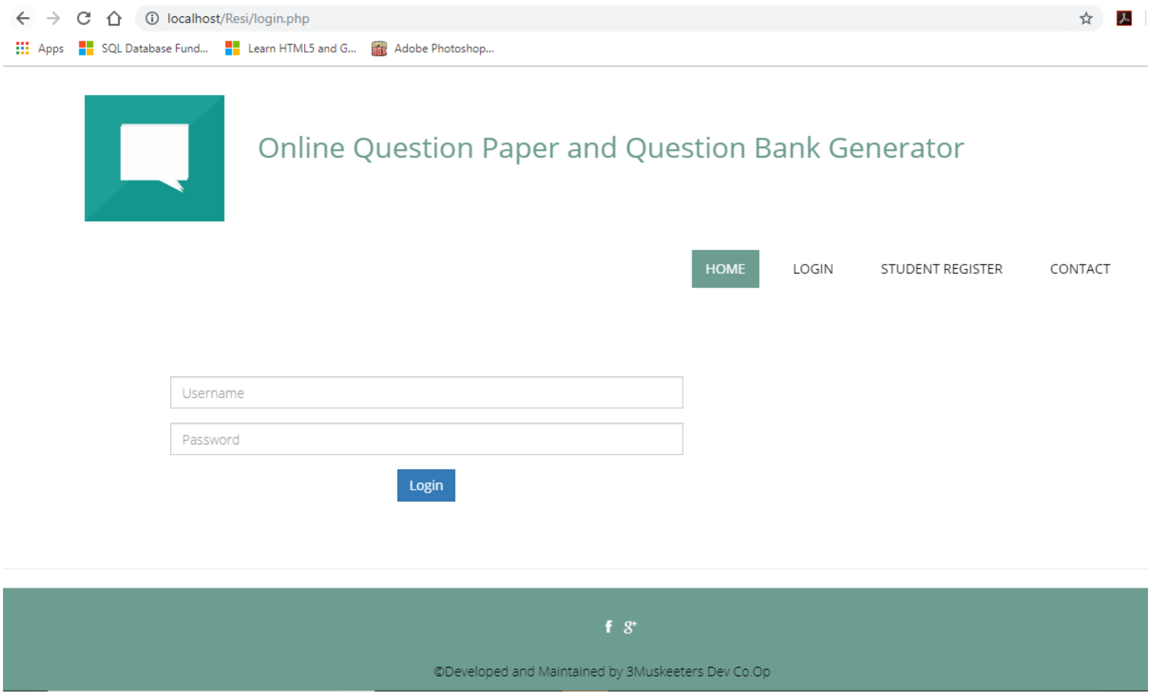
Types of user interfaces

The various types of user interfaces include:

- graphical user interface (GUI)
- command line interface (CLI)
- menu-driven user interface
- touch user interface
- voice user interface (VUI)
- form-based user interface
- natural language user interface

Websites such as Airbnb, Dropbox and Virgin America display strong user interface design. Sites like these have created pleasant, easily operable, user-centred designs (UCD) that focus on the user and their needs.

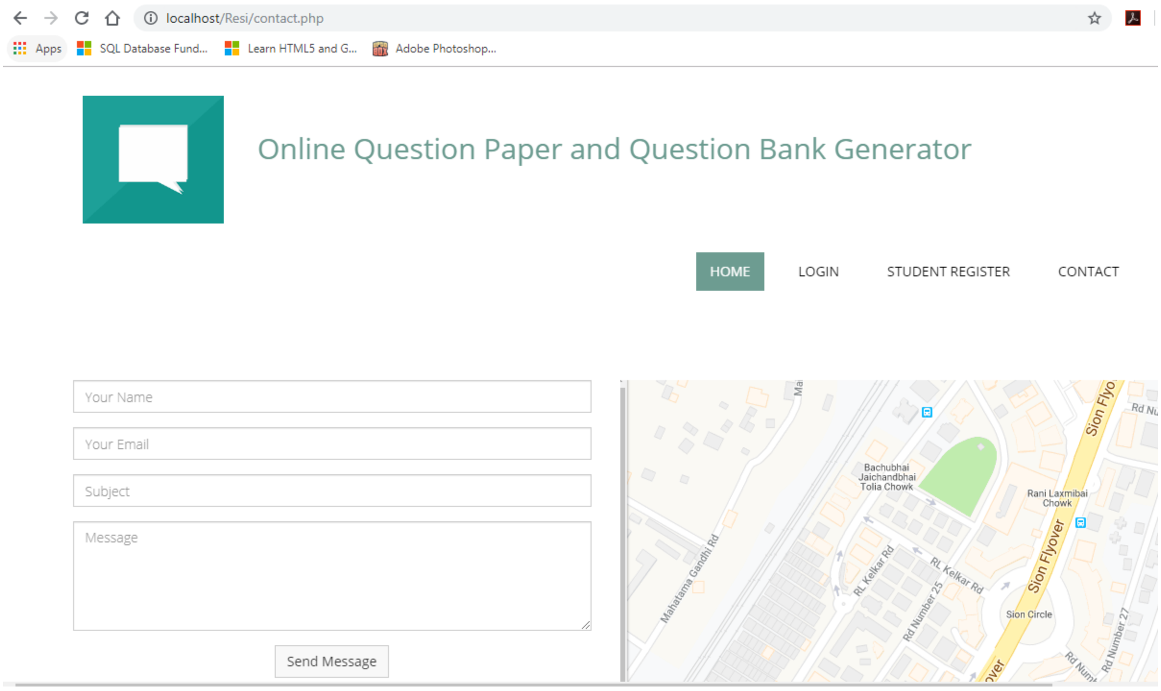




localhost/Resi/login.php

Online Question Paper and Question Bank Generator

©Developed and Maintained by 3Muskeeters Dev Co.Op

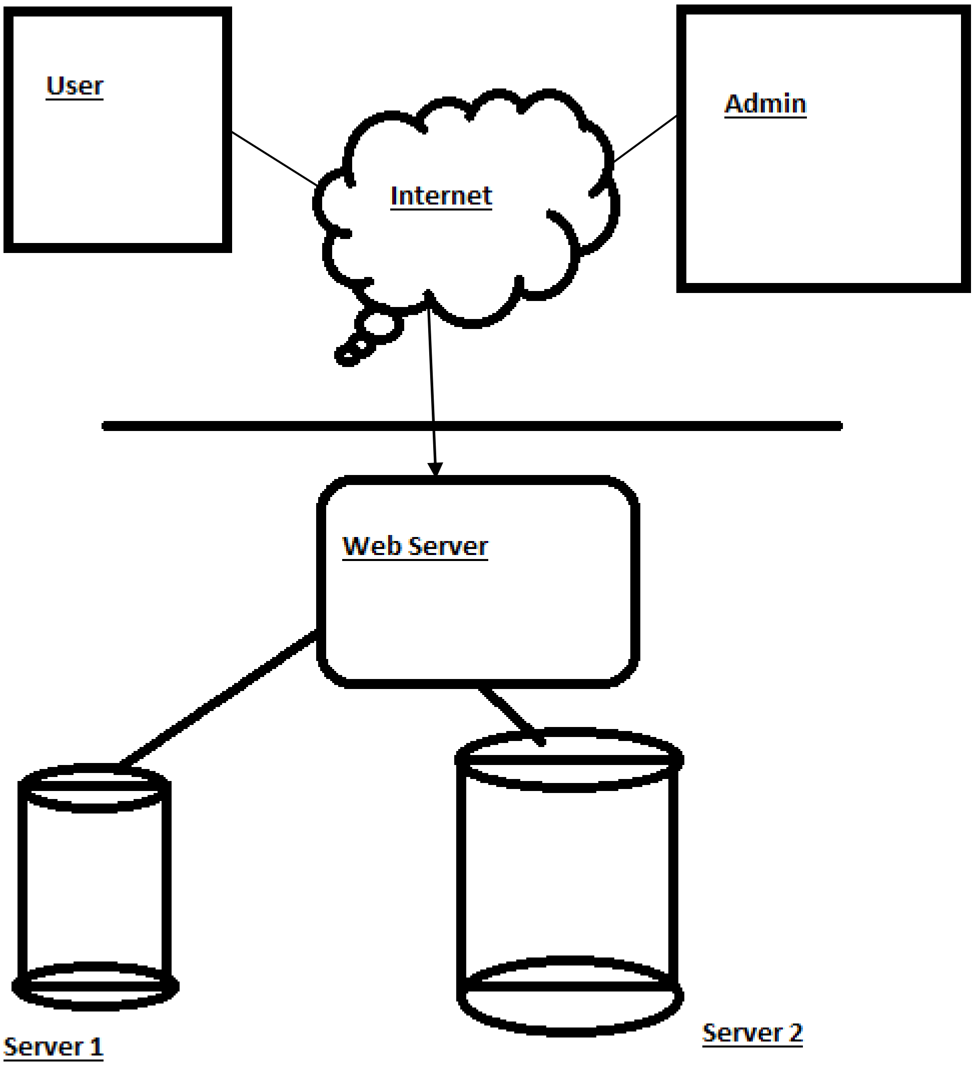


localhost/Resi/contact.php

Online Question Paper and Question Bank Generator

System Architecture

A system architecture diagram would be used to show the relationship between different components. Usually they are created for systems which include hardware and software and these are represented in the diagram to show the interaction between them



Security Mechanisms

The security mechanisms used in the project are as follows:

- RSA Algorithm for Encryption and Decryption
- The files are transferred in the form a zip file with a password set by the faculties and only shared between the concerned parties
- SHA-256 Algorithm for file transfer

Costs

- **Estimated Costs:** Rs.10,00,000 + GST.
- **Maintenance Policy:** First Three (03) Months of Maintenance after release would be free and thereafter charges would apply
- **Release Date:** The expected test release of the project is 15th February 2020, the original version would be released on satisfaction of the client on the test release

Bibliography

Definitions:

https://www.wikipedia.org/

https://google.com/

https://searchapparchitecture.techtarget.com/definition/user-interface-UI

https://medium.com/the-andela-way/system-design-in-software-development-f360ce6fcbb9

Theme Credits:

https://bootstrapmade.com/resi-free-bootstrap-html-template/