LABORATORY MANUAL
SOFTWARE ENGINEERING
LABORATORY

(R20CSE31L1)

III B. Tech I SEM (AIML)

2022 - 2023

Prepared by

Ms.B.K.N.Priyanka
Asst. Professor

Department of AIML

SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

An Autonomous Inistution Under UGC, New Delhi , Sheriguda (V), Ibrahimpatnam Ranga Reddy Dist – 501 510. T.S.

INDEX

COURSE NAME: Software Engineering Lab

COURSE CODE: R20CSE31L1

COURSE OBJECTIVES:

- To understand the software engineering methodologies involved in the phases for project development.
- To gain knowledge about open source tools used for implementing software engineering methods.
- To exercise developing product-startups implementing software engineering methods. Open source Tools: Star UML / UML Graph / Top cased
  .

COURSE OUTCOMES:

- Ability to identify the minimum requirements for the development of application.
- Ability to develop, maintain efficient, reliable and cost effective software solution..
- Ability to critically thinking and evaluate assumptions and arguments.

SOFTWARE ENGINEERING LAB (R20CSE31L1)

Prepare the following documents and develop the software project startup, prototypemodel, using software engineering methodology for at least two real time scenarios or for the sample experiments.

•Problem Analysis and Project Planning -Thorough study of the problem–Identify Project scope, Objectives and Infrastructure.

•Software Requirement Analysis –Describe the individual Phases/modules of the project and Identify deliverables. Identify functional and non-functional requirements.

•Data Modeling –Use work products –data dictionary.

•Software Designing -Develop use case diagrams and activity diagrams, build and test class diagrams, sequence diagrams and add interface to class diagrams.

•Prototype model –Develop the prototype of the product.

The SRS and prototype model should be submitted for end semester examination.

List of Sample Experiments:

1. Course management system(CMS)

A course management system(CMS) is a collection of software tools providing an online environment for course interactions. A CMS typically includes a variety of online tools and environments, such as:

•An area for faculty posting of class materials such as course syllabus and handouts

•An area for student posting of papers and other assignments

•A grade book where faculty can record grades and each student can view his or her grades

•An integrated email tool allowing participants to send announcement email messages to the entire class or to a subset of the entire class

•A chat tool allowing synchronous communication among class participants

•A threaded discussion board allowing asynchronous communication among participants.

In addition, a CMS is typically integrated with other databases in the university so that students enrolled in a particular course are automatically registered in the CMS as participants in that course.

The Course Management System (CMS) is a web application for department personnel, Academic Senate, and Registrar staff to view, enter, and manage course information formerly Submitted via paper. Departments can use CMS to create new course proposals, submit changes for existing courses, and track the progress of proposals as they move through the stages of online approval.

2. Easy Leave

This project is aimed at developing a web based Leave Management Tool, which is of importance to either an organization or a college.

The Easy Leave is an Intranet based application that can be accessed throughout the Organization or a specified group/Dept. This system can be used to automate the workflow of leave applications and their approvals. The periodic crediting of leave is also automated.

There are features like notifications, cancellation of leave, automatic approval of leave, report generators etc in this Tool.

Functional components of the project:

There are registered people in the system. Some are approvers. An approver can also be a requestor. In an organization, the hierarchy could be Engineers/Managers/Business Managers/Managing Director etc. In a college, it could be Lecturer/Professor/Head of the Department/Dean/Principal etc.

Following is a list of functionalities of the system: A person should be able to

•login to the system through the first page of the application

•change the password after logging into the system

•see his/her eligibility details (like how many days of leave he/she is eligible for etc)

•query the leave balance

•see his/her leave history since the time he/she joined the company/college

•apply for leave, specifying the from and to dates, reason for taking leave, address for communication while on leave and his/her superior's email id

•see his/her current leave applications and the leave applications that are submitted to him/her for approval or cancellation

•approve/reject the leave applications that are submitted to him/her
•withdraw his/her leave application (which has not been approved yet)
•Cancel his/her leave (which has been already approved). This will need to be approved by his/her Superior
•get help about the leave system on how to use the different features of the system
•As soon as a leave application /cancellation request /withdrawal /approval /rejection /password-change is made by the person, an automatic email should be sent to the person and his superior giving details about the action
•The number of days of leave (as per the assumed leave policy) should be automatically credited to everybody and a notification regarding the same be sent to them automatically
•An automatic leave-approval facility for leave applications which are older than 2 weeks should be there. Notification about the automatic leave approval should be sent to the person as well as his superior

3.E-Bidding

Auctions are among the latest economic institutions in place. They have been used since antiquity to sell a wide variety of goods, and their basic form has remained unchanged. In this dissertation, we explore the efficiency of common auctions when values are interdependent-the value to a particular bidder may depend on information available only to others-and asymmetric. In this setting, it is well known that sealed-bid auctions do not achieve efficient allocations in general since they do not allow the information held by different bidders to be shared.

Typically, in an auction, say of the kind used to sell art, the auctioneer sets a relatively low initial price. This price is then increased until only one bidder is willing to buy the object, and the exact manner in which this is done varies. In my model a bidder who drops out at some price can "reenter" at a higher price.

With the invention of E-commerce technologies over the Internet the opportunity to bid from the comfort of one’s own home has seen a change like never seen before. Within the span of a few short years, what may have began as an experimental idea has grown to an immensely popular hobby, and in some cases, a means of livelihood, the Auction Patrol gathers

tremendous response every day, all day. With the point and click of the mouse, one may bid on an item they may need or just want, and in moments they find that either they are the top bidder or someone else wants it more, and you're outbid! The excitement of an auction all from the comfort of home is a completely different experience.
Society cannot seem to escape the criminal element in the physical world, and so it is the same with Auction Patrols. This is one area where in a question can be raised as to how safe Auction Patrols.

Proposed system
To generate the quick reports
To make accuracy and efficient calculations
To provide proper information briefly
To provide data security
To provide huge maintenance of records
Flexibility of transactions can be completed in time

4. Electronic Cash counter
This project is mainly developed for the Account Division of a Banking sector to provide better interface of the entire banking transactions. This system is aimed to give a better out look to the user interfaces and to implement all the banking transactions like:
•Supply of Account Information
•New Account Creations
•Deposits
•Withdraws
•Cheque book issues
•Stop payments
•Transfer of accounts
•Report Generations.

Proposed System:

The development of the new system contains the following activities, which try to automate the entire process keeping in view of the database integration approach.

•User friendliness is provided in the application with various controls.

•The system makes the overall project management much easier and flexible.

•Readily upload the latest updates, allows user to download the alerts by clicking the URL.

•There is no risk of data mismanagement at any level while the project development is under process.

•It provides high level of security with different level of authentication

REFERENCE BOOKS:

1. Roger S.Pressman, Software engineering- A practitioner's Approach, McGraw-Hill International Edition, 6th edition, 2001.
2. Ian Sommerville, Software engineering, Pearson education Asia, 6th edition, 2000
3. Unified modeling language- Grady booch

UML INTRODUCTION

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

- UML was created by Object Management Group (OMG) and UML
- Specification draft was proposed to the OMG in January 1997.
- OMG is continuously putting effort to make a truly industry standard.
- UML stands for Unified Modeling Language.
- UML is different from the other common programming languages like C++, Java, and COBOL etc.
- UML is a pictorial language used to make software blue prints.

GOALS OF UML

- A picture is worth a thousand words, this absolutely fits while discussing about UML.
- UML diagrams are not only made for developers but also for business users, common people and anybody interested to understand the system.
- The system can be a software or non software. So it must be clear that.
- UML is not a development method rather it accompanies with processes to make a successful system.

CONCEPTUAL MODEL OF UML

- A conceptual model can be defined as a model which is made of Concepts and their relationships.

- A conceptual model is the first step before drawing a UML diagram. It helps to understand the entities in the real world and how they interact with each other. Conceptual model of UML can be mastered by learning.

The following three major elements:

- UML building blocks.
- Rules to connect the building blocks.
- Common mechanisms of UML.

Object Oriented Analysis and Design

The purpose of OO analysis and design can described as:

1. Identifying the objects of a system.

2. Identify their relationships.

3. Make a design which can be converted to executables using OO languages.

OO Analysis --> OO Design --> OO implementation using OO languages.

- During object oriented analysis the most important purpose is to identify objects and describing them in a proper way. If these objects are identified efficiently then the next job of design is easy. The objects should be identified with responsibilities. Responsibilities are the functions performed by the object. Each and every object has some type of responsibilities to be performed. When these responsibilities are collaborated the purpose of the system is fulfilled.

- The second phase is object oriented design. During this phase emphasis is given upon the requirements and their fulfillment. In this stage the objects are collaborated according to their intended association. After the association is complete the design is also complete.

- The third phase is object oriented implementation. In this phase the design is implemented using object oriented languages like Java, C++ etc.

BUILDING BLOCKS

The building blocks of UML can be defined as:

1. Things
2. Relationships
3. Diagrams

(1) Things

Things are the most important building blocks of UML. Things can be:

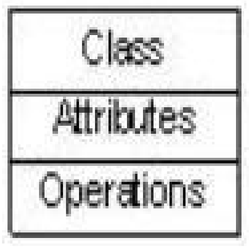
- Structural
- Behavioral
- Grouping
- Annotational

Structural things

The **Structural things** define the static part of the model. They represent physical and conceptual elements. Following are the brief descriptions of the structural things.

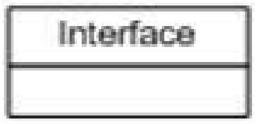
Class

Class represents set of objects having similar responsibilities.



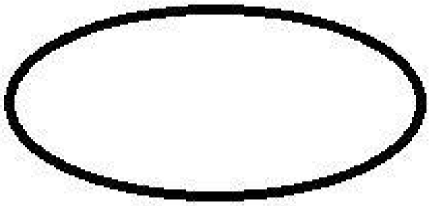
Interface

Interface defines a set of operations which specify the responsibility of a class.



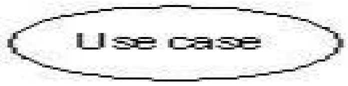
Collaboration

Collaboration defines interaction between elements



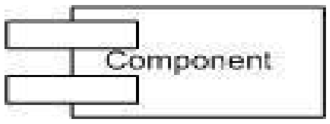
Use case

Use case represents a set of actions performed by a system for a specific goal.



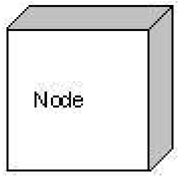
Component

Component describes physical part of a system.



Node

A node can be defined as a physical element that exists at run time.

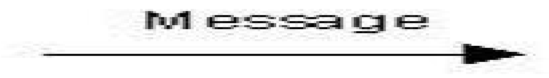


Behavioral things

A behavioral thing consists of the dynamic parts of UML models. Following are the behavioral things:

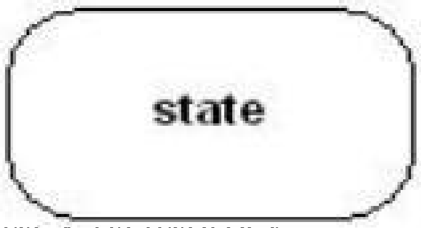
Interaction

Interaction is defined as a behavior that consists of a group of messages exchanged among elements to accomplish a specific task.



State machine

State machine is useful when the state of an object in its life cycle is important. It defines the sequence of states an object goes through in response to events. Events are external factors responsible for state change.

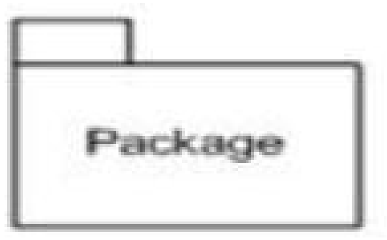


Grouping things

Grouping things can be defined as a mechanism to group elements of a UML model together. There is only one grouping thing available

Package

Package is the only one grouping thing available for gathering structural **and behavioral things**

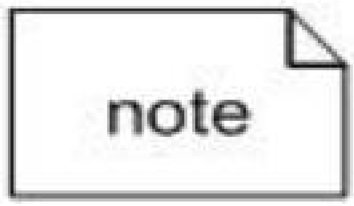


Annotational things

Annotational things can be defined as a mechanism to capture remarks, descriptions, and comments of UML model elements. **Note** is the only one Annotational thing available

Note:

A note is used to render comments, constraints etc of an UML element.



(2) Relationships

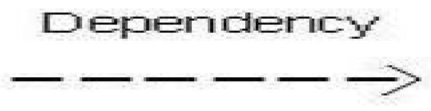
Relationships are another most important building block of UML. It shows how elements are associated with each other and this association describes the functionality of an application.

There are four kinds of relationships available.

- Dependency
- Association
- Generalization.
- Realization

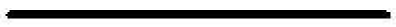
Dependency

Dependency is a relationship between two things in which change in one element also affects the other one.



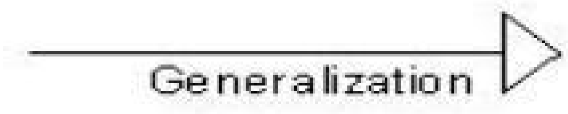
Association

Association is basically a set of links that connects elements of an UML model. It also describes how many objects are taking part in that relationship.



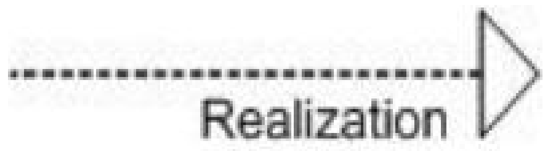
Generalization

Generalization can be defined as a relationship which connects a specialized element with a generalized element. It basically describes inheritance relationship in the world of objects.



Realization

Realization can be defined as a relationship in which two elements are connected. One element describes some responsibility which is not implemented and the other one implements them. This relationship exists in case of interfaces



(3) Diagrams

Each UML diagram is designed to let developers and customers view a software system from a different perspective and in varying degrees of abstraction. UML diagrams commonly created in visual modeling tools include

Use Case Diagram displays the relationship among actors and use cases

Class Diagram models class structure and contents using design elements such as classes, packages and objects. It also displays relationships such as containment, inheritance, associations and others.

Interaction Diagrams

- **Sequence Diagram** displays the time sequence of the objects participating in the interaction. This consists of the vertical dimension (time) and horizontal dimension (different objects).
- **Collaboration Diagram** displays an interaction organized around the objects and their links to one another. Numbers are used to show the sequence of messages.

State Diagram displays the sequences of states that an object of an interaction goes through during its life in response to received stimuli, together with its responses and actions.

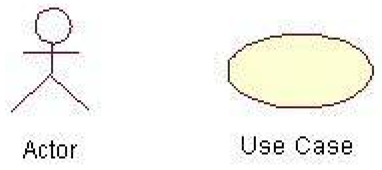
Activity Diagram displays a special state diagram where most of the states are action states and most of the transitions are triggered by completion of the actions in the source states. This diagram focuses on flows driven by internal processing.

Physical Diagrams

- **Component Diagram** displays the high level packaged structure of the code itself. Dependencies among components are shown, including source code components, binary code components, and executable components. Some components exist at compile time, at link time, at run times well as at more than one time.
- **Deployment Diagram** displays the configuration of run-time processing elements and the software components, processes, and objects that live on them. Software component instances represent run-time manifestations of code units.

Use Case Diagrams

A use case is a set of scenarios that describing an interaction between a user and a system. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors.



An actor is represents a user or another system that will interact with the system you are modeling. A use case is an external view of the system that represents some action the user might perform in order to complete a task.

When to Use: Use Cases Diagrams

Use cases are used in almost every project. These are helpful in exposing requirements and planning the project. During the initial stage of a project most use cases should be defined, but as the project continues more might become visible.

Modeling steps for Use case Diagram

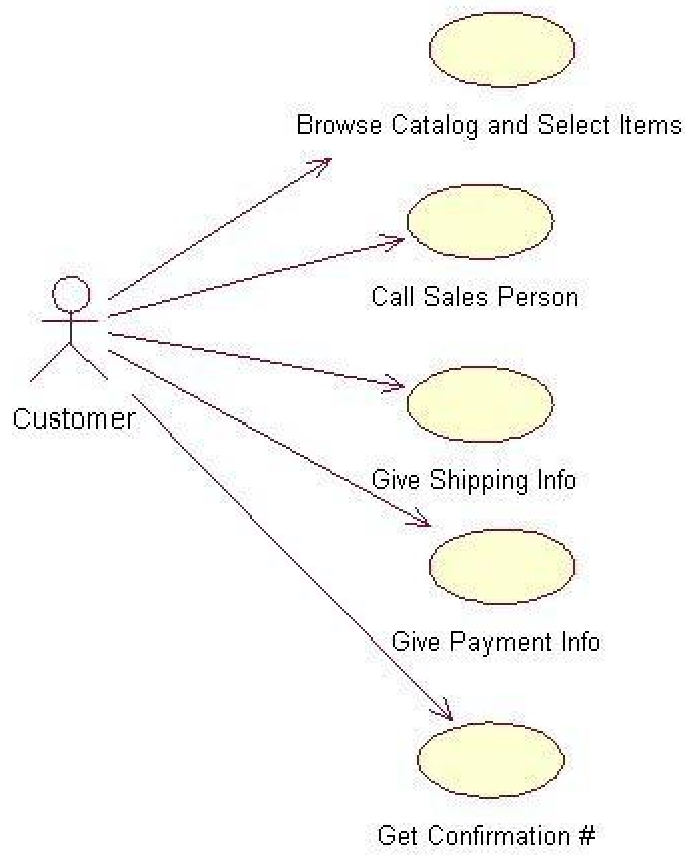
1. Draw the lines around the system and actors lie outside the system.
2. Identify the actors which are interacting with the system.
3. Separate the generalized and specialized actors.
4. Identify the functionality the way of interacting actors with system and specify the behavior of actor.
5. Functionality or behavior of actors is considered as use cases.
6. Specify the generalized and specialized use cases.
7. Se the relationship among the use cases and in between actor and use cases.
8. Adorn with constraints and notes.
9. If necessary, use collaborations to realize use cases.

How to Draw: Use Cases Diagrams

Use cases are a relatively easy UML diagram to draw, but this is a very simplified example. This example is only meant as an introduction to the UML and use cases. Start by listing a sequence of steps a user might take in order to complete an action. For example a user placing an order with a sales company might follow these steps.

1. Browse catalog and select items.
2. Call sales representative.
3. Supply shipping information.
4. Supply payment information.
5. Receive conformation number from salesperson.

These steps would generate this simple use case diagram:



This example shows the customer as a actor because the customer is using the ordering system. The diagram takes the simple steps listed above and shows them as actions the customer might perform. The salesperson could also be included in this use case diagram because the salesperson is also interacting with the ordering system.

From this simple diagram the requirements of the ordering system can easily be derived. The system will need to be able to perform actions for all of the use cases listed. As the project progresses other use cases might appear. The customer might have a need to add an item to an order that has already been placed. This diagram can easily be expanded until a complete description of the ordering system is derived capturing all of the requirements that the system will need to perform.

The <<*extends*>> Relationship

- <<Extends>> relationships represent exceptional or seldom invoked cases.
- The exceptional event flows are factored out of the main event flow for clarity.
- Use cases representing exceptional flows can extend more than one use case.
- The direction of a <<extends>> relationship is to the extended use case

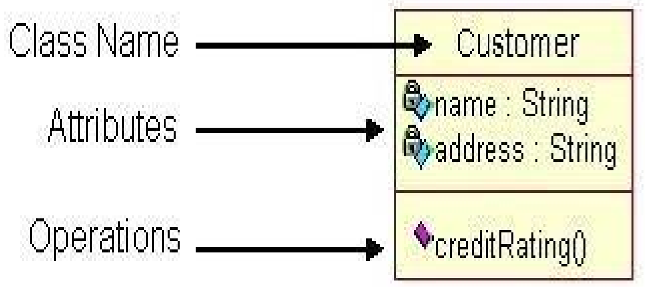
The <<*includes*>> Relationship

- <<Includes>> relationship represents behavior that is factored out of the use case.
- <<Includes>> behavior is factored out for reuse, not because it is an exception.
- The direction of a <<includes>> relationship is to the using use case (unlike <<extends>> relationships).

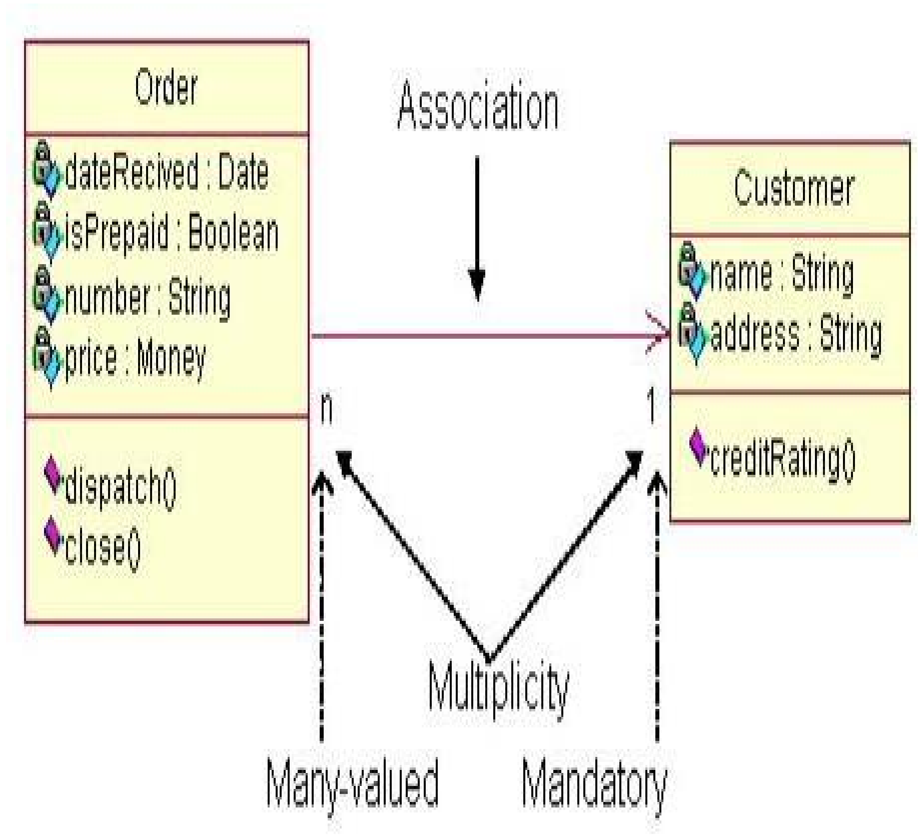
Class Diagrams

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 three different perspectives when designing a system, conceptual, specification, and implementation. These perspectives become evident as the diagram is created and help solidify the design. This example is only meant as an introduction to the UML and class diagrams. Classes are composed of three things: a name, attributes, and operations.

Below is an example of a class.

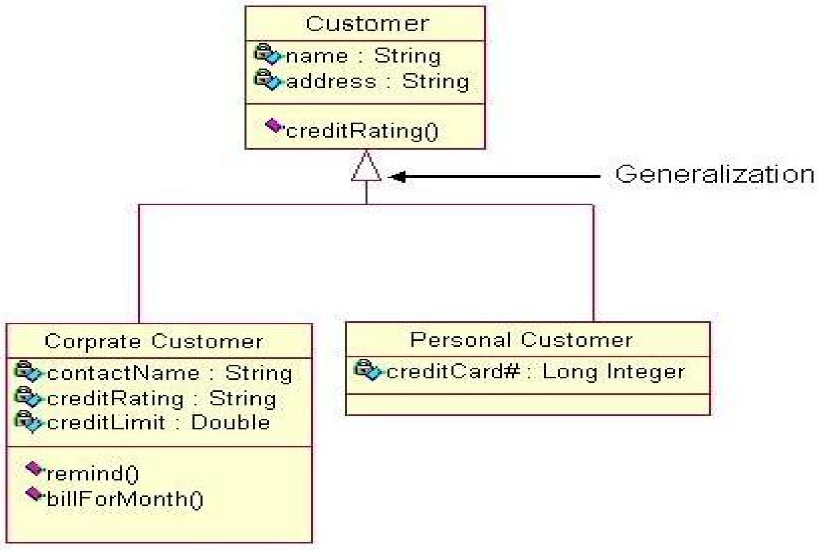


Class diagrams also display relationships such as containment, inheritance, associations and others. Below is an example of an associative relationship:



The association relationship is the most common relationship in a class diagram. The association shows the relationship between instances of classes. For example, the class Order is associated with the class Customer. The multiplicity of the association denotes the number of objects that can participate in then relationship. For example, an Order object can be associated to only one customer, but a customer can be associated to many orders. Another common relationship in

Class diagrams is a generalization. A generalization is used when two classes are similar, but have some differences.



In this example the classes Corporate Customer and Personal Customer have some similarities such as name and address, but each class has some of its own attributes and operations. The class Customer is a general form of both the Corporate Customer and Personal Customerclasses. This allows the designers to just use the Customer class for modules and do not require in-depth representation of each type of customer.

When to Use: Class Diagrams

Class diagrams are used in nearly all Object Oriented software designs. Use them to describe the Classes of the system and their relationships to each other.

Modeling steps for Class Diagrams

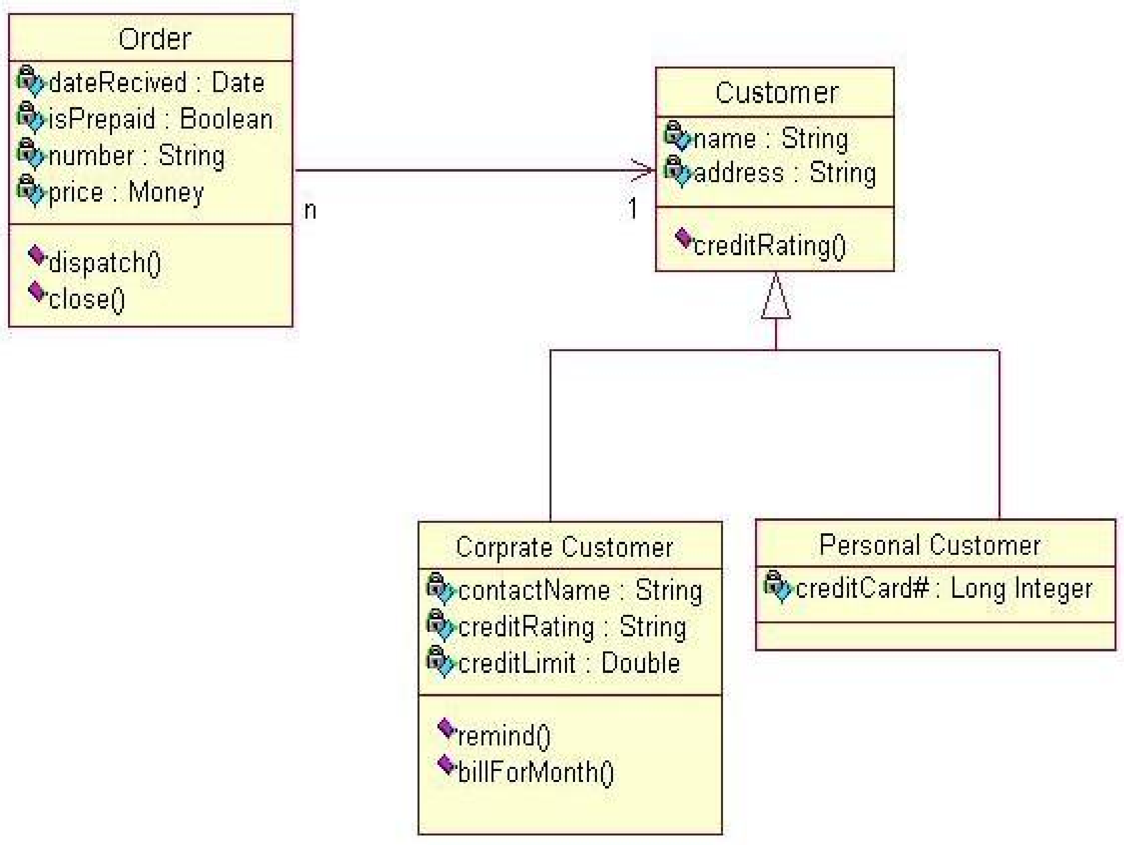
1. Identity the things that are interacting with class diagram.
2. Set the attributes and operations.
3. Set the responsibilities.
4. Identify the generalization and specification classes.
5. Set the relationship among all the things.
6. Adorn with tagged values, constraints and notes.

How to Draw: Class Diagrams

Class diagrams are some of the most difficult UML diagrams to draw. To draw detailed and useful diagrams a person would have to study UML and Object Oriented principles for a long time. Therefore, this page will give a very high level overview of the process.

Before drawing a class diagram consider the three different perspectives of the system the diagram will present; conceptual, specification, and implementation. Try not to focus on one perspective and try see how they all work together.

When designing classes consider what attributes and operations it will have. Then try to determine how instances of the classes will interact with each other. These are the very first steps of many in developing a class diagram. However, using just these basic techniques one can develop a complete view of the software system.



Interaction Diagrams

Interaction diagrams model the behavior of use cases by describing the way groups of objects interact to complete the task. The two kinds of interaction diagrams are **sequence** and **collaboration** diagrams. This example is only meant as an introduction to the UML and interaction diagrams.

When to Use: Interaction Diagrams

Interaction diagrams are used when you want to model the behavior of several objects in a use case. They demonstrate how the objects collaborate for the behavior. Interaction diagrams do not give a in depth representation of the behavior. If you want to see what a specific object is doing for several use cases use a state diagram. To see a particular behavior over many use cases or threads use an activity diagrams.

How to Draw: Interaction Diagrams

Sequence diagrams, collaboration diagrams, or both diagrams can be used to demonstrate the interaction of objects in a use case. Sequence diagrams generally show the sequence of events that occur. Collaboration diagrams demonstrate how objects are statically connected. Both diagrams are relatively simple to draw and contain similar elements.

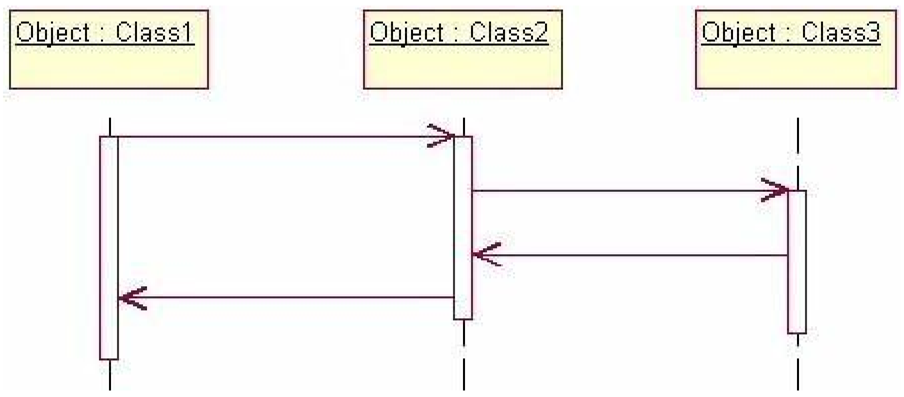
Sequence diagrams:

Sequence diagrams demonstrate the behavior of objects in a use case by describing the objects and the messages they pass. the diagrams are read left to right and descending. The example below shows an object of class 1 start the behavior by sending a message to an object of class 2. Messages pass between the different objects until the object of class 1 receives the final message

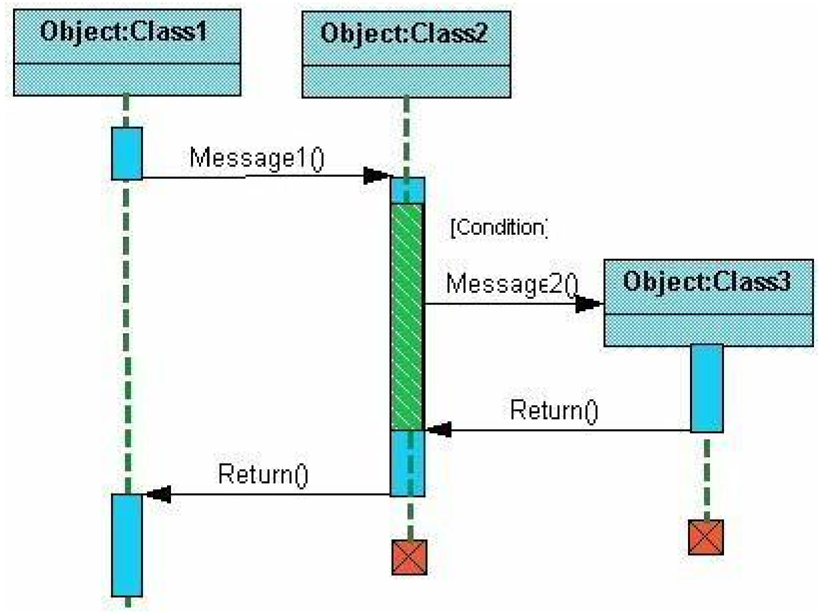
Modeling steps for Sequence Diagrams

1. Set the context for the interactions, system, subsystem, classes, object or use cases.
2. Set the stages for the interactions by identifying objects which are placed as actions in interaction diagrams.

3. Lay them out along the X-axis by placing the important object at the left side and others in the next subsequent.
4. Set the lifelines for each and every object by sending create and destroy messages.
5. Start the message which is initiating interactions and place all other messages in the increasing order of items.
6. Specify the time and space constraints.
7. Set the pre and post conditions.

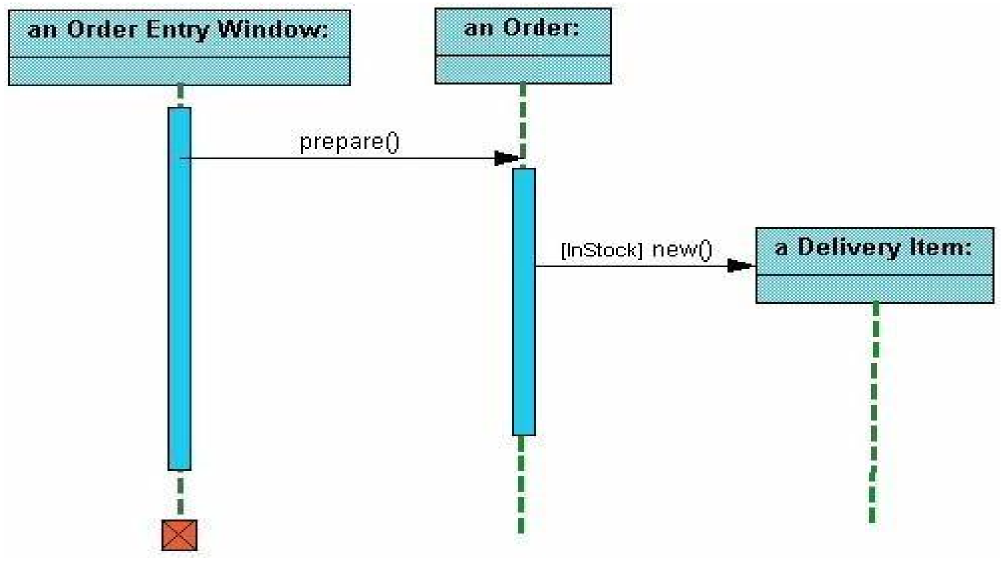


Below is a slightly more complex example. The light blue vertical rectangles the objects activation while the green vertical dashed lines represent the life of the object. The green vertical rectangles represent when a particular object has control. The ☒ represents when the object is destroyed. This diagrams also shows conditions for messages to be sent to other object. The condition is listed between brackets next to the message. For example, a [condition] has to be met before the object of class 2 can send a message() to the object of class 3.

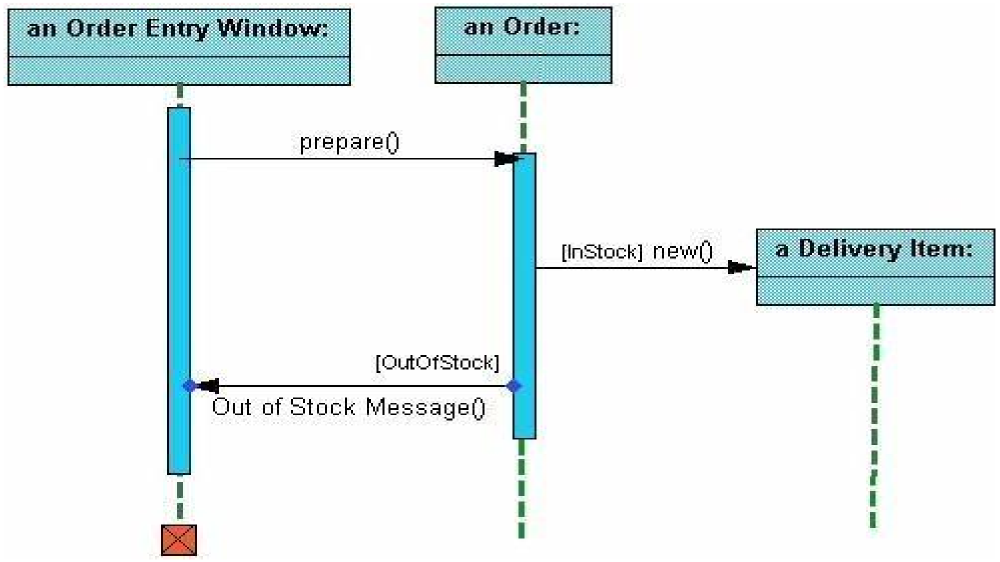


The next diagram shows the beginning of a sequence diagram for placing an order. The object an Order Entry Window is created and sends a message to an Order object to prepare the order. Notice the the names of the objects are followed by a colon. The names of the classes the objects belong to do not have to be listed. However the colon is required to denote that it is the name of an object following the objectName:className naming system.

Next the Order object checks to see if the item is in stock and if the [InStock] condition is met it sends a message to create an new Delivery Item object.



The next diagrams adds another conditional message to the Order object. If the item is [OutOfStock] it sends a message back to the Order Entry Window object stating that the object is out of stack.



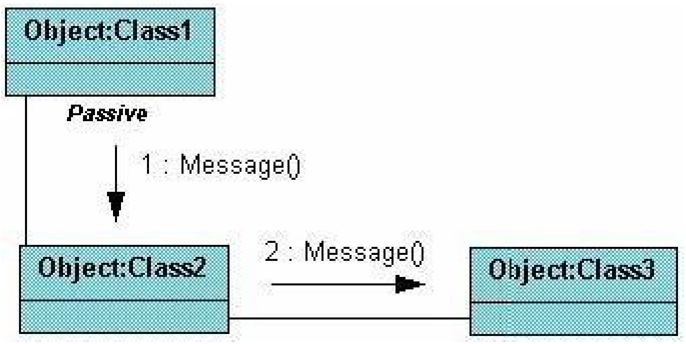
This simple diagram shows the sequence that messages are passed between objects to complete a use case for ordering an item.

Collaboration diagrams

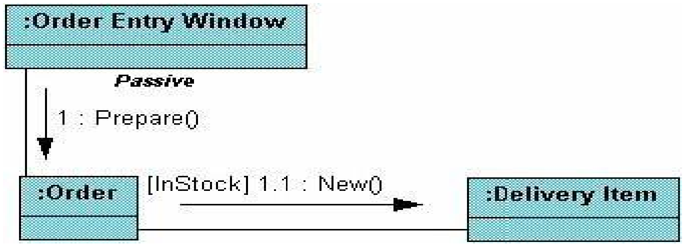
Collaboration diagrams are also relatively easy to draw. They show the relationship between objects and the order of messages passed between them. The objects are listed as icons and arrows indicate the messages being passed between them. The numbers next to the messages are called sequence numbers. As the name suggests, they show the sequence of the messages as they are passed between the objects. There are many acceptable sequence numbering schemes in UML. A simple 1, 2, 3... format can be used, as the example below shows, or for more detailed and complex diagrams a 1, 1.1 ,1.2, 1.2.1... scheme can be used.

Modeling steps for Collaboration Diagrams

1. Set the context for interaction, whether it is ystem, subsystem, operation or class or one scenario of use case or collaboration.
2. Identify the objects that play a role in the interaction. Lay them as vertices in graph, placing important objects in centre and neigboring objects to outside.
3. Set the initial properties of each of these objects. If the attributes or tagged values of an object changes in significant ways over the interaction, place a duplicate object, update with these new values and connect them by a message stereotyped as become or copy.
4. Specify the links among these objects.Lay the association links first represent structural connection lay out other links and adorn with stereotypes.
5. Starting with the message that initiates this interaction, attach each subsequent message to appropriate link, setting sequence number as appropriate.



The example below shows a simple collaboration diagram for the placing an order use case. This time the names of the objects appear after the colon, such as :Order Entry Window following the objectName:className naming convention. This time the class name is shown to demonstrate that all of objects of that class will behave the same way.



State chart diagram

State diagrams are used to describe the behavior of a system. State diagrams describe all of the possible states of an object as events occur. Each diagram usually represents objects of a single class and track the different states of its objects through the system.

Modeling steps for State chart Diagram

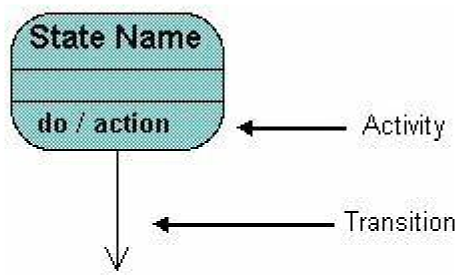
1. Choose the context for state machine, whether it is a class ,a use case, or the system as a whole.
2. Choose the initial & final states of the objects.
3. Decide on the stable states of the object by considering the conditions in which the object may exist for some identifiable period of time. Start with the high level states of the objects & only then consider its possible substates.
4. Decide on the meaningful partial ordering of stable states over the lifetime of the object.
5. Decide on the events that may trigger a transition from state to state. Model these events as triggers to transitions that move from one legal ordering of states to another.
6. Attach actions to these transitions and/or to these states.
7. Consider ways to simplify your machine by using substates, branches, forks, joins and history states.
8. Check that all states are reachable under some combination of events.
9. Check that no state is a dead from which no combination of events will transition the object out of that state.
10. Trace through the state machine, either manually or by using tools, to check it against expected sequence of events & their responses.

When to Use: State Diagrams

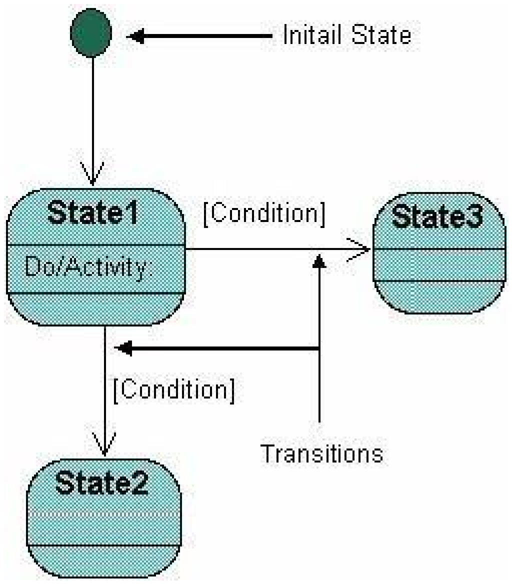
Use state diagrams to demonstrate the behavior of an object through many use cases of the system. Only use state diagrams for classes where it is necessary to understand the behavior of the object through the entire system. Not all classes will require a state diagram and state diagrams are not useful for describing the collaboration of all objects in a use case. State diagrams are other combined with other diagrams such as interaction diagrams and activity diagrams.

How to Draw: State Diagrams

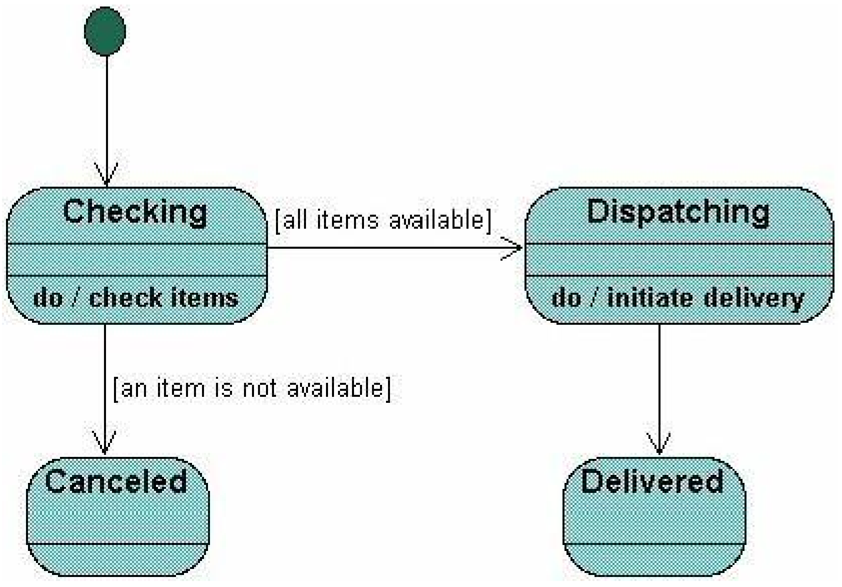
State diagrams have very few elements. The basic elements are rounded boxes representing the state of the object and arrows indicting the transition to the next state. The activity section of the state symbol depicts what activities the object will be doing while it is in that state.



All state diagrams being with an initial state of the object. This is the state of the object when it is created. After the initial state the object begins changing states. Conditions based on the activities can determine what the next state the object transitions to.

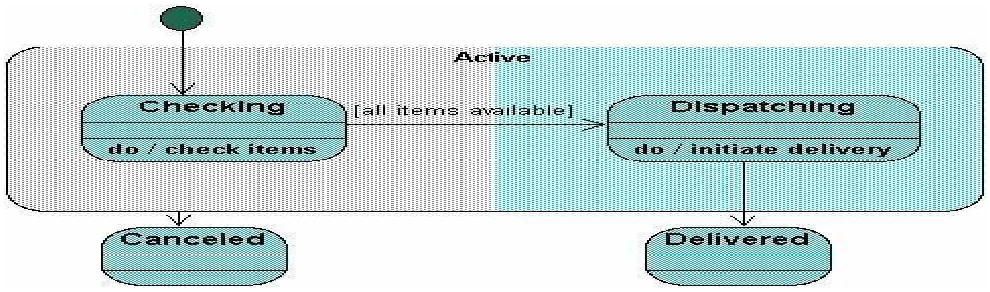


Below is an example of a state diagram might look like for an Order object. When the object enters the Checking state it performs the activity "check items." After the activity is completed the object transitions to the next state based on the conditions [all items available] or [an item is not available]. If an item is not available the order is canceled. If all items are available then the order is dispatched. When the object transitions to the Dispatching state the activity "initiate delivery" is performed. After this activity is complete the object transitions again to the Delivered state.



State diagrams can also show a super-state for the object. A super-state is used when many transitions lead to the certain state. Instead of showing all of the transitions from each state to the redundant state a super-state can be used to show that all of the states inside of the super-state can transition to the redundant state. This helps make the state diagram easier to read.

The diagram below shows a super-state. Both the Checking and Dispatching states can transition into the Canceled state, so a transition is shown from a super-state named Active to the state Cancel. By contrast, the state Dispatching can only transition to the Delivered state, so we show an arrow only from the Dispatching state to the Delivered state.



Activity Diagram

Activity diagrams describe the workflow behavior of a system. Activity diagrams are similar to state diagrams because activities are the state of doing something. The diagrams describe the state of activities by showing the sequence of activities performed. Activity diagrams can show activities that are conditional or parallel.

Modeling steps for Activity Diagrams

1. Select the object that have high level responsibilities.
2. These objects may be real or abstract. In either case, create a swimlane for each important object.
3. Identify the precondition of initial state and post conditions of final state.
4. Beginning at initial state, specify the activities and actions and render them as activity states or action states.
5. For complicated actions, or for a set of actions that appear multiple times, collapse these states and provide separate activity diagram.

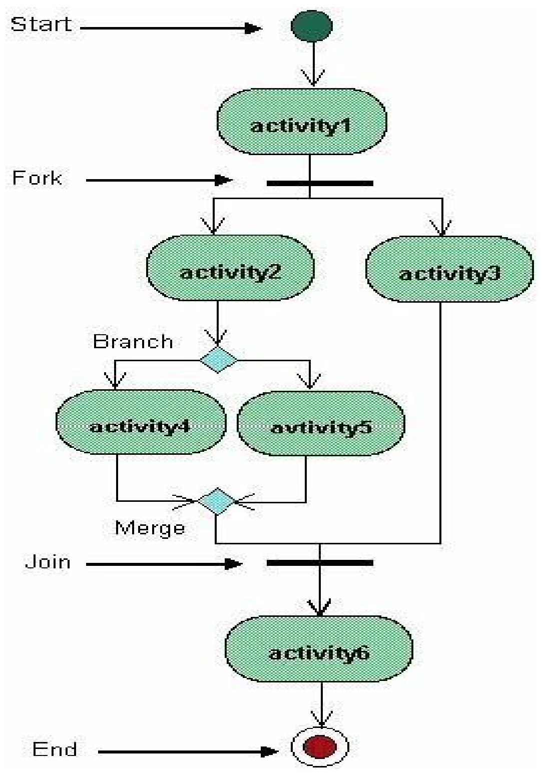
6. Render the transitions that connect these activities and action states.
7. Start with sequential flows, consider branching, fork and joining.
8. Adorn with notes tagged values and so on.

When to Use: Activity Diagrams

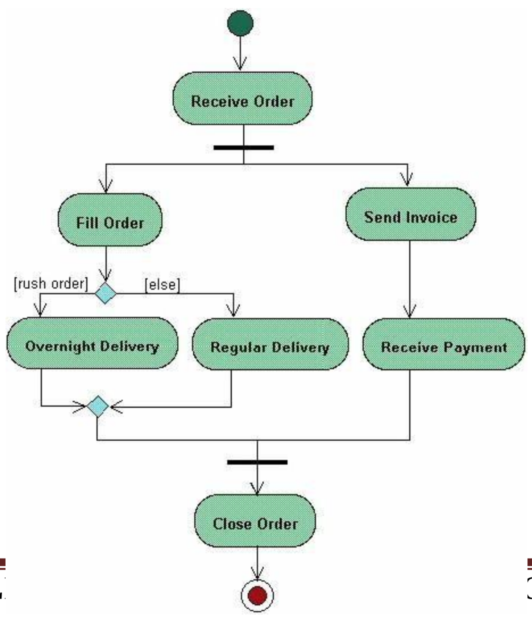
Activity diagrams should be used in conjunction with other modeling techniques such as interaction diagrams and state diagrams. The main reason to use activity diagrams is to model the workflow behind the system being designed. Activity Diagrams are also useful for analyzing a use case by describing what actions need to take place and when they should occur; describing a complicated sequential algorithm; and modeling applications with parallel processes. However, activity diagrams should not take the place of interaction diagrams and state diagrams. Activity diagrams do not give detail about how objects behave or how objects collaborate.

How to Draw: Activity Diagrams

Activity diagrams show the flow of activities through the system. Diagrams are read from top to bottom and have branches and forks to describe conditions and parallel activities. A fork is used when multiple activities are occurring at the same time. The diagram below shows a fork after activity1. This indicates that both activity2 and activity3 are occurring at the same time. After activity2 there is a branch. The branch describes what activities will take place based on a set of conditions. All branches at some point are followed by a merge to indicate the end of the conditional behavior started by that branch. After the merge all of the parallel activities must be combined by a join before transitioning into the final activity state.



Below is a possible activity diagram for processing an order. The diagram shows the flow of actions in the system's workflow. Once the order is received the activities split into two parallel sets of activities. One side fills and sends the order while the other handles the billing. On the Fill Order side, the method of delivery is decided conditionally. Depending on the condition either the Overnight Delivery activity or the Regular Delivery activity is performed. Finally the parallel activities combine to close the order.



Experiment 1: COURSE MANAGEMENT SYSTEM (CMS)

A **course management system (CMS)** is a collection of software tools providing an online environment for course interactions. A CMS typically includes a variety of online tools and environments, such as:

- An area for faculty posting of class materials such as course syllabus and handouts
- An area for student posting of papers and other assignments
- A grade book where faculty can record grades and each student can view his or her grades
- An integrated email tool allowing participants to send announcement email messages to the entire class or to a subset of the entire class
- A chat tool allowing synchronous communication among class participants
- A threaded discussion board allowing asynchronous communication among participants.

In addition, a CMS is typically integrated with other databases in the university so that students enrolled in a particular course are automatically registered in the CMS as participants in that course.

The Course Management System (CMS) is a web application for department personnel, Academic Senate, and Registrar staff to view, enter, and manage course information formerly Submitted via paper. Departments can use CMS to create new course proposals, submit changes for existing courses, and track the progress of proposals as they move through the stages of online approval.

Problem Analysis and Project Planning

A course management system is a set of tools that enables an online environment for course interaction i.e. to create online course content and post it on the Web without having to handle HTML or other programming languages.

Course management system become an integral a part of the upper education system.

They create teaching and course management easier by providing a framework and set of tools for faculties and for students. The executive aspects of such systems could include class rosters (a group of people or things) and therefore the ability to record students' grades. With relevance the teaching aspects, however, it might include learning objects, class exercises, quizzes and tests. The CMS might also include tools for real-time chat, integrated email tool allowing participants to send announcement email messages to entire class or to a subset of the entire class. The CMS tool additionally focuses on all aspects of teaching, learning and teacher-student interaction.

Software Requirement Analysis

(1)Module Summary:

(1.1)Administrator Module:

Admin can produce accounts for college students and faculties and make course programmed list and add faculties and students to it course list.

Admin can produce course details exploitation course creation kind that consists in fact name, course id, and choose student. Using Student creator kind student details are entered to information. User name, adapt username, password, given name and name, ID. After accounts are produced supported every students and instructors are divided and accessorial to list exploitation create missing students kind.

(1.2)Faculty Module:

It can check student's papers, their assignments and assign grades for work. This module accommodates preparation menu, choose student for grades.

(1.3)Students Module: Student can register with application or the proposed system and login with user name and password. He will check and submit assignment and his/her grade. Every student can have id.

(2) Functional and Non-Functional Requirements

(2.1)Functional Requirements:

(2.1.1) Creating Courses

Integration with registration system: The system shall periodically upload the latest registrar's classes list to determine courses that offered in the current semester.

The system shall generate course for each class that registered and determine the current set of students that enrolled in that class.

The system shall allow course instructor to update course content.

(2.1.2)Grade Management

a. Allow grades to be entered online: The system shall allow instructors to enter and modify grades online.
b. Allow students to access their grades online: The system shall allow student to log in their account and check their grades at any time.
c. The system shall provide statistical information such as averages, standard deviation, and median about student's grades.
d. Track and Handle Re-grade Requests: The system shall be able to track and handle requests for re- grades, and all information about re-grades shall be available to the student, and the course instructor.

(2.1.3)Paper and Assignment Submission

a. Accept submissions in multiple formats: The system shall accept submissions in multiple formats, including .zip, .cpp, .txt, .doc,etc.
b. Support for late submissions: The system shall provide information about late submissions, and also disallow submissions after a certain period of time.
c. Integration with grade management: The homework submission system shall be integrated with the grade management by using online grading templates that can be filled out, and automatically annotating code with line numbers.
   1. Assignment grades can be automatically posted to student account.
   2. Grader comments can be sent along with the grades.

(2.1.4)Create Accounts

a. The system shall automatically create accounts for each class.
   1. Create one account for course instructor regardless to the number of classes that he/she teaches.
   2. The account username is course name and its number.
   3. The account password is the same password that in Academic Information System (AIS).
   4. Any change in the password in AIS the system shall reflect it on the instructor account password in CMS.
   5. Create one account for each student that registered in this class.

6. The account username is course name and its number.
7. The account password is the same password that in Student Information System (SIS).
8. Any change in the password in SIS the system shall reflect it on the student account password in CMS.

b. Instructor account contain the classes that he/she teach, each class contain list of student that ordered based on student serial number.

c. Instructor can modify student grades from his/her account.

(2.2)Non-Functional Requirements:

(2.2.1)Response Time

a. Average response time shall be less than 2 second.

(2.2.2) Throughput

a. The system shall accommodate 1000 booked per minute.

(2.2.3) Recovery Time

a. In case of a system failure, redundant system shall resume operations within 30 sec.
b. Average repair time shall be less than 1 hour.

(2.2.4)Start-up/Shutdown Time

a. The system shall be operational within 1 minute of starting-up.

(2.2.5) Capacity

a. The system accommodates 4000 concurrent users.

(2.2.6)Utilization of Resources

a. The system shall store in the database no more than one million transactions.
b. If the database grows over this limit, old transaction shall be backed up and deleted from the operational database.

(2.2.7) Security

a. Firewall Protection: The course management software system shall run inside a firewall.
b. Support different roles: The system shall support different roles for users, such as Instructors, Students, and administrative staff, the user logged in with given role should only be allowed access consistent with that role. For example a student shall only be allowed to see he/she grades not to modify it.

(2.2.8) Reliability

a. The system shall not be down more 2 times in year.

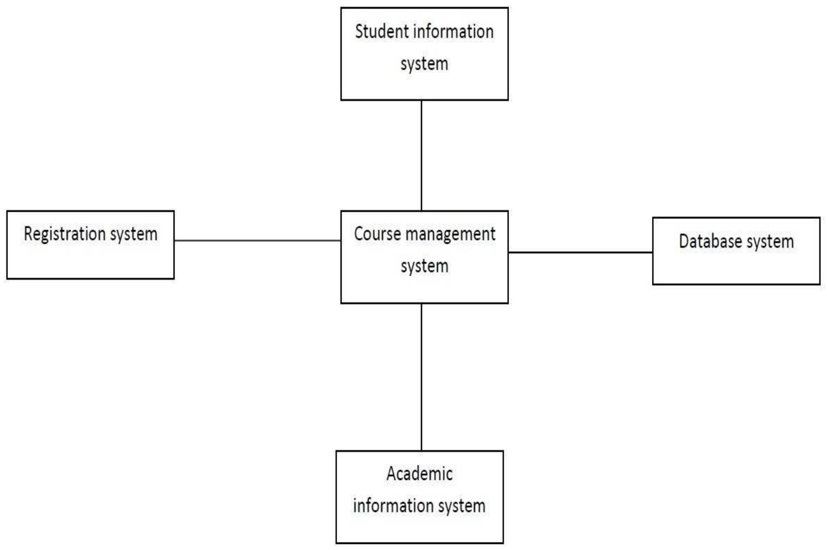
(2.2.9) Scalability

a. Scaling the system to large number of users: large courses will have hundreds of students.
b. The system shall be able to handle the load for such courses, especially near assignment deadlines when many students can be expected to access the course management system.

<u>Data Modeling</u>

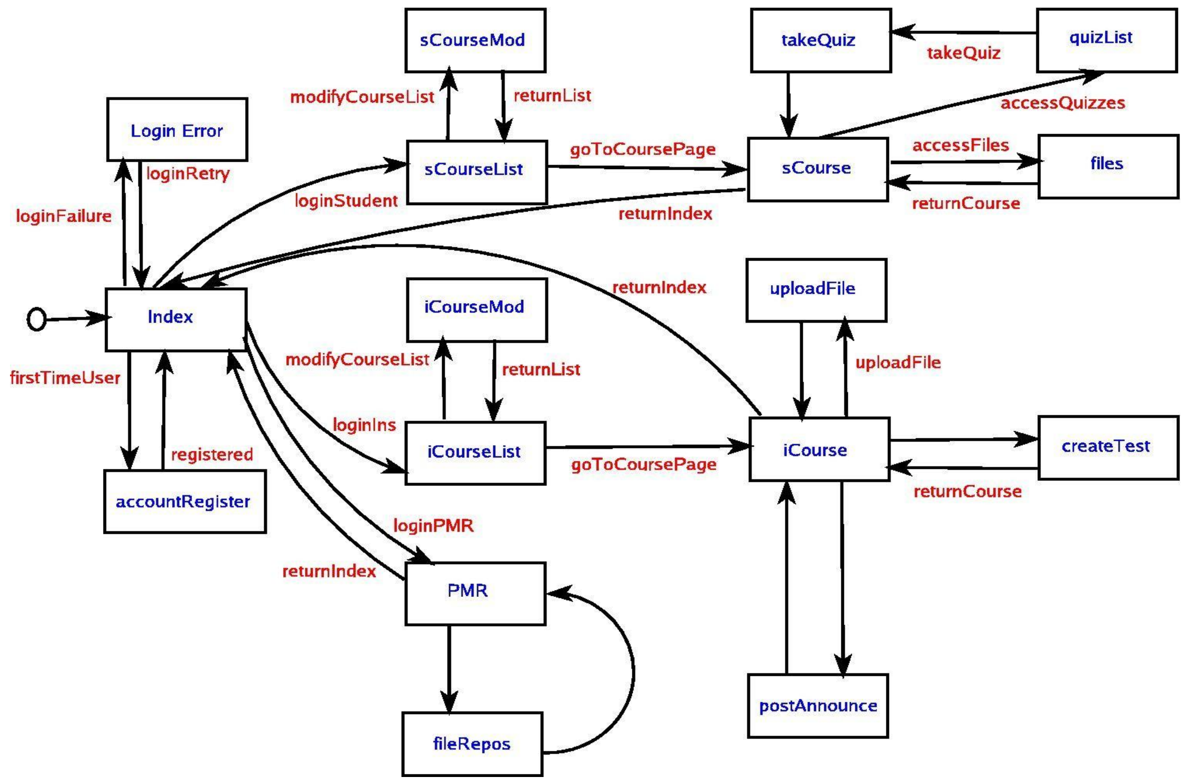
(1)Product Perspective

The system will be operating within university environment. This environment has anther systems that will interact with this system so we need interfaces between these systems.



(2) Flow Chart

The below diagram will provide the overall flow of the project.



(3) Data Dictionary

(3.1)StudentDetails

FIELD NAME	TYPE	CONSTRAINTS
Sid	Varchar2	Primary key
Name	Varchar2	
Roll_No	Varchar2	Notnull
Regulation	Varchar	
Courseid	Number	Foreign key
grade	Char	
Fid	Varchar2	Foreign Key

(3.2)CourseDetails

FIELD NAME	TYPE	CONSTRAINTS
Courseid	Number	Primary key
CourseName	Varchar2	
Start_date	Date	

End_date	Date	
Subject	Varchar2	not null

(3.3)FacultyDetails

FIELD NAME	TYPE	CONSTRAINTS
Fid	Varchar2	Primary key
Name	Varchar2	
Courseid	Number	Foreign Key
Designation	Varchar	
Subject	Varchar	

(3.4)LoginDetails

FIELD NAME	TYPE	CONSTRAINTS
Userid	Varchar2	Unique
Password	Varchar2	Not null

Software Designing

UML

UML stands for Unified Modeling Language. This object-oriented system of notation has evolved from the work of Grady Booch, James Rum Baugh, Ivar Jacobson, and the Rational Software Corporation. These renowned computer scientists fused their respective technologies into a single, standardized model. Today, UML is accepted by the Object Management Group (OMG) as the standard for modeling object oriented programs.

UML Diagrams

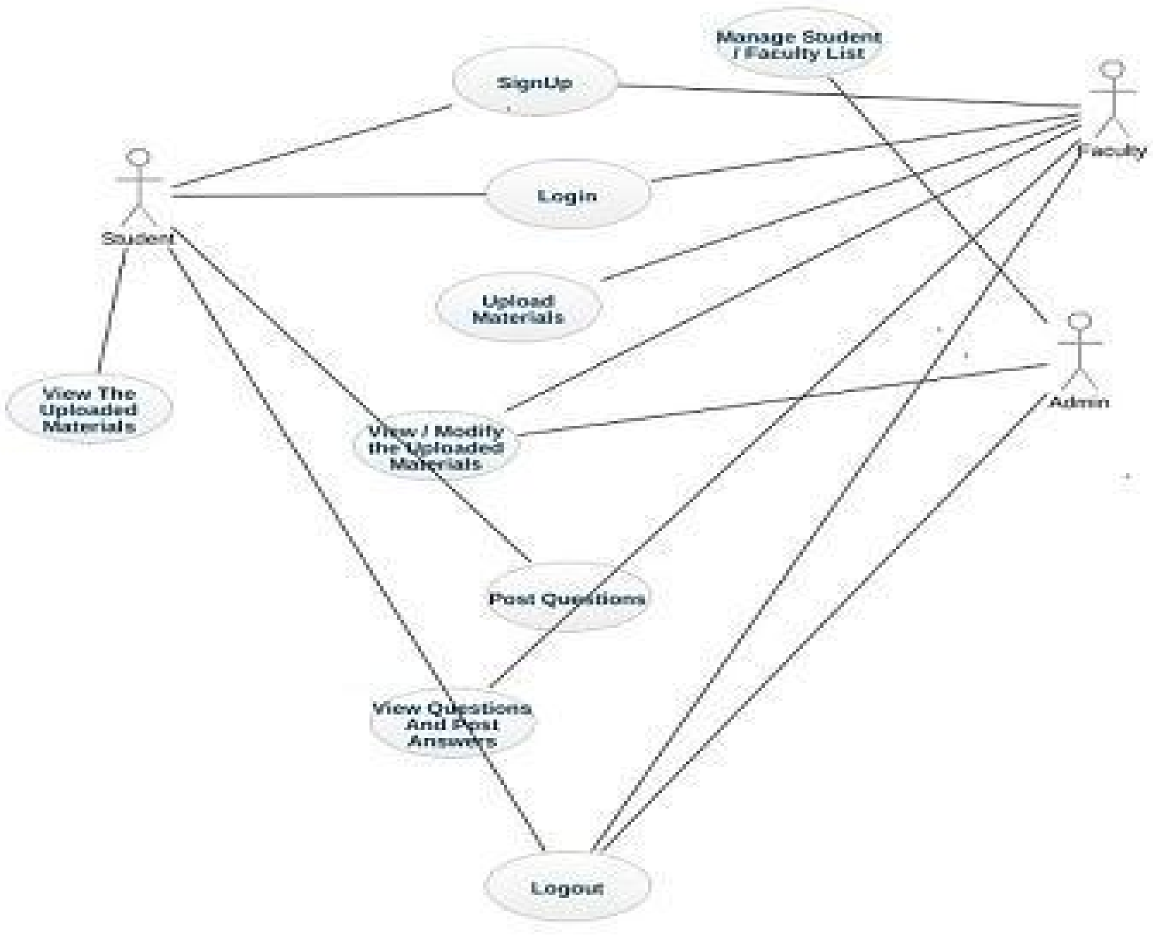
UML defines nine types of diagrams: class (package), object, use case, sequence, collaboration, state chart, activity, component, and deployment diagram.

(1) Use Case Diagram

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

The purposes of use case diagrams can be defined as follows −

- Used to gather the requirements of a system.
- Used to get an outside view of a system.
- Identify the external and internal factors influencing the system.
- Show the interaction among the requirements is actors.

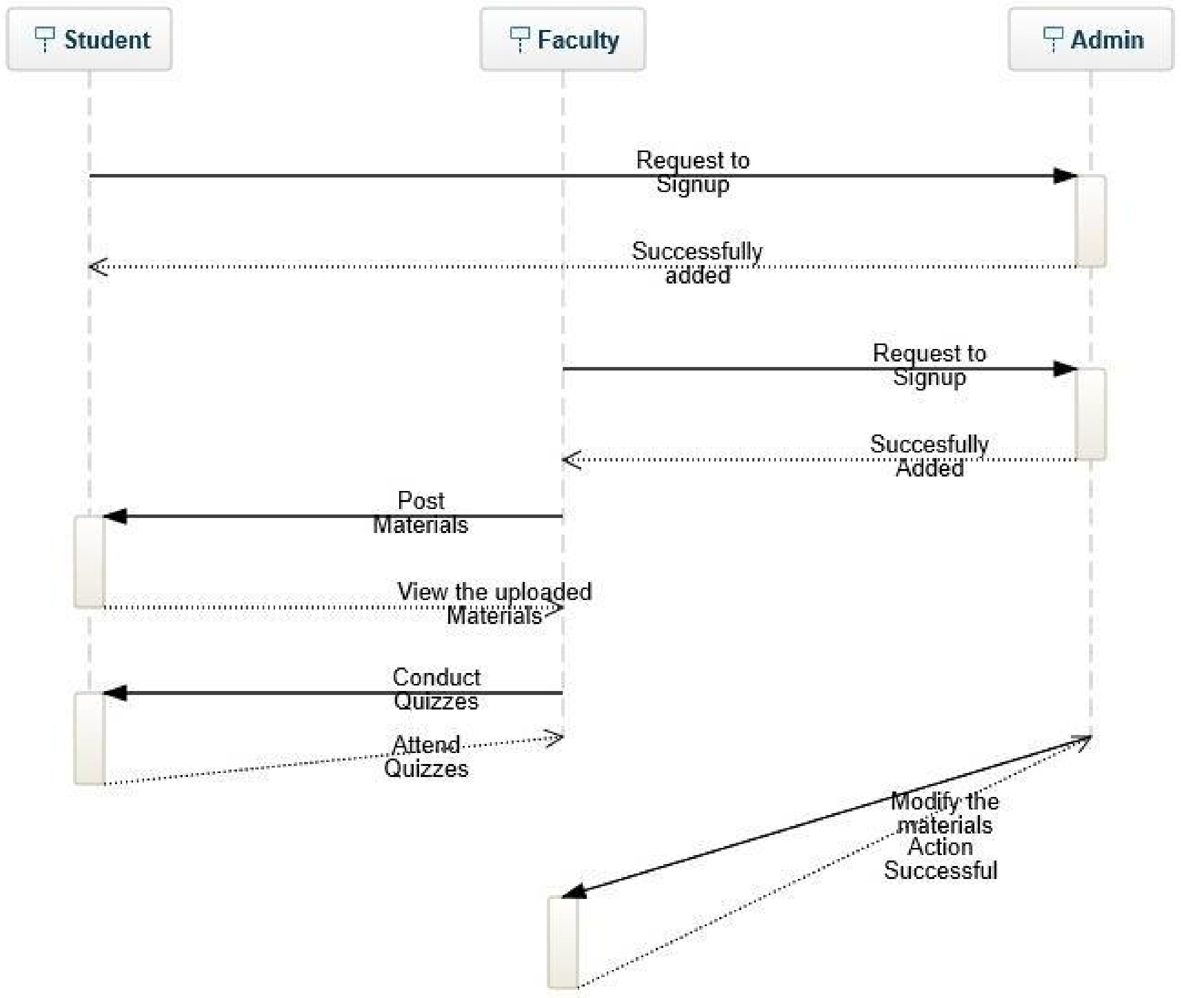

Manage Student / Faculty List
SignUp
Faculty
Student
Login
Upload Materials
Admin
View The Uploaded Materials
View / Modify the Uploaded Materials
Post Questions
View Questions And Post Answers
Logout

Sequence Diagram

This interactive behavior is represented in UML by Sequence **diagram**. Sequence diagram emphasizes on time sequence of messages that send and receive messages.

Following things are to be identified clearly before drawing the sequence diagram

- Objects taking part in the interaction.
- Message flows among the objects.
- The sequence in which the messages are flowing.
- Object organization.

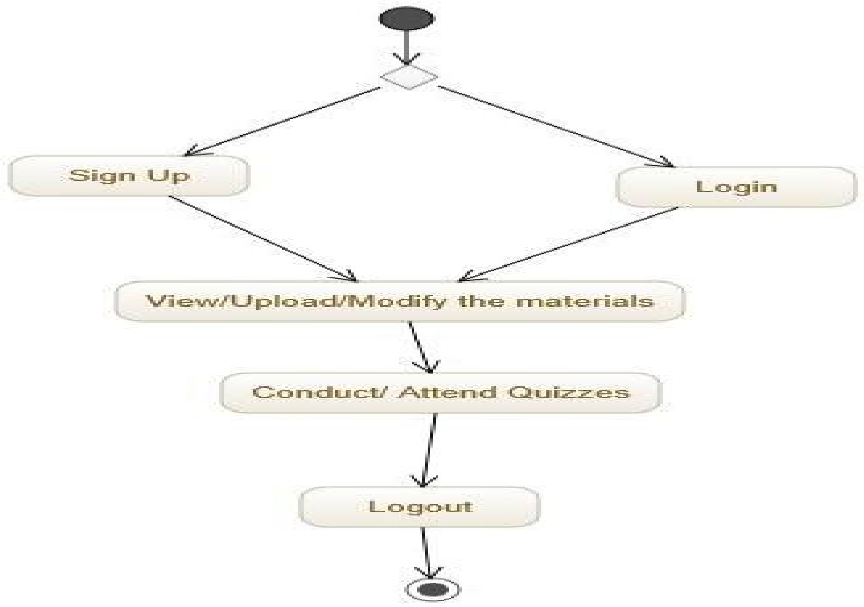


Activity Diagram

The basic purposes of activity diagrams are to captures the dynamic behavior of the system. Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

The purpose of an activity diagram can be described as −

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

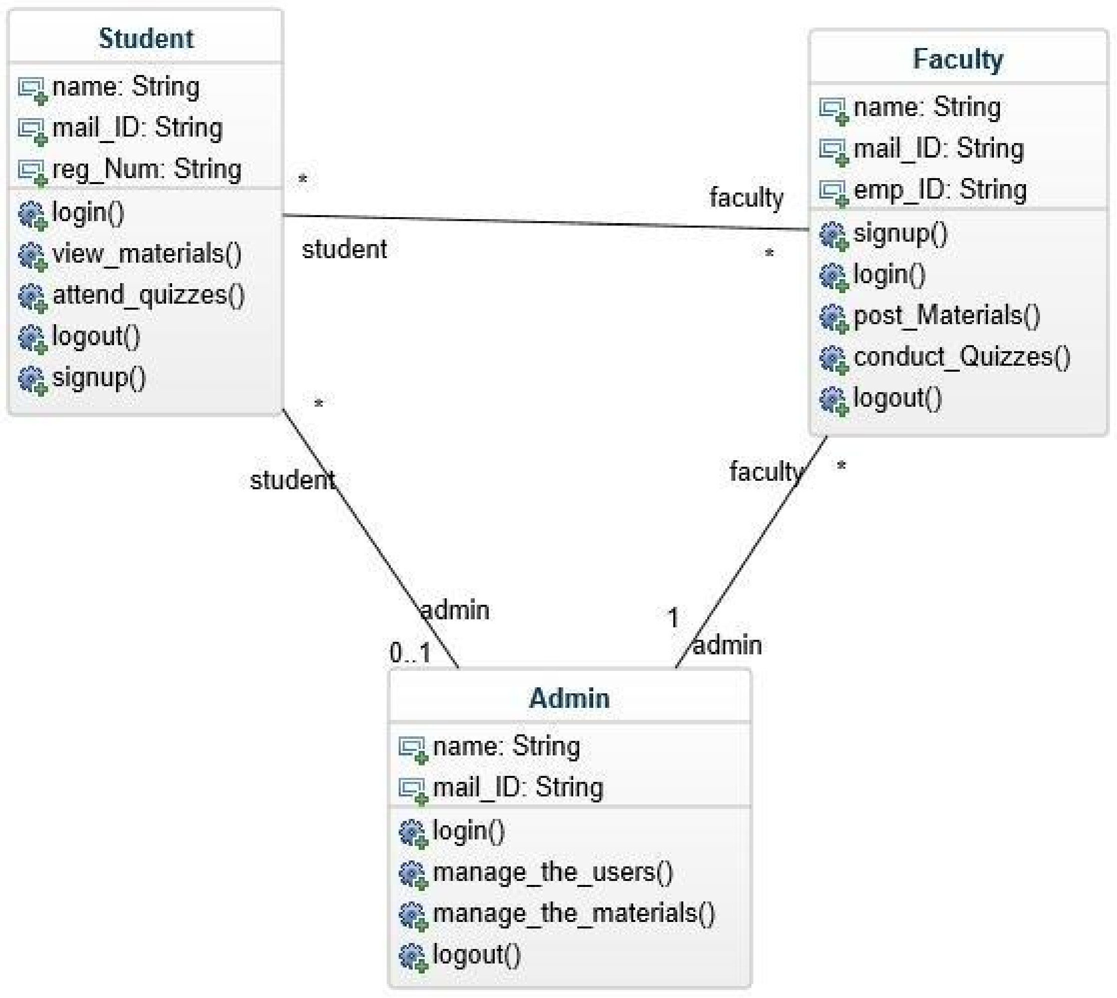


Class Diagram

The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

The purpose of the class diagram can be summarized as −

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering.

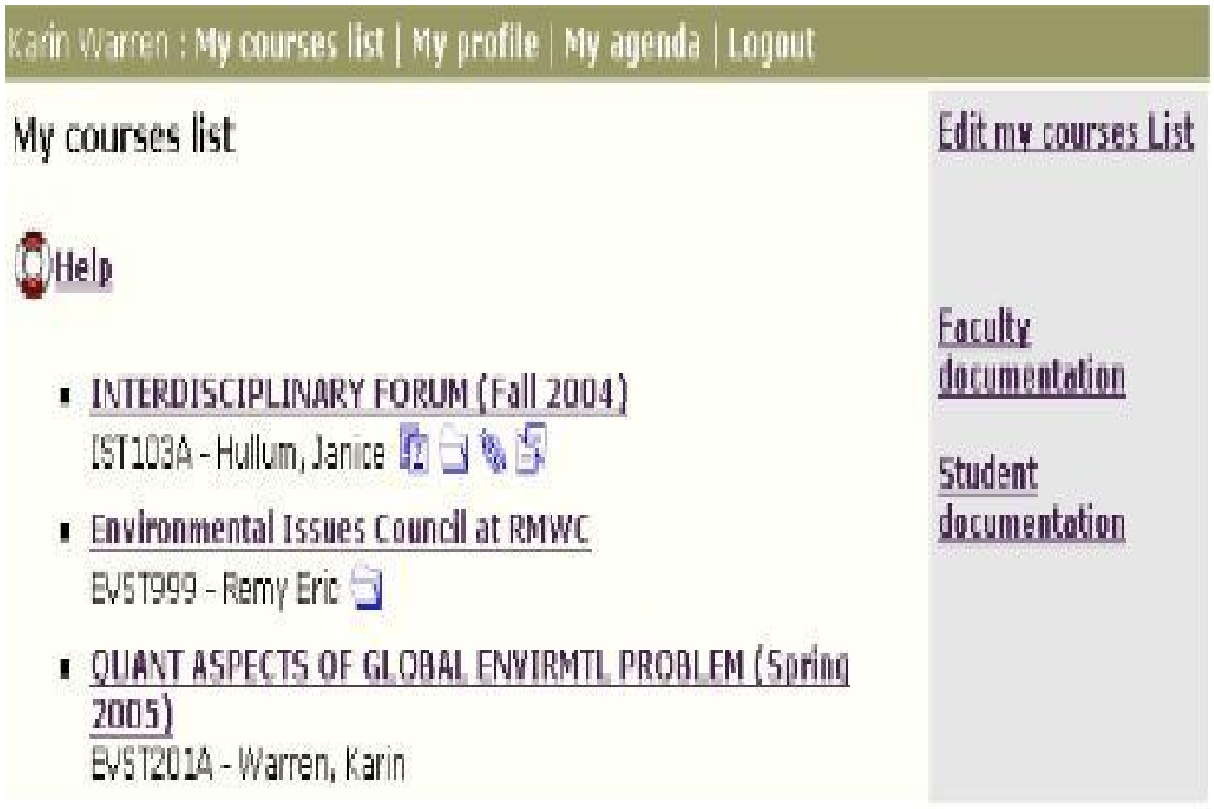


Prototype model

Prototype is a working model of software with some limited functionality. The prototype does not always hold the exact logic used in the actual software application and is an extra effort to be considered under effort estimation.

Prototyping is used to allow the users evaluate developer proposals and try them out before implementation. It also helps understand the requirements which are user specific and may not have been considered by the developer during product design.

To get course List



Karin Warren : My courses list | My profile | My agenda | Logout

My courses list

Help

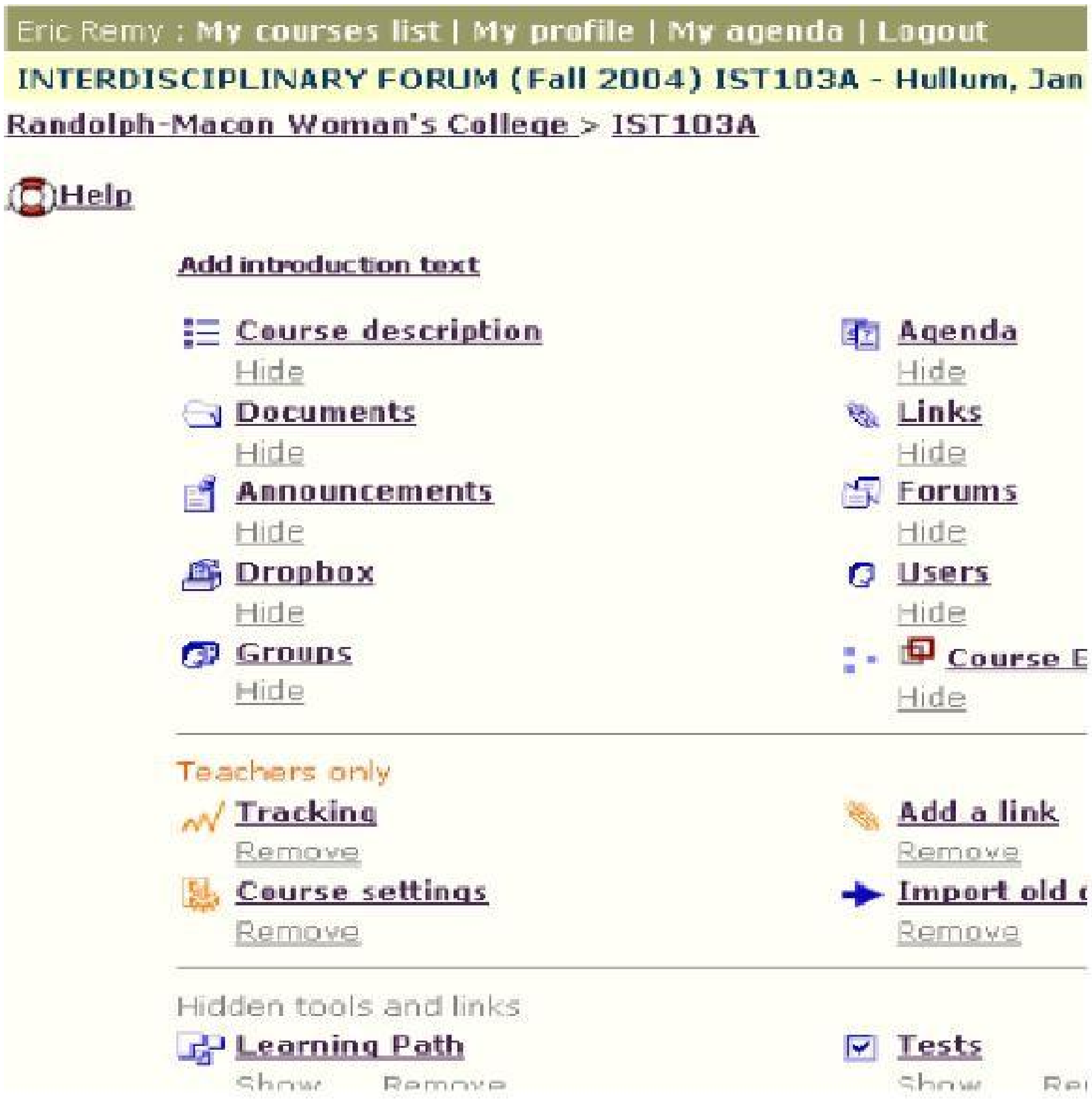
- INTERDISCIPLINARY FORUM (Fall 2004)
  IST103A - Hullum, Janice
- Environmental Issues Council at RMWC
  EVST999 - Remy Eric
- QUANT ASPECTS OF GLOBAL ENVIRMTL PROBLEM (Spring 2005)
  EVST201A - Warren, Karin

Edit my courses List

Faculty documentation

Student documentation

Following fields are available in this project



Eric Remy : My courses list | My profile | My agenda | Logout

INTERDISCIPLINARY FORUM (Fall 2004) IST103A - Hullum, Jan

Randolph-Macon Woman's College > IST103A

Help

Add introduction text

Course description
Hide
Documents
Hide
Announcements
Hide
Dropbox
Hide
Groups
Hide

Agenda
Hide
Links
Hide
Forums
Hide
Users
Hide
Course E
Hide

Teachers only

Tracking
Remove
Course settings
Remove

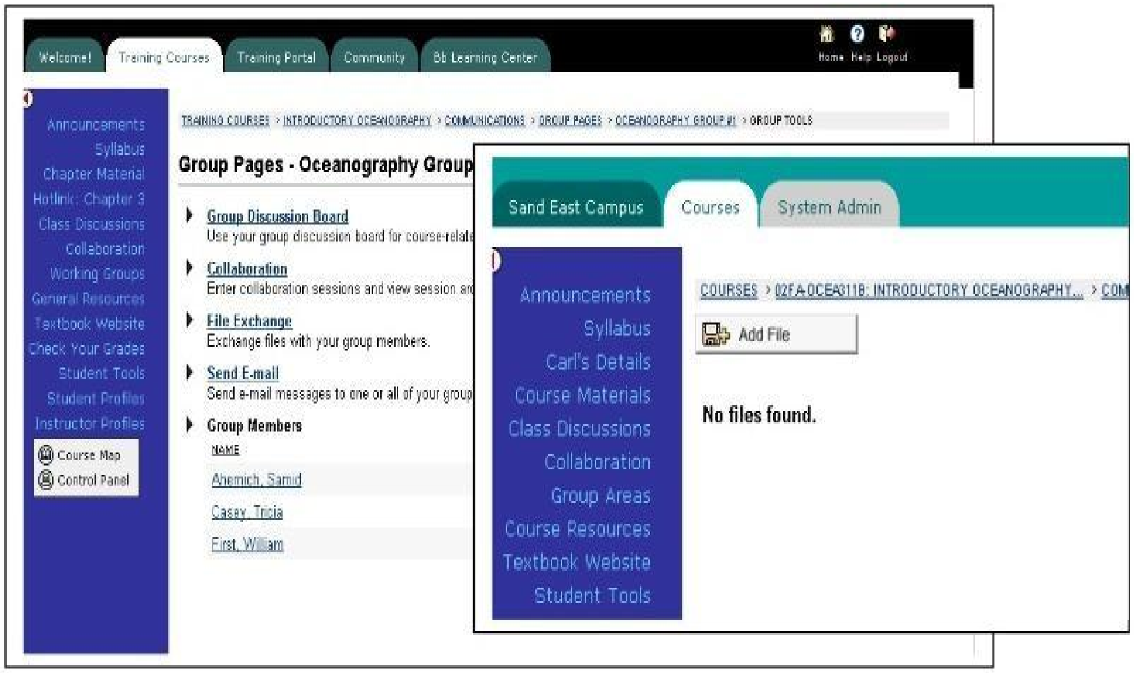
Add a link
Remove
Import old
Remove

Hidden tools and links

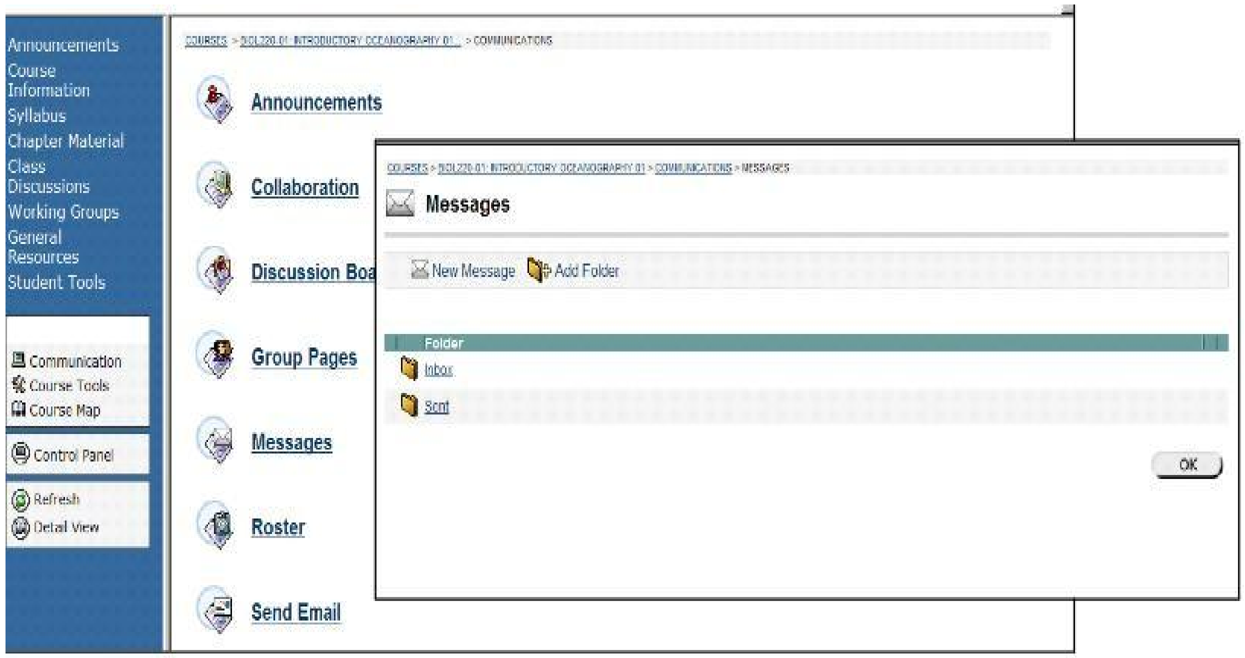
Learning Path
Show Remove

Tests
Show Re

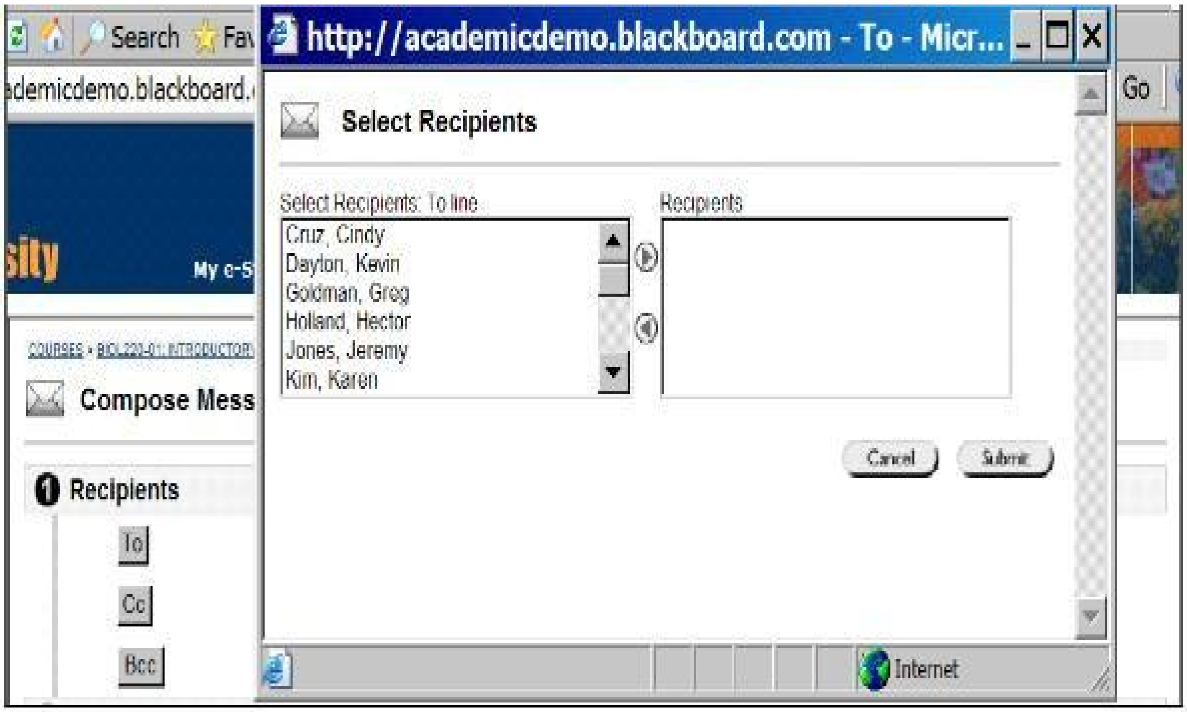
File exchange – enables learners to upload files from their local computers and share these files with instructors or other students



Internal asynchronous messaging – mail that can be sent and read from within an online course



users select recipients, compose messages, attach files to messages and send to other users as shown in the following picture


http://academicdemo.blackboard.com - To - Micr...
Select Recipients
Select Recipients: To line
Recipients
Cruz, Cindy
Dayton, Kevin
Goldman, Greg
Holland, Hector
Jones, Jeremy
Kim, Karen
Cancel
Submit
Compose Mess
Recipients
To
Cc
Bcc
Internet

Experiment 2: Easy Leave

This project is aimed at developing a web based Leave Management Tool, which is of importance to either an organization or a college. The Easy Leave is an Intranet based application that can be accessed throughout the Organization or a specified group/Dept. This system can be used to automate the workflow of leave applications and their approvals. The periodic crediting of leave is also automated. There are features like notifications, cancellation of leave, automatic approval of leave, report generators etc in this Tool.

Functional components of the project:

There are registered people in the system. Some are approvers. An approver can also be a requestor. In an organization, the hierarchy could be Engineers/Managers/Business Managers/Managing Director etc. In a college, it could be Lecturer/Professor/Head of the Department/Dean/Principal etc.

Following is a list of functionalities of the system: A person should be able to

•login to the system through the first page of the application

•change the password after logging into the system

•see his/her eligibility details (like how many days of leave he/she is eligible for etc)

•query the leave balance

•see his/her leave history since the time he/she joined the company/college

•apply for leave, specifying the form and to dates, reason for taking leave, address for communication while on leave and his/her superior's email id

•see his/her current leave applications and the leave applications that are submitted to him/her for approval or cancellation

•approve/reject the leave applications that are submitted to him/her

•withdraw his/her leave application (which has not been approved yet)

•Cancel his/her leave (which has been already approved). This will need to be approved by his/her Superior

•get help about the leave system on how to use the different features of the system

•As soon as a leave application /cancellation request /withdrawal /approval /rejection /password-change is made by the person, an automatic email should be sent to the person and his superior giving details about the action

•The number of days of leave (as per the assumed leave policy) should be automatically credited to everybody and a notification regarding the same be sent to them automatically

•An automatic leave-approval facility for leave applications which are older than 2 weeks should be there. Notification about the automatic leave approval should be sent to the person as well as his superior

Problem Analysis and Project Planning

In the existing Leave Record Management System, every College/Department follows manual procedure in which faculty enters information in a record book. At the end of each month/session, Administration Department calculates leave/s of every member which is a time taking process and there are chances of losing data or errors in the records. This module is a single leave management system that is critical for HR tasks and keeps the record of vital information regarding working hours and leaves. It intelligently adapts to HR policy of the management and allows employees and their line managers to manage leaves and replacements (if required).

In this module, Head of Department (HOD) will have permissions to look after data of every faculty member of their department.HOD can approve leave through this application and can view leave information of every individual. This application can be used in a college to reduce processing work load. This project's main idea is to develop an online centralized application connected to database which will maintain faculty leaves, notices information and their replacements (if needed). Leave management application will reduce paperwork and maintain record in a more efficient & systematic way. This module will also help to calculate the number of leaves taken monthly/annually and help gather data with respect to number of hours' worked, thereby helping in calculating the work hours by the HR Department.

Software Requirement Analysis

In the existing paper work related to leave management, leaves are maintained using the attendance register for staff. The staff needs to submit their leaves manually to their respective authorities. This increases the paperwork & maintaining the records becomes tedious.

Maintaining notices in the records also increases the paperwork. The main objective of the proposed system is to decrease the paperwork and help in easier record maintenance by having a particular centralized Database System, where Leaves and Notices are maintained. The proposed system automates the existing system. It decreases the paperwork and enables easier record maintenance. It also reduces chances of Data loss. This module intelligently adapts to HR policy of the management &allows employees and their line managers to manage leaves and replacements for better scheduling of workload. The application basically contains the given modules:

Module:

1) STAFF MODULE: It consist of two types of faculties

a) Teaching

b) Non-teaching

2) HOD MODULE: It consists of Head of the Department/Manager Body which takes critical decision related to HR.

3) ADMINISTRATION MODULE: It calculates leaves & maintains records.

Objective:

- To automate the existing leave management in educational institutes
- To decrease the paperwork and enable the process with efficient, reliable record maintenance by using centralized database, thereby reducing chances of data loss
- To provide for an automated leave management system that intelligently adapts to HR policy of the organization and allows employees and their line managers to manage leaves and replacements for better scheduling of work load & processes.

Functional Requirements:

•login to the system through the first page of the application

•change the password after logging into the system

•see his/her eligibility details (like how many days of leave he/she is eligible for etc)

•query the leave balance

•see his/her leave history since the time he/she joined the company/college

•apply for leave, specifying the form and to dates, reason for taking leave, and address for communication while on leave and his/her superior's email id

•see his/her current leave applications and the leave applications that are submitted to him/her for approval or cancellation

•approve/reject the leave applications that are submitted to him/her

•withdraw his/her leave application (which has not been approved yet)

•Cancel his/her leave (which has been already approved). This will need to be approved by his/her Superior

•get help about the leave system on how to use the different features of the system

•As soon as a leave application /cancellation request /withdrawal /approval /rejection /password-change is made by the person, an automatic email should be sent to the person and his superior giving details about the action

•The number of days of leave (as per the assumed leave policy) should be automatically credited to everybody and a notification regarding the same be sent to them automatically

•An automatic leave-approval facility for leave applications which are older than 2 weeks should be there. Notification about the automatic leave approval should be sent to the person as well as his superior

Non-Functional Requirements:

Security

a. Firewall Protection: The Easy leave software system shall run inside a firewall.

b. Support different roles: The system shall support different roles for users, such as Lecturer/Professor/Head of the Department/Dean/Principal, the user logged in with given role should only be allowed access consistent with that role.

Scalability

a. Scaling the system to large number of users: As faculties are going to use easy leave server every time to apply leaves.

b. The system should able to operate properly when the web application is accessed by many users at a single time.

Utilization of Resources

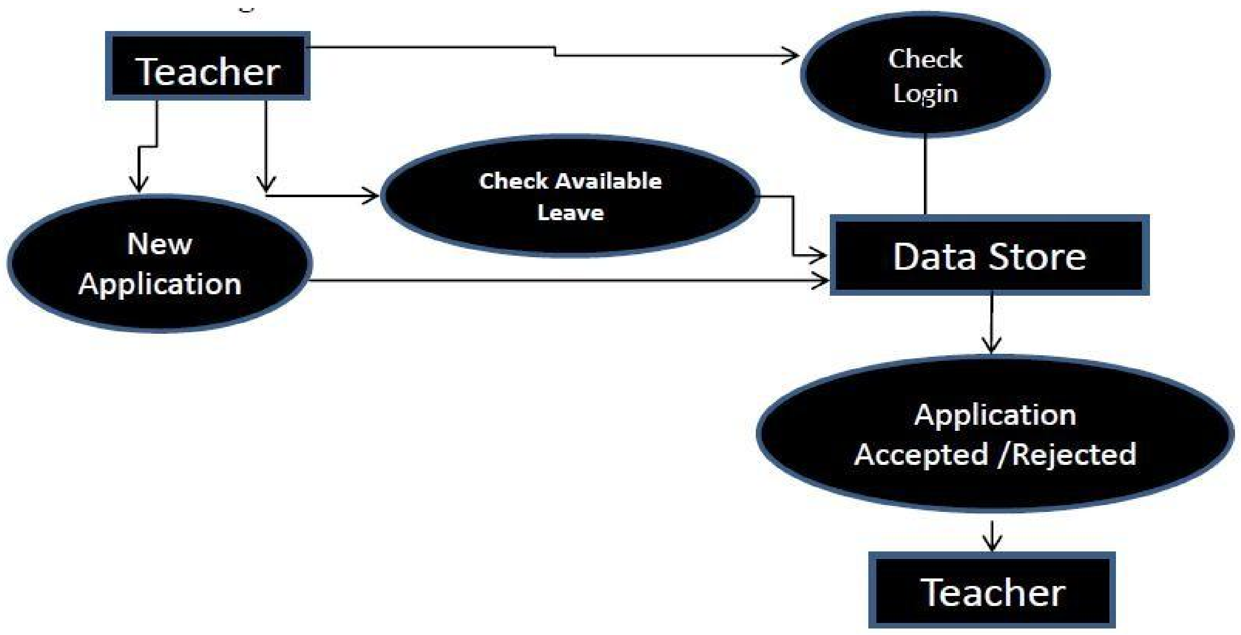
a. The system shall store in the database no more than one million transactions.

b. If the database grows over this limit, old transaction shall be backed up and deleted from the operational database.

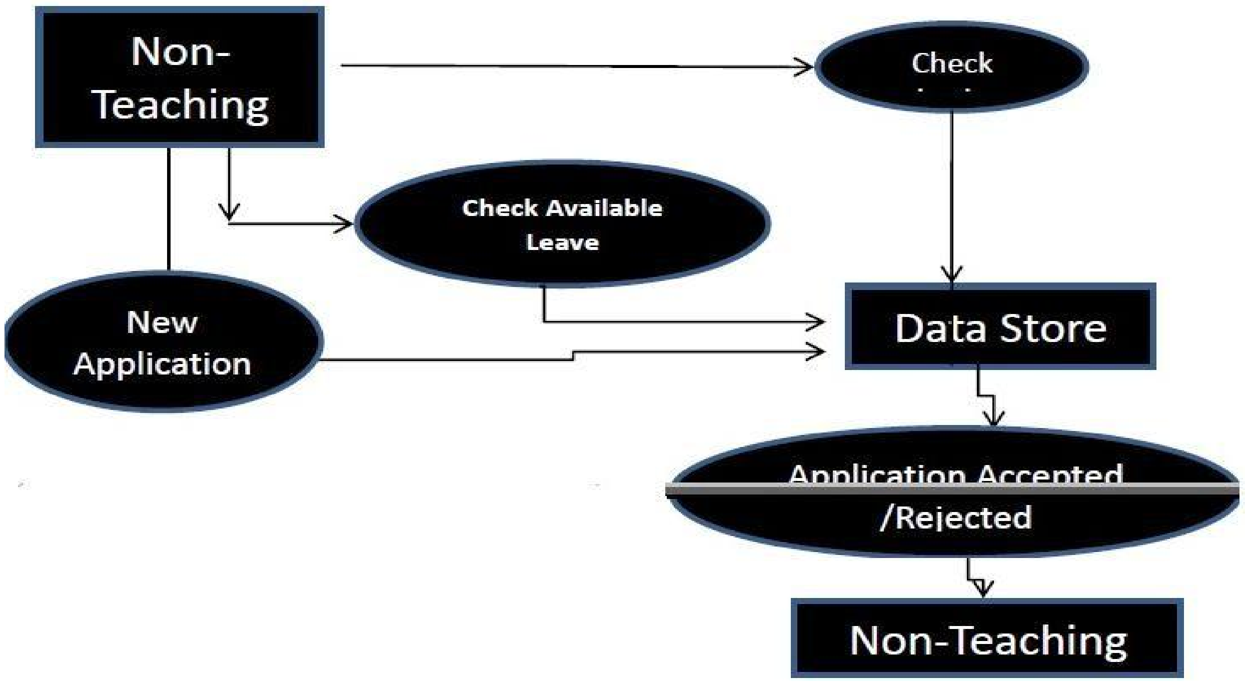
<u>Data Modeling</u>

1. Data Flow Diagram

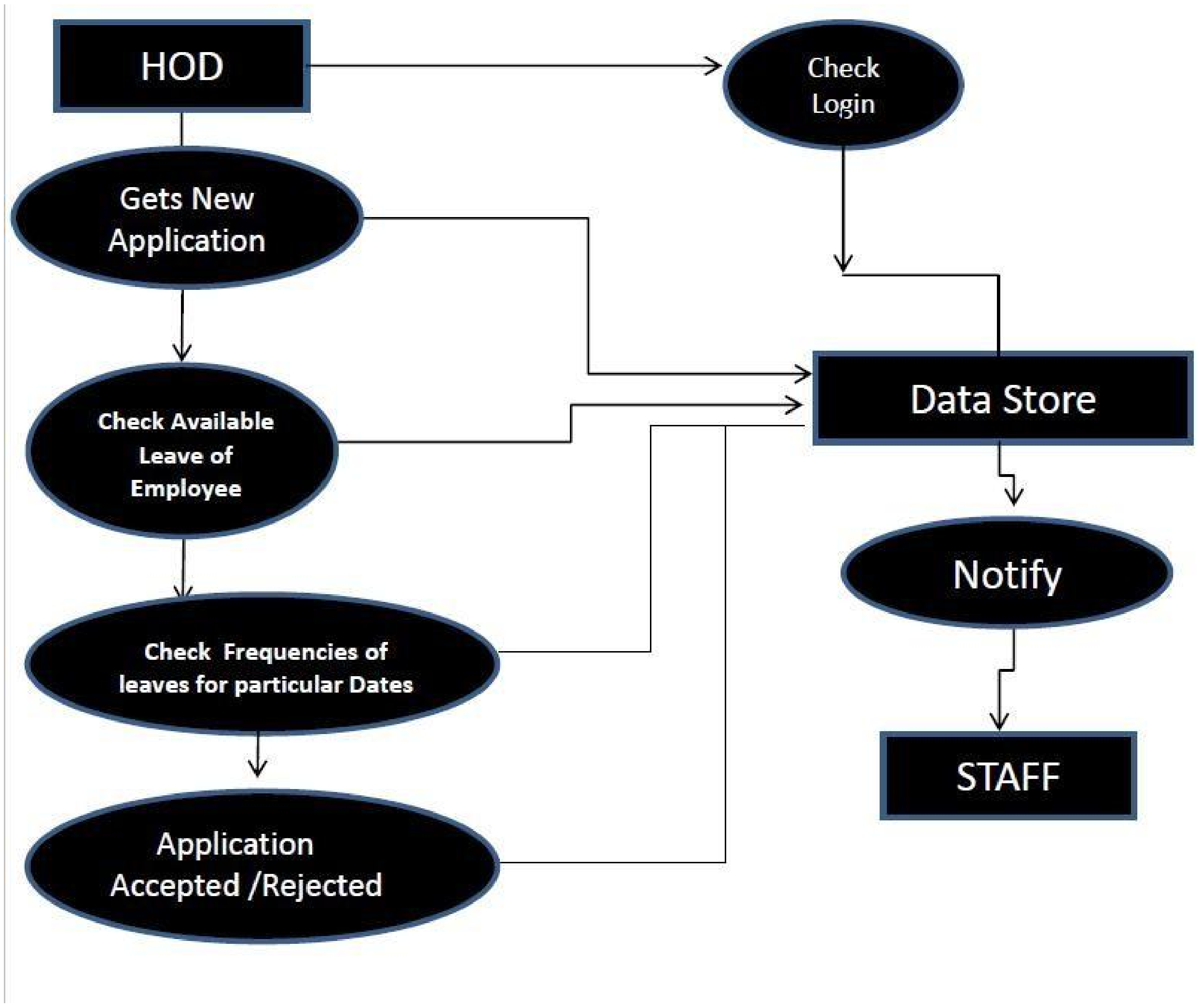
a. DFD for teaching staff



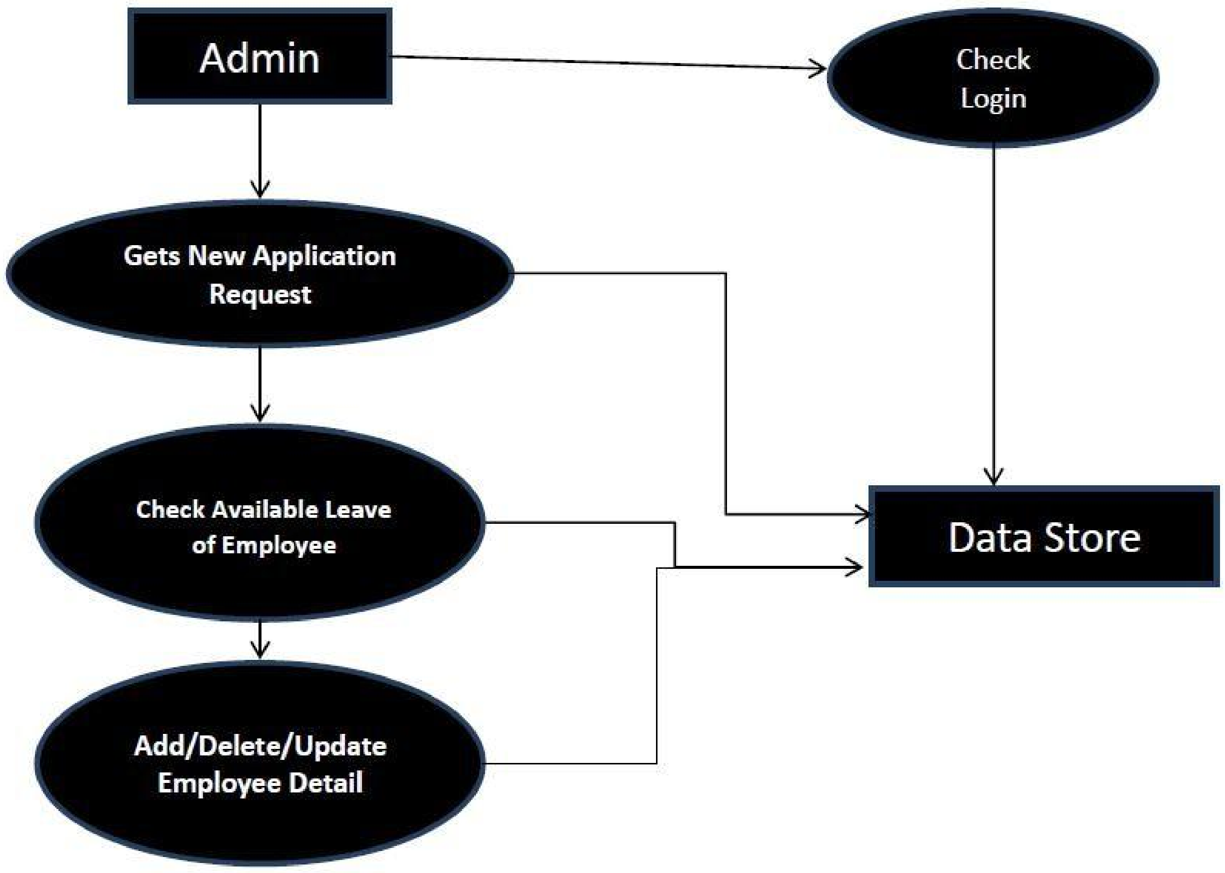
b. DFD for non-teaching staff


Non-Teaching
Check
Check Available Leave
New Application
Data Store
Application Accepted /Rejected
Non-Teaching

c. DFD for HOD


HOD
Check Login
Gets New Application
Check Available Leave of Employee
Check Frequencies of leaves for particular Dates
Application Accepted /Rejected
Data Store
Notify
STAFF

d. DFD for Admin


Admin
Check Login
Gets New Application Request
Check Available Leave of Employee
Add/Delete/Update Employee Detail
Data Store

2. Data Dictionary

2.1 Staff Details

FIELD NAME	TYPE	CONSTRAINTS
staffID	Number	Primary key
Name	Varchar2	
DeptId	Number	Foreign key
Email	Varchar2	
phone	Number	unique
DOJ	Date	

2.2 LeavesDetails

FIELD NAME	TYPE	CONSTRAINTS
Staffid	Number	Foreign key
TotalCL	Number	
usedCL	Number	
BalanceCL	Number	
TotalCCL	Number	
usedCCL	Number	
BalanceCCL	Number	

2.3 LeaveInfo

FIELD NAME	TYPE	CONSTRAINTS
Staffid	Number	Foreign key

NoOfDays	Number	
TypeOfLeave	Varchar2	
FromDate	Date	
ToDate	Date	
HODStatus	char	
PrincipalStatus	char	
AdminStatus	char	

2.4 Adjustments

FIELD NAME	TYPE	CONSTRAINTS
FacultyId	Number	Foreign key
ToId	Number	
Class	Varchar2	
DeptId	Number	Foreign key
Hour	Number	
Status	char	

2.5 DeptCode

FIELD NAME	TYPE	CONSTRAINTS
DeptId	Number	Primary key
DeptName	Varchar2	

2.6 HodDetails

FIELD NAME	TYPE	CONSTRAINTS
StaffId	Number	Foreign key
DeptId	Number	Foreign key

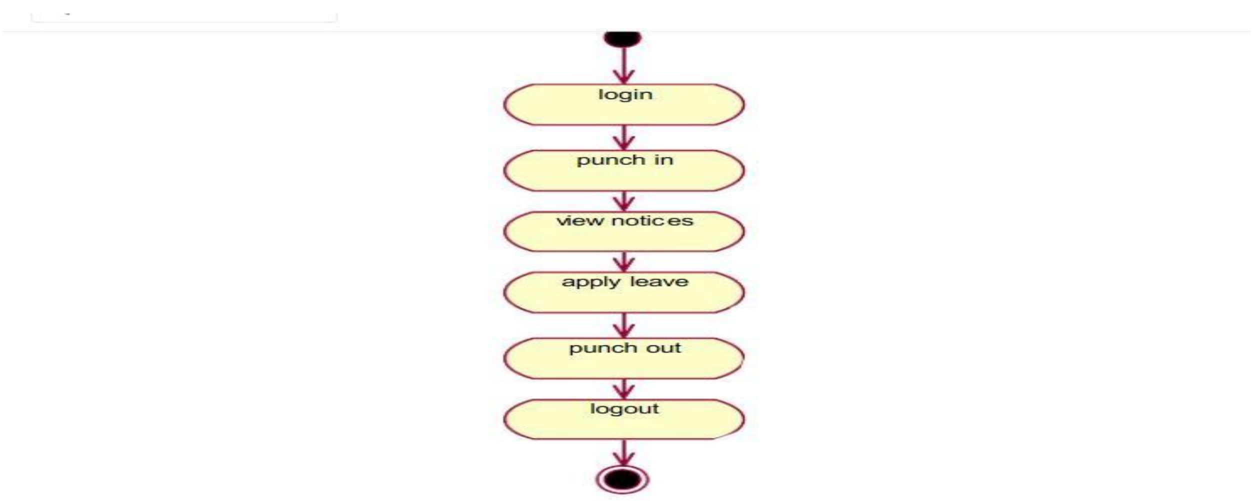
2.7 PrincipalDetails

FIELD NAME	TYPE	CONSTRAINTS
StaffId	Number	Foreign key
DeptId	Number	Foreign key

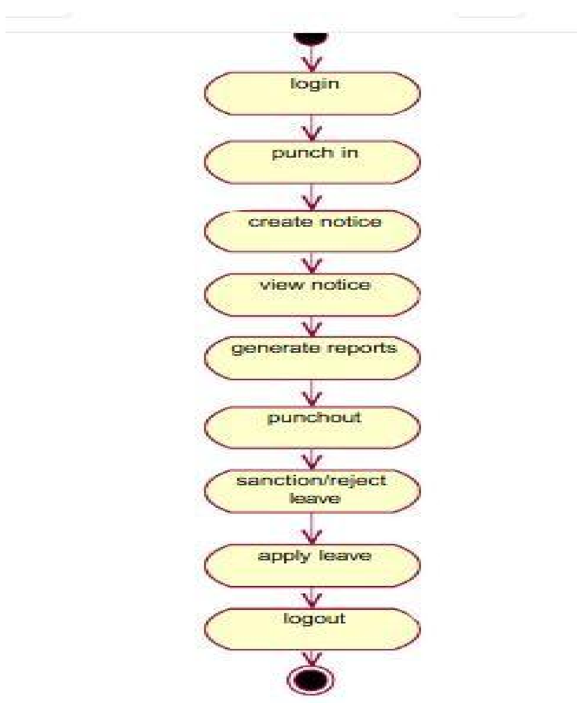
SOFTWARE DESIGNING

UML DIAGRAMS

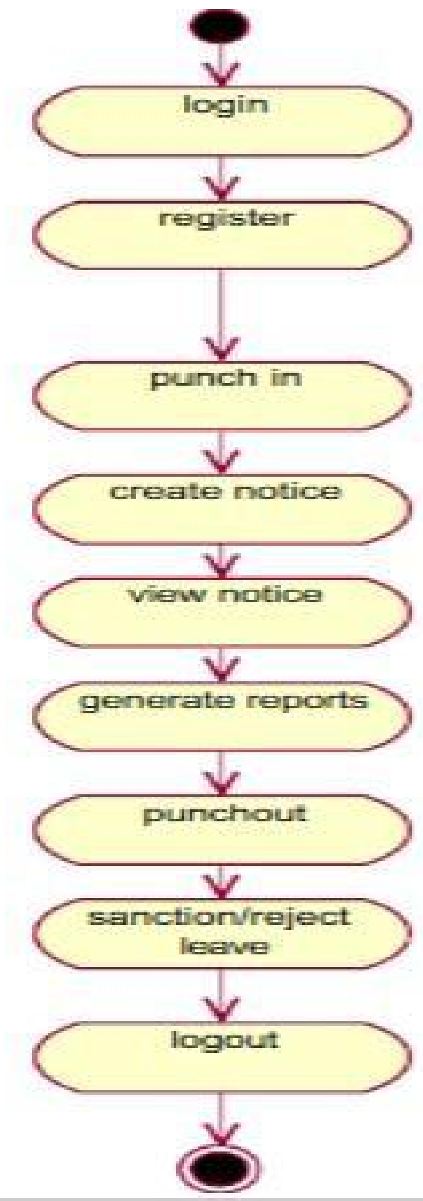
Activity diagram for employee/staff:



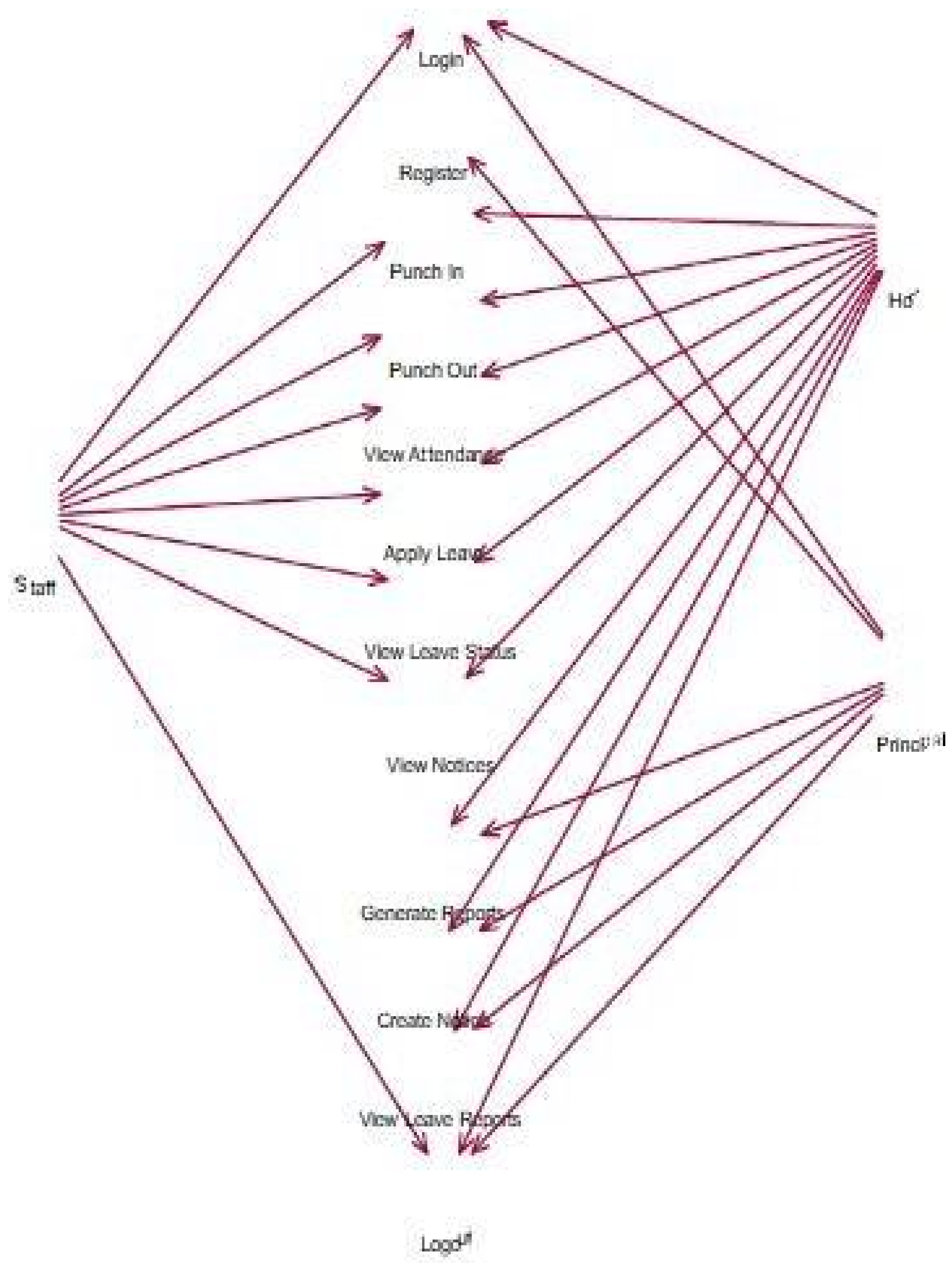
Activity diagram for hod:



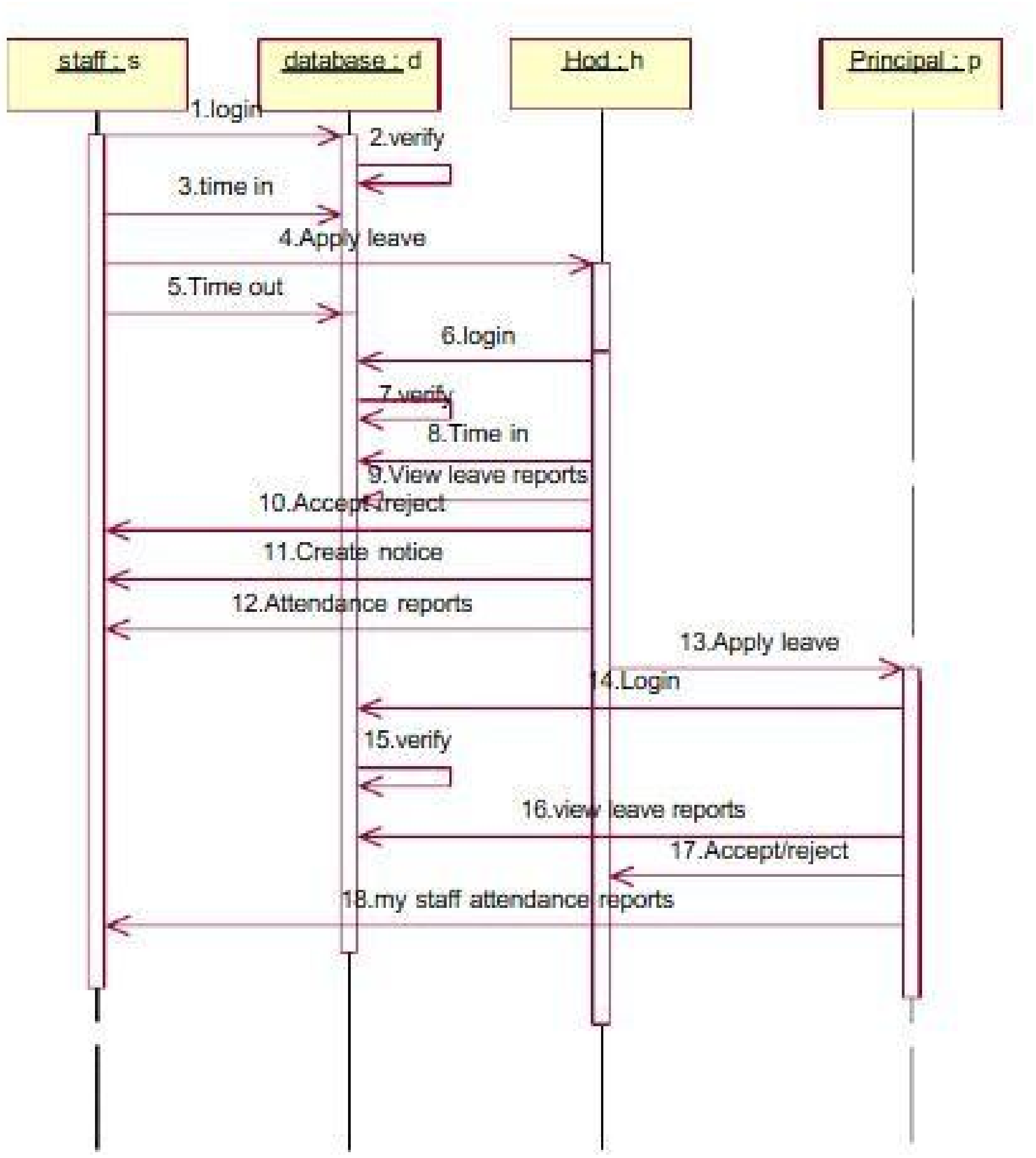
Activity diagram for accountant:



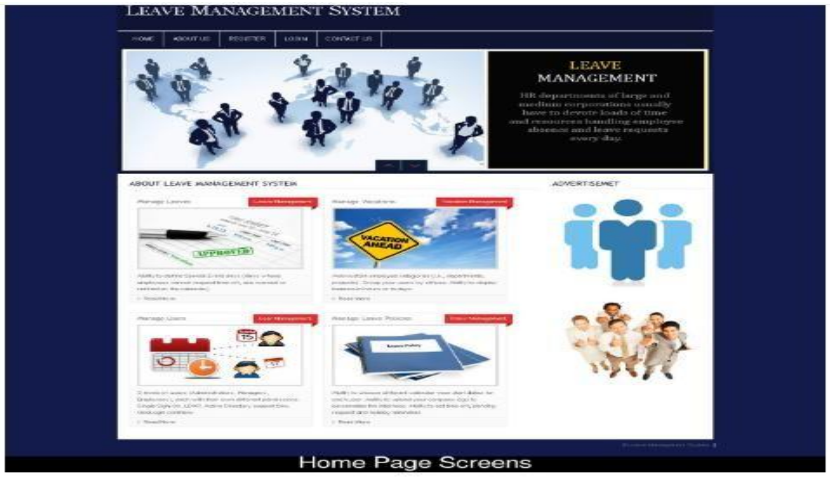
Usecase diagrams:



Sequence diagram:

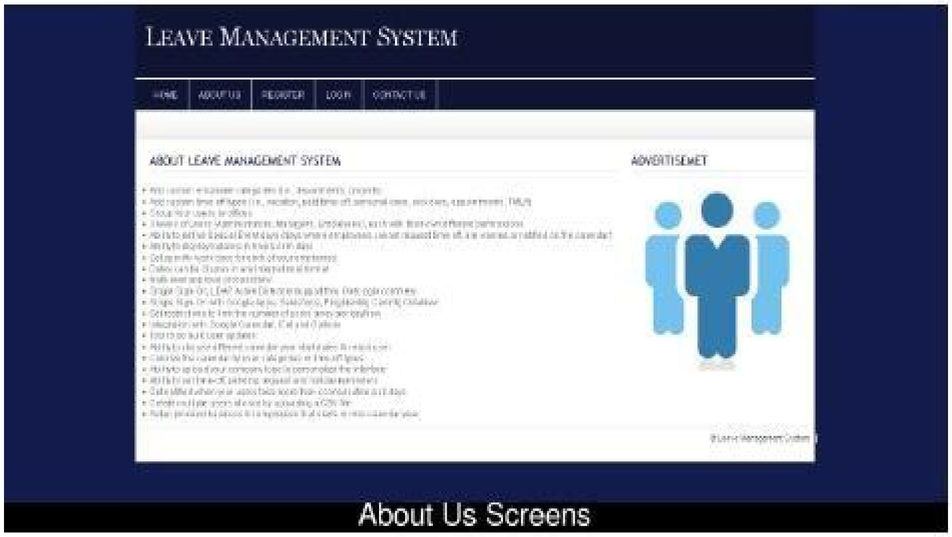


Prototype :



HR departments of large and medium corporations usually have to devote loads of time and resources handling employee absence and leave requests every day.

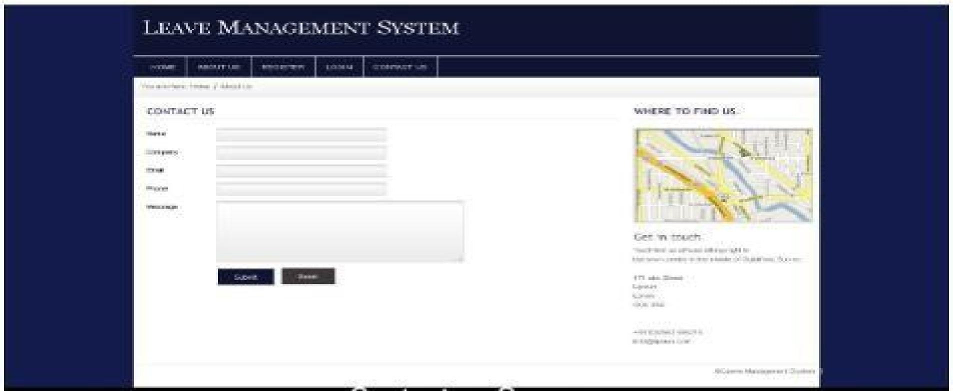
Home Page Screens



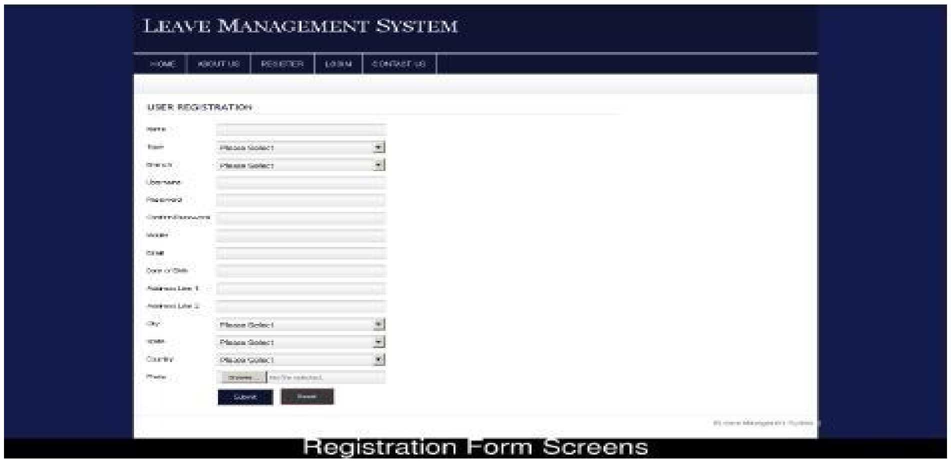
ABOUT LEAVE MANAGEMENT SYSTEM

- Add custom employee categories (i.e., departments, projects)
- Add custom time off types (i.e., vacation, paid time off, personal care, sick days, appointments, TMLR)
- Group your users by offices
- 3 levels of users: Administrators, Managers, Employees, each with their own different permissions
- Ability to add special Events (days where employees cannot request time off, i.e. [illegible]) to the calendar
- Ability to display vacations in hours or in days
- [illegible]
- [illegible]
- [illegible]
- Single Sign On: LDAP Active Directory support [illegible]
- Single Sign On with Google Apps, Salesforce, [illegible]
- [illegible]
- Integration with Google Calendar, iCal and Outlook
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]
- [illegible]

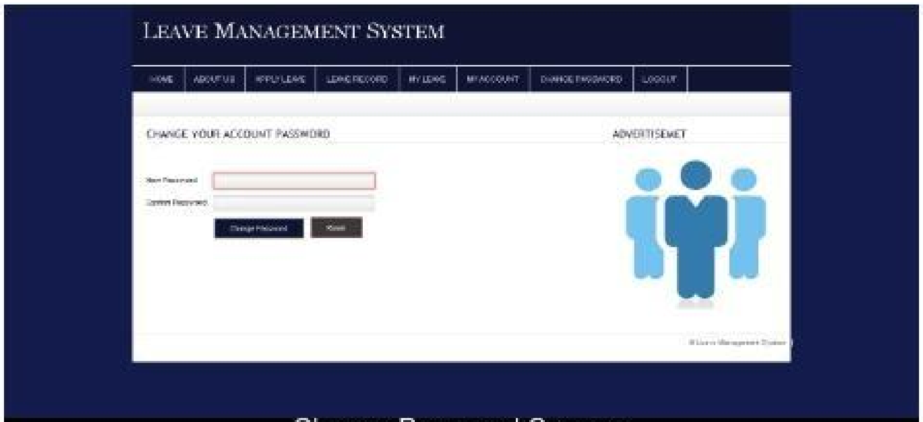
About Us Screens



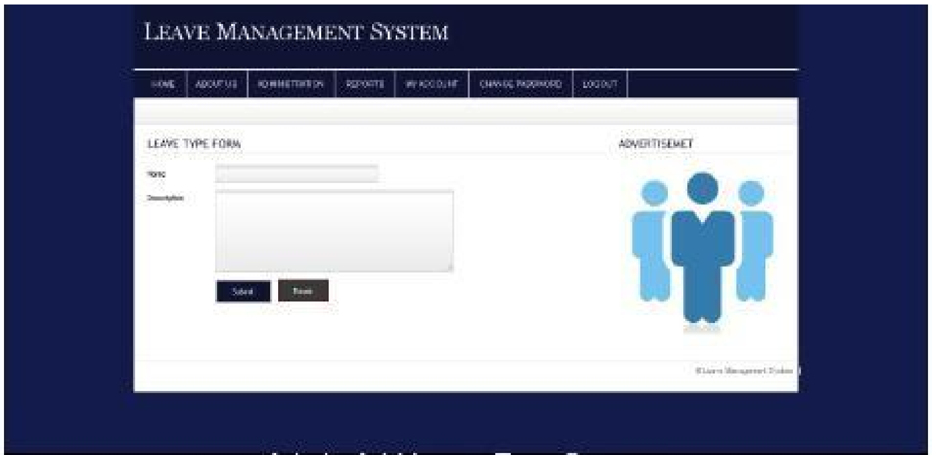
Contact us Screens



Registration Form Screens



Change Password Screens

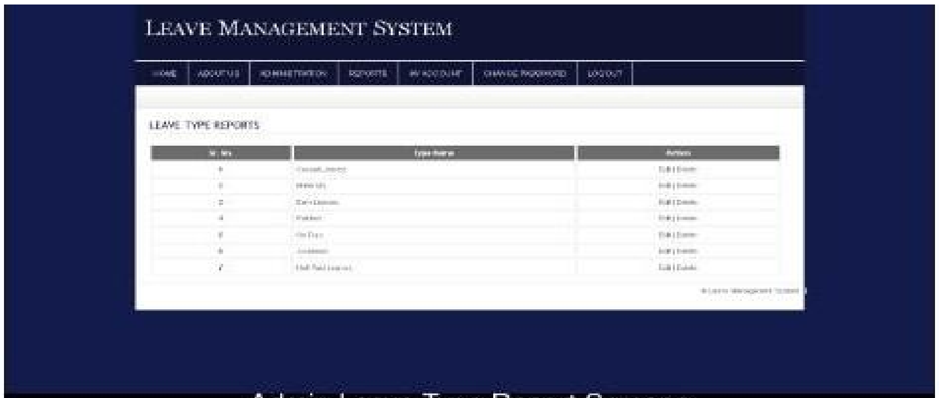


LEAVE TYPE FORM

Name	
Description	

© Leave Management System

Admin Add Leave Type Screens

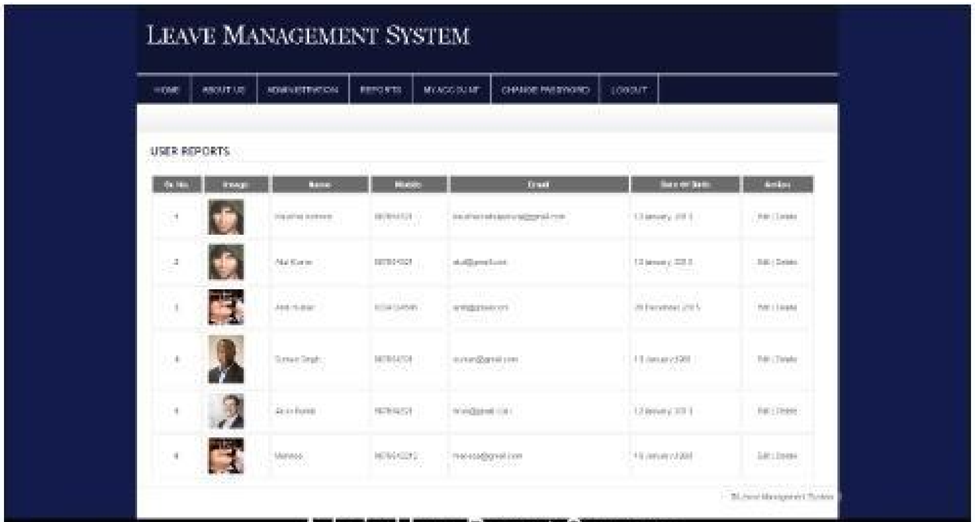


LEAVE TYPE REPORTS

Sr. No	Type Name	Action
1	Casual Leave	Edit | Delete
2	[illegible]	Edit | Delete
3	Earn Leaves	Edit | Delete
4	[illegible]	Edit | Delete
5	On Duty	Edit | Delete
6	[illegible]	Edit | Delete
7	Half Paid Leave	Edit | Delete

© Leave Management System

Admin Leave Type Report Screens

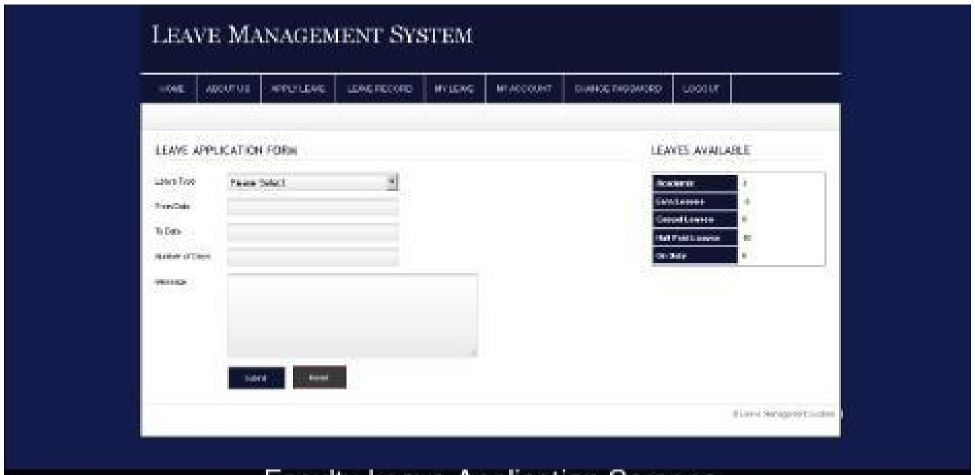


USER REPORTS

Sr. No	Image	Name	Mobile	Email	Date of Birth	Action
1		[illegible]	[illegible]	[illegible]@gmail.com	[illegible] January [illegible]	Edit | Delete
2		Atul Kumar	[illegible]	atul@gmail.com	[illegible] January [illegible]	Edit | Delete
3		[illegible]	[illegible]	[illegible]	[illegible] December [illegible]	Edit | Delete
4		Suman Singh	[illegible]	suman@gmail.com	[illegible] January [illegible]	Edit | Delete
5		[illegible]	[illegible]	[illegible]@gmail.com	[illegible] January [illegible]	Edit | Delete
6		[illegible]	[illegible]	[illegible]@gmail.com	[illegible] January [illegible]	Edit | Delete

© Leave Management System

Admin User Report Screens


LEAVE MANAGEMENT SYSTEM

LEAVE APPLICATION FORM

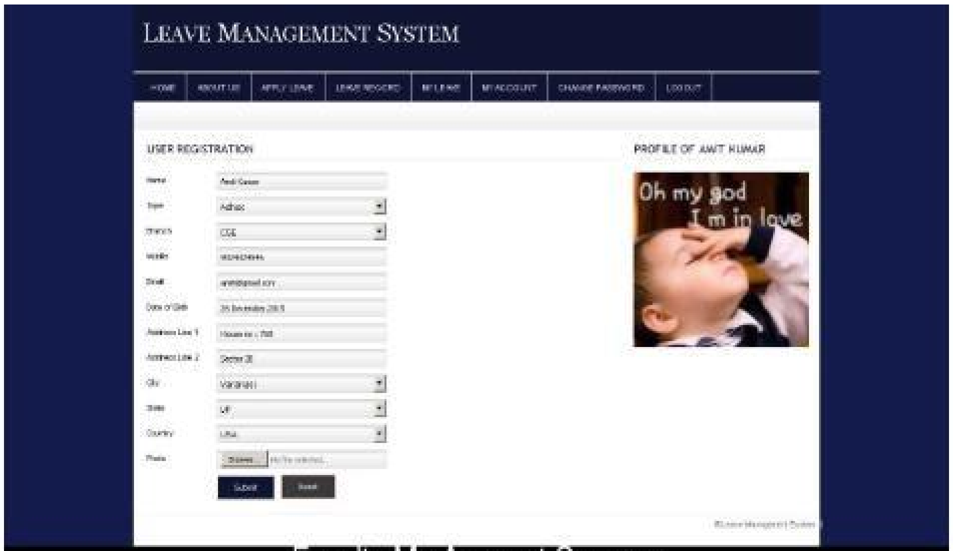
Leave Type	Please Select
From Date	
To Date	
Number of Days	
Message	

LEAVES AVAILABLE

[illegible]	[illegible]
Earn Leaves	[illegible]
Casual Leaves	[illegible]
Half Paid Leaves	[illegible]
On Duty	[illegible]

© Leave Management System

Faculty Leave Application Screens


LEAVE MANAGEMENT SYSTEM

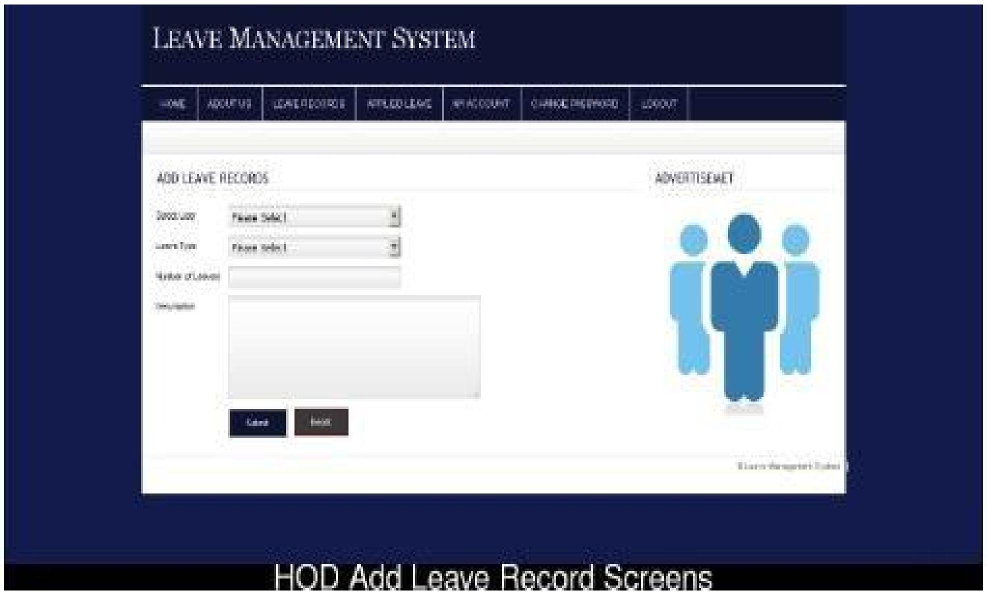
USER REGISTRATION

Name	[illegible]
Type	Adhoc
Branch	CSE
Mobile	[illegible]
Email	[illegible]
Date of Birth	[illegible]
Address Line 1	[illegible]
Address Line 2	Sector [illegible]
City	Varanasi
State	UP
Country	[illegible]
Photo	No file selected.

PROFILE OF AMIT KUMAR

© Leave Management System

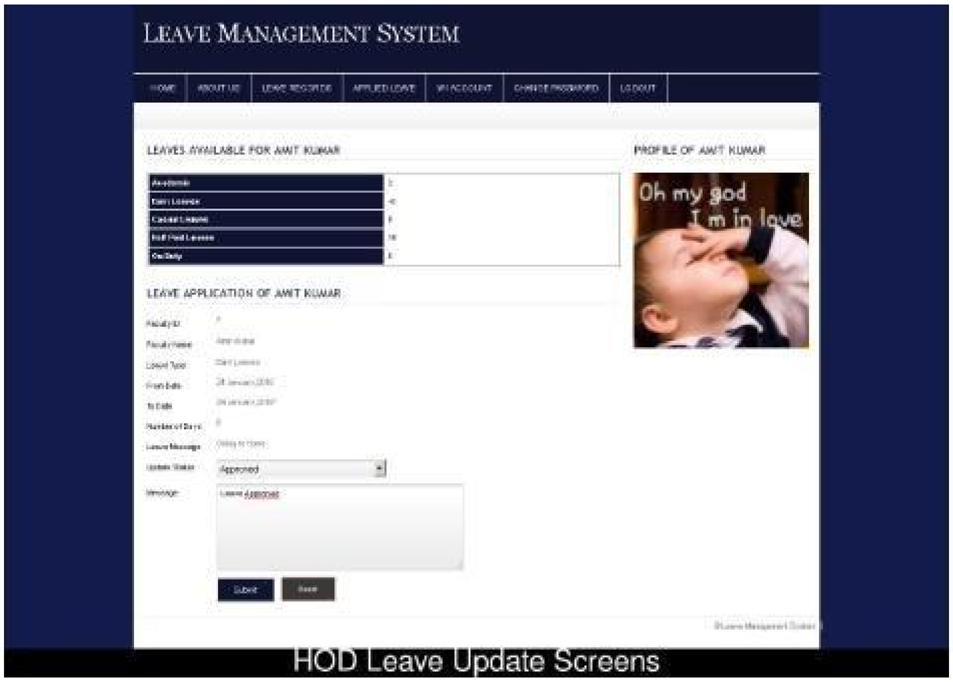
Faculty My Account Screens



ADD LEAVE RECORDS

Select User	Please Select
Leave Type	Please Select
Number of Leaves	
Description	

HOD Add Leave Record Screens



LEAVES AVAILABLE FOR AMIT KUMAR

Academic	[illegible]
Earn Leaves	[illegible]
Casual Leaves	[illegible]
Half Paid Leaves	[illegible]
On Duty	[illegible]

LEAVE APPLICATION OF AMIT KUMAR

Faculty ID	[illegible]
Faculty Name	Amit Kumar
Leave Type	Earn Leaves
From Date	[illegible] January [illegible]
To Date	[illegible] January [illegible]
Number of Days	[illegible]
Leave Message	[illegible]
Update Status	Approved
Message	Leave Approved

HOD Leave Update Screens

Experiment 3: E-Bidding

Auctions are among the latest economic institutions in place. They have been used since antiquity to sell a wide variety of goods, and their basic form has remained unchanged. In this dissertation, we explore the efficiency of common auctions when values are interdependent-the value to a particular bidder may depend on information available only to others-and asymmetric. In this setting, it is well known that sealed-bid auctions do not achieve efficient allocations in general since they do not allow the information held by different bidders to be shared.

Typically, in an auction, say of the kind used to sell art, the auctioneer sets a relatively low initial price. This price is then increased until only one bidder is willing to buy the object, and the exact manner in which this is done varies. In my model a bidder who drops out at some price can "reenter" at a higher price.

With the invention of E-commerce technologies over the Internet the opportunity to bid from the comfort of one’s own home has seen a change like never seen before. Within the span of a few short years, what may have began as an experimental idea has grown to an immensely popular hobby, and in some cases, a means of livelihood, the Auction Patrol gathers tremendous response every day, all day. With the point and click of the mouse, one may bid on an item they may need or just want, and in moments they find that either they are the top bidder or someone else wants it more, and you're outbid! The excitement of an auction all from the comfort of home is a completely different experience. Society cannot seem to escape the criminal element in the physical world, and so it is the same with Auction Patrols. This is one area where in a question can be raised as to how safe Auction Patrols.

Proposed system

To generate the quick reports

To make accuracy and efficient calculations

To provide proper information briefly

To provide data security

To provide huge maintenance of records

Flexibility of transactions can be completed in time

Problem Analysis and Project Planning

An **Auction** is Latin work which means augment. Auction is a bid, a process of selling; buying and services offered take place. There are several different types of auctions and certain rules exist for each auction. There are variations for an auction which may include minimum price limit, maximum price limit and time limitations etc. Depending upon the auction method bidder can participate remotely or in person. Remote auction include participating through telephone, mail, and internet. Shopping online has widely grown; online auction system is increasing rapidly. Online auction is becoming more and more popular in electronic commerce and hence it should system must increase its quality and security.

The online auction system is a model where we participate in a bid for products and service. This auction is made easier by using online software which can regulate processes involved. There are several different auction methods or types and one of the most popular methods is English auction system. This system has been designed to be highly-scalable and capable of supporting large numbers of bidders in an active auction. Online Auctioning System has several other names such as e-Auctions, electronic auction etc. The requirement for online auction or online bidding can be more accurately specified by the client. It should be healthy and will be a good practice when it is made more transparent as a matter of fact.

Online Bidding has become more wide spread in all sorts of industrial usage. It not only includes the product or goods to be sold, it also has services which can be provided. Due to their low cost this expansion made the system to grow. Online bidding has become a standard method for procurement process. Bidders can be maintained in a single database according to the preference, and they can be monitored. User’s data can be maintained in a confidential way for validity and integrity of contractual documentation. Neat reporting reduces paperwork, postage, photocopying and time beneficial. Multiple bidders can be communicated with a great ease. This system allows multiple bids by single users. Online bidding is based upon lowest or the highest price which is initiated but not the best value for the product. Although there is a chance to fix the criteria against the fact expected to have desired value by the seller.

1.1 OVERVIEW

The Objective is to develop a user-friendly auctioning site where any kind of product can be auctioned and provide value-added services to the bidders and the sellers. The products will be authenticated and the site provides a safe environment for online users:

Secure registration of all users including a personal profile Administrators would authorize the product to auction, set auction dates and Minimum auction amount for that product.

Prior to each bid, the user's bank or credit account must be authenticated for available balance required for the bid.

Complete Search/Site Map of the entire site for easy access.

Discussion forums for users to interact with other users to know about the product's value and originality.

Online Legal Documentation to avoid disputes. Guidance to the users about the same must be available.

Rare articles may be withheld by owner on the advice of the administrator to be
thrown open in special auctions held by the site so as to increase the bid-values.

Software Requirement Analysis

Modules:

1. **Login:**

Login Module includes various utilities like User Registration, Authentication, Change Password and Forgot Password.

2. **Category Management:**

This module provides all facilities to admin for managing the Category.

3. **Package Management:**

This module provides all facilities to admin for managing the Package.

4. **Search:**

Search Module Provides Category wise Search of items.

5. **Auction:**

In This Module Seller can Upload their Products for Auction, Bidders can bid for the Products finally Admin decides the Winner based on Highest Bidding Price.

6. **Report:**

Report Generation Module can generate reports of past Auctions, Sellers and Bidders.

Users:

1. Admin
2. Seller
3. Bidder

1. Admin

¬ Admin can manage user and product.

¬ Admin can manage category.

¬ Admin can send the update to the seller and bidder.

¬ Admin can manage biding.

¬ Admin can manage package.

¬ Admin can generate the whole system work report.

2. Seller

¬ Seller can upload auction product.

¬ Seller can set the starting prize of the item.

¬ Seller can view the bid information for there items.

¬ Seller can bid for product.

3. Bidder

¬ Bidder can also search the items.

¬ Bidder can buy package for auction.

¬ Bidder can view detail of product.

¬ Bidder can bid on particular product.

¬ Bidder can also modify the bidding prize.

Functional Requirements:

Each user type admin or user needs to register him or her as a user or an admin for accessing the user's necessary information. They also have email, username and password. They can login into the system from the web using their email and password.

Admin needs to login to the system to operate the system. Admin has an individual or unique login email, password and a user level. Through this email and password admin can login into the system.

Admin can update all product pages. An admin can insert a new product with details and can update the product information through edit option.

Admin can delete user from user panel. It can have the full access of user's bid list.

Admin can have access in the bid page.

Users can look for a product from a selected category.

User can add a product to the site with full details of that product.

They can see their products and bided list through their account page.

Users can edit their profiles.

Non-Functional Requirements:

1) Performance Requirements

1.1 Performance

The system must be interactive and the delays involved must be less .So in every action-response of the system, there are no immediate delays. In case of opening windows forms, of popping error messages and saving the settings or sessions there is delay much below 2 seconds, In case of opening databases, sorting questions and evaluation there are no delays and the operation is performed in less than 2 seconds for opening ,sorting, computing, posting > 95% of the files. Also when connecting to the server the delay is based editing on the distance of the 2 systems and the configuration between them so there is high probability that there will be or not a successful connection in less than 20 seconds for sake of good communication.

1.2 Safety

Information transmission should be securely transmitted to server without any changes in information

1.3 Reliability

As the system provides the right tools for discussion, problem solving it must be made sure that the system is reliable in its operations and for securing the sensitive details.

2) Software Quality Attributes

2.1 Availability

If the internet service gets disrupted while sending information to the server, the information can be sending again for verification.

2.2 Security

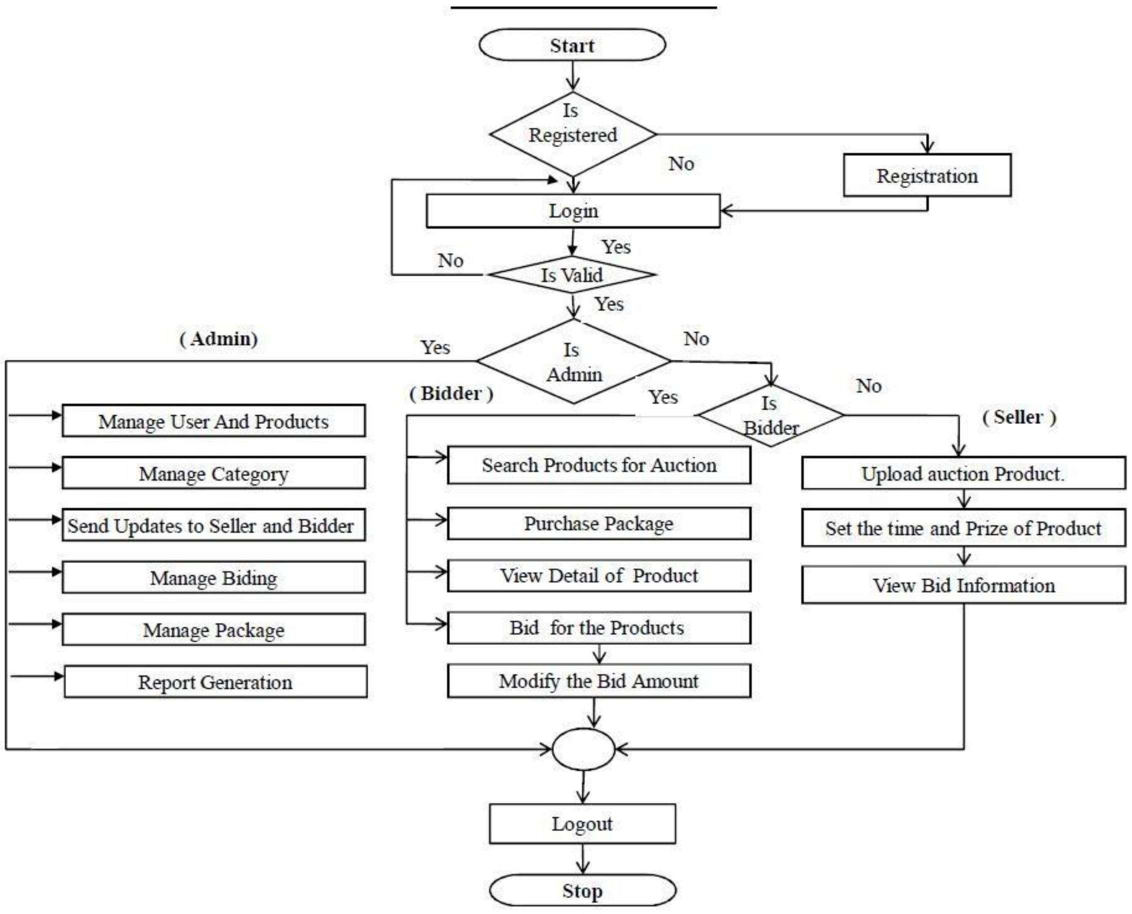
The main security concern is for users account hence proper login mechanism should be used to avoid hacking. The tablet id registration is way to spam check for increasing the security. Hence, security is provided from unwanted use of recognition software.

2.3 Usability

As the system is easy to handle and navigates in the most expected way with no delays. In that case the system program reacts accordingly and transverses quickly between its states.

Data Modeling

(1) Data Flow Diagram



(2) Data Dictionary

(2.1) UserInformation

Field Name	Type	Constraint
User_id	Int	Primary key
User_name	Varchar	Unique
First_name	Varchar	
Last_name	Varchar	
Gender	Varchar	
Email	Varchar	unique
Mobile	Varchar	
password	Varchar	
level	int	

(2.2) Product Information

Field Name	Type	Constraint
P_id	Int	Primary key
User_id	Int	Foreign key
User_name	Varchar	

Title	Varchar	
Category	Varchar	
Brand	Varchar	
Description	Text	
Inti_price	Float	
Time	Date	
Image	Text	
status	varchar	

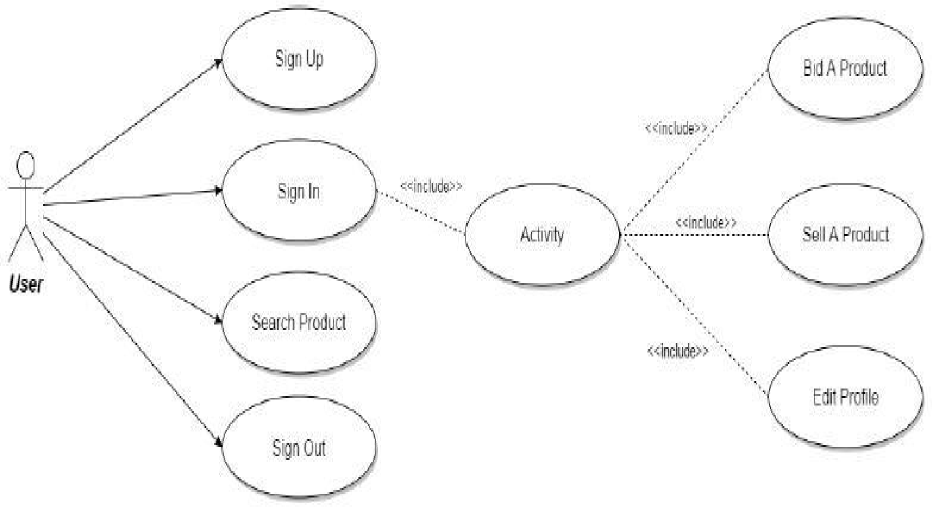
(2.3) BIddingInformation

Field Name	Type	constraint
Bid_id	Int	Primary key
User_id	Int	Foreign key
Bid_init	Float	
Bid_price	Float	
P_id	int	Foreign key

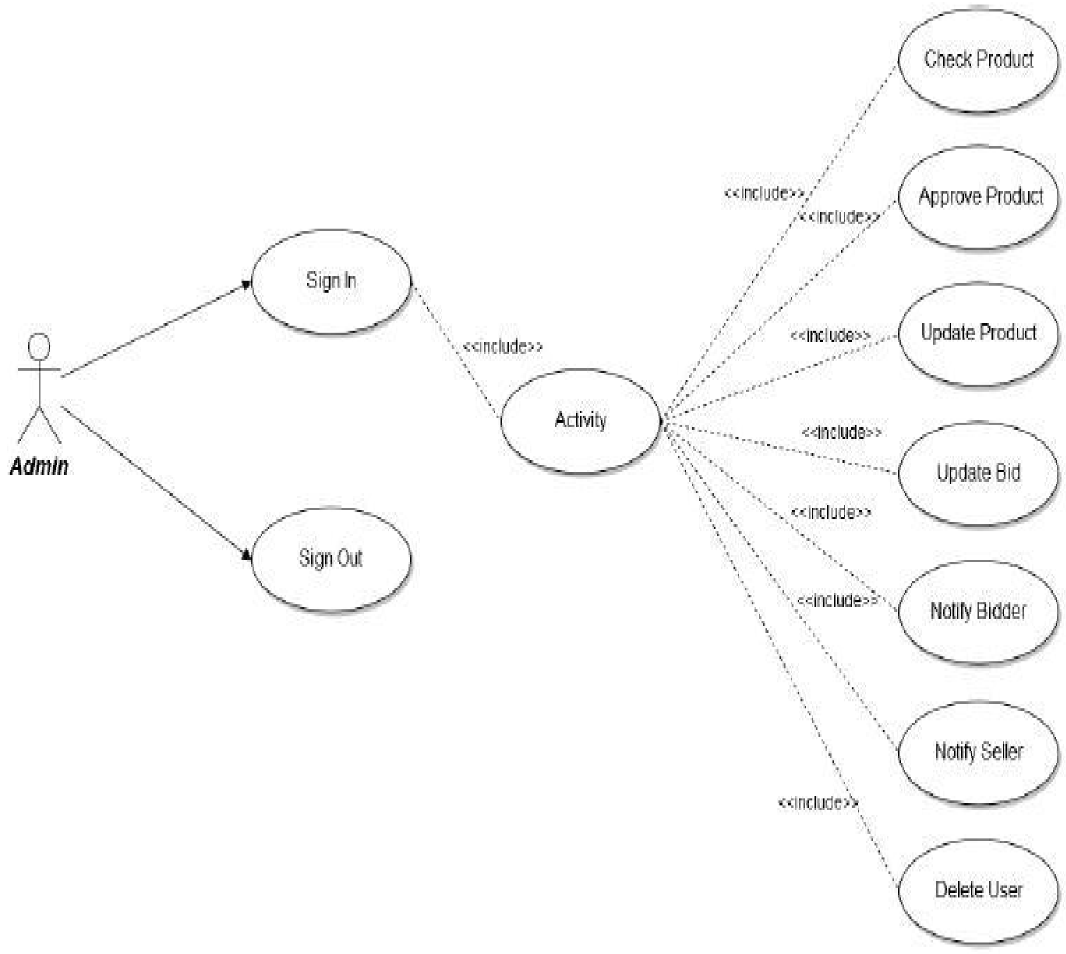
Software Designing

(1) Use case Diagram

Use Case Diagram for User panel

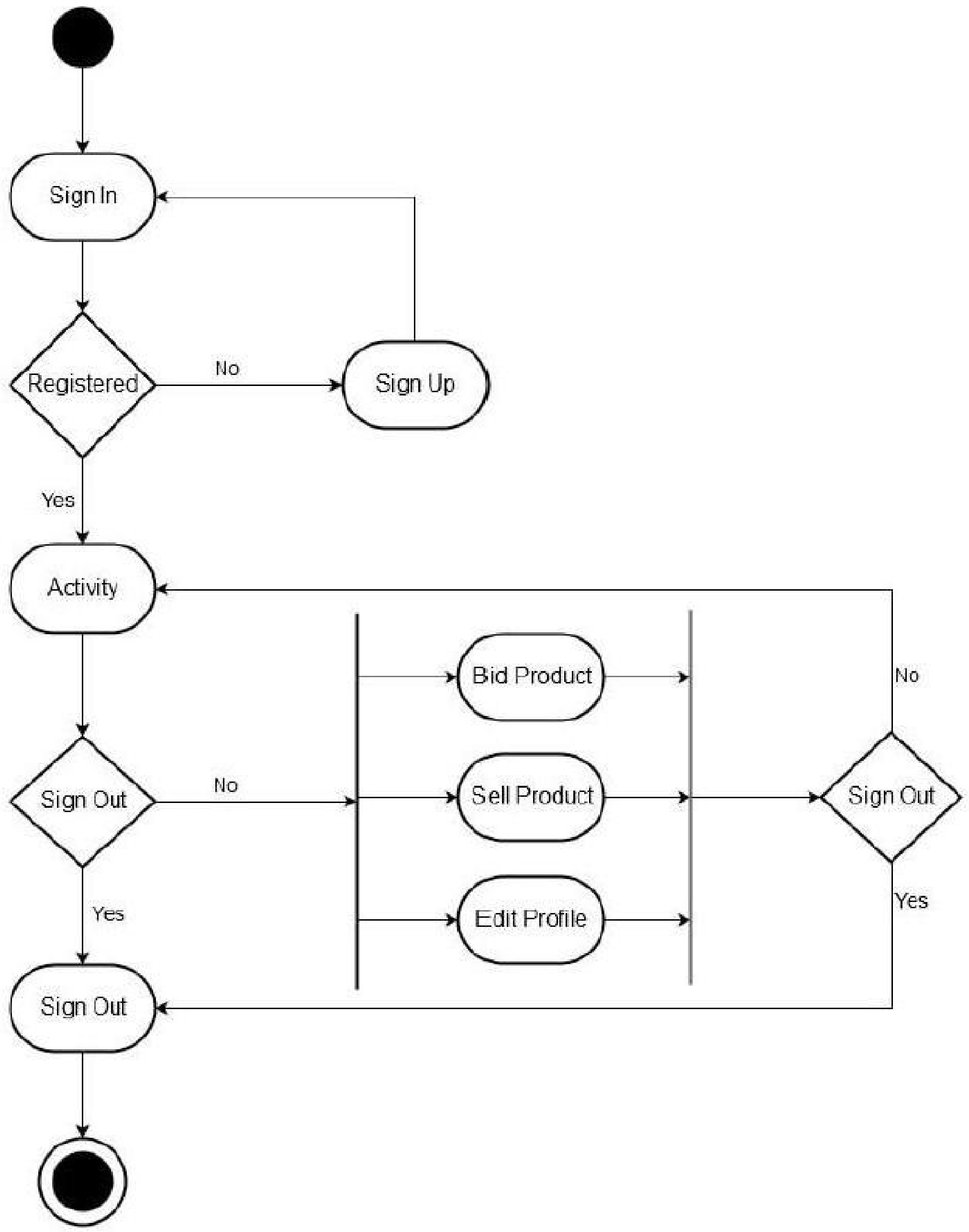


Use Case Diagram for Administrative panel

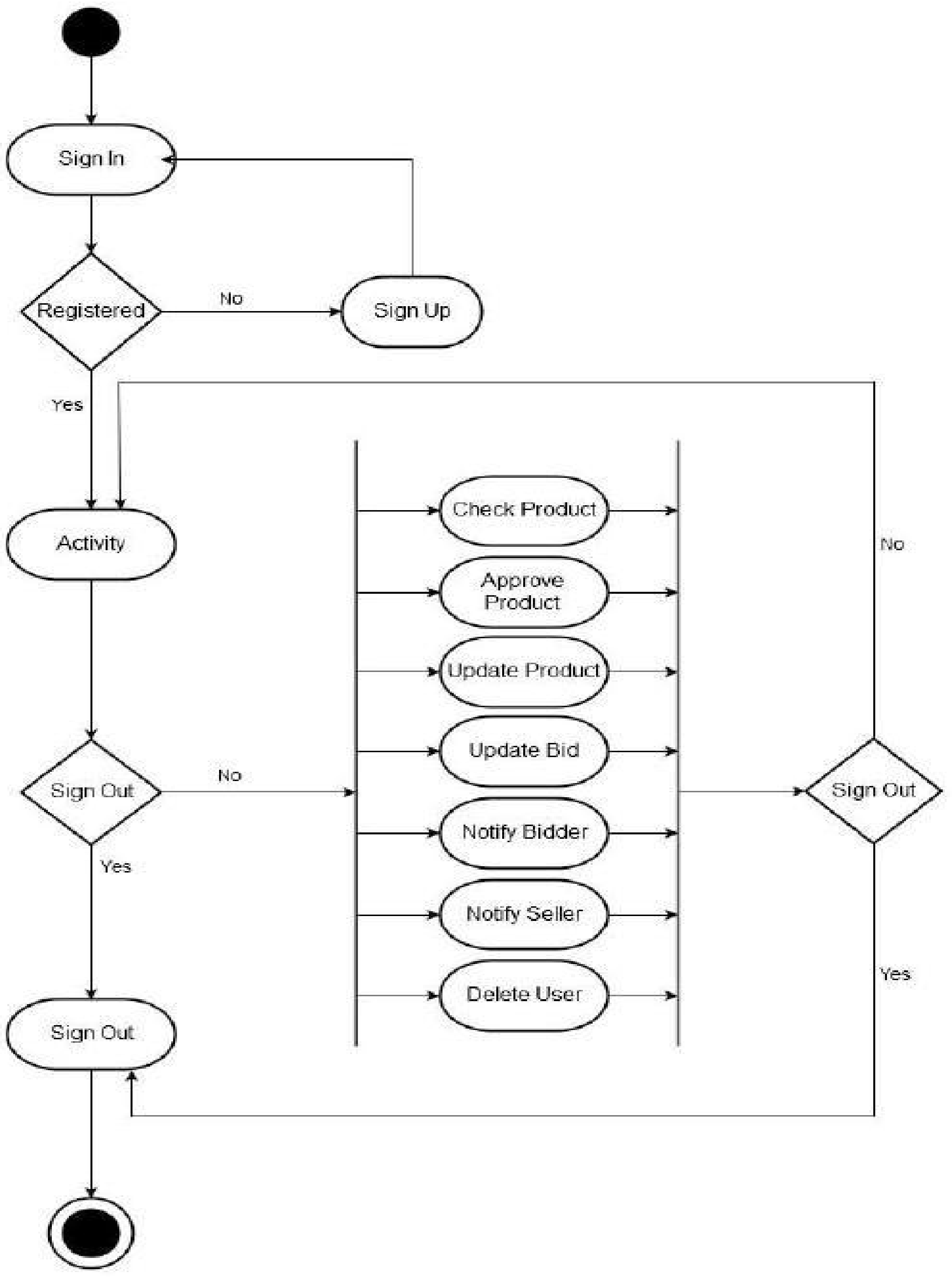

Admin
Sign In
Sign Out
<<include>>
Activity
Check Product
Approve Product
Update Product
Update Bid
Notify Bidder
Notify Seller
Delete User

2) Activity Diagram

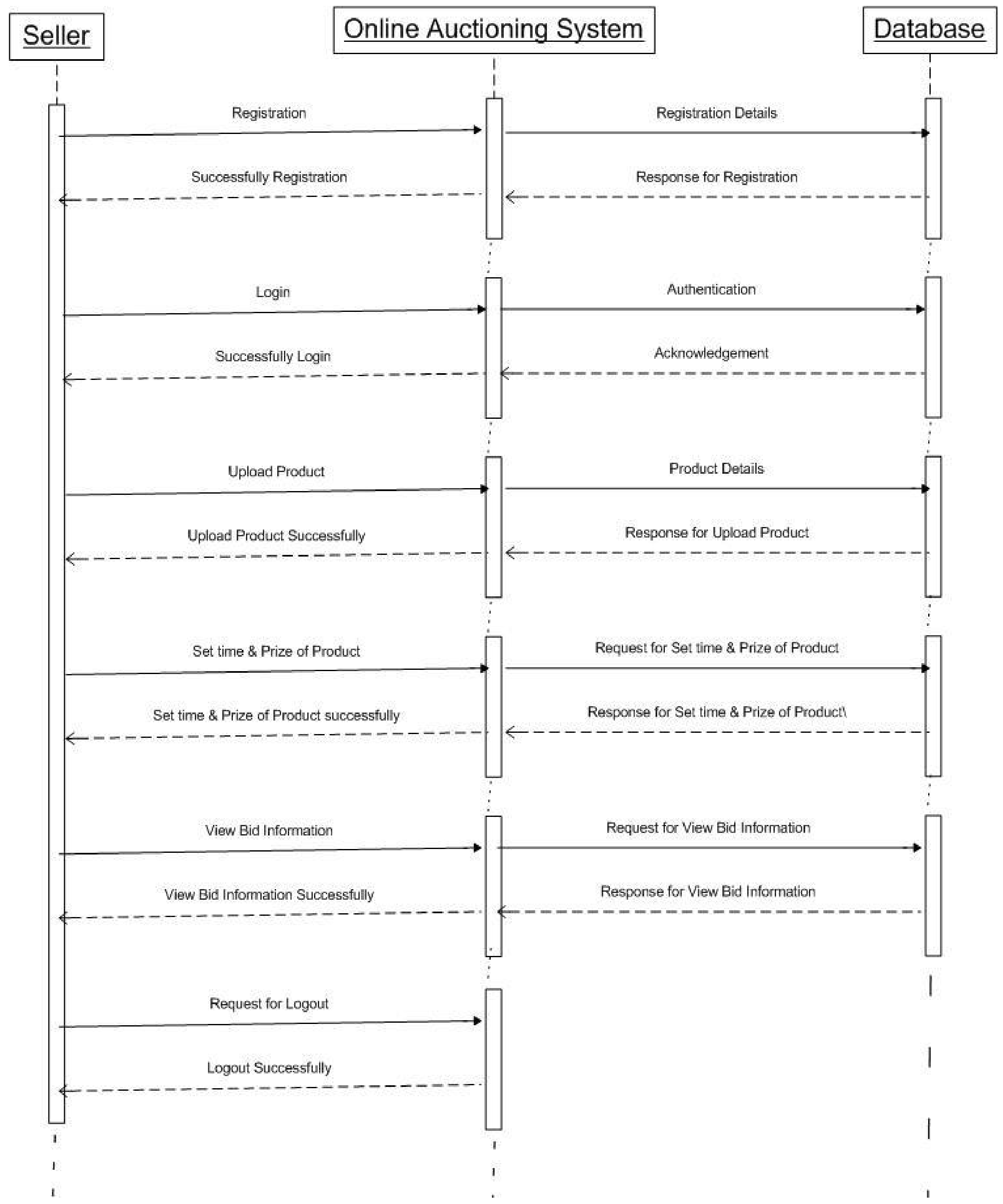
Activity Diagram for User panel



Activity Diagram for Admin panel



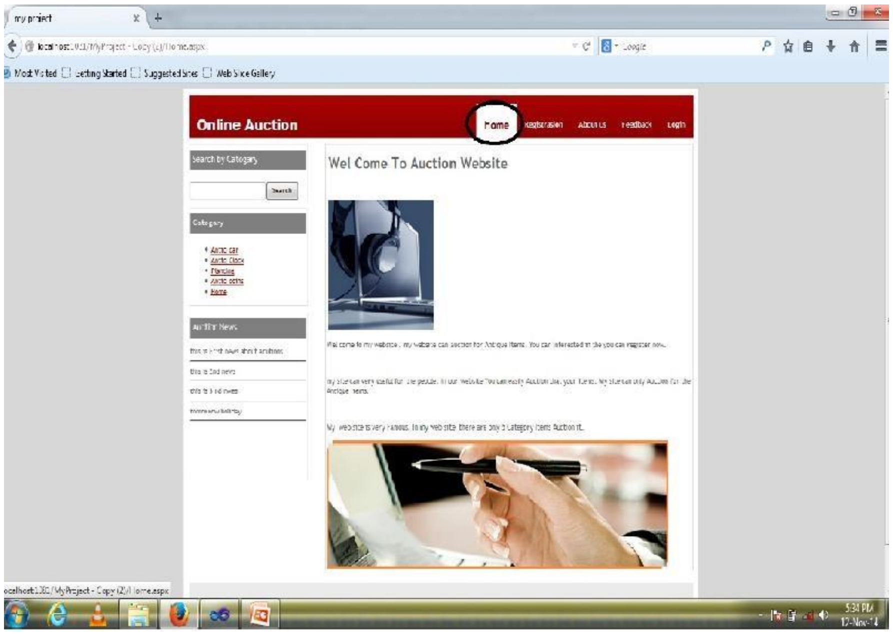
2)Sequence Diagram


Seller
Online Auctioning System
Database
Registration
Registration Details
Successfully Registration
Response for Registration
Login
Authentication
Successfully Login
Acknowledgement
Upload Product
Product Details
Upload Product Successfully
Response for Upload Product
Set time & Prize of Product
Request for Set time & Prize of Product
Set time & Prize of Product successfully
Response for Set time & Prize of Product\
View Bid Information
Request for View Bid Information
View Bid Information Successfully
Response for View Bid Information
Request for Logout
Logout Successfully

<u>Prototype models</u>:

1. Home Page:

This Home Page is open When Customer can Open the Site.



Wel Come To Auction Website

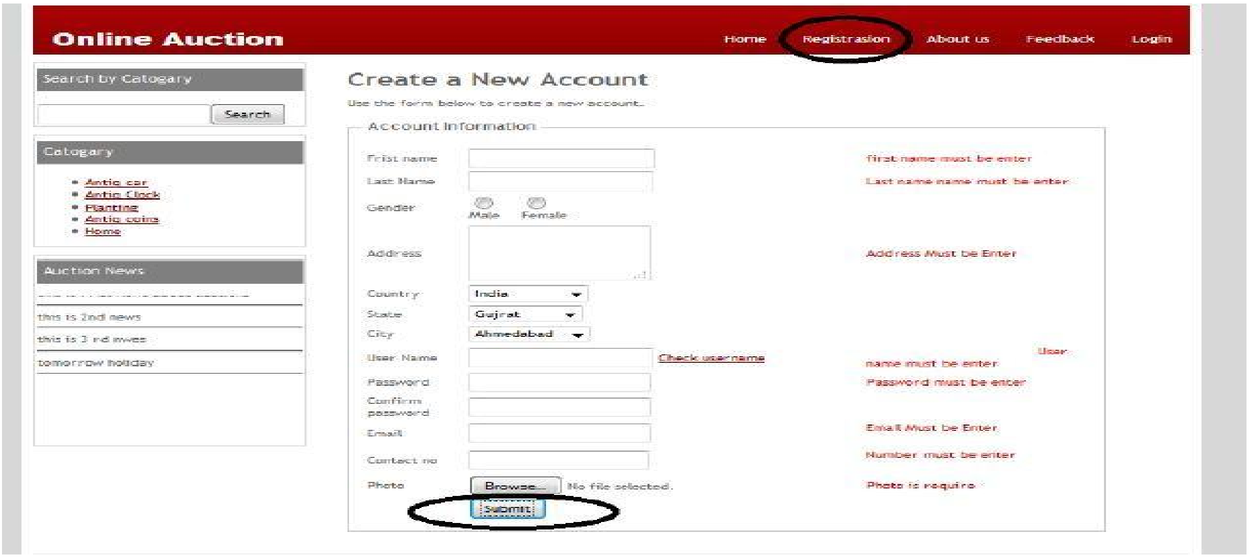
Wel come to my website , my website can auction for Antique Items. You can interested in die you can register now.

my site can very useful for the people. In our Website You can easily Auction the your Items. My site can only Auction for the Antique Items.

My website is very Famous. In my web site there are only 5 Category items Auction it.

2. Registration Form:

This page is used to customer can Registration here. But customer not enter data so error will be occur.



Create a New Account

Use the form below to create a new account.

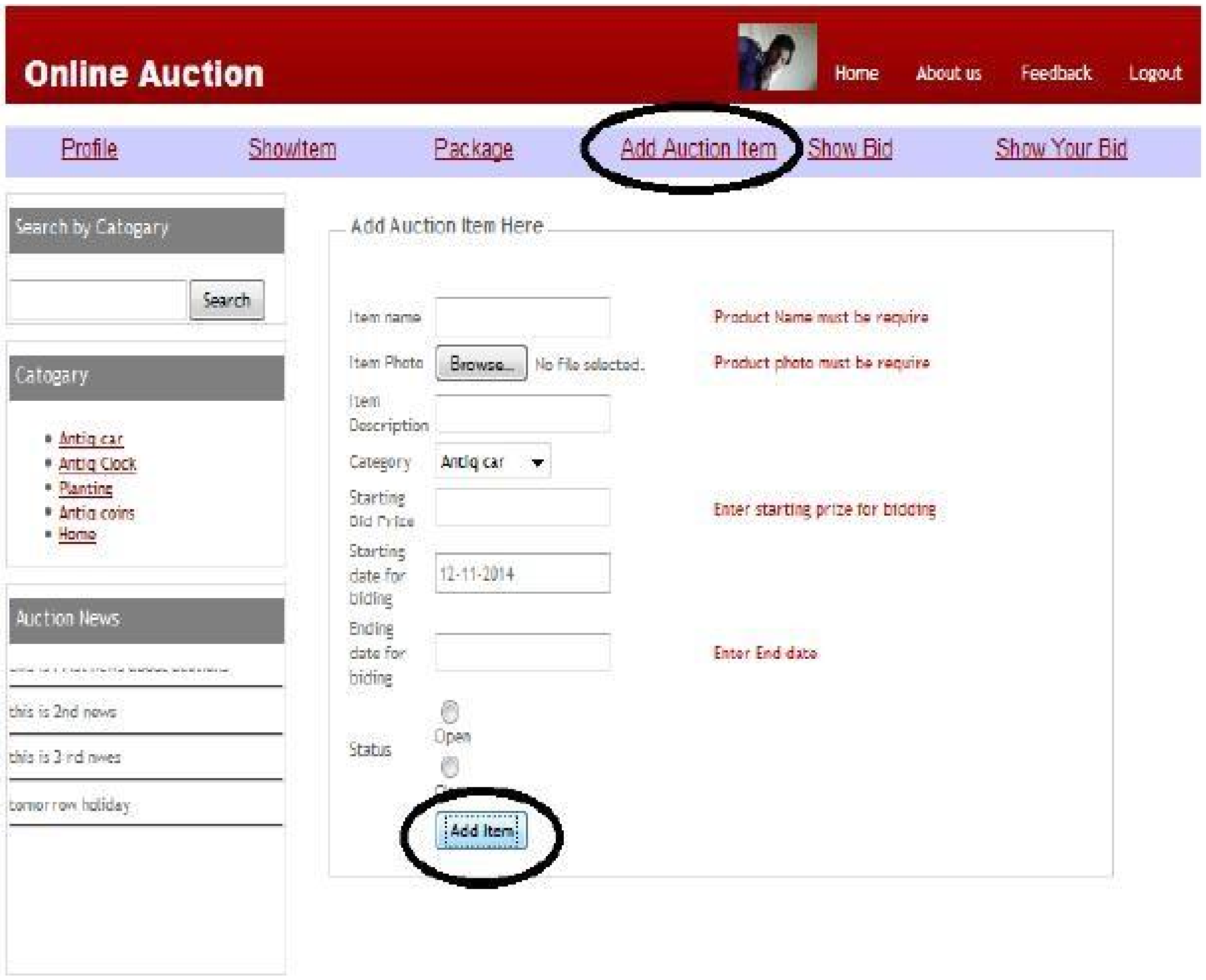
Account Information

Frist name		first name must be enter
Last Name		Last name name must be enter
Gender	Male Female	
Address		Address Must be Enter
Country	India	
State	Gujrat	
City	Ahmedabad	
User Name	Check username	User name must be enter
Password		Password must be enter
Confirm password		
Email		Email Must be Enter
Contact no		Number must be enter
Photo	Browse... No file selected.	Photo is require

Submit

3. Add Auction Item:

This page for user can not enter some data into the fields error will be occur.



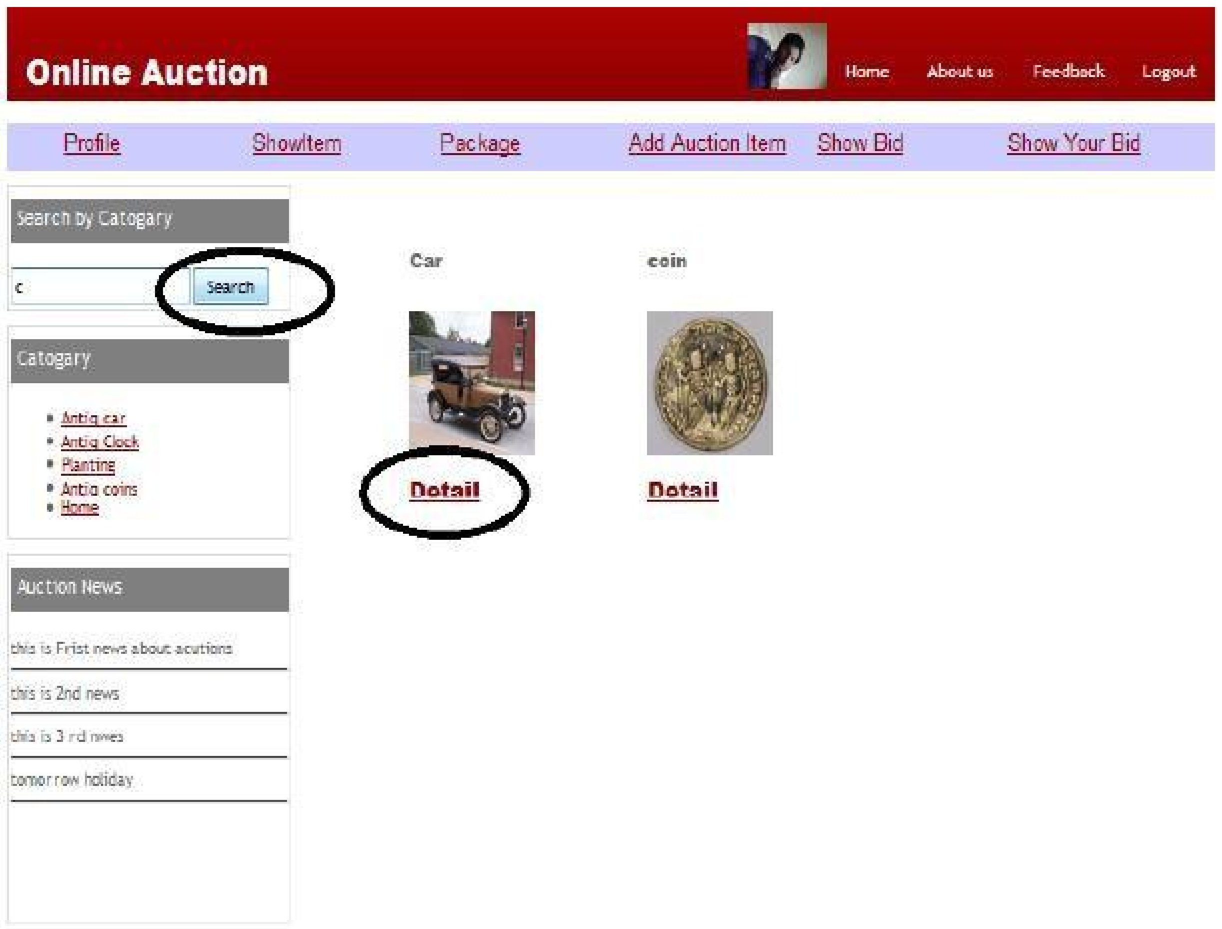
Add Auction Item Here

Item name		Product Name must be require
Item Photo	Browse... No File selected.	Product photo must be require
Item Description		
Category	Antiq car	
Starting Bid Price		Enter starting prize for bidding
Starting date for biding	12-11-2014	
Ending date for biding		Enter End date
Status	Open	

Add Item

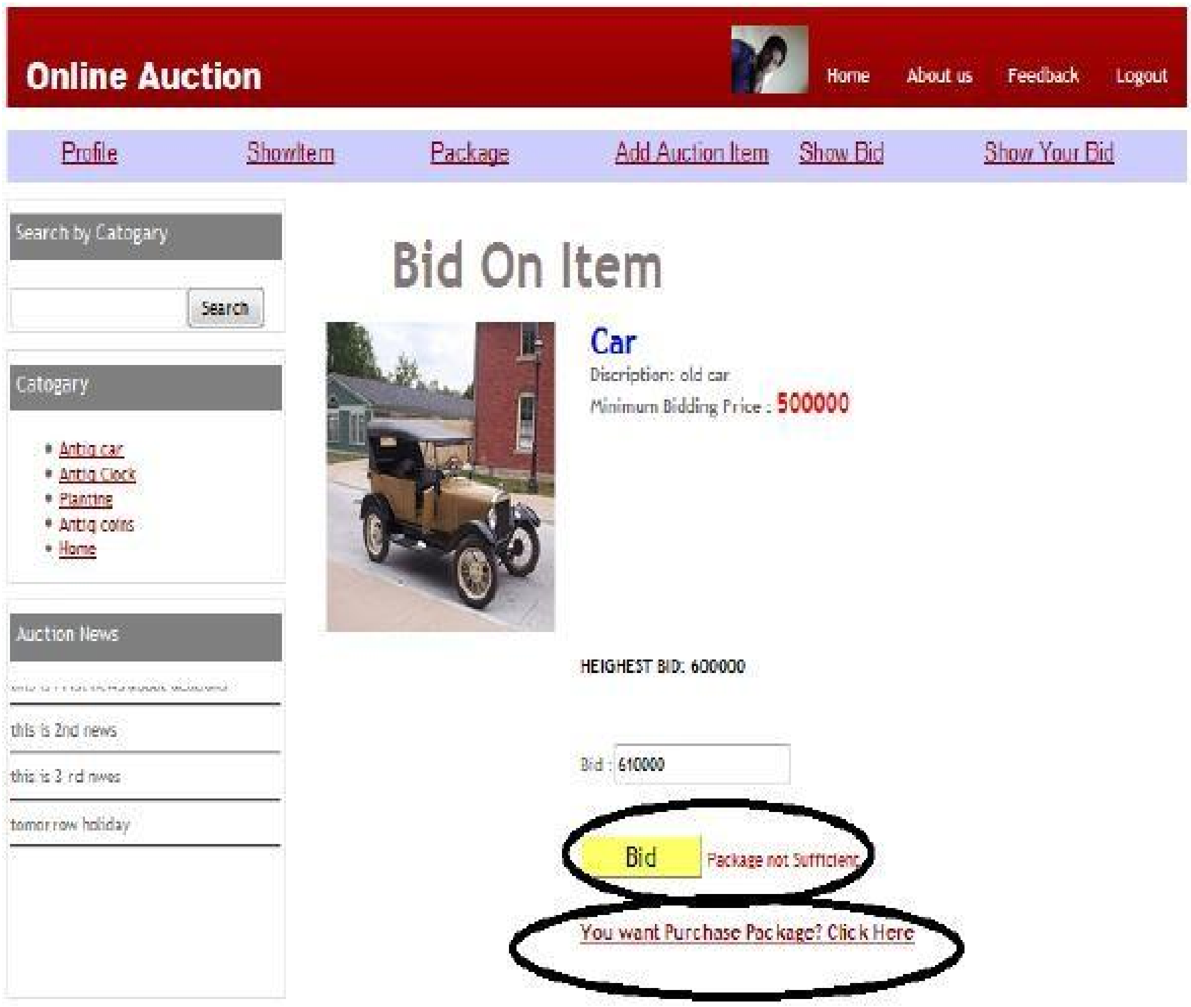
4. Search Item:

This page for user can search Items.



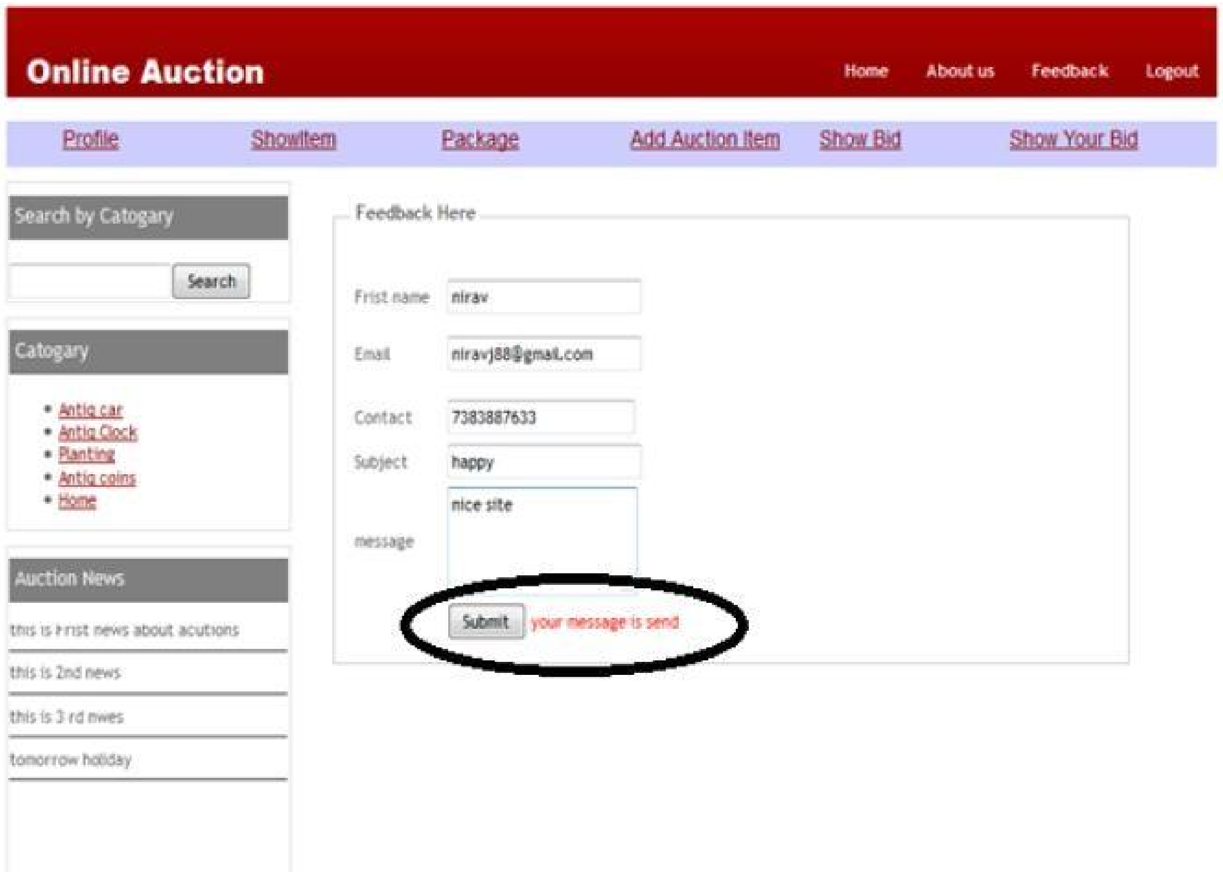
5. Bid On Item:

This page for user can Bid On the Particular Item then package not available so error will be occur.



6. Contact us :

This page for user have Any Query to Contact to the Company.



Feedback Here

Frist name: nirav

Email: niravj88@gmail.com

Contact: 7383887633

Subject: happy

message: nice site

your message is send

Experiment 4: Electronic Cash counter

This project is mainly developed for the Account Division of a Banking sector to provide better interface of the entire banking transactions. This system is aimed to give a better out look to the user interfaces and to implement all the banking transactions like:

•Supply of Account Information

•New Account Creations

•Deposits

•Withdraws

•Cheque book issues

•Stop payments

•Transfer of accounts

•Report Generations.

Proposed System:

The development of the new system contains the following activities, which try to automate the entire process keeping in view of the database integration approach.

•User friendliness is provided in the application with various controls.

•The system makes the overall project management much easier and flexible.

•Readily upload the latest updates, allows user to download the alerts by clicking the URL.

•There is no risk of data mismanagement at any level while the project development is under process.

•It provides high level of security with different level of authentication.

Problem Analysis and Project Planning

(1)Project Scope:

Internet Banking System refers to systems that enable bank customers to Access accounts and general Information on bank products and services through a personal computer or other intelligent device.

The chances and threats that the internet symbolizes is no longer news to the present day banking sector. No traditional bank would dare face investment analysts without an Internet strategy. The main intention behind the commencement of electronic banking services is to provide the customers with an alternative that is more responsive and with less expensive options. With options just a click away, customers have more control than ever. Their expectations are usability and real-time answers. They also want personal attention and highly customized products and services. Internet banking identifies a particular set of technological solutions for the development and the distribution of financial services, which rely upon the open architecture of the Internet. With the implementation of internet banking system, it maintain a direct relationship with the end users via the web and are able to provide a personal characterization to the interface, by offering additional customized services.

(2)Objectives:

The objective of this project is limited to the activities of the operations unit of the banking system which includes opening of Account, Deposit and withdraw of funds, Electronic funds transfer, Cheque balance and Monthly statement.

Software Requirement Analysis

(1)Module Description:

The Electronic cash counter Application project will be divided into 2 modules namely:

1. Bank Account
2. Bank Account Administrator

Bank Account

In this module the customer is allowed to logon to the website and can access his/her account by getting user name and password which will be verified with the server and the database. Once he/she gets verified then they are allowed to view their personal account and perform operations such as change of address, paying bills online, viewing transactions and transferring money into other accounts. Once the customer finishes the task the update information instantly gets stored into the database. The customer is then allowed to sign out from his/her account.

Bank Account Administrator

In this module the administrator is allowed to log on to the website and can access his/her administrative account by using the user name and password which will then be verified with the database. Once he/she gets verified the administrative interface will be displayed, where the administrator can perform operations for both new customers and existing customers. Administrator will help a new customer in opening their account by taking complete information from them. Administrator provides services like withdrawal, deposit, transfer and deleting customer during the time of closing the account. In this module administrator provides great customer service to the customers who want to do phone banking or teller banking. The interface for administrator will be both very users friendly and efficient. The data gets stored in the database instantly when the administrator hits the submit button.

(2) Functional Requirements:

- Customer can request details of the last ‘n’ number of transactions he has performed on any account.
- Customer can make a funds transfer to another account in the same bank.
- Customer can request for cheque book
- Customer can view his monthly statement. She/he can also take print out of the same.
- Customer can make Electronic Fund Transfer’s to accounts at their and other banks.
- The system is providing balance enquiry facility

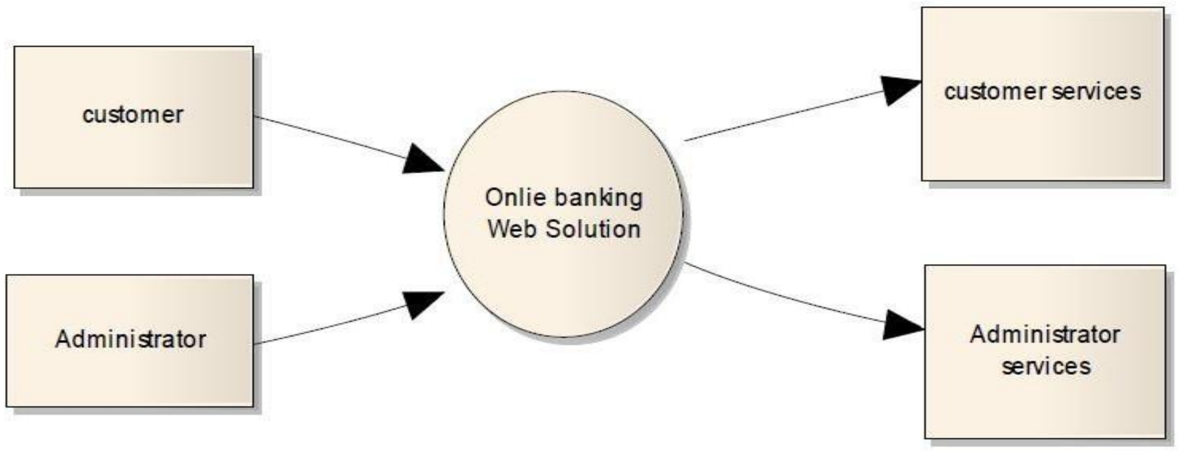
(3) Non-Functional Requirements:

Those requirements which are not the functionalities of a system but are the characteristics of a system are called the non-functionalities.

- Secure access of confidential data. Secure socket layer can be used.
- 24X7 availability
- Better component design to get better performance at peak time
- Flexible service based architecture will be highly desirable for future extensions.

Data Modeling

1) Context Level Diagram



Data Dictionary

Customer table

Name	Null?	Type
Customer_id (PK)	NOT NULL	INTEGER
Cust_first_name		VARCHAR2(20)
Cust_last_name		VARCHAR2(20)
DOB		VARCHAR(10)
Gender		VARCHAR2(2)

Login table

Name	Null?	Type
Customer_id (FK)	NOT NULL	INTEGER
Password		VARCHAR2(30)
Username		VARCHAR2(30)

Customer Detail table

Name	Null?	Type
Customer_id (FK)	NOT NULL	INTEGER
City		VARCHAR2(20)
State		VARCHAR2(20)
Zip		VARCHAR2(20)

Phone Number		NUMBER(10)
Email id		VARCHAR2(20)

Credit Card table

Name	**Null?**	**Type**
Request Number	NOT NULL	INTEGER
Name		VARCHAR2(30)
Profession		VARCHAR2(30)
Annual Income		INTEGER
Address		VARCHAR2(30)
City		VARCHAR2(30)
Telephone Number		VARCHAR2(30)
Card type		VARCHAR2(30)

Account table

Name	**Null?**	**Type**

Account Number (PK)	NOT NULL	NUMBER(8)
Customer_id (FK)	NOT NULL	INTEGER
Min_Balance		NUMBER(8)
Current_ balance		NUMBER(8)
Recommended_ by		VARCHAR2(20)
Nominee		VARCHAR2(20)
Type_of_account		VARCHAR2(20)
Date_of_opening		VARCHAR2(20)
Date_of_access		VARCHAR2(20)

Branch locator table

Name	**Null?**	**Type**
Location	NOT NULL	VARCHAR2(30)
Branch_city		VARCHAR2(20)

Employee table

Name	**Null?**	**Type**
Employee_id (PK)	NOT NULL	NUMBER(10)
Name		VARCHAR2(20)
Working_from		VARCHAR2(20)
Age		NUMBER(10)

Transaction(transfer-funds) table

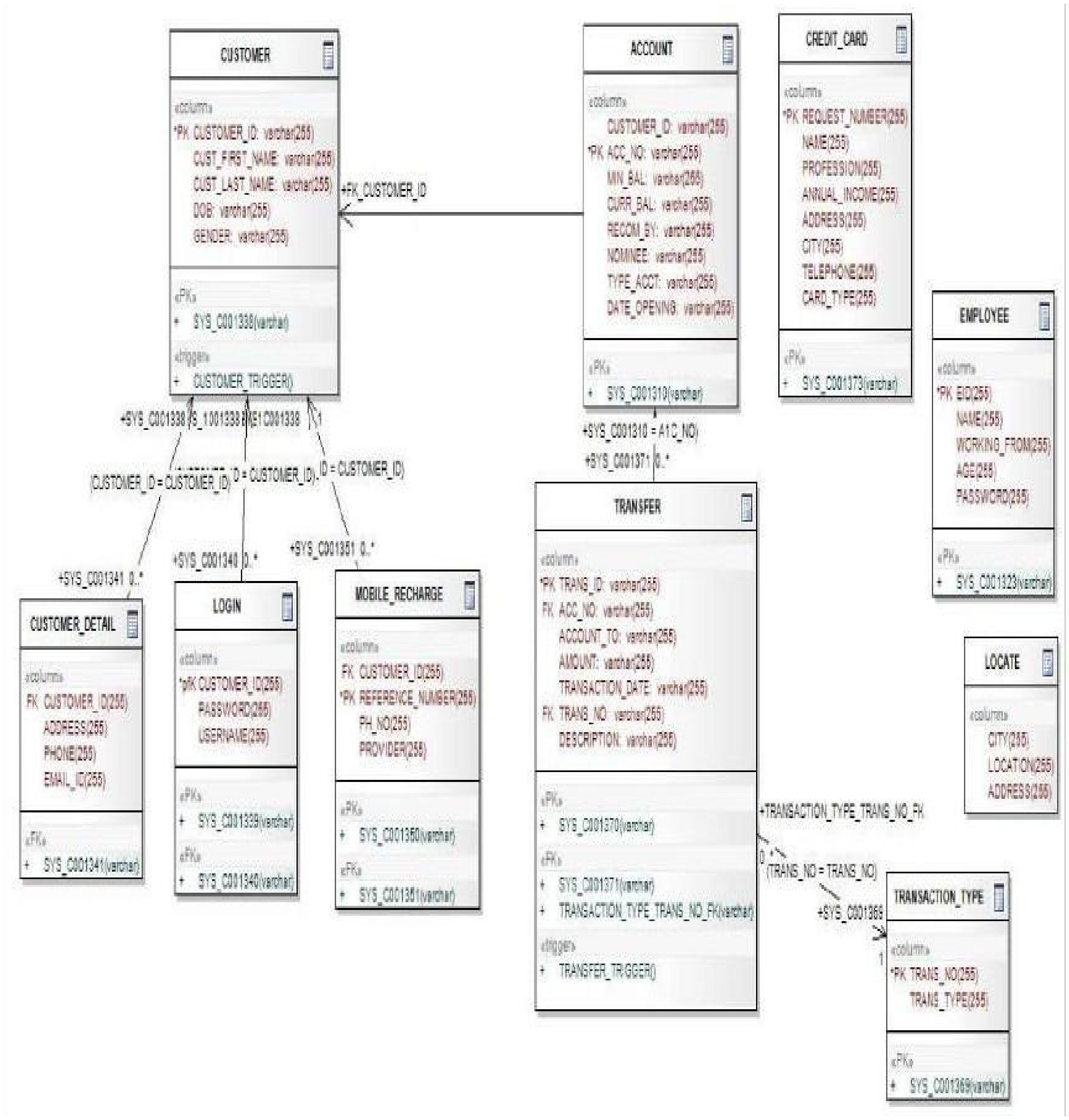
Name	**Null?**	**Type**
Trans_id	NOT NULL	NUMBER(10)
Acc_no		NUMBER(10)
Account_to		NUMBER(10)
Amount		NUMBER(10)
Transaction_date		VARCHAR2(20)
Trans_no		INTEGER
description		VARCHAR2(30)

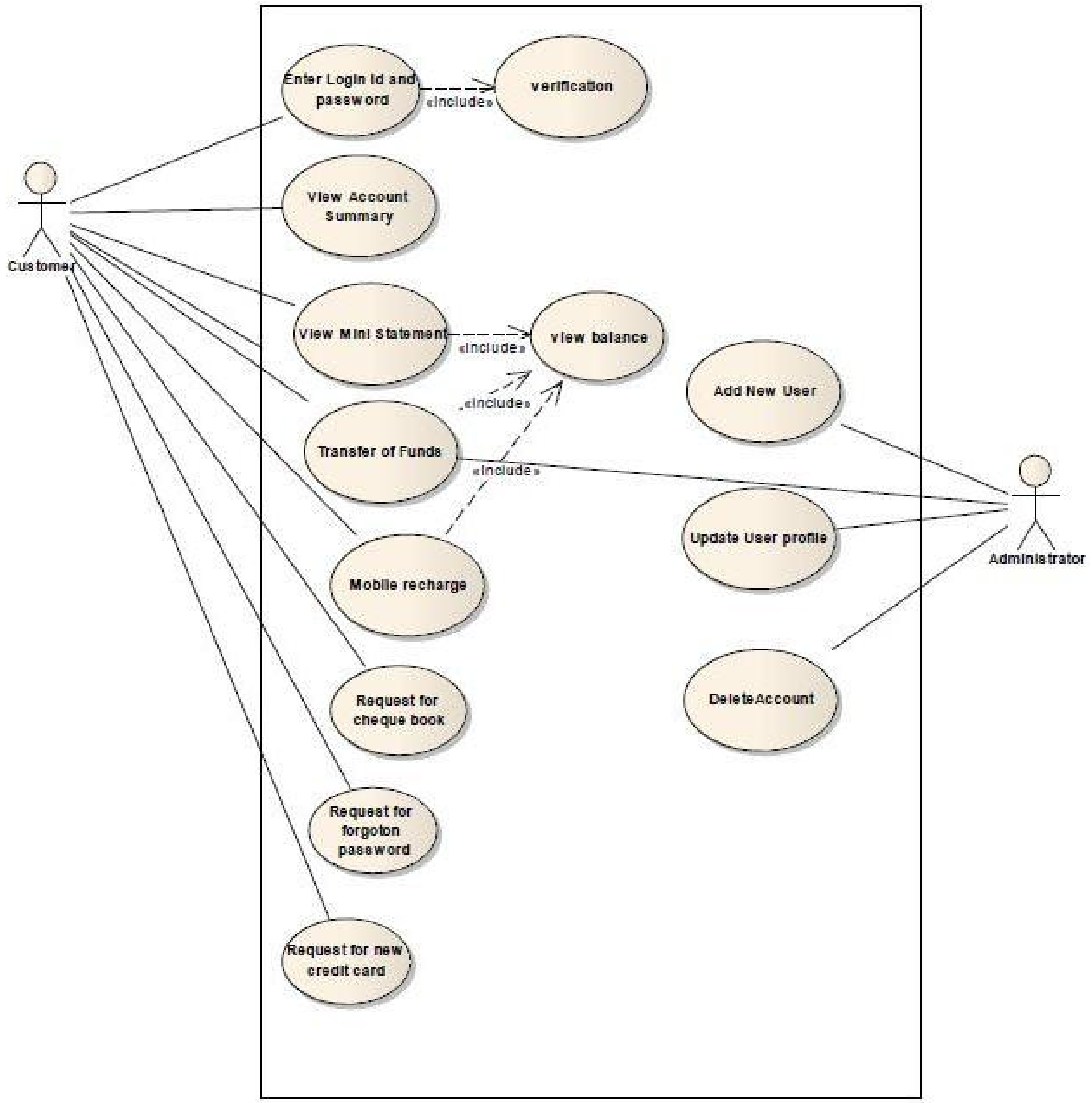
Transaction type table

Name	Null?	Type
Transaction Number (PK)	NOT NULL	INTEGER
Account Number (FK)	NOT NULL	INTEGER

Software Designing

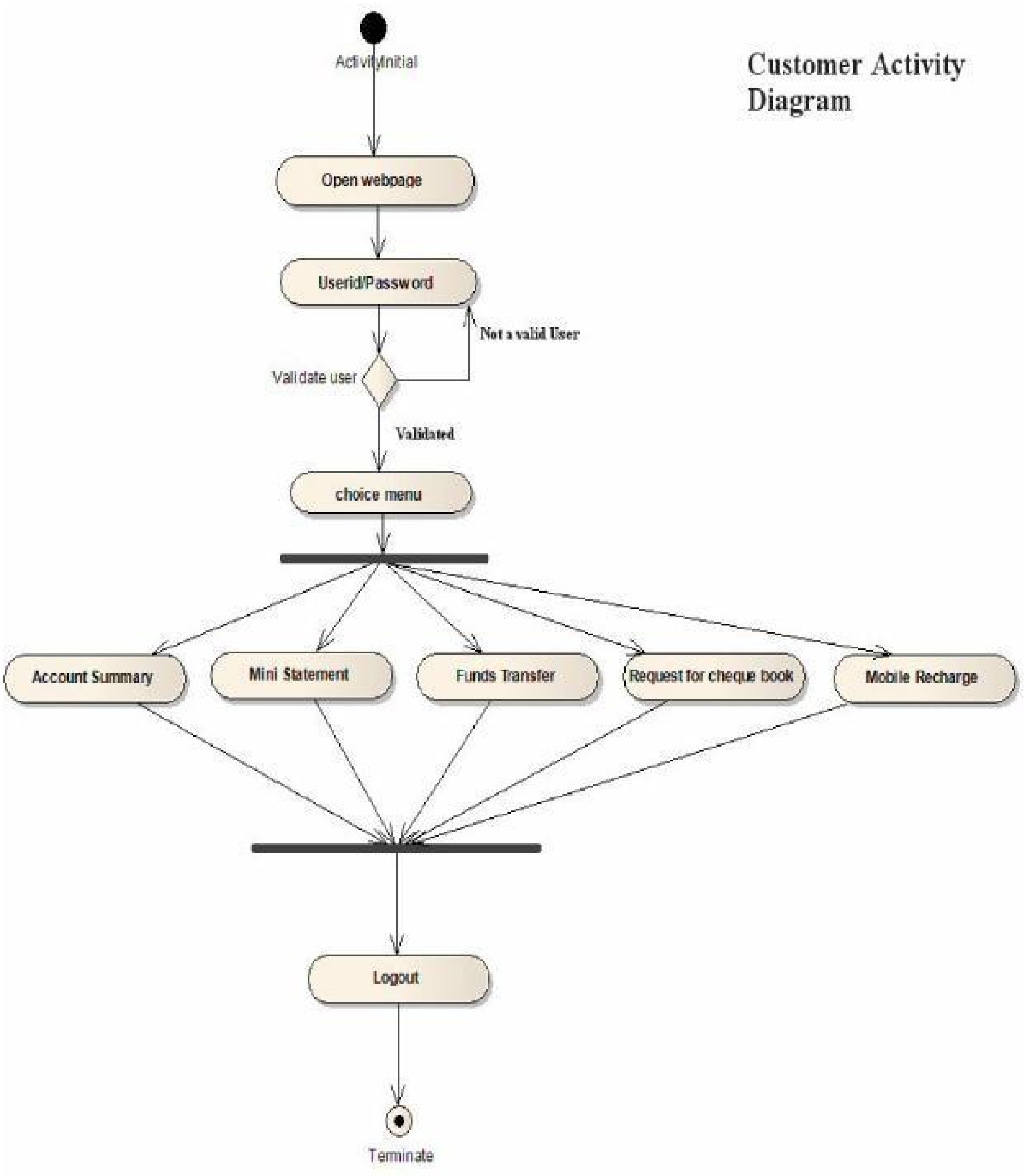
1) Class diagram:



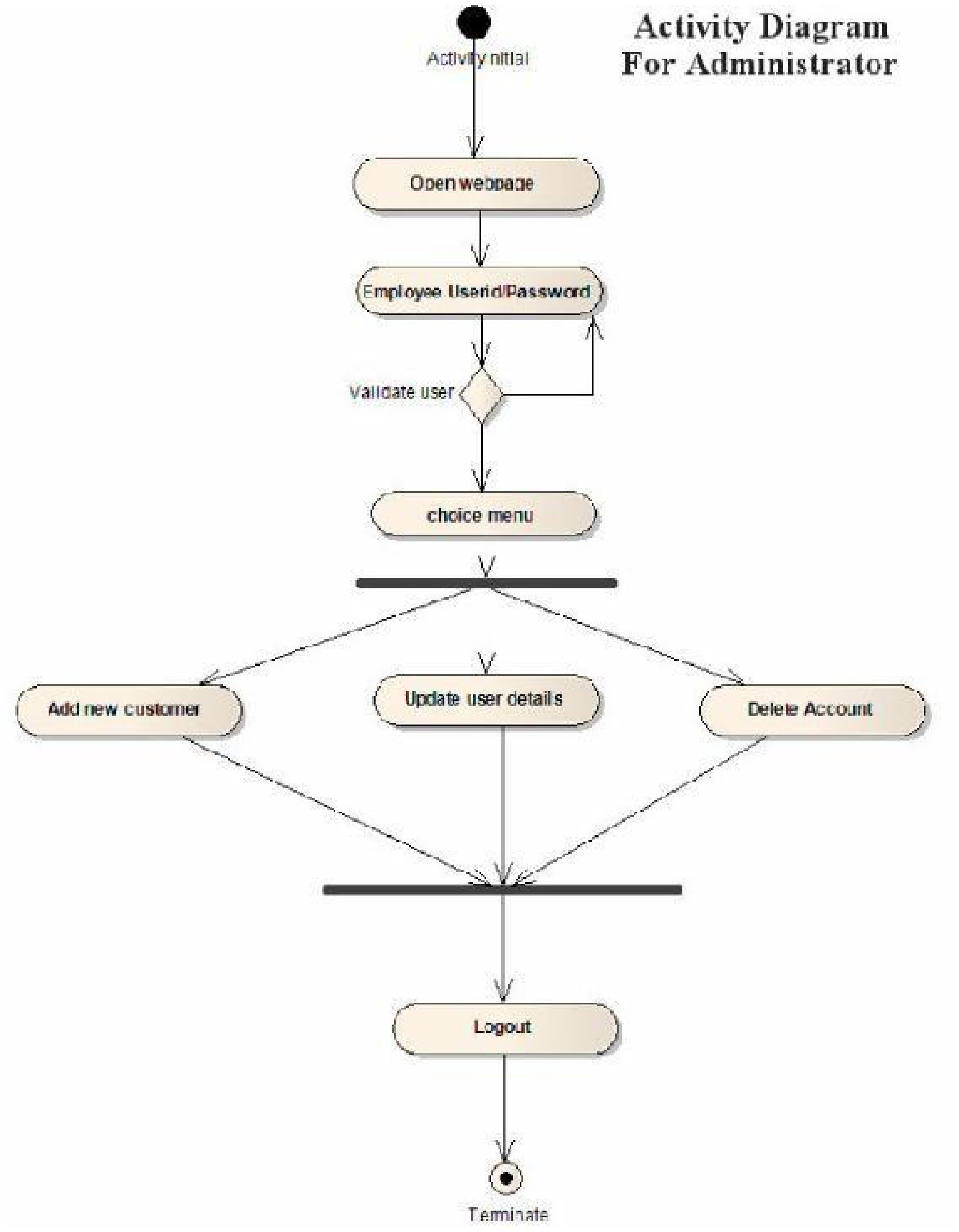

Enter Login Id and password
«include»
verification
View Account Summary
Customer
View Mini Statement
«include»
view balance
«include»
Transfer of Funds
«include»
Mobile recharge
Request for cheque book
Request for forgoton password
Request for new credit card
Add New User
Update User profile
Administrator
DeleteAccount

3)Activity Diagram

(3.1)Customer Activity Diagram



(3.2)Activity Diagram for Administrator



Prototype model

Prototype is a working model of software with some limited functionality. The prototype does not always hold the exact logic used in the actual software application and is an extra effort to be considered under effort estimation.

Prototyping is used to allow the users evaluate developer proposals and try them out before implementation. It also helps understand the requirements which are user specific and may not have been considered by the developer during product design.