**K.S.R COLLEGE OF ENGINEERING**

**(**AUTONOMOUS) THOKKAVADI POST, TIRUCHENGODE-637215

NAMAKKAL DIST, TAMILNADU



# RECORD NOTE BOOK

**REG NO:**

Certified that this is the bonafide record of work done by Selvan/Selvi……………………………….…of the………….

Semester………………………………………Branch during the year……………..in the laboratory.

## Staff-in-charge Head of the Department

Submitted for the university practical Examination on ……………………

Internal Examiner External Examiner

## CONTENTS

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.NO** | **DATE** | **NAME OF THE EXPIREMENT** | **PAGE NO** | **MARKS AWARDED** | **SIGNATURE** |
| 1 |  | PAYROLL SYSTEM |  |  |  |
| 2 |  | ONLINE SHOPPING |  |  |  |
| 3 |  | BANKING SYSTEMS |  |  |  |
| 4 |  | TEXT EDITOR |  |  |  |
| 5 |  | ONLINE VOTING SYSTEM |  |  |  |
| 6 |  | LIBRARY MANAGEMENT SYSTEM |  |  |  |

### EX.NO:1 DATE :

**Aim:**

**PAYROLL SYSTEM**

To prepare necessary documents and to develop the PAYROLL

SYSTEM with UML diagrams using Software Engineering Methodology

### PROGRAM ANALYSIS AND PROJECT PLANNING

**Problem Statement:**

This project PAYROLL SYSTEM is to develop an application to analyze the salary of the employee in a company. Admin can access all details in the application such as salary details of an employee. And can also update the salary details of an employee in a company. If the employee want the salary details the he want to request admin.

### SOFTWARE REQUIREMENT ANALYSIS

**The Modules in the Project:**

1. Admin Login.
2. Change password
3. Add or update record
4. Calculate salary.
5. Delete record.
6. View records.

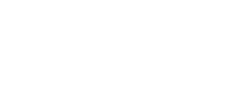
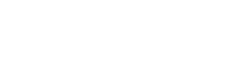
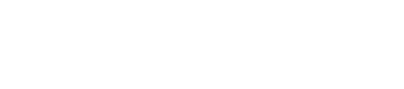
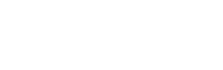
The first module is the user should login as an administrator with his login credentials such as username and password. Then the application verifies the username and password, then allow the user to continue.The second module allows the user to change current password to new password by taking input as current password, new password, confirm password.The third module is to add or update record by taking required employee details.The fourth module is to calculate salary for the entered input. The fifth module allows admin to remove a record from database.

### DATA MODELING

**Data Dictionary:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Field size | Default value | Validation |
| EmpId EmpName Experience BasicSalary Allowances Nod  Nol  pf  Tax  Salary | Identification Number  Name of the Employee  Experience of the Employee  Basic salary of the Employee  Other than Basic Salary  Number of days per month  Number of leave taken by Employee  Provident fund of the Employee  Tax of basic salary Take Home Salary | Integer String String  Number  Number  Integer  Integer Integer float Integer | 10  20  10  10  10  5  5  10  10  10 | NULL NULL NULL  NULL  NULL  NULL  NULL NULL NULL NULL | (0-9)  (A-Z) or (a-z) (0-9)  (0-9)  (0-9)  (0-9)  (0-9)  (0-9)  (0-9)  (0-9) |

**USE CASE DAIGRAM:**



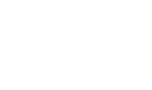
**Login**

**Change password**

**Administrator**

Add or Update record

Delete record



View list

**Use case Diagram:**

Use case diagram is a graph of actors, set of use cases enclosed by a system boundary, communication (participation) association between the actors and the use cases and a generalization among the use cases.

### Actor:

An actor represent a set of roles that user of a use case play when interacting with the use cases. Actor identified here is Administrator and Staff.

### Use case:

A use case is a description of a set of sequence of actions that a system performs to yield result of value to an actor.

The Use Cases described are,

* 1. Login
  2. Display Salary
  3. Calculate Current Salary
  4. Add New Records
  5. Update Records

The Login use case is to describe that, the user should choose his/her category whether he/she is a administrator or staff.

The use case Display Salary describes that, the system displays the salary detail of the staff from the database.

The use case Calculate Current Salary is to describe that, the administrator or the staff can calculate the current salary of basic salary and daily allowance available with the database.

The Add New records use case describes that; the administrator can add new records to the database.

The Update Records use case describes that the Administrator can update the records.

### SEQUENCE DIAGRAM

Sequence diagrams are easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and its environment. A sequence diagram shows an interaction arranged in a time sequence.

The objects used in this sequence diagram are,

1. Login
2. Display
3. Add
4. Update
5. Controller
6. Database



**login**

**display**

**controller**

**database**

**Enter as admin**

**send data**



**display salary**

**calculate current salary**

**send data**

**store data**

**send salary**

**display details**

### COLLOBORATION DIAGRAM :

**USER: ADMINISTRATOR**



**14: Update Records**

**Administrator**

**11: Add New Records**

**1: Enter as a administrator with name and password**

**5: Enter**

**10: Display Details**

**18: Display Details**

**15: Update Details**

**2: Send data**

**12: Send New Records**

**4: Valid Password**

**3: Check password**

**6: Calculate current salary**

**8: store data 13: Store New Records**

**16: Store Details**

**9: Send Salary**

**17: Send Details**

**7: send current salary**

**Login**

**Controller**

**Update**

**Database**

**Display**

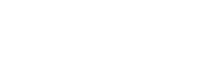
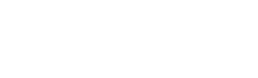
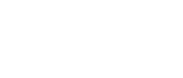
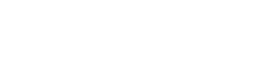
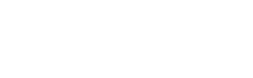
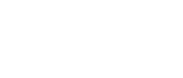
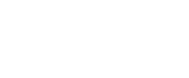
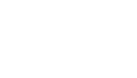
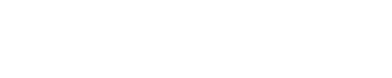
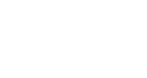
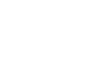
**Add**

A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchanged among the objects within the collaboration to achieve a desired outcome.

In this collaboration diagram, the objects are represented as rectangle, the actors are stick figures. Whereas the sequence diagram illustrates the object and actor interaction overtime, the collaboration diagram shows the object and actor interaction without reference to time.

In our PAYROLL SYSTEM each object interacts with each other or collaborates with each other; it gets represented by the solid line drawn between them.

### COLLOBORATION DIAGRAM: User: STAFF



**Staff**

**6: store data**

**1: Enter as a staff**

**7: Send Salary**

**5: send data**

**2: send data**

**4: calculate Current Salary**

**3: Display Salary**

**8: Display Details**

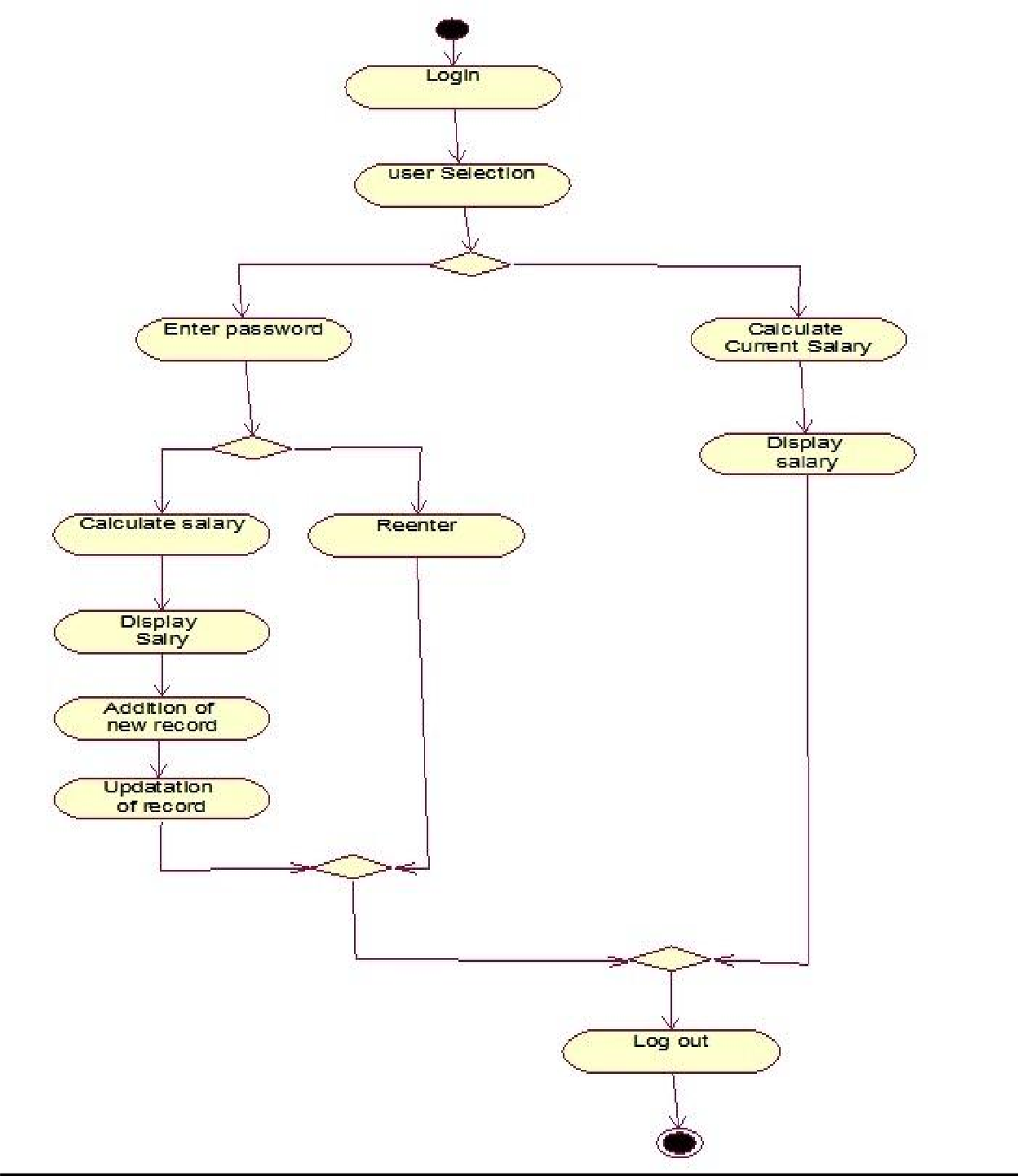
**display**

**controller**

**login**

**database**

**ACTIVITY DIAGRAM:**



The activity diagram describes the sequencing of activities with support for both conditional and parallel behavior.

The Activity diagram is used to describe the various activities taking place in an application. Here in our PAYROLL SYSTEM, we have various activities starting from login.

After login, the user selection activity gets performed, where the user can be an administrator or staff.

If the user is a administrator, then they have to enter their name and password and only when those details are valid they can access the system. They can calculate the current salary obtained by the staffs, they can add new records, and they can update the values of the records which gets stored in the database.

If the user is a staff then they can view their salary detail and they can calculate their current salary.

### CLASS DAIGRAM:4

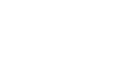
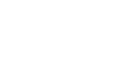
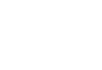
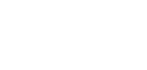
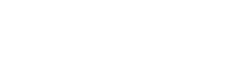
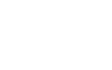
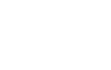
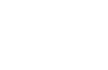
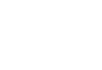
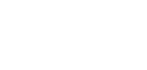
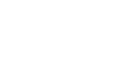
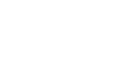
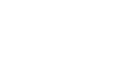
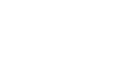
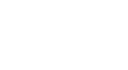
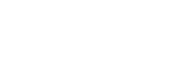
Class diagrams show the interactions between classes in the system. Class diagram also shows the attributes and operation of a class and the constraints that apply to the way objects are connected.

Classes contain information and behavior that acts on that information. Each class on class diagram is represented by rectangle divided into three sections. The first section shows the class name, second section shows the attributes the class contains and last section contains the operation of the class.

In our PAYROLL SYSTEM, the classes identified are

1. Login
2. Display
3. Add
4. Update
5. Controller
6. Database

Each class has its own attributes and operations.



**r**

**Add new record ()**

1

1

1

1

**Display()**

**Update() Add() Store()**

1

1

**Update**

**Update ()**

**ID.no Name Salary**

**Display ()**

**Add () Update () Display ()**

**ID.no Name Destination B.Salary C.salary**

**Id.no Name Password Salary**

**ID.no Name Destination B.Salary**

**C. Salary**

**Display**

**Controller**

**Database**

1

1

**Login ()**

**ID.no Name Salary**

**Administrato Staff**

**Add**

**Login**

Login class - The attributes defined is administrator and staff.

The method identified is login.

Display class - The attributes are id.no, name, destination, b.salary, c.salary.

The operation identified is Display.

Add class - The attributes are id.no, name, destination, b.salary, c.salary.

The operation defined is adding new record.

Update class - The attributes are id.no, name, salary,

The operation defined is update.

Controller class -The attributes it has is id.no, name, password, salary. The operations carried out by this class are added, update and display.

Database class - The attributes are id.no, name, destination, b.salary, c.salary. The operations defined are store, display, update, and add.

The Solid line between the classes shows the Association relationship between them.

### SOFTWARE DEVELOPMENT

**Login Form**:

Private Sub Command1\_Click()

If Text1.Text = "rv" And Text2.Text = "mk" Then payroll.Show

Else

MsgBox ("Invalid input") End If

End Sub

### Payroll Form:

Dim id As Integer

Private Sub cmdadd\_click()

If Adodc1.Recordset.RecordCount = 0 Then id = 0

Else Adodc1.Recordset.MoveLast

id = Mid(Adodc1.Recordset("empid"), 2, 3) + 1 End If

Adodc1.Recordset.AddNew textid = Format(id, "E000") textname.SetFocus

End Sub

Private Sub cmdexit\_click() End

End Sub

Private Sub cmdsave\_click()

Dim flag As Boolean flag = False

Select Case combodesig Case "manager"

If Val(textbasic.Text) > 15000 And Val(textbasic.Text) <= 25000 Then textda.Text = Val(textbasic) \* 0.08

texthra.Text = Val(textbasic) \* 0.03 textdeduct.Text = Val(textbasic) \* 0.01

textgross.Text = Val(textbasic) + Val(textda) + Val(texthra) textnet.Text = Val(textgross) - Val(textdeduct)

flag = True Else

MsgBox "input basic pay between 15k and 25k", vbInformation + vbOKOnly

End If

Case "engineer"

If Val(textbasic.Text) > 15000 And Val(textbasic.Text) <= 20000 Then textda.Text = Val(textbasic) \* 0.07

texthra.Text = Val(textbasic) \* 0.04 textdeduct.Text = Val(textbasic) \* 0.01

textgross.Text = Val(textbasic) + Val(textda) + Val(texthra) textnet.Text = Val(textgross) - Val(textdeduct)

flag = True Else

MsgBox "input basic pay between 15k and 20k", vbInformation + vbOKOnly End If

Case "clerk"

If Val(textbasic.Text) > 5000 And Val(textbasic.Text) <= 15000 Then textda.Text = Val(textbasic) \* 0.04

texthra.Text = Val(textbasic) \* 0.02 textdeduct.Text = Val(textbasic) \* 0.01

textgross.Text = Val(textbasic) + Val(textda) + Val(texthra) textnet.Text = Val(textgross) - Val(textdeduct)

flag = True Else

MsgBox "input basic pay between 5k and 15k", vbInformation + vbOKOnly End If

Case "messenger"

If Val(textbasic.Text) > 5000 And Val(textbasic.Text) <= 10000 Then textda.Text = Val(textbasic) \* 0.03

texthra.Text = Val(textbasic) \* 0.02 textdeduct.Text = Val(textbasic) \* 0.01

textgross.Text = Val(textbasic) + Val(textda) + Val(texthra) textnet.Text = Val(textgross) - Val(textdeduct)

flag = True Else

MsgBox "input basic pay between 5k and 10k", vbInformation + vbOKOnly End If

End Select If flag Then

Adodc1.Recordset.Update

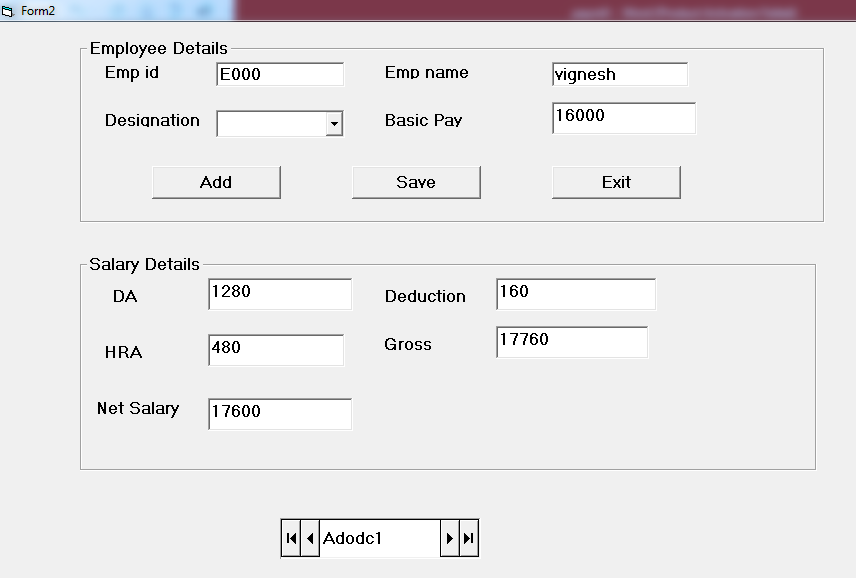
MsgBox "employee details saved", vbOKOnly End If

End Sub

### Output: Form1:



**Form2:**



**SOFTWARE TESTING:**

**TEST CASES:**

* If an unauthorized user attempts to access the system, the system should not allow them to access.
* If the password entered by the administrator is incorrect, the system should display the bad password message and allows them to reenter the password or to terminate the process.

### RESULT:

Thus the **PAYROLL SYSTEM** is developed with all necessary documents and UML diagrams using Software Engineering methodology.

### EX.NO:2 DATE :

**ONLINE SHOPPING**

**AIM**

To prepare necessary documents and to develop ONLINE SHOPPING with UML diagrams using Software Engineering Methodology.

### PROGRAM ANALYSIS AND PROJECT PLANNING

**Problem Statement:**

This project ONLINE SHOPPING is to develop an application for shopping. Initially a user should choose the category whether he\she is a Customer or Administrator. If the person is an Administrator then they can add the products. If the user is a Customer then he\she can register an account, then choose and buy the products through online.

### SOFTWARE REQUIREMENT ANALYSIS

**The Modules in the Project:**

* 1. Login.
  2. Add Products.
  3. User Registration.
  4. Choose Products.
  5. Calculate Total Amount.
  6. Online Payment.

The first module is Login in which the user has to login to the system as a Customer or Administrator.

The next module is Add Products, where if the user is an administrator, then he\she add the new products.

The next module is User Registration the new customer can register to create an account.

The next module is Choose Products, the customer choose the products.

The next module is Calculate Total Amount, here the total amount for the products are calculated.

The last module is the Online Payment the payment is transferred through online.

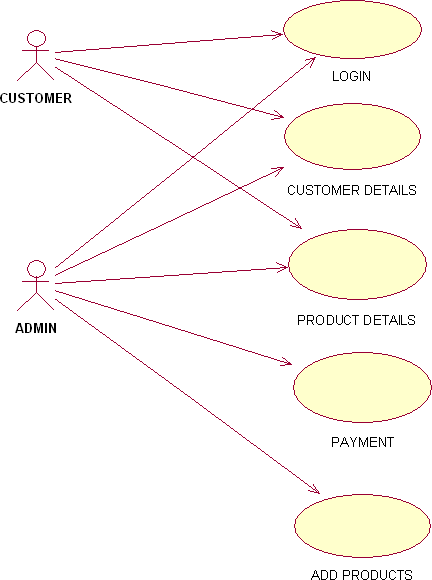
### DATA MODELING

**Data Dictionary:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data Type | Field size | Default value | Validation |
| Name Pass wd dob Gender Address Ph no  Mail id Pi d  P name  P type C name price  quantity | Name of the customer  Password for the customer  Date of Birth  Gender  Address of the customer Phone no of the customer Customer’s Mail id Product id  Name of the Product Type of the Product Company Name Price of the Product  Quantity of the Product | String  String  Date/Tim e  String String Double String Integer String String String  Currency  Integer | 50  50  8  50  70  8  50  2  50  50  50  8  2 | NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL NULL  NULL | (A-Z) or (a-z)  (A-Z) or (a-z) (0-9)  (A-Z) or (a-z)  (A-Z) or (a-z) (0-9)  (A-Z) or (a-z) (0-9)  (A-Z) or (a-z)  (A-Z) or (a-z)  (A-Z) or (a-z) (0-9)  (0-9) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| t payment  b name  acc no date | Type of Payment  Name of the Bank  Account number for the Bank  Current Date | String  String  Integer Date | 50  50  8  8 | NULL  NULL  NULL NULL | (A-Z) or (a-z)  (A-Z) or (a-z) (0-9)  (0-9) |

**USE CASE DIAGRAM:**



**Use case Diagram:**

Use case diagram is a graph of actors, set of use cases enclosed by a system boundary, communication association between the actors and the use cases and a generalization among the use cases.

### Actor:

An actor represent a set of roles that user of a use case play when interacting with the use cases. Actor identified here is Administrator and Customer.

### Use case:

A use case is a description of a set of sequence of actions that a system performs to yield result of value to an actor.

The Use Cases described are,

* + 1. Login
    2. Customer Details
    3. Product Details
    4. Payment
    5. Add Products
  + The Login use case is to describe that, the user should choose his/her category whether he/she is a administrator or customer.
  + The use case Customer Details describes that, the details of the customers are added into the database.
  + The use case Product Details describes that, it displays the list of products from the database.
  + The Payment use case describes that, the payment options for online shopping.
  + The Add Products use case describes that the Administrator can add the new products.

### SEQUENCE DIAGRAM User: ADMIN



: ADMIN

Database

Add

Login

Enter as Admin

Login succeed

Add data

Send data

Store data

### Sequence Diagram:

Sequence diagrams are easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and its environment. A sequence diagram shows an interaction arranged in a time sequence.

The objects used in this sequence diagram are,

1. Login
2. Add
3. Display
4. Controller
5. Payment
6. Database

### SEQUENCE DIAGRAM User: CUSTOMER



Registration

login

Display

Controller

Payment

Database

: CUSTOMER

Enter as new customer

Add details

Send details

Store details

Enter as customer

Check details

Login succeed

send data

Choose datails

Enter quantity

Calculate Total amount

Enter payment Details

Send payment details

Store payment details

### Basic Flow:

The administrator or customer enters their name and password, and the password gets checked by the system. After confirmation of the password the system allows them to access.

### Alternate Flow:

* The actor Administrator and Customer are the persons who interacts with the system.
* The object Login makes the administrator/ customer to enter.
* The object Add will add the new products into the database.
* The object Display will display the product details from the database.
* The object Payment will allows to choose the mode of payment.
* The object Database will store all the product details & customer details.

### COLLABORATION DIAGRAM : USER: ADMIN

: ADMIN



1: Enter as Admin

5: Add New

Login

Add

6: Send Product Details

2: Send

4: Valid

7: Store Product Details

3: Verify Username & Password

Database

Controller

### COLLABORATION DIAGRAM: User: CUSTOMER



6: Store data

: CUSTOMER

...

Database

1: Enter as customer

5

: Send data

Login

7: Send Product details

Payment

2: Send data

4: Send data

3: Calculate Total Amount 8: Display Amount & Product

Controller

display

### Collaboration Diagram:

A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchanged among the objects within the collaboration to achieve a desired outcome.

Collaboration diagram shows exactly the same information as the sequence diagram. However, collaboration diagram shows this information in a different way and with different purpose.

In this collaboration diagram, the objects are represented as rectangle, the actors are stick figures. Whereas the sequence diagram illustrates the object and actor interaction overtime, the collaboration diagram shows the object and actor interaction without reference to time.

In our ONLINE SHOPPING each object interacts with each other or collaborates with each other; it gets represented by the solid line drawn between them.

### ACTIVITY DIAGRAM:

The activity diagram describes the sequencing of activities with support for both conditional and parallel behavior.

The Activity diagram is used to describe the various activities taking place in an application. Here in our ONLINE SHOPPING, we have various activities starting from login.

After login, the user selection activity gets performed, where the user can be an administrator or customer.

If the user is a administrator, then they have to enter their name and password and only when those details are valid they can access the system. They can add the new products which gets stored in the database.

If the user is a customer, then they login into the system using valid username and password, after login they can choose the products and also buy the products.

LOGIN



LOGIN SELECTION

ADMIN CUSTOMER

ADD PRODUCT

NEW CUSTOMER USER

REGISTRATION

DISPLAY PRODUCT DETAILS

UPDATING PRODUCT RECORDS

UPDATING RECORDS

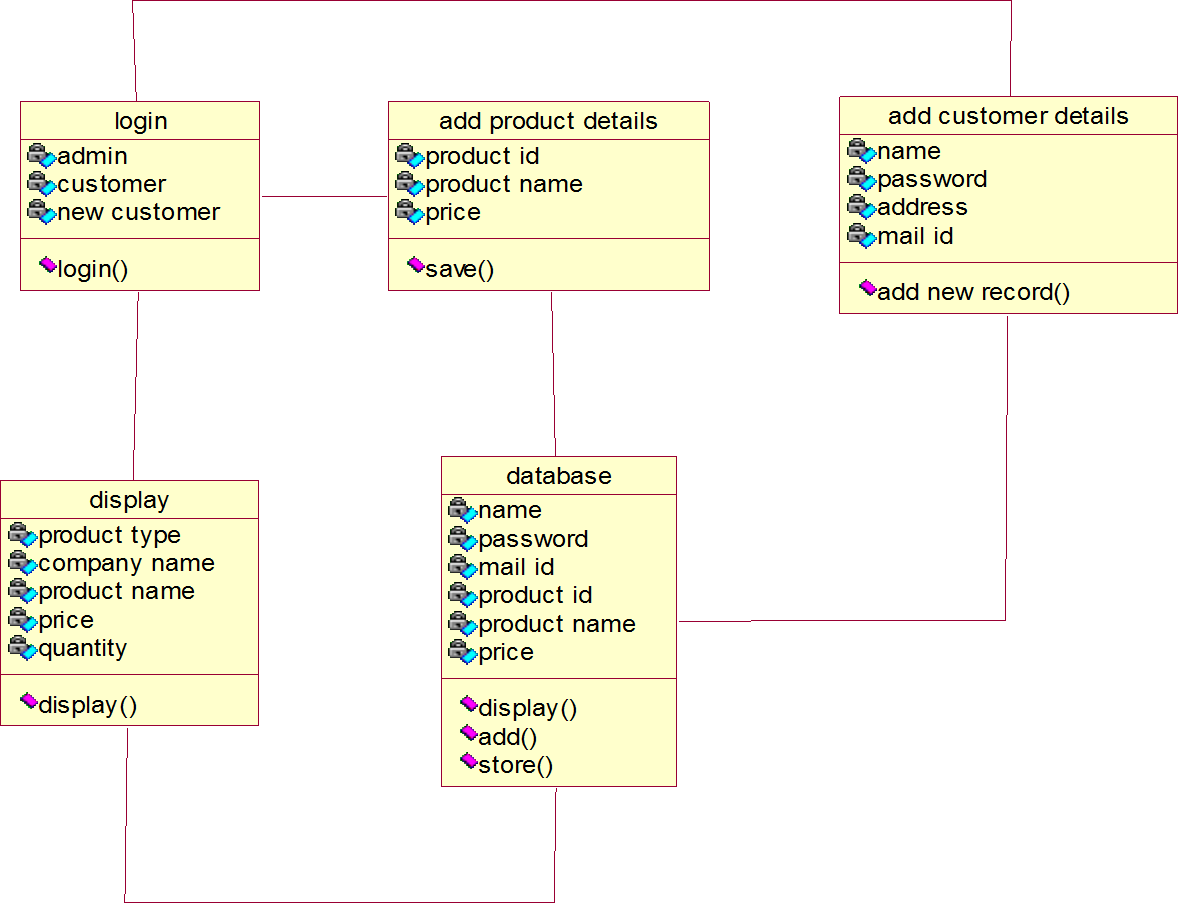
CHOOSE PRODUCT

CALCULATE TOTAL

PAYMENT

LOGOUT

### CLASS DIAGRAM:



**Class Diagram:**

Class diagrams show the interactions between classes in the system. Class diagram also shows the attributes and operation of a class and the constraints that apply to the way objects are connected.

Classes contain information and behavior that acts on that information. Each class on class diagram is represented by rectangle divided into three sections. The first section shows the class name, second section shows the attributes the class contains and last section contains the operation of the class.

In our ONLINE SHOPPING, the classes identified are

1. Login
2. Add Product Details
3. Add Customer Details
4. Display
5. Database

Each class has its own attributes and operations.

**Login class** - The attributes defined are administrator, customer. and new customer .The method identified is login.

**Add Product Details class** - The attributes are product id, product name, Price.The operation defined is saving new record.

**Add Customer Details class** - The attributes are name, password, address, mail .The operation defined is adding new record.

**Display class** - The attributes are product name, company name, price, quantity.The operation identified is Display.

**Database class** - The attributes are name, password, mail id, product name, price.The operations defined are display, add and store.

The Solid line between the classes shows the Association relationship between them.

### SOFTWARE DEVELOPMENT:

**Login Form:**

**Private Sub Command1\_Click()**

If Text1.Text = "rv" And Text2.Text = "mk" Then shopping.Show

Else

MsgBox ("Invalid input") End If

End Sub

### Shopping Form:

**Private Sub buybtn\_Click()** shoppingado.Recordset.Fields("name") = txtname.Text shoppingado.Recordset.Fields("date of birth") = txtdob.Text shoppingado.Recordset.Fields("gender") = txtgender.Text shoppingado.Recordset.Fields("Address") = txtadd.Text shoppingado.Recordset.Fields("Contact no") = txtphone.Text shoppingado.Recordset.Fields("mail") = txtmail.Text shoppingado.Recordset.Fields("product name") =

txtproduct.Text shoppingado.Recordset.Fields("price") = txtprice.Text

shoppingado.Recordset.Fields("quantity") = txtquantity.Text shoppingado.Recordset.Fields("total amount") = txtamount.Text

shoppingado.Recordset.Fields("payment") = txtpayment.Text shoppingado.Recordset.Fields("code word") = txtcode.Text shoppingado.Recordset.Update

MsgBox "User Shopping Successful" End Sub

### Private Sub Command2\_Click() End

End Sub

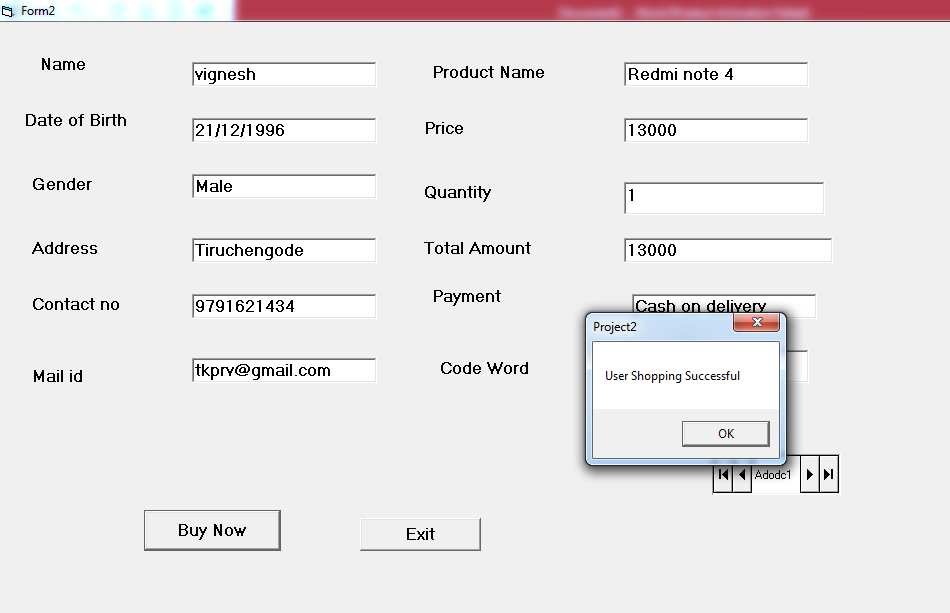
**Private Sub Form\_Load()** shoppingado.Recordset.AddNew End Sub

### OUTPUT:

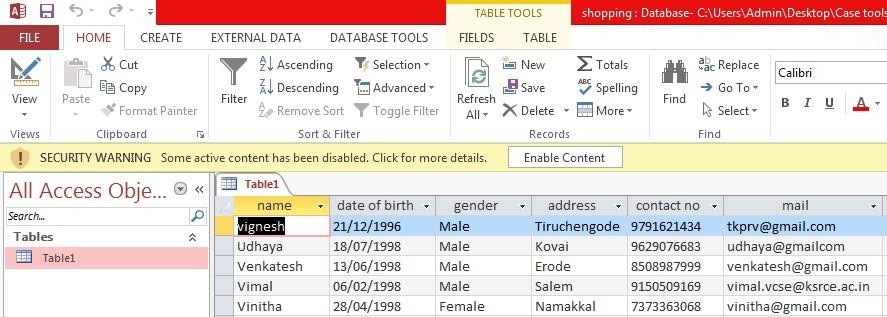
**Form1:**

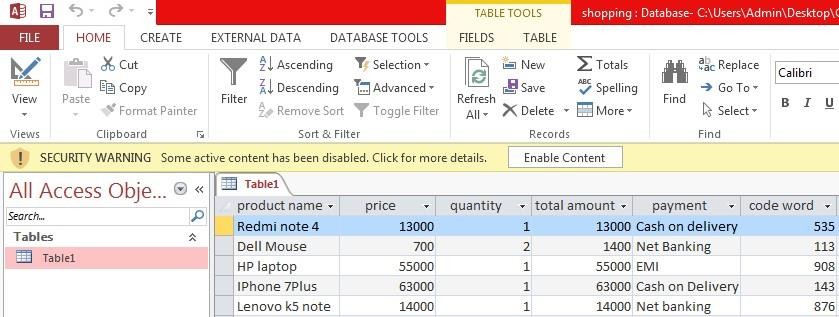


**Form2:**



**Database Form:**





**RESULT:**

Thus the ONLINE SHOPPING is developed with all necessary documents and UML diagrams using Software Engineering methodology.

### EX.NO:3 DATE

**BANKING SYSTEMS**

**AIM.:**

To develop a software for banking system by using software engineering methodology.

### ABSTRACT:

All the banks are centralized with computers nowadays. All the accounts maintained in the bank and also the transaction effected, including banking transaction, are to be processed by the computers in the bank. An BANKING accepts relevant cash card, interacts with user, communicates with central system to carry out the transaction such as amount deposit,withdraw,balance enquiry and prints receipts. The system to be designed and implement must include appropriate record keeping and security provisions. The system must handle concurrent access to the same amount.

### SYSTEM SPECIFICATION:

**Software Requirements:**

Operating system : Windows xP Front end : Visual Basic 6.0

Back end : Microsoft Access

### Hardware Requirements:

Processor : Intel Pentium @ 3.06GHz

RAM : 512MB DDR

Hard Disk : 80GB SATA

Monitor : 15’’TFT

Keyboard : Multimedia Keyboard

Mouse : USB Optical type

#### PROGRAM ANALYSIS AND PROJECT PLANNING

**PROBLEM STATEMENT:**

This application BANKING System helps the bank client to access his/her account. The client has to first enter his/her name and account id number. Once when the details are confirmed the client can access the account, if the details are wrong then the client has to reenter the details. Next the system should ask for the type of transaction to perform, where the client has to choose the type of transaction they wish to carry on. And based on the transaction chosen the actions have to be carried out. And based on their transaction their account has to be updated.

### SOFTWARE REQUIREMENT ANALYSIS

#### Modules.:

1. Card verification.
2. Selecting transaction.
3. Deposit.
4. Withdraw.

### Card verification:

This module is an Approval process in which the client has to enter his/her name and account ID number, these details will be referred with the database and the service will be provided if the details are correct.

### Selecting Transaction:

This module selects transaction in which the system displays the types of transaction (Deposit/Withdraw/Balance enquiry), the client has to choose any one, and the transaction will be carried on accordingly.

### Deposit:

This module is Deposit saving module where the client has to enter the amount to be deposited. The account will be updated automatically and saved.

### Withdraw:

This is a module where the client has to enter the amount to withdraw. After verifying that the funds are sufficient, the transaction is performed.

### DATA MODELING

**Data Dictionary:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Field Name** | **Description** | | | **Data Type** | **Field size** | **Default** | **Validation** |
| **value** |
| Name | Name of the client | |  | String | 20 | NULL | (A-Z) or |
|  |  | |  |  |  |  | (a-z) |
| D.O.B | Date of birth | |  | Date and | 10 | NULL | (0-9) |
|  |  | |  | time |  |  |  |
| Gender | Specify the gender | |  | string | 20 | NULL | (A-Z) or |
|  |  |  |  |  |  |  | (a-z) |
| Acc.no | Account | Number | of | Integer | 10 | NULL | (0-9) |
|  | the Client | |  |  |  |  |  |
| Balance | Amount | In the |  | Integer | 20 | NULL | (0-9) |
|  | client’s | |  |  |  |  |  |
|  | Account | |  |  |  |  |  |

**USE CASE DIAGRAM:**

Use case diagram is a graph that describes the association between the actors and the use cases and a generalization among the use cases.

### Actor:

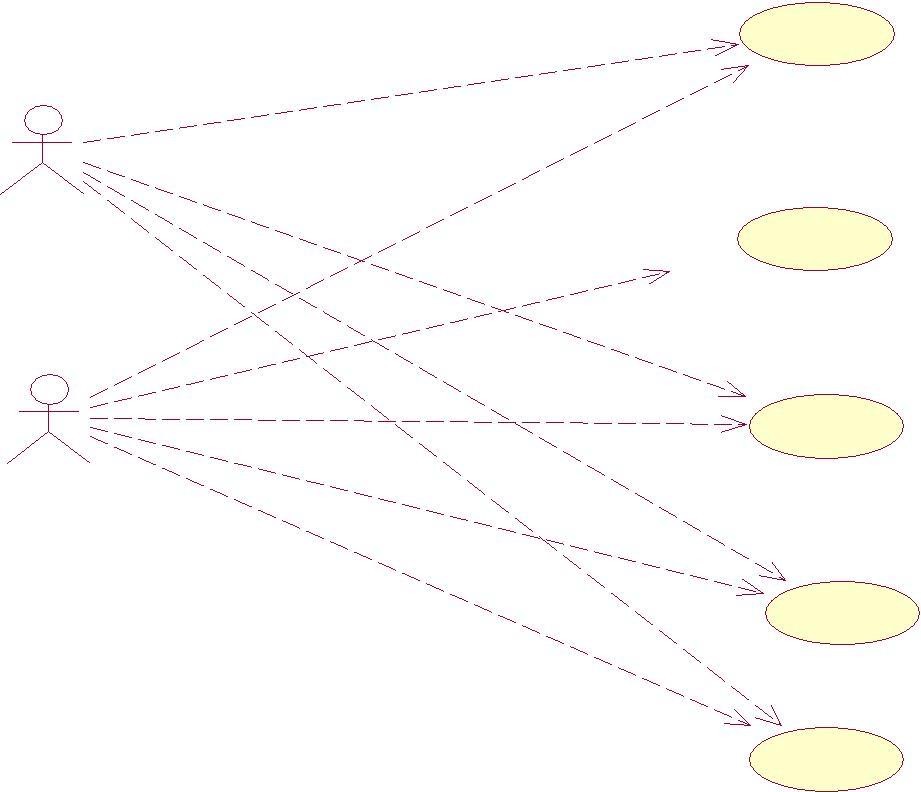
An actor represent a set of roles that user of a use case play when interacting with the use cases. Actor identified here is the user and banker.

### Use case:

A use case is a description of a set of sequence of actions that a system performs to yield result of value to an actor.

The Use Cases here it describes,

* + Customers insert a card and verify it. Then he selects the type of transaction.
  + BANKING performs the required transaction. After transaction completes customer takes the card.



ATM

user

Customer'sdetails

banker

deposit

loan

withdrawal

UML class diagram is the main static analysis diagram. This diagram shows the static structure of the model. This is a collection of static modeling elements, such as classes and their relationships, connected as a graph to each other and to their contents.

A class is drawn as a rectangle with three fields. The top field contains class name, the middle field contains attributes and the bottom field contains list of operations.

In our BANKING application, the classes are

1. New account
2. Deposit
3. Withdraw
4. Balance check
5. Transactions

Each class has its own attributes and properties

### New account class:

The attributes involved are name,acc.no,dob,.. The operations involved are created and cancel.

### Deposit class:

The attributes involved are account number and balance. The operations involved are add and exit.

### Withdraw class:

The attributes involved are account number and balance. The operations involved are save and cancel.

### Balance checking class:

This involves attributes like name, account number and balance amount. The operation involved is getbalance.

### Transactions:

This class involves attributes like name, account number, last transaction, date and amount in that account.

The operation involved is getdetails.

### ACTIVITY DIAGRAM:

In this, the states are activities representing the performance of operations and the transactions are triggered by the completion of the operations.This diagram provides the view of flows and what is going on inside a usecase or among several classes.

The Activity diagram is used to describe the various activities taking place in an application. Here in our BANKING SYSTEM, we have various activities starting from login.

After login the client has to select the transactions to be done. If it is deposit, then add the given amount with the balance amount and update the database.

If it is withdraw, then decrement the given amount with the balance amount and update the database.



start

check user details

select transaction to be done

get amount from user and add it with balance

get amount from user and do proper withdraw activity

produce the last transaction details

get the balance amount from the account

stop

If it is balance checking, then get the balance amount from the database.

If it is ministatement generation, then obtain all the details from the database.

### COLLABORATION DIAGRAM(ATM):

customer console

network to bank

5: select choice

9: another transaction

3: verified

8: give receipt

7: feasibility

4: choice menu

1: insert card

6: account information

2: validation

transacti on

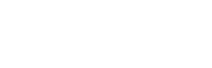
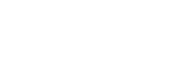
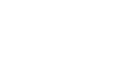
A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchanged among the objects within he collaboration to achieve a desired outcome.

llaboration diagrams show exactly same information as the sequence diagram. However, collaboration diagram show this information in a different way and with different purpose.

In this collaboration diagram, the objects are represented as rectangle, the actors are stick figures. Whereas the sequence diagram illustrates the object and actor interaction overtime, the collaboration diagram shows the object and actor interaction without reference to time.

In our BANKING SYSTEM, each object interacts with each other or collaborates with each other; it gets represented by the solid line drawn between them.

### CLASS DIAGRAM:



k

checkbalance()

accountnumber name

amount

balance chec

depositamount()

accountno amount

deposit

getdetails()

accountno name transaction date amount

ministatement

updateaccount()

accountno amount

withdraw

checkdetails() selectchoice()

name password

login

UML class diagram is the main static analysis diagram. This diagram shows the static structure of the model. This is a collection of static modeling elements, such as classes and their relationships, connected as a graph to each other and to their contents.

A class is drawn as a rectangle with three fields. The top field contains class name, the middle field contains attributes and the bottom field contains list of operations.

In our BANKING application, the classes are

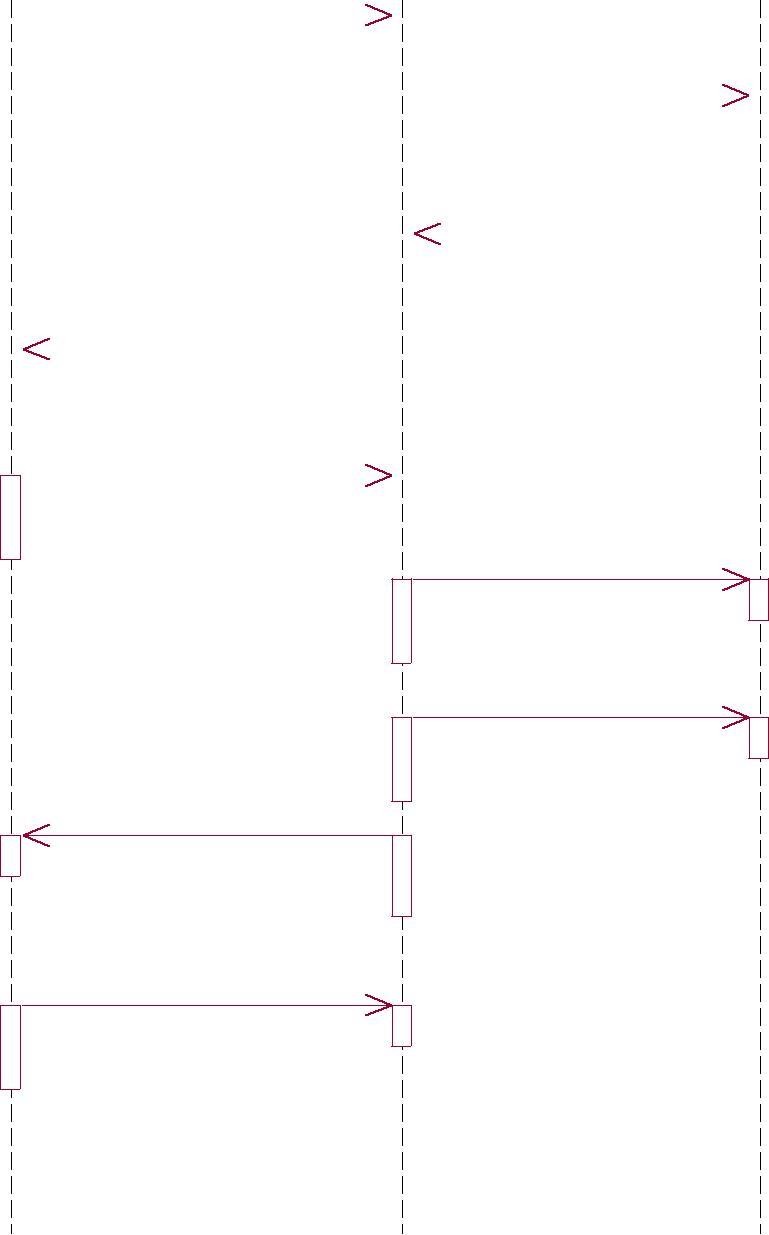
* 1. New account
  2. Deposit
  3. Withdraw
  4. Balance check
  5. Transactions

Each class has its own attributes and properties

### SEQUENCE DIAGRAM(ATM):

|  |  |  |  |
| --- | --- | --- | --- |
| customer | transaction |  | network to |
| console | 1.Insert card |  | bank |
|  |  |  |  |
|  | 2.Validation | | |
|  |  | | |
|  |  | | |
|  | 3.Verified | | |
|  |  | | |
|  | 4.Choice menu |  |  |
|  |  |  |  |
|  | 5.Transaction |  |  |

Sequence diagrams are easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and its environment. A sequence diagram shows an interaction arranged in a time sequence.



6.Account information

7.Feasibility

8.Give receipt

9.Another transaction

The objects used in this sequence diagram are, 1.customer console

1. transaction
2. network to bank

The object client enters the login information. The object login access the details from the database and enters into the application. The object control obtains the amount for the operation either it is deposit or withdraw and update them according to the balance amount.

### CODING:

**Login Form:**

Private Sub Command1\_Click()

If Text1.Text = "rv" And Text2.Text = "mk" Then banking.Show

Else

MsgBox ("Invalid input") End If

End Sub

### Banking Form:

Private Sub Command1\_Click() newaccount.Show

End Sub

### Account Form:

Private Sub crebtn\_Click() newaccountado.Recordset.Fields("Name") = txtname.Text newaccountado.Recordset.Fields("Date of Birth") = txtdob.Text newaccountado.Recordset.Fields("Gender") = txtgender.Text newaccountado.Recordset.Fields("Address") = txtadd.Text newaccountado.Recordset.Fields("Contact no") = txtphone.Text newaccountado.Recordset.Update

MsgBox "User newaccount Successful" withdraw.Show

End Sub

Private Sub Form\_Load() newaccountado.Recordset.AddNew End Sub

### Withdraw Form:

Private Sub Form\_Load() withdrawado.Recordset.AddNew End Sub

Private Sub okbtn\_Click() withdrawado.Recordset.Fields("Accountno") = txtaccno.Text

withdrawado.Recordset.Fields("Withdraw amount") = txtwithdraw.Text withdrawado.Recordset.Fields("Current Balance") = txtcurbal.Text withdrawado.Recordset.Fields("Minimum Balance") = txtminbal.Text withdrawado.Recordset.Fields("New Balance") = txtnewbal.Text withdrawado.Recordset.Update

MsgBox "User withdraw Successful" Deposit.Show

End Sub

### Deposit Form:

Private Sub addbtn\_Click() depositado.Recordset.Fields("Account no") = txtaccno.Text depositado.Recordset.Fields("Current Balance") = txtcurbal.Text depositado.Recordset.Fields("Deposit amount") = txtdepamt.Text depositado.Recordset.Fields("New Balance") = txtnewbal.Text depositado.Recordset.Update

MsgBox "User deposit Successful" Transaction.Show

End Sub

Private Sub Form\_Load() depositado.Recordset.AddNew End Sub

### Transaction Form:

Private Sub Exitbtn\_Click() End

End Sub

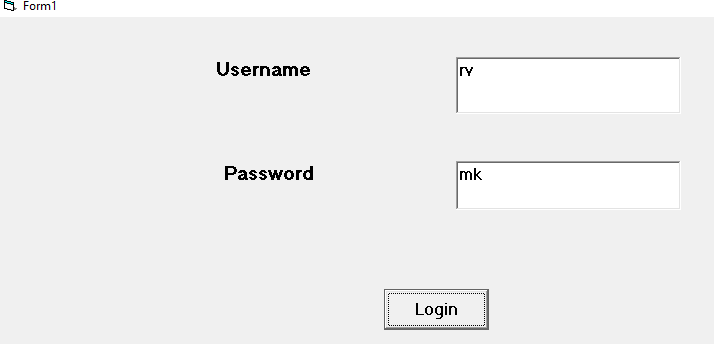
Private Sub Form\_Load() transactionado.Recordset.AddNew End Sub

Private Sub transactionbtn\_Click() transactionado.Recordset.Fields("Account no") = txtaccno.Text transactionado.Recordset.Fields("Receiver Account no") = txtrecaccno.Text

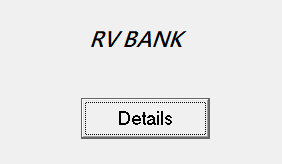
transactionado.Recordset.Fields("Current Balance") = txtcurbal.Text transactionado.Recordset.Fields("Minimum Balance") = txtminbal.Text transactionado.Recordset.Fields("transaction Amount") = txttransamt.Text transactionado.Recordset.Update

MsgBox "User transaction Successful" End Sub

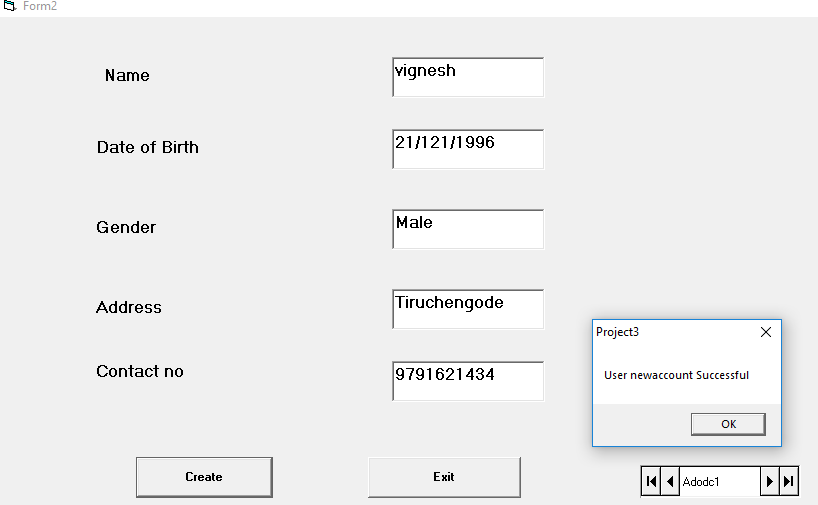
### Output: Form1:



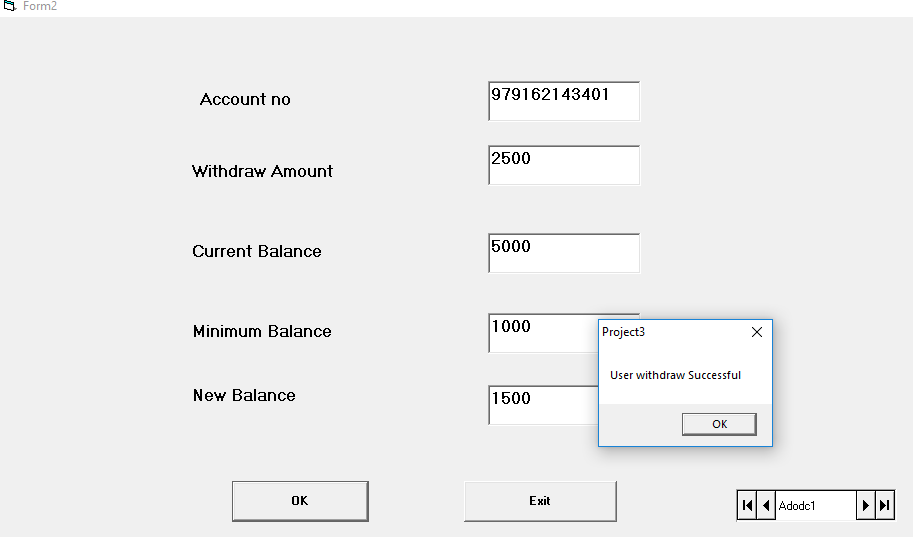
**Form2:**



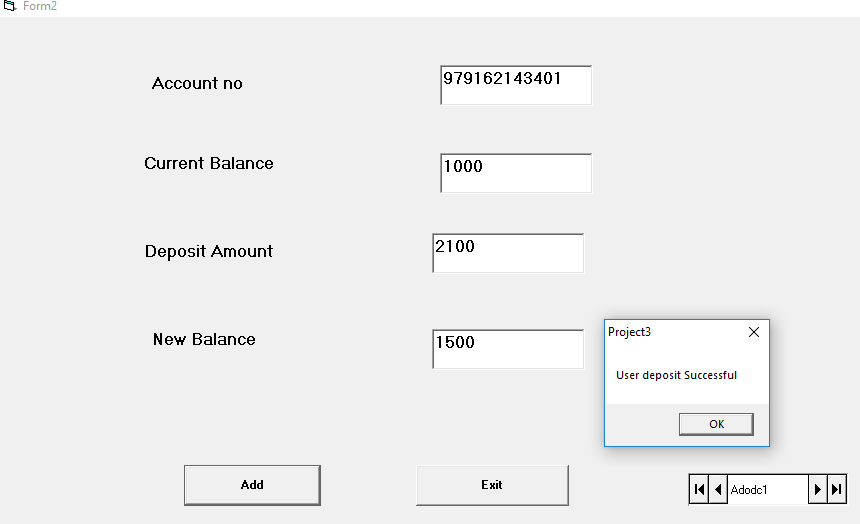
**Form3:**



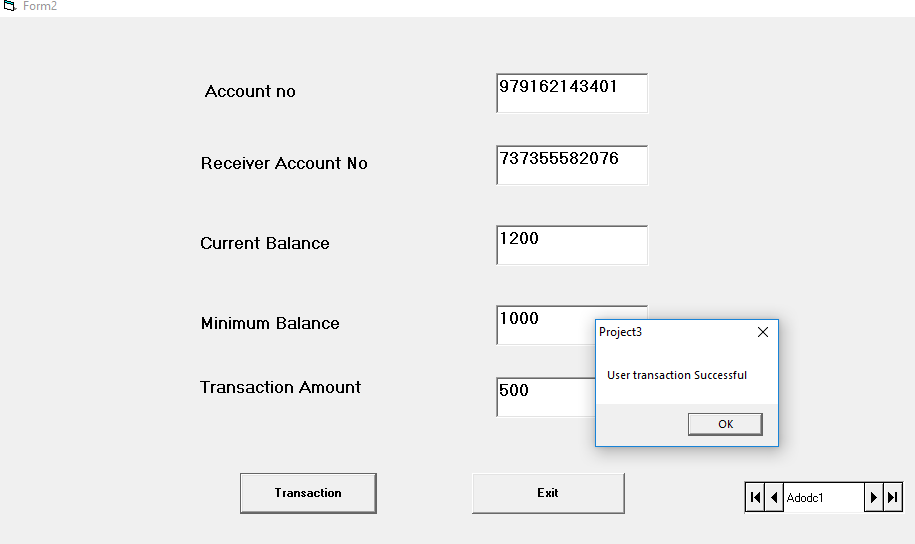
Form4:



### Form5:

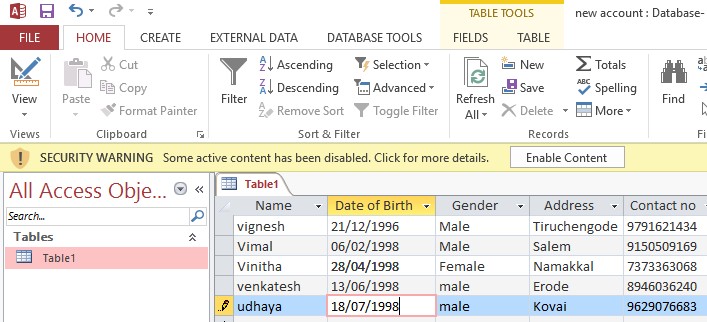


**Form6:**

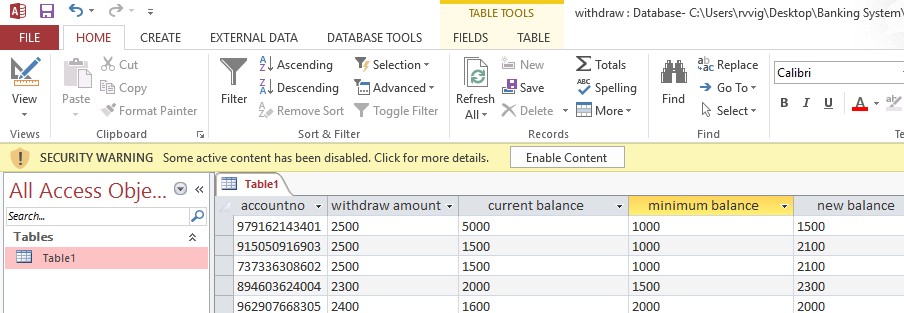


**Database:**

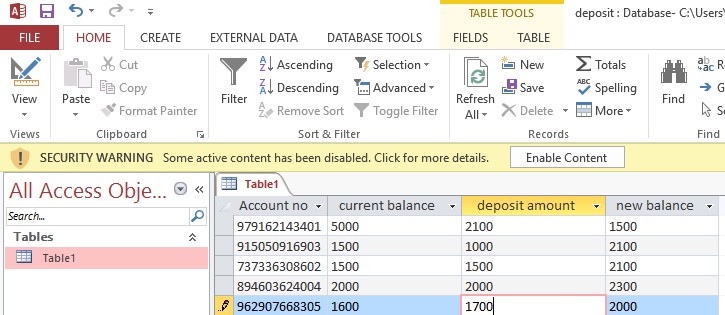
**New Account Form**



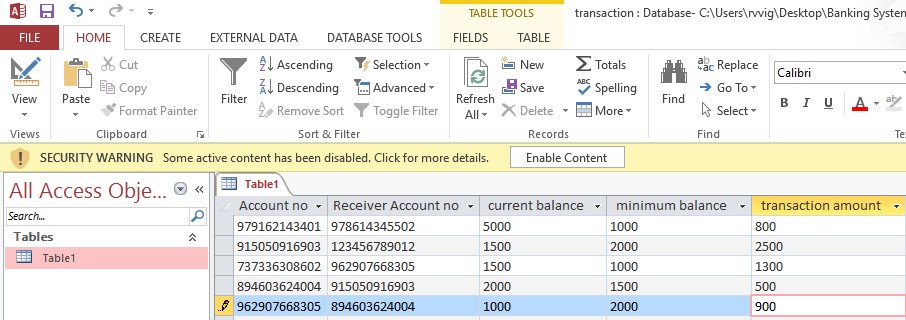
**WithDraw Form**



**Deposit Form**



### Transaction Form



**Result:**

Thus the **BANKING SYSTEM** is developed with all necessary documents and UML diagrams using Software Engineering methodology.

### EX.NO:4 DATE :

**TEXT EDITOR**

**AIM**

To prepare necessary documents and to develop the “Text Editor” with usecase diagrams using software engineering methodology.

### PROBLEM STATEMENT

The project Text Editor is to develop an application for allows the user to type the text with different font styles, type, size and vibrant colors. It contains facility to save the typed text as document and to open the existing document for reading and writing.

### SOFTWARE REQUIREMENT ANALYSIS

1. Editor Form
2. Font Style Frame
3. Save Frame
4. Open Frame

The Editor form contains the components that allow the user to type the text and the menu bar that provides various options.

form.

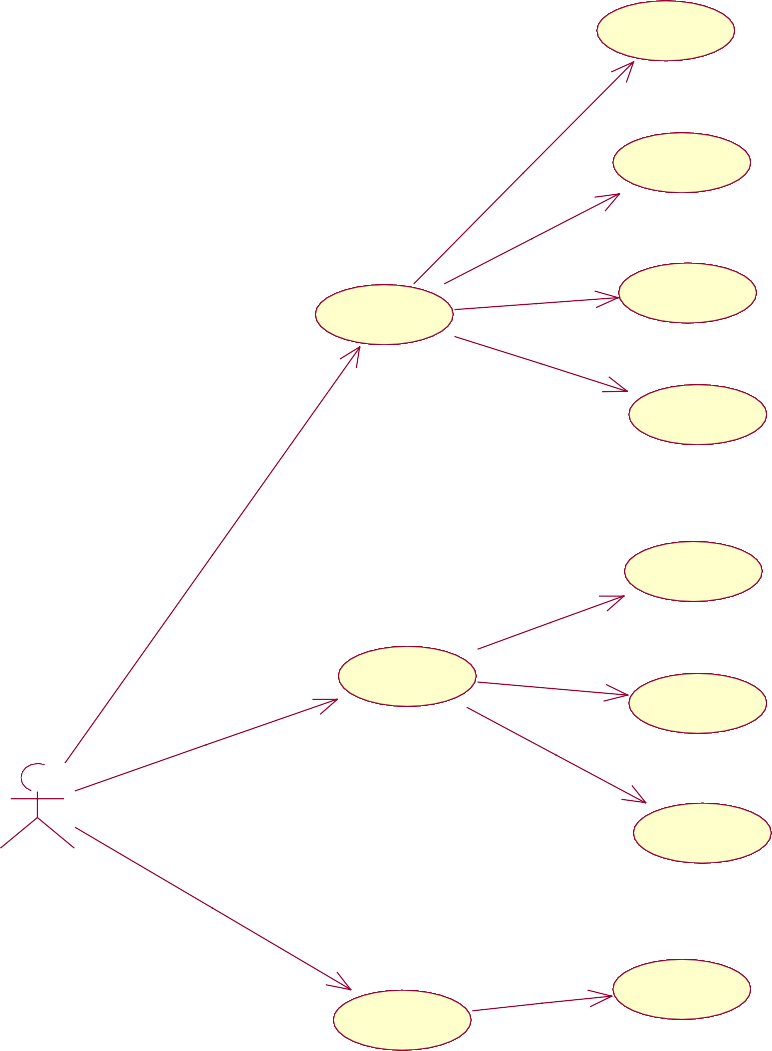
The Font style, Save and Open frames are the members of the Editor

The Font Style frame allows the user to enrich their document with

different font color, style, type and size of the text.

The Save frame allows the user to save their typed text as document and the Open frame allows the user to open the existing document for reading and writing.

### USE CASE DIAGRAM

new

open

file

save

exit

cut

edit

copy

user

paste

font size

format

font style

### Use case Diagram:

Use case diagram is a graph of actors, set of use cases enclosed by a system boundary, communication (participation) association between the actors and the use cases and a generalization among the use cases.

### Use case:

A use case is a description of a set of sequence of actions that a system performs to yield result of value to an actor.

The Use Cases described are,

* 1. New File
  2. Type Text
  3. Format Text
  4. Change Style
  5. Open, Save File

The File use case is to describe that, the user should Open and Save the files or create a new File.

### SEQUENCE DIAGRAM:



: User

1: Enter as user

2: New

3: Save

5: Copy

6: Cut

7: Paste

8: Font Size

9: Fon

10: Exit

Storage

t Style

4: Open

Edit

Workspace

Format

File

Sequence diagrams are easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and its environment.A sequence diagram shows an interaction arranged in a time sequence.

The objects used in this sequence diagram are,

* + 1. File
    2. Workspace
    3. Edit
    4. Format
    5. Storage

### BASIC FLOW:

The user opens the application and application allows the user to type

the text.

### ALTERNATE FLOW:

The object File makes the user to select new file or to open the existing file and to save the file.

The object Workspace allows the users to type text, from the workspace the users access all other object.

The object Edit used to cut, copy and paste the selected text.

The object Format allows the user to change the font style, color and its

type.

The object Storage is to store the typed text in the disk as file.

### COLLOBORATION DIAGRAM

A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchanged among the objects within the collaboration to achieve a desired outcome.

Collaboration diagram shows exactly the same information as the sequence diagram. However, collaboration diagram shows this information in a different way and with different purpose.

In this collaboration diagram, the objects are represented as rectangle, the actors are stick figures. Whereas the sequence diagram illustrates the object and actor interaction overtime, the collaboration diagram shows the object and actor interaction without reference to time.

2: editmenu 3: copy/text 5: paste



edit

paste

copy

20.formatmenu

format

|  |  |
| --- | --- |
| : u | ser |
|  | 1: |
| file | |
| ew  .crea o.o..pe | 1  tefil n file |
| controlle r | |

filename

4: fontstyle

16 displayfile

font

13: displayfile

10: savefile

9: savefile

12: n

display

17

1: open e

6: entertext

save

7: savefile

8: savefile

18

14: read/write

19.sendtext 15.sendtext

In our TEXT EDITOR each object interacts with each other or collaborates with each other; it gets represented by the solid line drawn between them.

### ACTIVITY DIAGRAM

The activity diagram describes the sequencing of activities with support for both conditional and parallel behavior.

The Activity diagram is used to describe the various activities taking place in an application. Here in our TEXT EDITOR, we have various activities starting from File.

After user opens the application he/she choose file menu to create new file or to open the existing file then user read/write. User chooses the Edit menu to cut, copy and paste the selected text.

Menu Selection

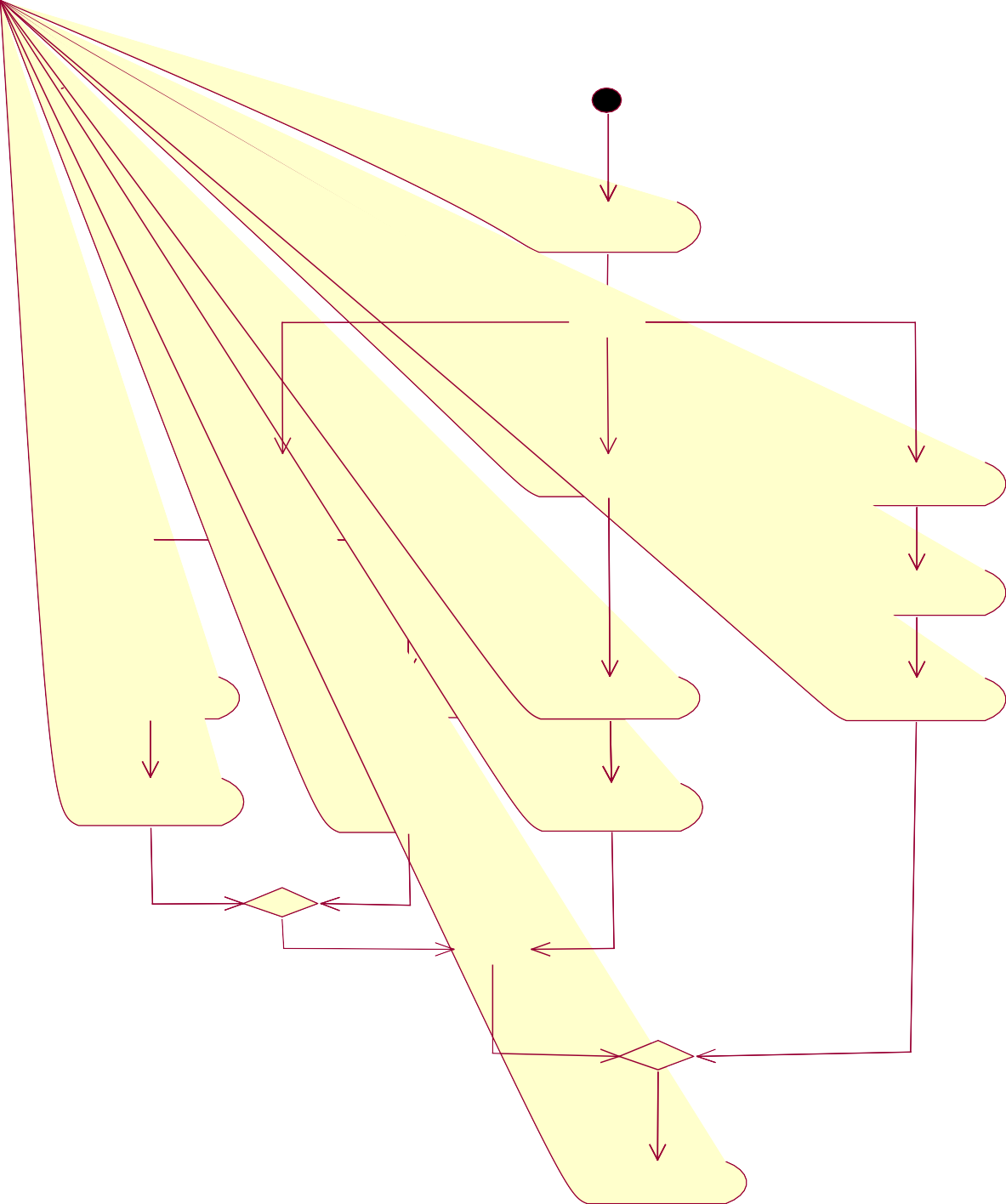
File Edit Format

Font Size

New Open

Cut/Copy

Font Style



Type Text Read/Write Paste

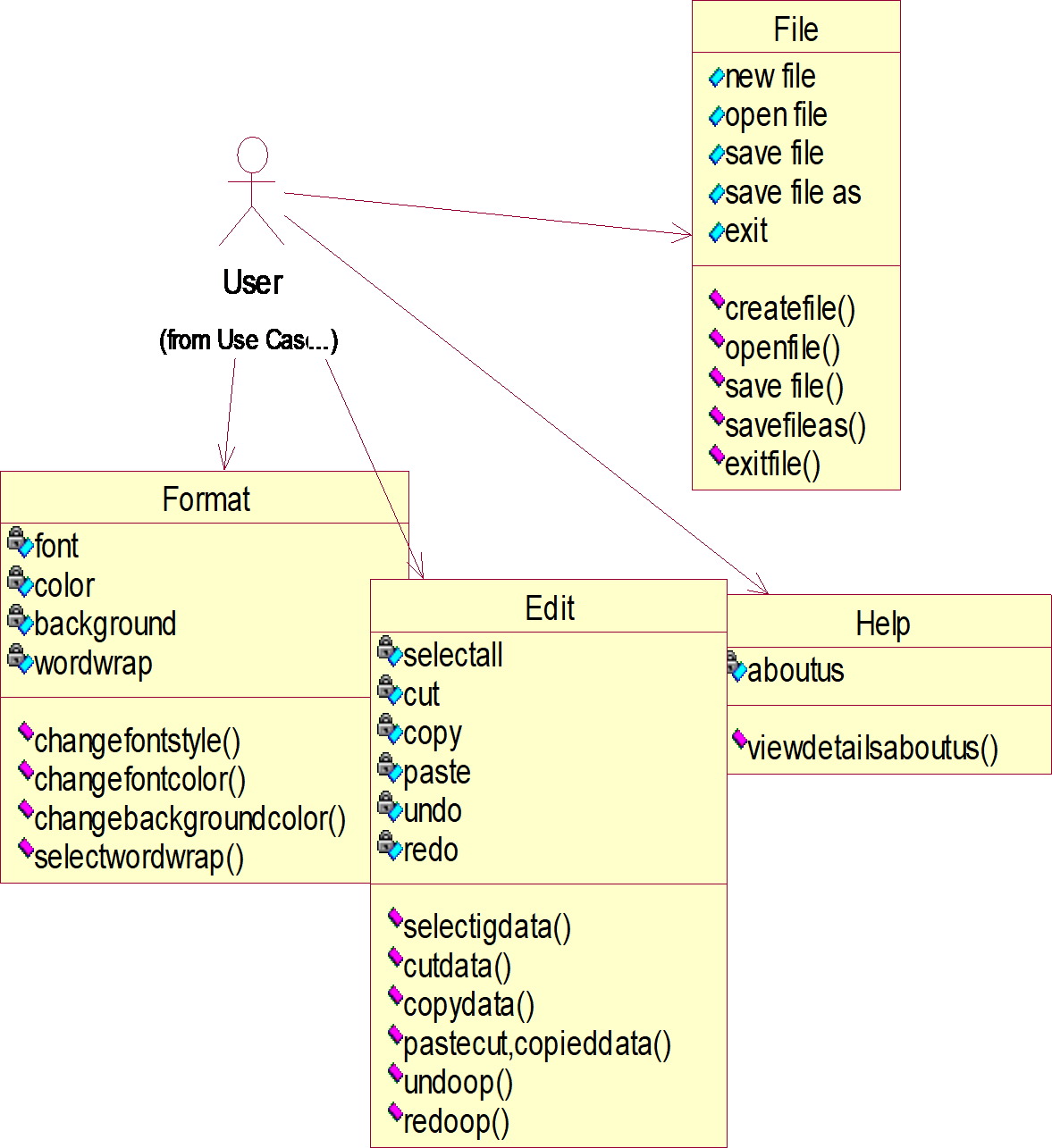
Save

By choosing Format the user can change the font style, color and its type. The object Storage is to store the typed text in the disk as file.

### CLASS DIAGRAM:

Class diagrams show the interactions between classes in the system. Class diagram also shows the attributes and operation of a class and the constraints that apply to the way objects are connected.

Classes contain information and behavior that acts on that information. Each class on class diagram is represented by rectangle divided into three sections. The first section shows the class name, second section shows the attributes the class contains and last section contains the operation of the class.



### SOFTWARE DEVELOPMENT

**Fore color:**

Private Sub mnuTxtColor\_Click() cd.ShowColor FrmChild.TextBox.ForeColor = cd.Color End Sub

### Background Color:

Private Sub mnuBackColor\_Click() cd.ShowColor FrmChild.TextBox.BackColor = cd.Color End Sub

### Fonts:

Private Sub mnuFont\_Click() cd.Flags = cd1CFBoth cd.ShowFont

FrmChild.TextBox.Font = cd.FontName FrmChild.TextBox.FontItalic = cd.FontItalic FrmChild.TextBox.FontBold = cd.FontBold FrmChild.TextBox.FontSize = cd.FontSize End Sub

### Exit:

Private Sub mnuExit\_Click() End

End Sub

### SOFTWARE TESTING:

**TEST CASES**

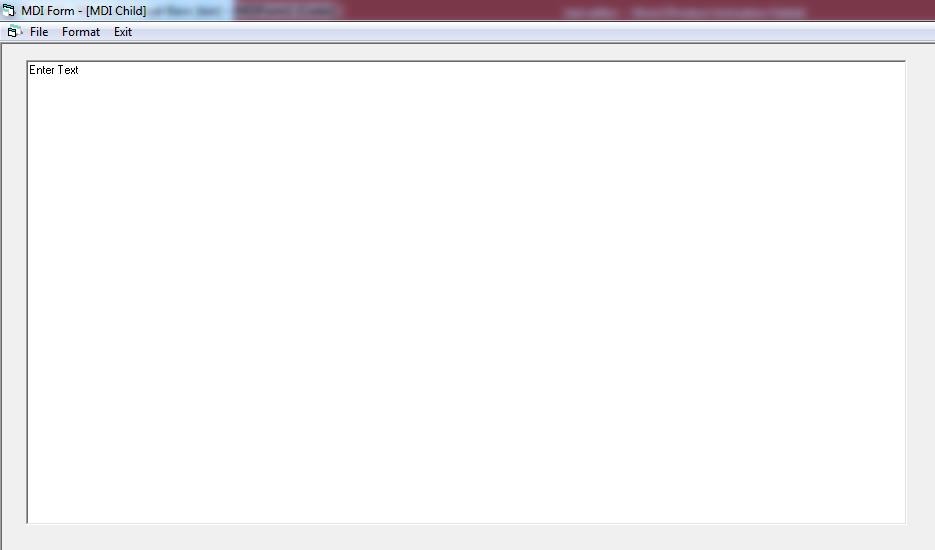
The sample text is typed and its font style, font color and font type are changed.

The new file is created and the typed text is stored in disk as file. The existing file is opened and it allows to read or write the file content.

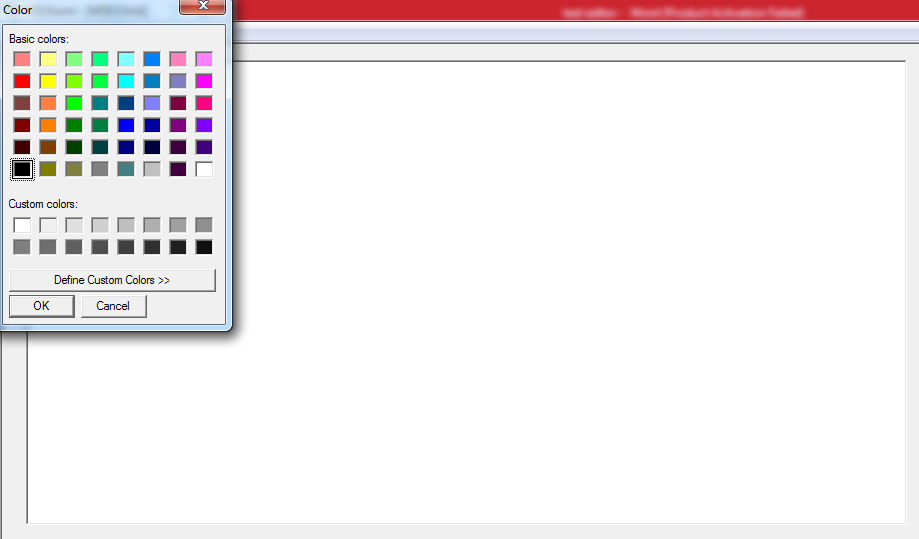
The developed application gets tested with the sample inputs.

**OUTPUT:**

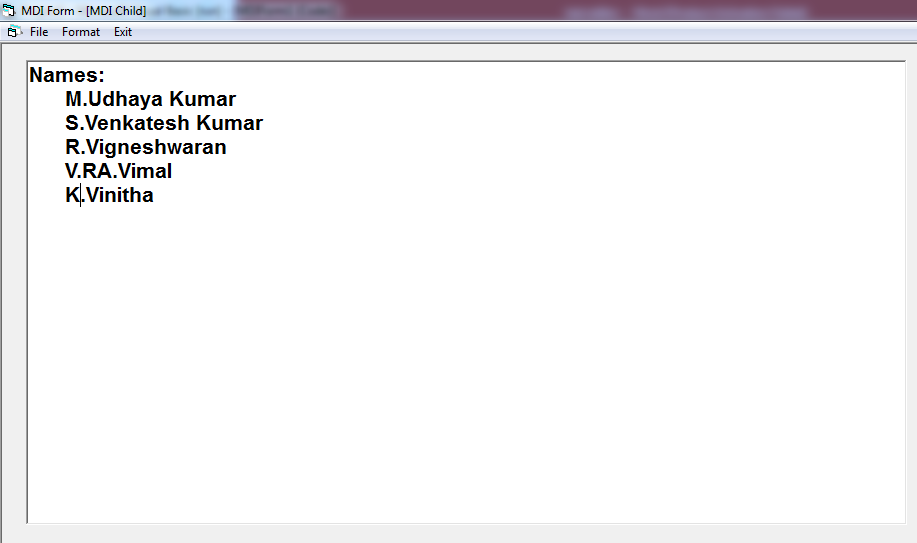
### Form1:



**Form2:**



**Form3:**



**RESULT:**

Thus the Text Editor is developed with all necessary documents

and usecase diagrams using software engineering methodology.

### EX. NO:5 DATE :

**ONLINE VOTING SYSTEM**

**AIM:**

To prepare necessary documents and to develop the project **Course Registration** S**ystem** with UML diagrams using Software Engineering Methodology.

### PROGARM ANALYSIS AND PROJECT PLANNING

This project **COURSE REGISTRATION SYSTEM** is to develop an application to make the student registration in computer centers. First the students select their needed course from the available courses. Then submit their details with their qualification. If there is wanted course available we allocate a seat and reduce one from that. These functions are done in student login. In administration login the authorized person can do the operations of the delete, update and search.

### SOFTWARE REQUIREMENT ANALYSIS

**The Modules in the Project:**

1. Login student 1.1 course detail 1.2 Student details
2. Login Administration
   1. view
   2. update
   3. exit

The student login provides the options are course details.

In this login a student can choose the given course and fill up the student details for registration such as

1. Name
2. Gender
3. Place
4. Date of birth
5. Qualification

In the administration login ,the management uses it for the purpose of making an admission etc. They are

1. Amount
2. Payment details
3. Student details
4. Name of the bank

Some function in the administration can be given below:

1. View -helps in viewing the student details.
2. Update -helps in updating the latest function.admission.
3. Exit -helps in the normal exit

### SYSTEM SPECIFICATION SOFTWARE REQUIRMENTS: DATA DICTIONARY:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Description | Data | Field Size | Default | Validation |
| Type | Value |
| Name | Name of the | string | 20 | Null | (A-Z)or(a-z) |
| Student |
| Gender | Gender of | string | 10 | Null | (A-Z)or(a-z) |
| the student |
| Place | Address of | string | 20 | Null | (A-Z)or - |
| the student |
| Dob | Date of birth | integer | 03 | Null | - |
| of a student |
| qualification | Course | string | 50 | Null | - |
| studied by |
| the student |

**HARDWARE REQUIRMENTS:**

o Operating system:windows xp

* o Front end:Microsoft Visual Studio 2005
  + Back end:MS-Acess
* o Processor:Intel pentium IV
* o RAM:512MB
* o Harddisk:80GB
* o Monitor:17”LCD
* o Keyboard:multimedia keyboard
  + Mouse: USB optical mouse

## USE CASE DIAGRAM:



login

user

course detail

register

Admin

view

update

**Use Case Diagram:**

Use case diagram is a graph of actors, set of use cases enclosed by a system boundary, communication (participation) association between the actors and the cases and a generalization among the use cases.

### Actors:

An actor represents a set of roles that user of a use case play when interacting with the use cases. Actor identified here is Staff, Student and Administrator.

### Use case:

A use case is a description of a set of sequence of action that a system performs to yield result of value to an actor.

The Use Cases described are

* 1. Login
  2. Course Details
  3. Register
  4. View
  5. Update

1. The Login use case is to describe that, the user should choose his/her category whether he/she is user or administrator.
2. The use case display the course details and the offered with their instruction and also about the payment.
3. The use case collects the information about the user such as name, address, contact no, E=mail id, etc, and stored the details in the data base
4. The View Details use case describes that the administrator views the details about the user and update the payment details in the registration form.
5. The update details are confirmation details of user requisition to the course registration, and he is allowed to attain the course.

### SEQUENCE DIAGRAM FOR USER:

user

login

course detail

database

enter password

verify password

request c

send available course

select course

register

conformation(reff no)

ourse detail

**SEQUENCE DIAGRAM FOR ADMIN:**

admin

login

database

enter password

verify password

enter reff no

display student detail

update check detail

Sequence diagrams are easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and its environment. A sequence diagram shows an interaction arranged in a time sequence. Here we use two sequence diagrams one is for administrator and another is for user.

The objects used in this sequence diagram of user are,

1. Student
2. Login
3. Course Detail
4. Register

The student is the course selector. They first select their course and fill the registration form and then submit. After submitting the administrator modifies and updates the registration.

The objects used in this sequence diagram of administrator are,

* 1. Admin
  2. Login
  3. View
  4. Update

### BASIC FLOW:

The user has to login and fill up the registration form for their course need and submit it. For each registration a reference number is given. The administrator views the details with that reference number and he updates the details with additional information.

### COLLABORATION DAIGRAM: COLLABRATION DIAGRAM FOR USER

1: enter password

2: verify password

uest course detail select course

7: conformation(reff no)

6: register

database

login

|  |  |  |  |
| --- | --- | --- | --- |
|  | user | |  |
| abl  ecours | | 3:req  5:  e | |
| course detail | | | |
|  | | | |

4: send avail

A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchanged among the objects within the collaboration to achieve a desired outcome.

Collaboration diagram shows exactly the same information as the sequence diagram. However, collaboration diagram shows this information in a different way and with different purpose.

In this collaboration diagram, the objects are represented as rectangle, the actors are stick figures, whereas the sequence diagram illustrates the object and actor interaction overtime, the collaboration diagram shows the object and actor interaction without reference to time.

In our course registration, each object interacts with each other or collaborates with each other; it gets represented by solid line drawn between them.

### COLLABRATION DIAGRAM FOR ADMINSTRATOR

4: display studen

1: enter password

2: verify password

login

|  |  |
| --- | --- |
| admin | |
| et tdail | 5: |
| database | |

3: enter reff no update check detail

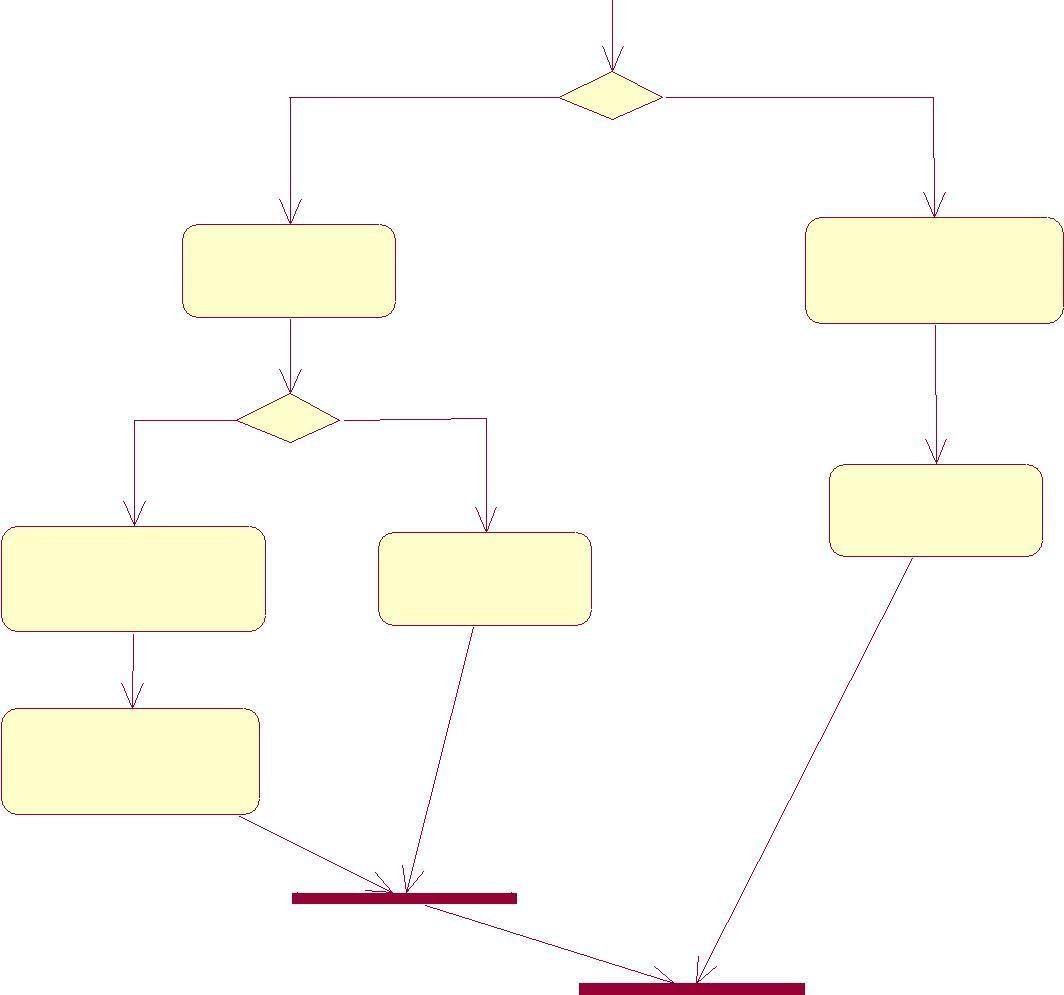
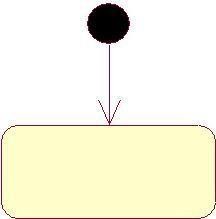
### ACTIVITY DIAGRAM:

|  |  |
| --- | --- |
| admin | user |

|  |  |
| --- | --- |
| enter ref. | view course |
| no | detail |

|  |  |
| --- | --- |
| view student | cannot |
| Details | view detail |

The activity diagram describes the sequence of activities with support for both conditional and parallel behavior.The activity diagram is used to describe the various activities taking place in an application.Here in our COURSE REGISTERATION SYSTEM,we have various activities strating from login.



login

register

update check detail

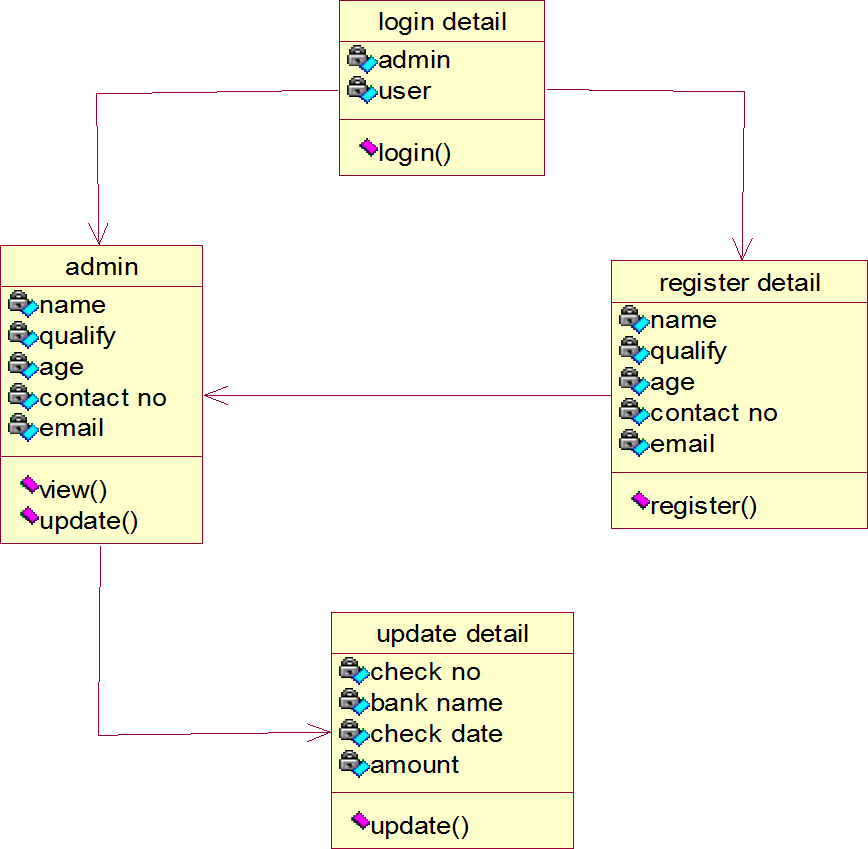
After login,the user selection activity gets performed,where the user can be a user or admin.

If the user is a student,then they have to enter their password and those details are valid they can access the system.They can register the any course and get the reference number.

### CLASS DIAGRAM:



|  |
| --- |
|  |
|  |
|  |

The class diagram involves various classes used in project and their attributes.It also explains various class members details.

The four class in this project are,

1. Login detail
2. Admin detail
3. Register detail
4. Update detail

Each class has its own attributes and operations. Login class-the attributes defined is user and admin

Admin class-It contains the attributes of the student and own details. Register class-It contains the attributes of the student

Update class-the attributes of students can be added to the admin class

### SOFTWAREDEVELOPMENT

**Login Form:**

Private Sub Command1\_Click()

If Text1.Text = "rv" And Text2.Text

= "mk" Then coursedetail.Show

Else

MsgBox ("Invalid input") End If

End Sub

### Course Detail:

Private Sub Command1\_Click() If Option1.Value = True Then instructions.Show

ElseIf Option2.Value = True Then instructions.Show

ElseIf Option3.Value = True Then instructions.Show

ElseIf Option4.Value = True Then instructions.Show

End If End Sub

### Instruction Form:

Private Sub Command1\_Click() If Button = Click\_here Then registration.Show

End If End Sub

Private Sub Command2\_Click() If Button = Back Then coursedetail.Show

End If End Sub

### Registration Form:

Private Sub Form\_Load() registerado.Recordset.AddNew End Sub

Private Sub regbtn\_Click() registerado.Recordset.Fields("Rolln o") = txtroll.Text registerado.Recordset.Fields("Name ") = txtname.Text registerado.Recordset.Fields("Class ") = txtclass.Text registerado.Recordset.Fields("Addr ess") = txtadd.Text registerado.Recordset.Fields("mail"

) = txtmail.Text registerado.Recordset.Fields("Conta ct") = txtphone.Text registerado.Recordset.Update MsgBox "User Registration Successful"

report.Show End Sub

### Exit Form:

Private Sub Command1\_Click() End

End Sub

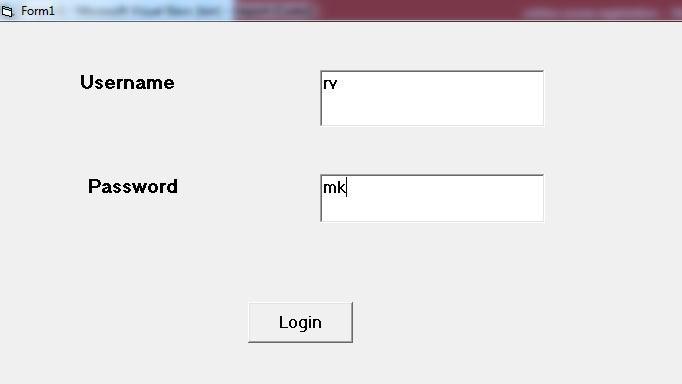
### SOFTWARE TESTING:

**TEST CASES:**

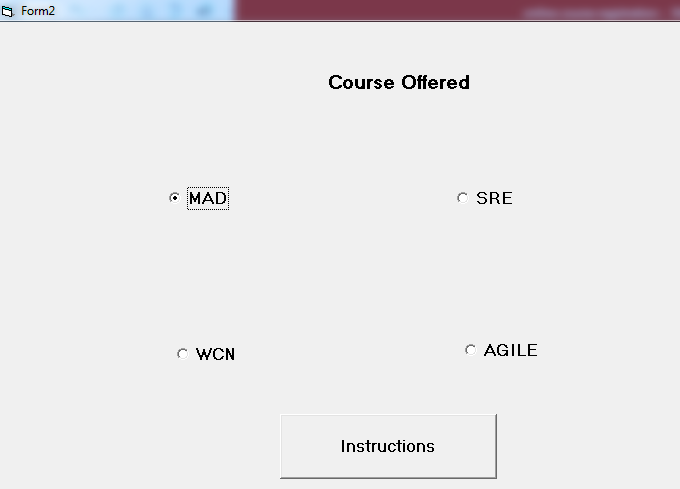
* + If an unauthorized user attempts to access the system, the system should not allow them to access.
  + If the password entered by the user is incorrect, the system should display the bad password message and allows them to reenter the password or to terminate the process.

### OUTPUT :

**Form1:**



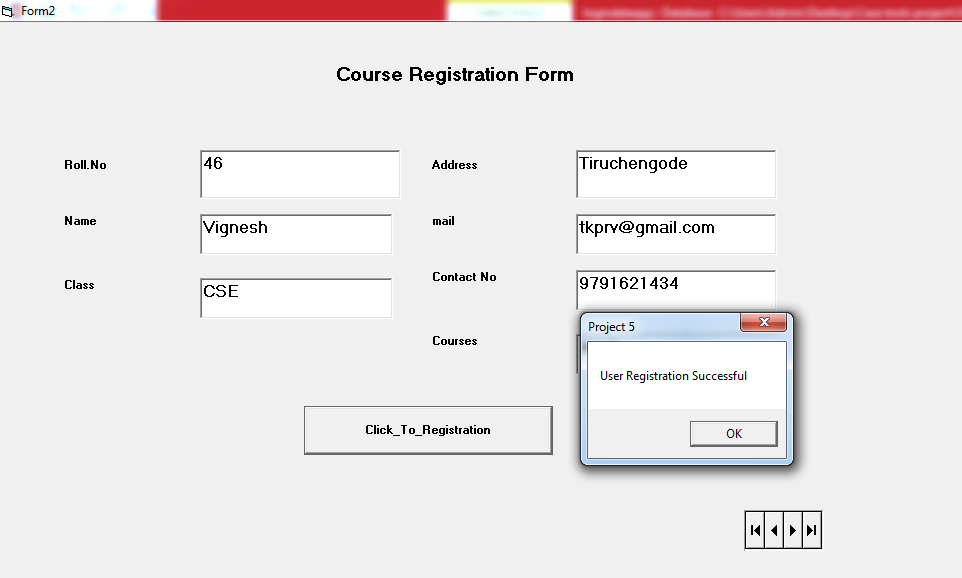
**Form2:**



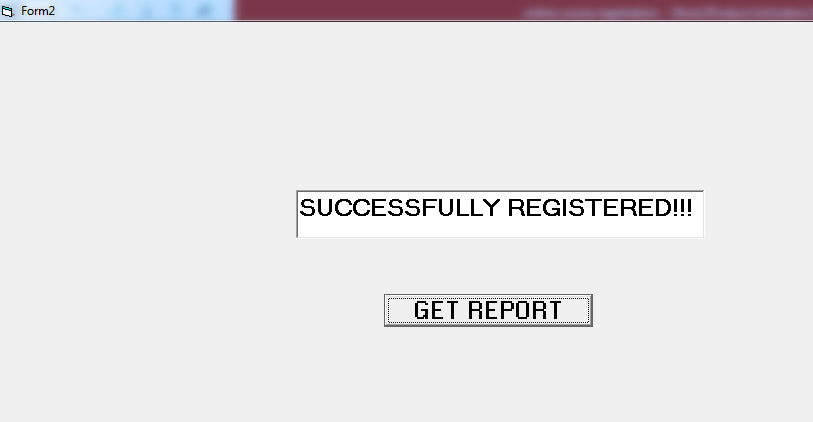
**Form3:**



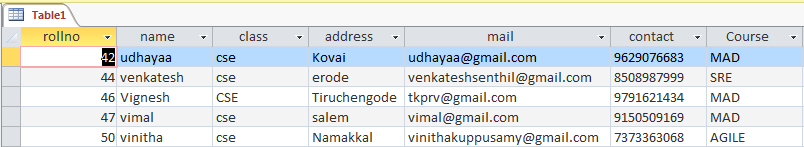
**Form4:**



**Form5:**



**Database:**



**RESULT:**

Thus the **Online Voting** as **Course Registration System** is developed with all documents and UML diagrams using Rational Rose software Engineering methodology.

### EX.NO:6 DATE :

**LIBRARY MANAGEMENT SYSTEM**

**Aim:**

To prepare necessary documents and to develop the LIBRARY

MANAGEMENT SYSTEM with UML diagrams using Software Engineering Methodology.

### PROGRAM ANALYSIS AND PROJECT PLANNING

**Problem Statement:**

This project LIBRARY MANAGEMENT SYSTEM is to develop an application to analyze the books details. Initially the user should login. After login to lend the book the user should click book lending command button. If the user wants to return the book then must choose the book returning command button. If the user wants to view the details of book lending and returning history then must click the report command button.

### SOFTWARE REQUIREMENT ANALYSIS

**The Modules in the Project:**

1. Login.
2. Password Checking
3. Book lending
4. Book returning
5. Report

The first module is Login in which the user has to login to the system.

The next module is Password Checking, where the checking of name and password occurs if correct then only the user can lend or return the book.

The next module is Book lending. This module is useful to lend the book from library.

The next module is Book returning. This module is useful to return the book to the library.

The last module is Report. This module is to view the history.

### DATA MODELING

**Data Dictionary:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Description | Data | Field | Default | Validation |
| Name | Type | size | Value |
| ID.NO | Identification Number | Integer | 10 | NULL | (0-9) |
| Name | Name of the user | String | 20 | NULL | (A-Z) or (a-z) |
| Date | Date | Integer | 10 | NULL | (MM/DD/YYYY) |
| Author | Name of the author | String | 10 | NULL | (A-Z) or (a-z) |
| Book | Name of the book | String | 10 | NULL | (A-Z) or (a-z) |

**USE CASE DIAGRAM:**



sign in

student

issue books

take books

admin

return books

search books

book details



**Use case Diagram:**

Use case diagram is a graph of actors, set of use cases enclosed by a system boundary, communication (participation) association between the actors and the use cases and a generalization among the use cases.

### Actor:

An actor represent a set of roles that user of a use case play when interacting with the use cases. Actor identified here is user.

### Use case:

A use case is a description of a set of sequence of actions that a system performs to yield result of value to an actor.

The Use Cases described are,

* 1. Login
  2. Book lending 3.Book returning

4. Report

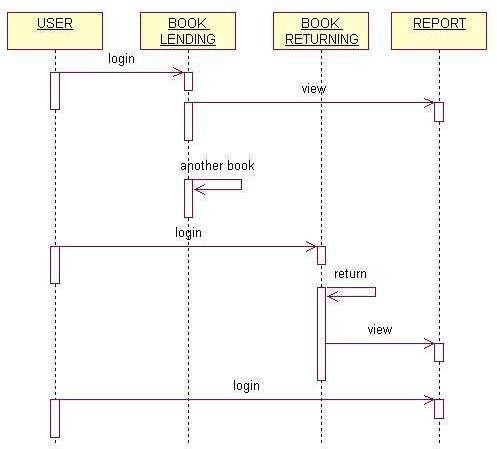
The Login use case is to describe that, the user should choose his/her category whether he/she is a administrator or staff.

### SEQUENCE DIAGRAM

Sequence diagrams are easy and intuitive way of describing the behavior of a system by viewing the interaction between the system and its environment. A sequence diagram shows an interaction arranged in a time sequence.

The objects used in this sequence diagram are,

1. Login
2. Display
3. Add
4. Update
5. Controller
6. Database



### BASIC FLOW:

The administrator enters their name and password, and the password gets checked by the system. After confirmation of the password the system allows them to access.

### ALTERNATE FLOW:

If the password entered by the administrator is invalid then they has to reenter or quit from the process.

The actor Administrator and Staff are the persons who interacts with the system.

The object Login makes the administrator/staff to enter.

The object Database will store all the salary of the staff. This will also store the details of new records added by the administrator. It will also update the values changed by the administrator.

The object Display will display the salary of the staffs from the database. The object Add will add the new records in the database.

The object Update will update the details in the database.

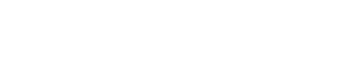
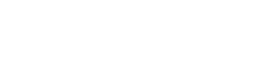
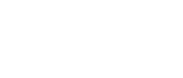
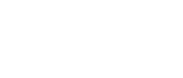
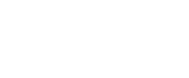
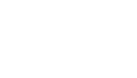
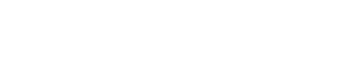
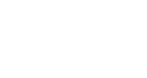
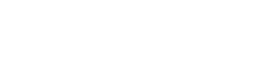
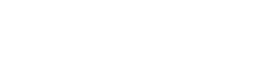
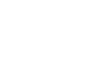
### COLLOBORATION DIAGRAM:

A collaboration diagram represents a collaboration, which is a set of objects related in a particular context, and interaction, which is a set of messages exchanged among the objects within the collaboration to achieve a desired outcome.

Collaboration diagram shows exactly the same information as the sequence diagram. However, collaboration diagram shows this information in a different way and with different purpose.

In this collaboration diagram, the objects are represented as rectangle, the actors are stick figures. Whereas the sequence diagram illustrates the object and actor interaction overtime, the collaboration diagram shows the object and actor interaction without reference to time.

In our LIBRARY MANAGEMENT SYSTEM each object interacts with each other or collaborates with each other; it gets represented by the solid line drawn between them.



**user**

**6: store data**

**1: Enter as a user**

**7: Send details of book**

**5: send data**

**2: send data**

**4: finds details of book**

**3: Display details**

**8: Display Details**

**display**

**controller**

**login**

**database**

### ACTIVITY DIAGRAM:

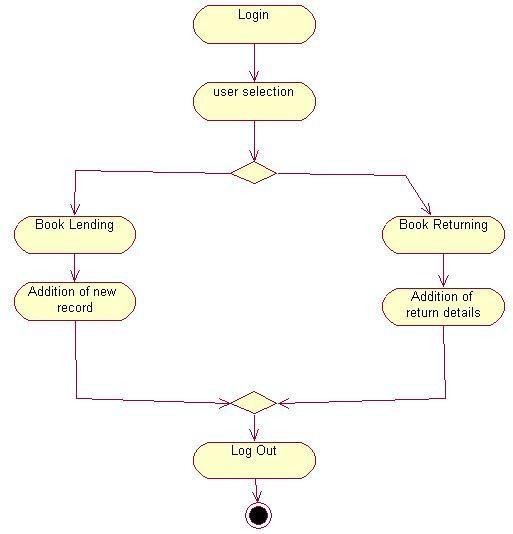
The activity diagram describes the sequencing of activities with support for both conditional and parallel behavior.

The Activity diagram is used to describe the various activities taking place in an application.

After login, the user selection activity gets performed, where the user can be an administrator or staff. If the user is a administrator, then they have to enter their name and password and only when those details are valid they can access the system.

They can calculate the current salary obtained by the staffs, they can add new records, and they can update the values of the records which gets stored in the database.

If the user is a staff then they can view their salary detail and they can calculate their current salary.

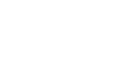
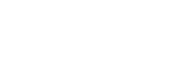
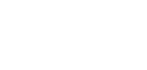
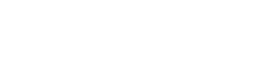
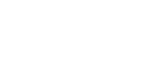
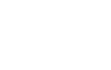
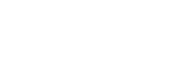
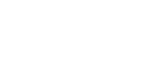
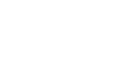
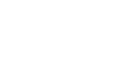
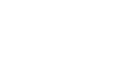
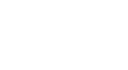


### CLASS DAIGRAM:

Class diagrams show the interactions between classes in the system. Class diagram also shows the attributes and operation of a class and the constraints that apply to the way objects are connected.

Classes contain information and behavior that acts on that information.

Each class on class diagram is represented by rectangle divided into three sections.



1

1

1

1

**Display()**

**Add() Store()**

1

1

**Display ()**

**Add ()**

**Display ()**

**ID.no Name Book Name**

**Author Name date**

**Id.no Name Password**

**ID.no Name Book Name**

**Author Name Date**

**Display**

**Controller**

**Database**

1

1

**Add new record ()**

**Login ()**

**ID.no Name Book Name**

**user**

**Add**

**Login**

The first section shows the class name, second section shows the attributes the class contains and last section contains the operation of the class.

In our LIBRARY MANAGEMENT SYSTEM, the classes identified are

1. Login
2. Display
3. Add
4. Controller
5. Database

Each class has its own attributes and operations.

Login class - The attributes defined is user.

The method identified is login.

Display class - The attributes are name, id, book name, author name, date.

The operation identified is Display.

Add class - The attributes are name, id, book name, author name, date.

The operation defined is adding new record.

Controller class - The attributes it has is id.no, name, and password.

The operations carried out by this class are added, display.

The Solid line between the classes shows the Association relationship between them.

### SOFTWARE DEVELOPMENT

**Login Form:**

Private Sub Command1\_Click()

If Text1.Text = "library" And Text2.Text = "cse" Then books.Show

Else

MsgBox ("Invalid input") End If

End Sub

### Lending Books:

**Private Sub Command1\_Click()**

lendingbooks.Show End Sub

**Private Sub Form\_Load()** lendingbooksado.Recordset.AddNew End Sub

**Private Sub Conbtn\_Click()** lendingbooksado.Recordset.Fields("Regno") = txtreg.Text lendingbooksado.Recordset.Fields("Name") = txtname.Text lendingbooksado.Recordset.Fields("Book name") = txtbook.Text lendingbooksado.Recordset.Fields("Author name") = txtauthor.Text lendingbooksado.Recordset.Fields("Date of lending books") = txtlendingbooks.Text

lendingbooksado.Recordset.Update MsgBox "User lendingbooks Successful" returnbooks.Show

End Sub

### Return Books:

**Private Sub Form\_Load()** returnbooksado.Recordset.AddNew End Sub

**Private Sub Confbtn\_Click()** returnbooksado.Recordset.Fields("Regno") = txtreg.Text returnbooksado.Recordset.Fields("Name") = txtname.Text returnbooksado.Recordset.Fields("Book name") = txtbook.Text returnbooksado.Recordset.Fields("Author name") = txtauthor.Text

returnbooksado.Recordset.Fields("Date of Return Books") = txtreturnbooks.Text returnbooksado.Recordset.Update

MsgBox "User returnbooks Successful" End Sub

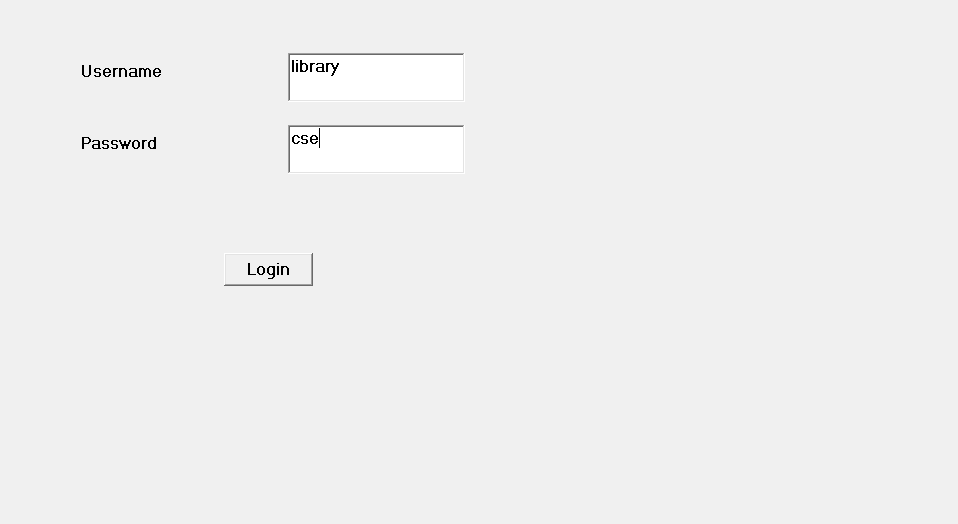
### Exit:

**Private Sub exitbtn\_Click()**

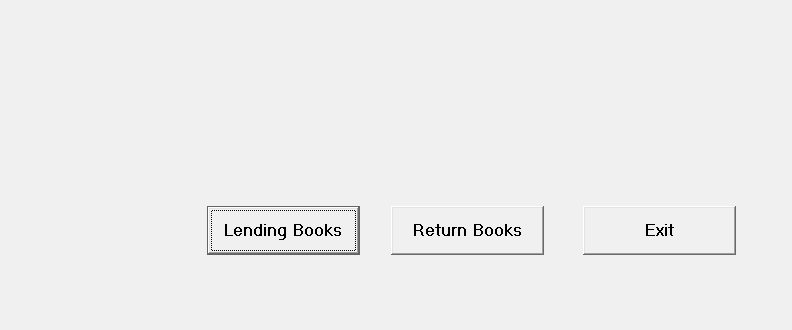
End End Sub

### OUTPUT:

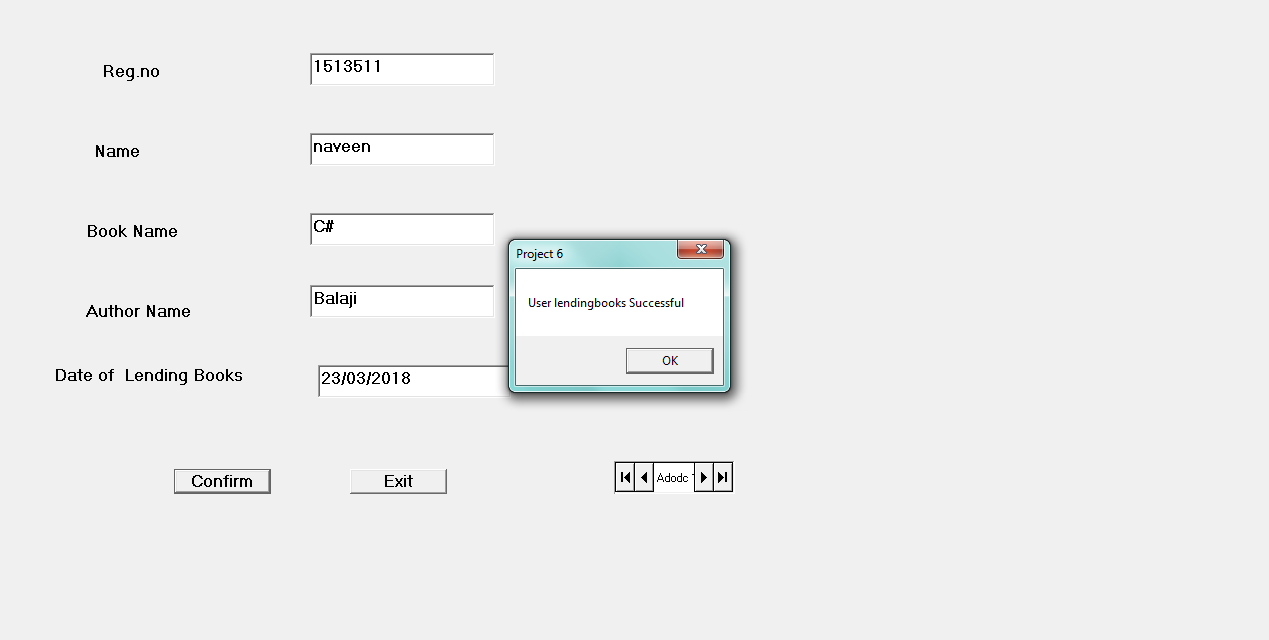
**Form1:**



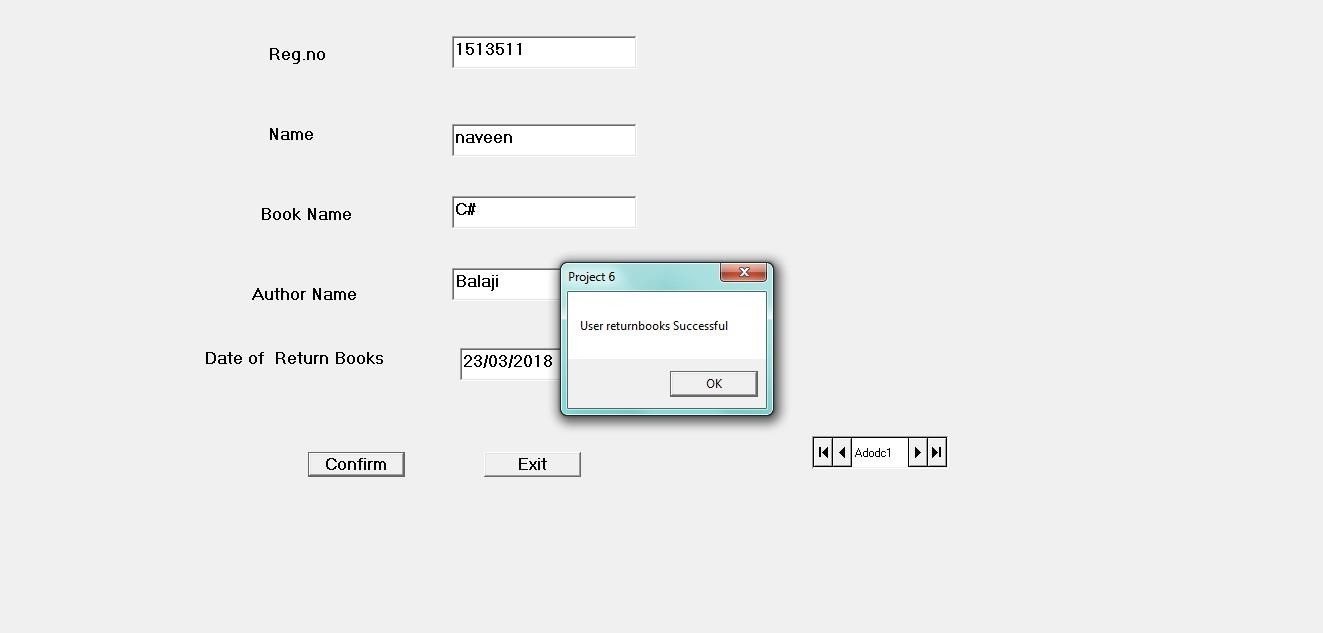
**Form2:**



**Form3:**

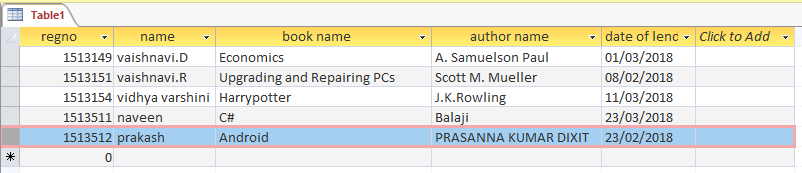


**Form4:**

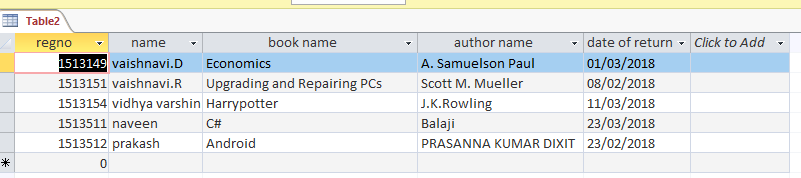


**Database:**

**Lending Books:**



**Return Books:**



**RESULT:**

Thus the **LIBRARY MANAGEMENT SYSTEM** is developed with all necessary documents and UML diagrams using Software Engineering methodology.