



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

Loan Management System for Jewels and Credit cards

SOFTWARE DESIGN AND DEVELOPMENT (CSE1005)

PROJECT COMPONENT

Submitted To

Prof. Satish C J

Associate Professor

School of Computer Science and Engineering

FINAL REVIEW REPORT

Submitted by

SAARTHAK AGARWAL (18BCB0056)	AKANCHA AGARWAL (17BCB0016)
GANGAVARAPU PRANEETH (18BCE2042)	VIDHI AGARWAL (18BCE2190)
SIDDHARTHA MONDAL (18BCB0145)	ANUPAMA GAUTAM (18BCB0019)
SREYAN BISWAS (18BCB0070)	AKILAN N (18BCB0157)
ABHISHEK ANAND (17BCB0073)	VAIBHAV SINGH(18BCE2313)

1.	Introduction.....	3
2.	Project SCOPE.....	3
3.	Key Contacts and Stakeholders.....	3
4.	Project Resource Requirements.....	3
4.1.	Software Resource Requirements.....	3
5.	SRS Contents.....	5
5.1.	Revision History.....	6
5.2.	User Requirements.....	6
5.2.1.	Product Functions.....	6
5.3.	Functional Requirements.....	12
5.3.1.	Use Case Diagrams.....	12
5.4.	Use Case Scenario Description.....	17
5.5.	Non Functional Requirements.....	58
5.5.1.	Usability.....	58
5.5.2.	Security.....	59
5.5.3.	Performance.....	59
5.5.4.	Capacity.....	60
5.5.5.	Recovery.....	60
5.5.6.	Availability.....	60
5.5.7.	Reliability.....	60
5.5.8.	Maintainability.....	60
5.5.9.	Portability.....	60
5.5.10.	Privacy.....	61
5.6.	Prototypes/Screenshots.....	62
6.	DESIGN CONTENTS.....	71
6.1.	ER DIAGRAM.....	72
6.2.	STATE CHART.....	72
6.3.	SEQUENCE DIAGRAM.....	76
6.4.	CLASS DIAGRAM.....	79
6.5.	DATA FLOW DIAGRAM.....	80
7.	Implementation.....	91

1. Introduction

Customer experience is the grand basis of competition in today's business world. Every bank focuses on automation reduce redundant physical tasks and to enhance customer experience. As a result the bank can gain great credibility in the eyes of the industry by reducing man-made errors and a proper system can take the bank to new heights. The Loan Collection management system is one such system that focuses on effective utilization of resources and enhancing the customer experience towards Loan management for Jewel and Credit Card Loans

2. Project SCOPE

Loan Collection management for Jewel Loans and credit card loans aims at automation of the following processes:

1. Credit Card and Jewel Loan Collection
2. Bankruptcy handling for credit card and jewel loans
3. Auction of Jewels
4. Reports and Statuses on Credit and Jewel Loan Collections

3. Key Contacts and Stakeholders

Name	Registration Number	Phone Number
Saarthak Agarwal	18BCB0056	9953446627
Akancha Agarwal	17BCB0016	7017366581
Gangavarapu Praneeth	18BCE2042	9003120412

Vidhi Agarwal	18BCE2190	9674931885
Siddhartha Mondal	18BCB0145	8981527584
Anupama Gautam	18BCB0019	9007546429
Sreyan Biswas	18BCB0070	7602212508
Akilan N	18BCB0157	6382115328
Abhishek Anand	17BCB0073	9693843478
Vaibhav Singh	18BCE2313	8449688817

4. Project Resource Requirements

4.1 Software Resource Requirements

- Windows XP (Service Pack3)
- mysql
- HTML 5
- JavaScript
- PHP5.5
- Apache Web Server
- Rational Rose Enterprise Edition
- 2-Plan
- ArgoUML
- IDE (like Eclipse)

5. SRS CONTENTS

Revision History

Name	Date	Reason For Changes	Version
Final documented completion with index and proper labelling of data	03/06/20 – 05/06/20	Final project Document content organizing	Version 1.0

Overall Description

2.1. Product Functions

The following are the user requirements for the system :

2.1.1. If the user is a Loan Officer and wants to view the decease status for Loan borrowers, he can view the following details for a particular loan **(18BCB0056 - SAARTHAK AGARWAL)**

- Borrower ID/Loan ID
- Name of Borrower
- Amount Borrowed
- Amount repaid
- Decease status
- Date of demise
- Co-borrower/ Guarantor ID
- Name of Guarantor

2.1.2. If the user is a Loan Officer/collector or Branch Manager and wants to view the repayment status report for various loans whose borrowers have deceased, they can view the following details with respect to a particular loan **(18BCB0056 - SAARTHAK AGARWAL)**

- Borrower/Loan ID
- Amount Borrowed
- Number of installments left
- Total amount repaid
- Amount left to be paid (Amount borrowed - Total amount repaid)
- Date of demise of previous Borrower

(User Requirements for team member 2 - 18BCE2042)

2.1.3. If the user desires to become a bidder. Then the user can apply to be a bidder by entering the following details. **(18BCE2042 - PRANEETH)**

1. First Name
2. Last Name
3. Email
4. Mobile Number
5. Pan Number
6. Aadhar Number
7. Address
8. Date of Birth (dd/mm/yyyy)

2.1.4. The User can view all the Bids arranged by the date the bid registration was opened. The Auction notification will have the following details. **(18BCE2042 - PRANEETH)**

1. Title

2. Reference Number
3. Bidder registration closing date
4. Bid Opening Date

(User Requirements for team member 3 - 18BCB0070)

2.1.5. The Loan Officer, collector, and the customer can have control over the payment of the loan dues . The filters are as follows: **(18BCB0070-SREYAN BISWAS)**

- Name of customer
- Loan Due
- Phone No. of the user
- Total loan Collected
- Option for delaying the payment by 2 months
- Access for removing the name of the user from the database.

The loan office collector after some time can pass the users data to a separate database wherein the users have the option to pay the money after 2 months or pay it now. The collector can contact the person for loan payment in his registered phone no. and after 9 months take appropriate actions.

(User Requirements for team member 4 - 18BCB0157)

2.1.6. If the user is a Customer and wants to apply for the bankruptcy, he can apply the following details for bankruptcy (18BCB0157 -AKILAN N)

- Customer id
- Type of loan
- Loan id
- Loan amount
- Bankruptcy reason
- Needed documents
- Applying bankrupt

2.1.7. After apply the bankruptcy, loan officer can check the needed details and he can approve or reject the application. Customers can login using his ID and check the bankruptcy status. They can enter their username and password for user verification. (18BCB0157 -AKILAN N)

- Log in
- Fill the bankruptcy form
- View the message(approve or not)
- Check remarks from loan officer

Loanofficer can login and view the customer's application and he can approve or reject. Bank manager can view the full details of approving process.

(User Requirements for team member 5 - 18BCE2190)

2.1.7 If the user is a Customer and participates in the auction, he can view the following (18BCE2190 - VIDHI AGARWAL)

- Image of the jewels
- Base price of the jewel
- Current bidded highest price on the jewel

- information about the jewel, that includes the name of metal used, weight, purity etc.

2.1.8 If the user is a Loan Officer/collector or Branch Manager and wants to view the status report for the auction, they can view the following details with respect to a particular customer (**VIDHI AGARWAL - 18BCE2190**)

- Name of customer
- Id no. of the product purchased
- Amount paid for the purchase

(User Requirements for team member 6 - 18BCB0019)

2.1.9 If the user is a Customer and wants to apply for the bankruptcy, he can apply the following details for bankruptcy. (**ANUPAMA GAUTAM-18BCB0019**)

- User login
- Type of loan
- Loan id
- Loan amount
- Bankruptcy reason
- Legal documents
- Applying bankrupt

2.1.10 After apply the bankruptcy, loan officer can check the needed details and he can approve or reject the application. Customers can login using his email ID and check the bankruptcy status. (**ANUPAMA GAUTAM**)

Log in

- Fill the bankruptcy form
- View the message(approve or not)
- Check remarks from loan officer

Loan officer(s) can login and view the customer's application and he can approve or reject. Bank manager(s) can view the full details of the approval process.

(User Requirements for team member 7 - 18BCB0145)

2.1.11 The loan officer(s) and the Collection officer(s) have access to the Customer details submitted by them on registration or during loan application. **(18BCB0145 - SIDDHARTHA MONDAL)**

The customers shall be contacted when the payments are due for more than 3 consecutive months and this done by the collection department. The Collection department can contact the customer by the following means :-

- Voice calling
- Emails
- Text messages / Whatsapp

The Collection department also has the permission and resources to record the conversation between the officer and the Customer (in the audio or text format). The details can be stored in the database as the '**Customer Payment Status**'.

There shall also be a status tracking of every customer who's details have been sent to the Collection department henceforth and a **detailed report** of the Payment Status of **such customers shall be maintained**.

(User Requirements for Team Member 8 – 17BCB0016)

2.1.12 If the user is the Bank Manager and wants to generate the report or view them for the individual collectors, he can view the following details for an individual collector **(17BCB0016 - AKANCHA AGARWAL)**

- Collector ID
- Collector Name
- Collector's Profile
- Amount collected (in chosen duration)
- List of defaulters assigned to individual collector

2.1.13. If the user is the Bank Manager and wants to generate the performance report of all the collectors in the duration selected, then he can view the following details separately for the credit and jewel loan collectors **(17BCB0016 - AKANCHAGARWAL)**

- Ranking of collectors (based on amount collected in given duration)
- Collector ID
- Collector Name
- Collector's Profile
- Amount collected (in chosen duration)
- List of defaulters assigned to individual collector

(User Requirements for team member 10 - 18BCE2313)

2.1.7 If the bank manager wants to view the report , they can find following details in

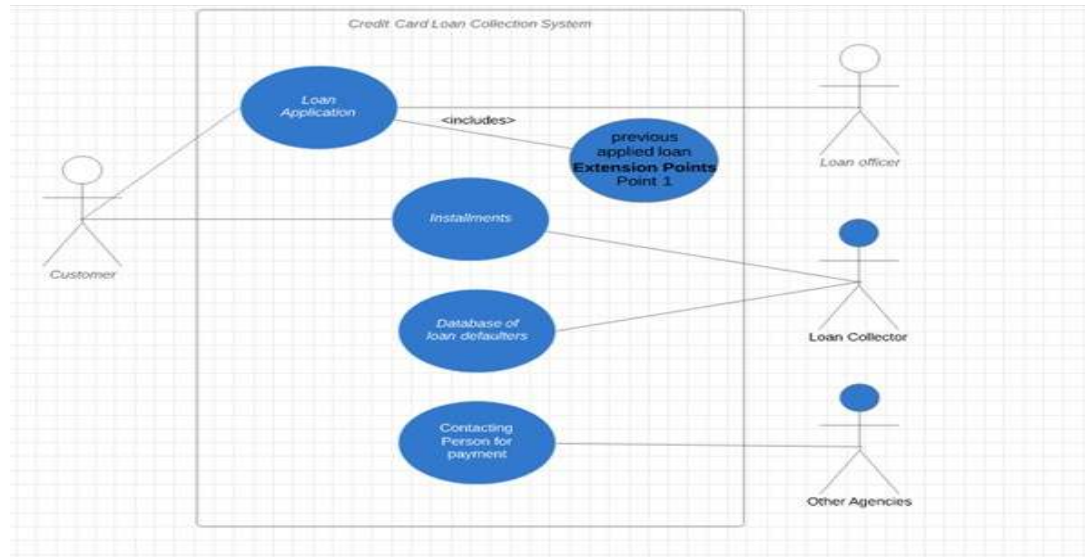
the report **(18BCE2313-VAIBHAV)**

- Report of the left debt on the consumer
- Report of the excess debt transferred to the consumer
- Report of the action

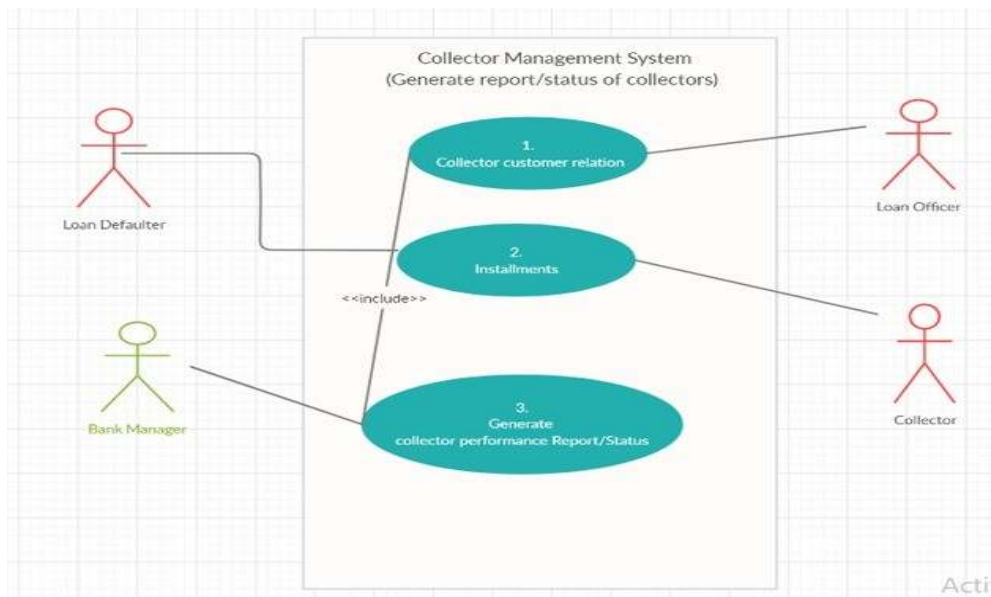
3. Functional Requirements

3.1. Use Case Diagrams ()

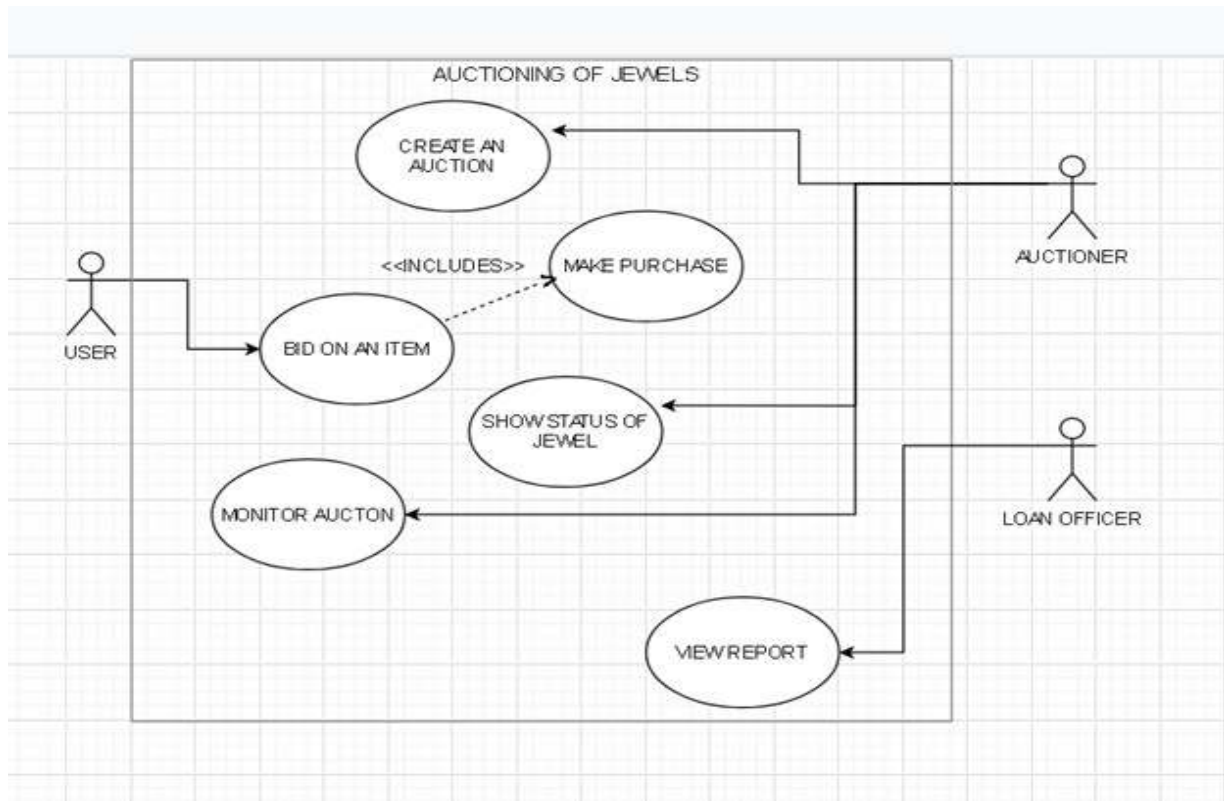
SREYAN BISWAS(18BCB0070)



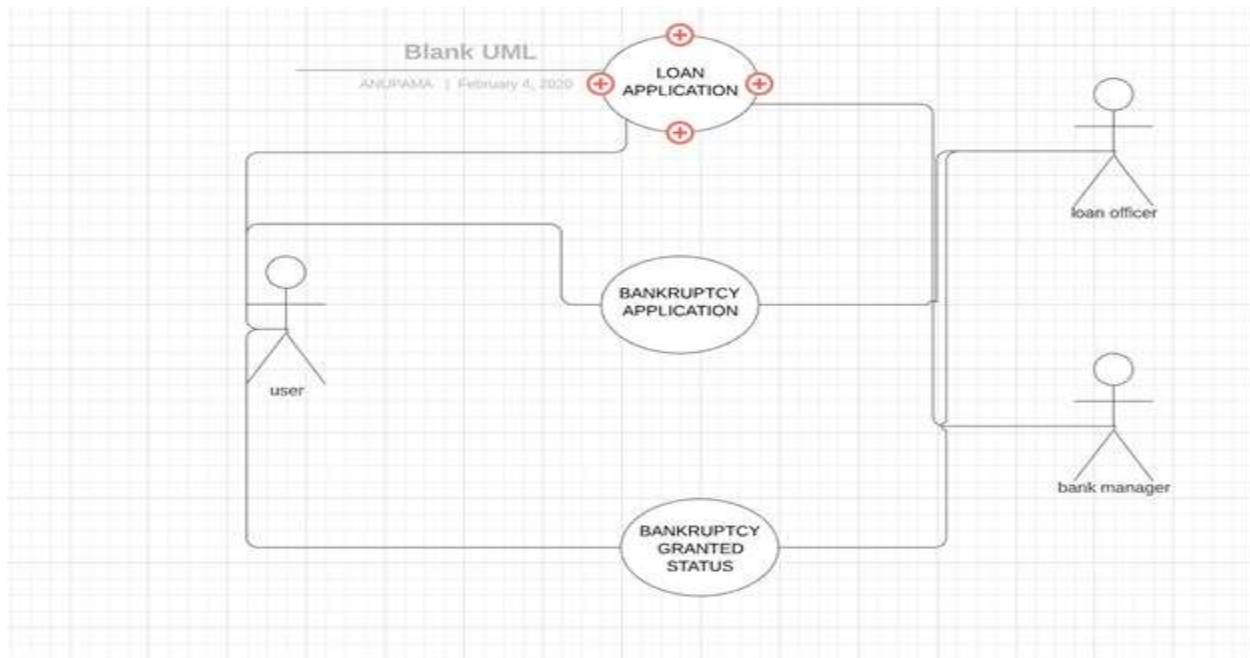
AKANCHA AGARWAL (17BCB0016)



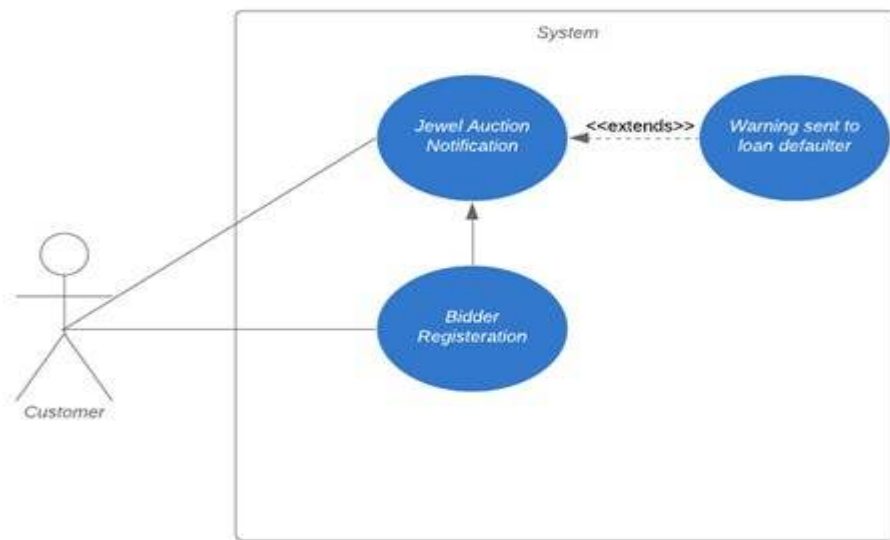
VIDHI AGARWAL (18BCE2190)



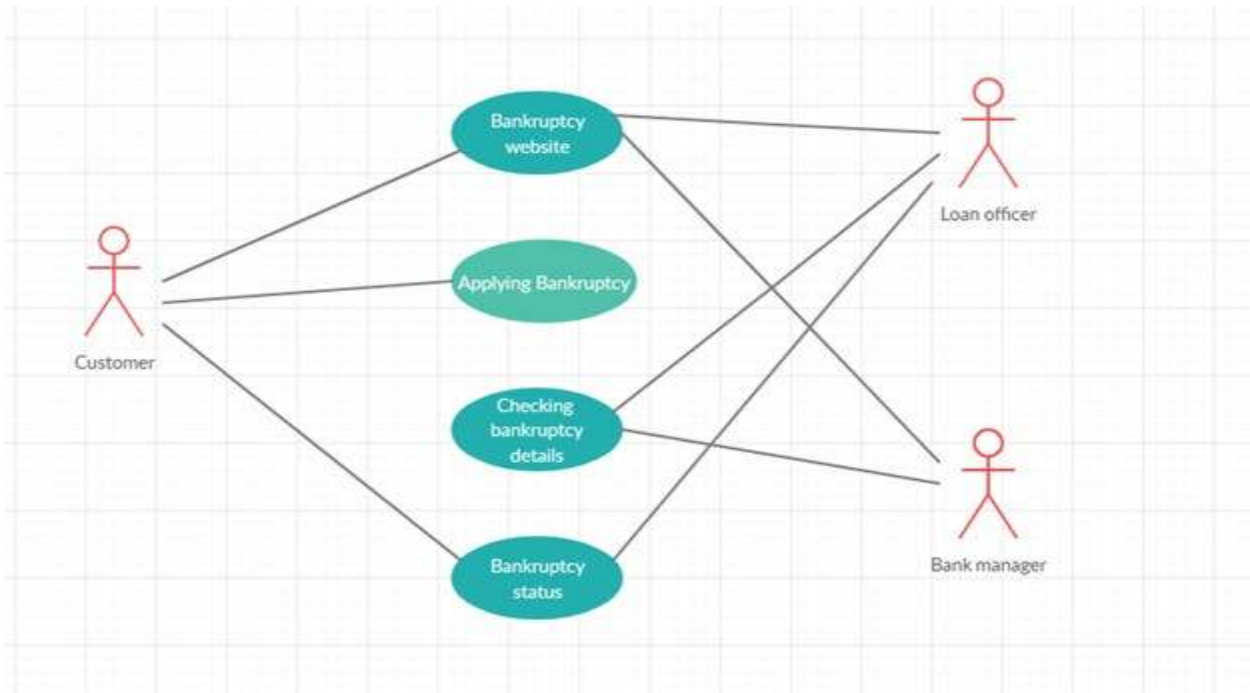
ANUPAMA GAUTAM(18BCB0019)



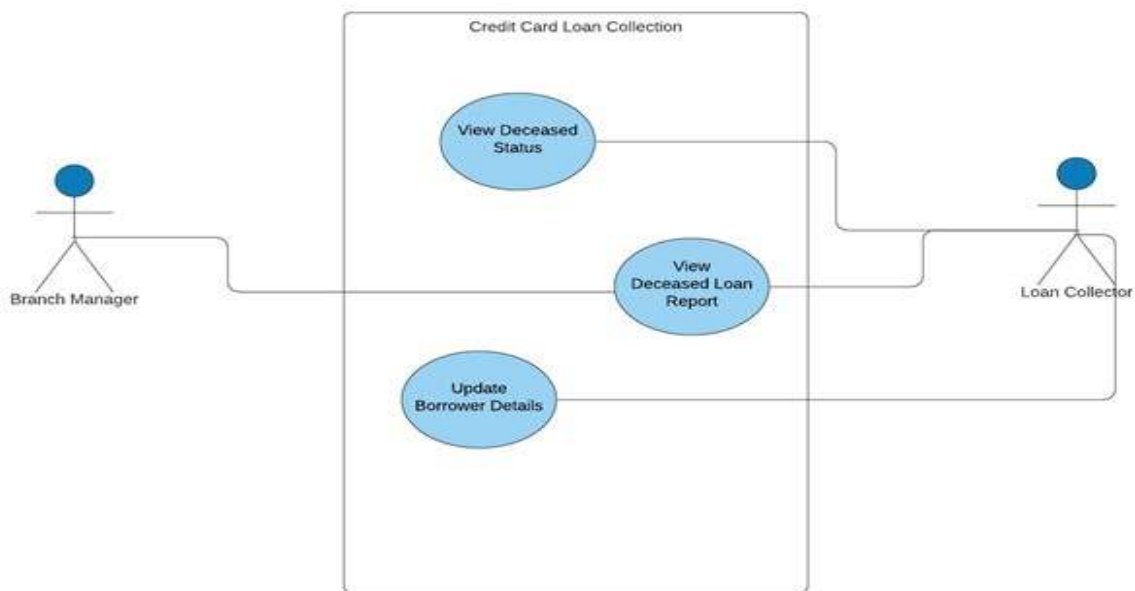
PRANEETH (18BCE2042)



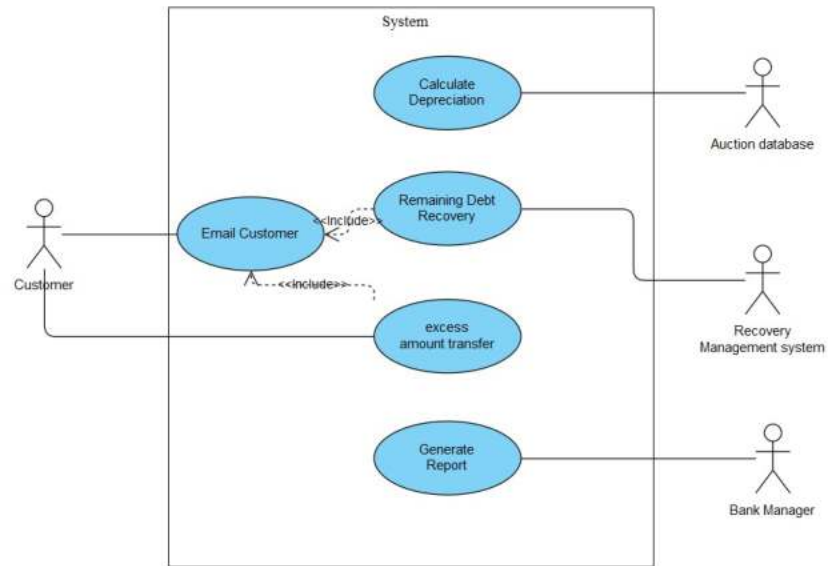
AKILAN N (18BCB0157)



SAARTHAK AGARWAL (18BCB0056)

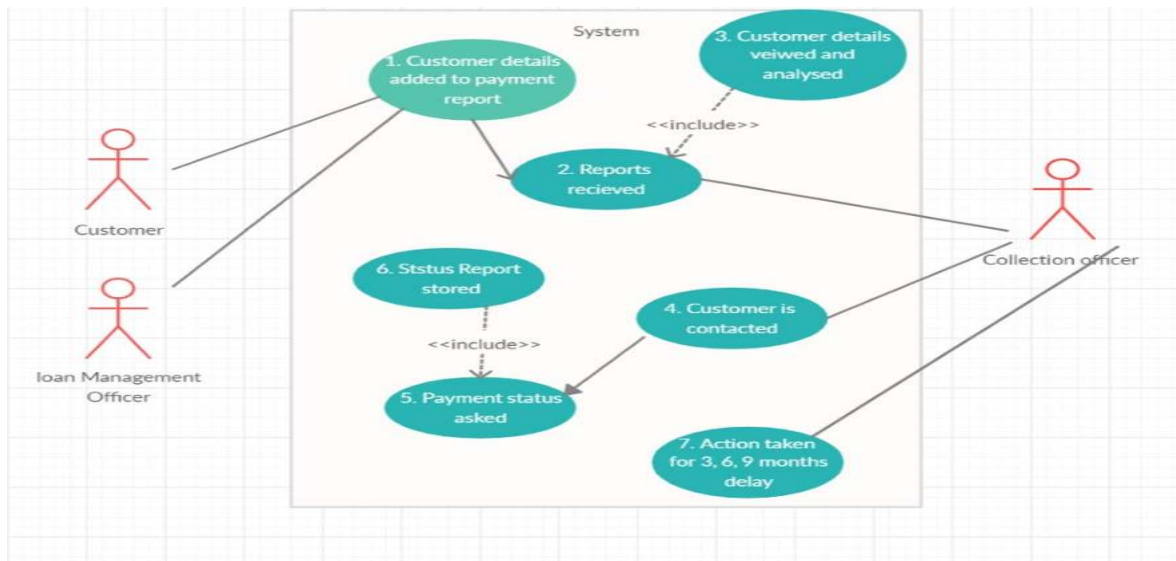


VAIBHAV SINGH (18BCE2313)



SIDDHARTHA MONDAL (18BCB0145)

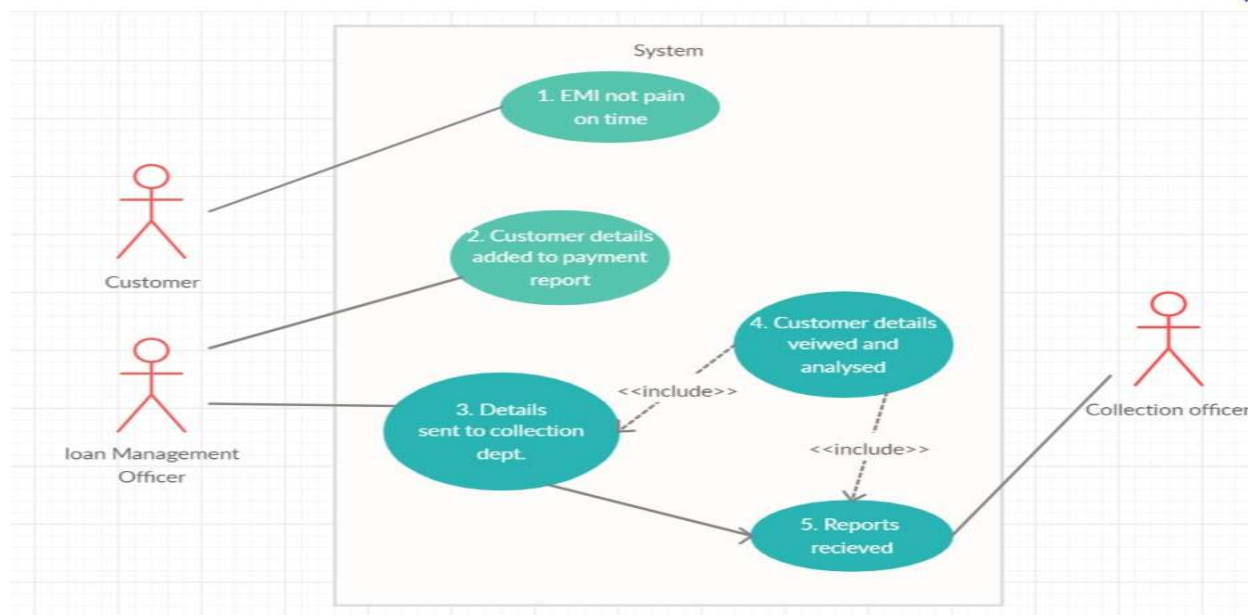
Status tracking in Collection Management System



ment System-

nage

SIDDHARTHA MONDAL (18BCB0145)

Collection Status Management System -

4. Use Case Scenario Description

USE CASE SCENARIO FOR 18BCB0019

Use Case ID	1 (18BCB0019 - ANUPAMA GAUTAM)
Use Case Name	LOAN APPLICATION
Summary	<p><u>customer,bank manager, loan officer will have their log in IDs. First they have to choose which user class they belong to. A login page will be provided. Where they can enter their username and password for user verification. The first page consists of following</u></p> <ul style="list-style-type: none"> • <u>bank ID</u>

	<ul style="list-style-type: none"> • <u>customer log in</u> • <u>loan officer Log In</u> • <u>bank Manager Log In</u> <p><u>customer,bank manager, loan officer will have their log in IDs.</u></p> <p><u>After choosing the user type they will be redirected to second page.</u></p> <p><u>where the user will get directed to loan application page Which consists of following</u></p> <ul style="list-style-type: none"> • <u>loan id(loan they have taken)</u> • <u>Enter Password</u>
<u>Preconditions</u>	<u>User must have his credentials in the database.</u>
<u>Success End Condition</u>	<u>You have logged in Successfully Message is displayed to the user.</u>
<u>Failed End Condition</u>	<u>If User could not provide his right details.</u>
<u>Primary, Secondary Actors</u>	<u>customer,, loan officer, bank Manager and programmers who will enter the details for verification.</u>
<u>Trigger</u>	<u>clicking on loan application we get directed to loan application</u>

	<u>page</u>	
<u>Description</u>	<u>Step</u>	<u>Action</u>
-	<u>1</u>	<u>The End user opens the first login Page.</u>
-	<u>2</u>	<u>End user will select his user type.</u>
-	<u>3</u>	<u>User will be redirected to the second page.</u>
-	<u>4</u>	<u>User will enter the user name and password for verification.</u>
-	<u>5</u>	<u>User then will click in the loan application</u>
-	<u>6</u>	<u>In case the user forgets his username. He will click on the “forgot password” link. Password will be sent to his registered Email ID.</u>

<u>Use Case ID</u>	<u>2 (18BCB0019 - ANUPAMA GAUTAM)</u>
<u>Use Case Name</u>	<u>BANKRUPTCY APPLICATION</u>

<u>Summary</u>	<p><u>Once the customer logins in his loan application page , an option for bankruptcy will be shown</u></p> <p><u>Customer can apply for the bankruptcy</u></p> <ul style="list-style-type: none"> • <u>Customer id</u> • <u>Type of loan</u> • <u>Loan id</u> • <u>Loan amount</u> • <u>Bankruptcy reason</u> • <u>Needed documents</u> • <u>Applying bankrupt</u> <p><u>Loan officer can approve or reject</u></p> <ul style="list-style-type: none"> • <u>Loan officer id</u> • <u>Checking applications</u> • <u>Customer</u> • <u>Needed documents</u> • <u>Approve or reject</u>
<u>Preconditions</u>	<p><u>User must have taken credit card loans in the database.</u></p>
<u>Success End Condition</u>	<p><u>You have logged in Successfully Message is displayed to the user.</u></p>

<u>Failed End Condition</u>	<u>If User didn't apply any loan</u>	
<u>Primary, Secondary Actors</u>	<u>customer,, loan officer, bank Manager who will enter the details for verification.</u>	
<u>Trigger</u>	<u>after filling the form the person has to wait for 30 working days to check its status.</u>	
<u>Description</u>	<u>Step</u>	<u>Action</u>
-	<u>1</u>	<u>The End user opens the first login Page.</u>
-	<u>2</u>	<u>End user will select his loan from which he needs relief.</u>
-	<u>3</u>	<u>User will select the loan relief options</u>
-	<u>4</u>	<u>User will click on bankruptcy option.</u>
-	<u>5</u>	<u>user will fill the form stating proper reason and proof</u>

-

-

-

<u>Use Case ID</u>	<u>3 (18BCB0019 - ANUPAMA GAUTAM)</u>
<u>Use Case Name</u>	<u>BANKRUPTCY RELIEF GRANTED STATUS</u>
<u>Summary</u>	<p><u>Once the customer logs in the bankruptcy application icon:</u></p> <ul style="list-style-type: none"> • <u>fills the form</u> • <u>amount to be paid</u> • <u>bankruptcy applied with details</u> <p><u>then the person has to wait for 30 days to view its status.</u></p> <p><u>Loan officer can login view the customer's application and he can approve or reject</u></p> <p><u>Bank manager can login and view the all the activities application process and approving process</u></p> <p>-</p>
<u>Preconditions</u>	<u>User must have proper legal proof of his bankruptcy</u>
<u>Success End Condition</u>	<u>You have logged in Successfully Message is displayed to the user.</u>
<u>Failed End Condition</u>	<u>If User didn't apply any loan or you failed to prove your bankruptcy.</u>
<u>Primary, Secondary</u>	<u>customer,, loan officer, bank Manager.</u>

<u>Actors</u>		
<u>Trigger</u>	<u>email will be recieved by the user stating his approval or rejection.</u>	
<u>Description</u>	<u>Step</u>	<u>Action</u>
	<u>1</u>	<u>End user will fill the form stating proper reason and proof</u>
	<u>2</u>	<u>User should upload legal affidavit in the site</u>
	<u>4</u>	<u>His status will be mailed in his mail id.</u>
	<u>5</u>	<u>he should open his email and see his approval from loan officer</u>

USE CASE SCENARIO FOR 18BCB0145

UseCase ID	3 (18BCB0145 - Siddhartha Mondal)
-------------------	--

Use Case name	Collection Status Management
Summary	The System will be working in such a manner that there will be a status tracking of each customer of the bank under the loan management team. The system should be such that after 3 simultaneous payment misses, the details of the particular customer shall be sent to the Collector management system.
Precondition	<ul style="list-style-type: none"> • The customer must be authorized and logged in. • The loan must be Sanctioned by the prior departments.
Success End Condition	After 3 consecutive failures in loan payment, the details of the customer must be sent to the Collection management department. <i>'Data will be sent'</i>
Failed End Condition	A customer who has not paid an EMI for 3 consecutive months is still not being dealt with by the loan collection management.
Primary, Secondary Actors	Loan Manager, Loan officers, Collection Management officers
Trigger	Automated system, triggered by the import of installment submission data. Failing of 3 consecutive EMIs triggers the system.

Description	<ol style="list-style-type: none"> 1. The payments of each Customer is observed. 2. On payment miss, the Customer is kept under surveillance. 3. If the Customer does not pay for 3 months straight, then then the details of this customer is sent to the Collection management team.
Use Case ID	4 (18BCB0145 - Siddhartha Mondal)
Use Case Name	<u>Status tracking in Collection Management System</u>
Summary	<p>For every Customer who has not been able to pay the EMI on time and passed the criteria of the USE CASE 3, the status of their payments shall be tracked. This includes :-</p> <ol style="list-style-type: none"> 1. Next payment date 2. Further payment delay 3. Contacting and asking for payment and recording the reply in a database. <p>There shall be different actions taken against the customer for the delay in payments by 3 months, 6 months and 9 months respectively. The actions taken and its results should also be stored and maintained for every customer.</p>

Preconditions	Customer has not paid EMI for 3 months consecutively.	
Success End Condition	An action is taken against each customer who is not paying the dues in time.	
Failed End Condition	A customer who isn't paying dues in time escapes without a reminder or any actions against him/her from the bank. A failure in storing the status report of each and every customer is also considered a failure.	
Primary, Secondary Actors	Loan management team, Collection officer, Customer.	
Trigger	When Customer doesn't pay in 3 months time an auto trigger is executed.	
Description	Step	Action
	1	Collection team receives data about the customer who isn't paying dues.

	2	The Collection team officer checks on the customer and contacts him for acting as a reminder.
	3	The customer's reply to the contact is recorded in a database.
	4	The Collection officer's check on the customer if he has paid the loan due.
	5	The customer is observed through the course of his EMI period.
	6	All delays and misses are recorded in the main database.
	7	For a delay of 3 months, 6 months and 9 months, separate action is taken by the collection officers.

-

USE CASE SCENARIO FOR 18BCE2042

-

Use Case ID	5 (Praneeth - 18BCE2042)
Use Case Name	Jewel Auction Notification
Summary	<p>This page will contain all the Bids arranged by the date the bid registration was opened. The Auction notification will have the following details.</p> <ol style="list-style-type: none"> 1. Title 2. Reference Number 3. Bidder registration closing date 4. Bid Opening Date <p>After clicking the desired item to be auctioned the user will be redirected to the bidder registration page.</p>
Preconditions	Items should be put up for Auction.
Success End Condition	Auction details are displayed with respect to the date the bidder registration it was opened
Failed End Condition	Auction details are not displayed.
Primary, Secondary Actors	Customer/User, Loan officer

Trigger	Clicking the Auction notification tab
Extension	Sends a message to the loan defaulter that his collateral (Jewel) will be auctioned in 5 days.

-

-

Use Case ID	6 (Praneeth - 18BCE2042)
Use Case Name	Bidder Registration
Summary	<p>A user can register to become a bidder for the selected auction.</p> <ol style="list-style-type: none"> 1. First Name 2. Last Name 3. Email 4. Mobile Number 5. Pan Number 6. Aadhar Number 7. Address 8. Date of Birth (dd/mm/yyyy)
Preconditions	<ol style="list-style-type: none"> 1) Items should be put up for Auction. 2) Bidder registration should be open for the selected

	auction	
Success End Condition	“Submitted” message will be displayed	
Failed End Condition	“Submission Failed” message will be displayed	
Primary, Secondary Actors	Customer	
Trigger	Clicking on the desired auction(Item)	
Description	Step	Action
	1	The End user selects the item or auction
	2	The user is directed to another page.
	3	<p>The page will consists of fields, where the user has to enter details of the following.</p> <ol style="list-style-type: none"> 1. First Name 2. Last Name 3. Email 4. Mobile Number

		5. Phone Number 6. Pan Number 7. Aadhar Number 8. Address 9. Date of Birth (dd/mm/yyyy)
	4	Then the “Submit” button is clicked.
	5	If submission is successful, “Submitted” message will be displayed
	6	If submission is a failure, “Submission Failed” message will be displayed

-

Use Case Description for 18BCE2190

Use Case ID	7 (18BCE2190- Vidhi Agarwal)
Use Case Name	Make Purchase
Summary	The user logs in to the system and bids on the desired item. The user enters his quoted amount. If at the end of the auction, the amount quoted is the highest, the user makes a purchase by entering his bank details.

Preconditions	<ol style="list-style-type: none"> 1. User must be logged in to the system. 2. User must have had the highest bid on an item. 	
Success End Condition	Payment successful. The jewel is successfully purchased by the customer.	
Failed End Condition	Payment is not successful. Message is displayed. The user is requested to initiate a payment again	
Primary, Secondary Actors	<p>Primary:</p> <p>User</p> <p>Secondary:</p> <p>Auctioner</p>	
Trigger	This use case is initiated by the user clicks on the purchase button..	
Description	Step	Action
	1	The End user selects the purchase tab.
	2	End user has to enter his bank account details.

	3	User enters the amount to be payed.
	4	Then the user clicks on the submit button.

Use Case ID	8 (18BCE2190- Vidhi Agarwal)
Use Case Name	Bid on an item
Summary	The user logs in to the system and bids on the desired item. The user enters his quoted amount. If at the end of the auction, the amount quoted is the highest, the user makes a purchase.
Preconditions	1. User must be logged in to the system.
Success End Condition	Bidded amount is reflected in the system
Failed End Condition	Bidded amount is not reflected in the system. User is asked to re enter the amount.
Primary, Secondary Actors	Primary: User Secondary:

	Auctioner	
Trigger	This use case is initiated by the user clicks on the bid button..	
Description	Step	Action
	1	The End user selects the jewel to bid on.
	2	End user has to enter his/her quoted amount.
	3	Then the user clicks on the bid button.

Use Case ID	9 (18BCE2190- Vidhi Agarwal)
Use Case Name	Create An Auction
Summary	An auction is arranged for all the loan defaulters, to have a chance to pay back their loans.
Preconditions	<ol style="list-style-type: none"> 1. User must have be authorized and logged in. 2. Users must have a loan overdue for over 9 months.

Success End Condition	Auction is created and displayed in the system.	
Failed End Condition	Auction is not created. Message displayed.	
Primary, Secondary Actors	<p>Primary Actors:</p> <ol style="list-style-type: none"> 1. User 2. Collection Management Officer <p>Secondary actors:</p> <ol style="list-style-type: none"> 1. Loan Officer 	
Trigger	This use case is initiated by the loan officer creating it..	
Description	Step	Action
	1	Collection management officer receives the data of customers who have pending loans for over 9 months
	2	He then creates an auction for all jewels mortgaged
	3	The collection management officer gives access to the pending loan payers.

	4	Jewel descriptions are added
--	---	------------------------------

Use Case ID	10 (18BCE2190- Vidhi Agarwal)
Use Case Name	Monitor auction
Summary	The auctioneer conducts the auction and monitors the system, and is active for help and support during the auction.He ensures smooth functioning of the system.
Preconditions	1. Auction is created.
Success End Condition	The user queries are resolved.
Failed End Condition	Chatbot server is down. Message is displayed.
Primary, Secondary Actors	Primary: User Auctioneer

Trigger	This use case is initiated by the user clicks on the chat button.	
Description	Step	Action
	1	The End user selects the chat icon.
	2	End user enters his queries.
	3	Auctioneer resolves the query.

Use Case ID	11 (18BCE2190- Vidhi Agarwal)
Use Case Name	Show status of jewel
Summary	The user logs in to the system and bids on the desired item. The user enters his quoted amount. All end users are shown the current highest bid.
Preconditions	1. User must be logged in to the system.

Success End Condition	User can view the live status of the jewel.	
Failed End Condition	Live status is not available. Message displayed	
Primary, Secondary Actors	Primary: User Secondary: Auctioneer	
Trigger	This use case is initiated by the user clicks on the purchase button..	
Description	Step	Action
	1	The End user selects the home tab.
	2	End user scrolls to his desired product.
	3	Users can then view the status of the jewel.

Use Case ID	12 (SREYAN BISWAS-18BCB0070)
-------------	------------------------------

Use Case Name	Loan Application
Summary	Applying for the loan online by filling a form and getting it approved by the officer.
Preconditions	The form should be filled properly, and should not have applied no history of not repaying loans
Success End Condition	Loan applied successfully and the loan approved by the officer
Failed End Condition	Form not successfully applied or history of not paying back loans, so application cancelled.
Primary, Secondary Actors	Primary actor is the customer(loan applier) Secondary actor is the loan officer granting the loan.
Trigger	Apply button after logging into the system, for loan.
Description	Step
	Fill the Form
	Apply for the loan

	Loan Officer checks appliers background and details
	If everything is fine then approve the loan.

Use Case ID	13 (18BCB0070-SREYAN BISWAS)
Use Case Name	Installments
Summary	The customer through the software needs to pay for their monthly installments.
Preconditions	The users should pay for their monthly installments without delay.
Success End Condition	The user on a regular basis every month pay for their installments.
Failed End Condition	The user fails to meet the deadlines of installment payment.
Primary, Secondary Actors	Primary actor is the customer Secondary actor is the loan collector

Trigger	Pressing the payment button for paying the installment for that month.
Description	Action
	Login to the system.
	Go to payment option for paying the installment, of that month or the due installments.
	Select the method of payment.
	Pay and the database of the user will be updated.

Use Case ID	14 (18BCB0070-SREYAN BISWAS)
Use Case name	Database of loan defaulters
Summary	All the person who have not repaid their loans ,their name along with contact details should be kept in a separate database system with.

Preconditions	The person should have defaulted for some months the loans
Success End Condition	The persons data is successfully added and the payment delay option by the user is working properly.
Failed End Condition	The data of the person not paying the loans is not added successfully, and even the checkbox is not working properly.
Primary, Secondary Actors	Primary actor is the loan collector.
Trigger	On not paying the loan collector updates the database and then the data is updated in the database of loan defaulters.
DESCRIPTION	Action
	Loan not paid for months.
	According to the no. of months not paid the database is updated.

USE-CASE SCENARIO FOR 18BCB0070

Use Case ID	15 (18BCB0070 - SREYAN BISWAS)
Use Case name	Contacting Person for payment
Summary	TESTE After every 3 months, if the person has not paid the money, then they are called or messaged for their repayment, else if they have option to delay for max. 2 months.
Preconditions	The person has not paid the loan and the checkbox have not been pressed for months delay.
Success End Condition	The person has been contacted and the loan paid for the person and his data has been removed from the database.
Failed End Condition	The person keeps on defaulting the payment, even after the multiple phone calls.
Primary, Secondary Actors	Primary actors are the agencies for which are responsible for calling the loan defaulters.
Trigger	The persons phone no. is noted down and the person not paying is called.

DESCRIPTION	Action
	Loan not paid for 3,6,9 months.
	Or selects to repay within 2 months by selecting in the checkbox.
	Person is accordingly called for repayment of the loan.
	If after 9 months of try the person refuses to pay back the loan then the persons trial is passed to another system.

USE-CASE SCENARIO FOR 17BCB0016- Akancha Agarwal

Use Case ID	16 (17BCB0016-Akancha Agarwal)
Use Case Name	Collector Customer Relation
Summary	Assigning collectors to a single or group of customers who are loan defaulters and store the information in the database.
Preconditions	Collectors and the customers must be registered with the system.
Success End	Successfully assigned the collector ("collector Name") to

Condition	("number") of loan defaulters.
Failed End Condition	Failed to assign the collector ("collector Name") to the loan defaulters due to failed connectivity.
Primary, Secondary Actors	Primary actor is the Loan Officer. Secondary actor is the Bank Manager.
Trigger	Pressing the "Assign" button for assigning a particular collector to a single or group of individuals.
Description	Action
	Login to the system.
	Go to loan defaulters section from the menu.
	Click on the Collector Customer relation button from the sub-menu.
	Select the collector and select the loan defaulters for which that collector has to be assigned.
	Click on the "Assign" button to successfully assign the collector to the loan defaulters.

Use Case ID	17 (17BCB0016-AKANCHHA AGARWAL)
Use Case Name	Installments
Summary	The customer through the software needs to pay for their monthly installments.
Preconditions	The users should pay for their monthly installments without delay.
Success Condition	The user on a regular basis every month pay for their installments.
Failed End Condition	The user fails to meet the deadlines of installment payment.
Primary, Secondary Actors	Primary actor is the customer Secondary actor is the loan collector
Trigger	Pressing the payment button for paying the installment for that month.
Description	Action
	Login to the system.

	Go to payment option for paying the installment, of that month or the due installments.
	Select the method of payment.
	Pay and the database of the user will be updated.

Use Case ID	18 (17BCB0016-AKANCHA AGARWAL)
Use Case Name	Generate Collector Performance Report/Status
Summary	The Bank Manager generates the collector performance Report based on the duration, he/she selects, which consist of the ranking of the collectors based on the amount they have collected in the duration selected by the Manager.
Preconditions	Bank Manager should have proper authentication.
Success Condition	End Bank Manager successfully generates and views the reports/statuses of all the collectors.
Failed End Condition	Failed to generate the report message displayed on the screen, asking to try again after declaring the reason (connectivity issue etc.).

Primary, Secondary Actors	Primary actor is the Bank Manager. Secondary actor is the Loan Officer
Trigger	Pressing the “Generate Report” button to view the report after selecting the duration.
Description	Action
	Login to the system.
	Go to the customer collector relation tab.
	Select the generate collector report link from the sub-menu.
	Select the duration for the collector performance.
	Click on “Generate Report” button to generate the report for the specified duration.

Use Case Description for 18BCB0056

Use Case ID	19 (18BCB0056- SAARTHAK AGARWAL)
Use Case	View Deceased Status

name	
Summary	<ul style="list-style-type: none"> • The Loan Officer/Collector will be available to see details of the Loan Borrowers and their respective status of being deceased. If Deceased, a column will contain the date of demise. • Another Column will contain the guarantor/co-borrower's unique ID.
Preconditions	<ul style="list-style-type: none"> • The Loan Officer/Collector must be authorized and logged in. • The list of borrowers and their deceased status will only be displayed when the user clicks the view status button.
Success End Condition	The list of all borrowers and their deceased statuses will be displayed.
Failed End Condition	An error message will be displayed.
Primary, Secondary Actors	Loan Officer/Collector
Trigger	This use case is initiated based on the clicking the view status button.

DESCRIPTION	Step	Action
	1.	The Collector clicks on the view status button.
	2.	The borrower details, decease status, date of decease, co-borrower's unique ID.
EXTENSIONS	Step	Branching Action
	1.	All required data is fetched from the database and displayed on the browser

Use Case ID	20(18BCB0056 - SAARTHAK AGARWAL)
Use Case name	Update borrower details
Summary	<ul style="list-style-type: none"> For all the borrowers that have a positive deceased status, all details and credentials will be updated. The details will be simply replaced by that of the co-borrowers/nominees and they will become responsible for further payments of the loan taken by the deceased borrower.

Preconditions	<ul style="list-style-type: none"> The Loan Officer/Collector must successfully log in to the system. Collector must Click the Update Borrower button to update all details in the database. 	
Success End Condition	'Borrower Update Successful' will be displayed as a message.	
Failed End Condition	'Failed to update database' will be displayed as a message.	
Primary, Secondary Actors	Loan Officer/Collector	
Trigger	The given use case is initiated by the click of the Update Borrower button.	
DESCRIPTION	Step	Action
	1.	The Loan Officer clicks on the Update Borrower button.
	2.	A message showing success or failure of update will be displayed
EXTENSIONS	Step	Branching Action

	1.	All entries in the table containing the borrower's information, corresponding to a deceased borrower will be replaced/updated with that of the co-borrower or nominee.
--	----	--

Use Case ID	21(18BCB0056- SAARTHAK AGARWAL)
Use Case name	View Deceased Loan Report
Summary	<ul style="list-style-type: none"> Detailing of amount borrowed, amount paid, duration of payback etc. which are the important details with respect to status of repayment of loan will be displayed
Preconditions	<ul style="list-style-type: none"> The User must be authorized and logged in. Click of the button Deceased Loan Report
Success End Condition	All loan related details and the status of their repayment are displayed
Failed End Condition	Error message will be displayed.
Primary, Secondary Actors	Loan Officer/Collector, Branch Manager
Trigger	The pressing of the Deceased Loan Report button.

DESCRIPTION	Step	Action
	1.	User clicks the <i>Deceased Loan Report</i> button.
	2.	The details of all loans whose borrowers had been deceased are displayed.
EXTENSIONS	Step	Branching Action
	1.	All required data is fetched to the browser from the database.

USE CASE DESCRIPTION (18BCE2313 - Vaibhav Singh)

Use Case ID	22(18BCE2313 - Vaibhav Singh)
Use Case Name	Remaining debt updation to loan collection system after Depreciation is calculated
Summary	The loan collection system will be updated when the recovered amount after the auction is less than the loaned amount
Preconditions	Depreciation amount already calculated
Success End Condition	The loan collection system will be updated and email would be generated from the system notifying customer about the remaining value and the branch's manager report would be

	updated	
Failed End Condition	The loan collection system would not be updated and the failed status would be updated in the report that would be sent to the branch manager	
Primary, Secondary Actors	primary actors - customer, loan collection system secondary - auction database	
Trigger	The process would be automatically triggered once the calculation of the depreciation amount is done	
Description	Step	Action
	1	The depreciation amount would be calculated
	2	The loan collection system would be updated
	3	email notification would be sent to the customer
	4	The managers report would be updated

Use Case ID	23(18BCE2313 - Vaibhav Singh)
--------------------	-------------------------------

Use Case Name	Excess debt transfer to customer after Depreciation is calculated	
Summary	The Excess debt transfer will be transferred to the customer when the recovered amount after the auction is more than the loaned amount	
Preconditions	Depreciation amount already calculated	
Success End Condition	The Excess debt transfer will be transferred and email would be generated from the system notifying customer about the excess value, the branch manager's report would be updated	
Failed End Condition	The loan collection system would not be updated and the failed status would be updated in the report that would be sent to the branch manager	
Primary, Secondary Actors	primary actors - customer, loan collection system secondary - auction database	
Trigger	The process would be automatically triggered once the calculation of the depreciation amount is done	
Description	Step	Action
	1	The depreciation amount would be calculated
	2	The Excess debt transfer will be transferred
	3	Email notification would be sent to the customer

	4	The managers report would be updated
--	---	--------------------------------------

Use Case ID	24(18BCE2313 - Vaibhav Singh)	
Use Case Name	Bank manager report generation	
Summary	After the process of transfer or updation is complete the report would be generated for the bank manager	
Preconditions	All the process of transfer or updation is complete for that day of auction	
Success End Condition	The report would be generated successfully	
Failed End Condition	The report would not be generated successfully and a mail would be sent to the software maintenance department briefing about the error	
Primary, Secondary Actors	primary actors - loan manager	
Trigger	The process would be triggered when the branch manager click on the view action report option	
Description	Step	Action

	1	All the process of transfer or updation is complete for that day of auction
	2	branch manager would click on the view action report option
	3	Report would be showed on screen

USE CASE SCENARIO FOR 18BCB0157

-

Use Case ID	25 (AKILAN N – 18BCB0157)
Use Case Name	Applying bankruptcy
Summary	<p>Customer, Loan officer and bank manager have their ID using login to the bankruptcy. First they have to choose user class belonged to. They can enter their user name and password for user verification.</p> <ul style="list-style-type: none"> • Bank id • Customer log in • Loan officer log in • Bank manager log in <p>Customer can apply for the bankruptcy</p> <ul style="list-style-type: none"> • Customer id

	<ul style="list-style-type: none"> • Type of loan • Loan id • Loan amount • Bankruptcy reason • Needed documents • Applying bankrupt <p>Loan officer can approve or reject</p> <ul style="list-style-type: none"> • Loan officer id • Checking applications • Customer • Needed documents • Approve or reject
Preconditions	Customer already had jewel loans
Success end condition	Applying bankruptcy successfully
Failed end condition	User could not have enough details. Error message displayed
Primary, Secondary actors	Customer, loan officer and bank manager

Trigger	User enters the user id and password, it moves to the next page here will click the loan option message will display “applying successfully”.
Description	Actions: (steps)
	1. User opens the first login Page
	2. User will select his user type.
	3. User will be redirected to the second page.
	4. User will enter the user name and password for verification 5. User then will click in the loan option (for bankruptcy) 6. User (customers) will apply for bankruptcy. User is loan officer can check and approve.

Use Case ID	26 (AKILAN N – 18BCB0157)
Use Case Name	Bankruptcy status
Summary	Customers can login using his ID and check the bankruptcy status. They can enter their username and password for user verification.

	<ul style="list-style-type: none"> · Log in · Fill the bankruptcy form · View the message(approve or not) · Check remarks from loan officer <p>Loan officer can login view the customer's application and he can approve or reject</p> <p>Bank manager can login and view the all the activities application process and approving process</p>
Preconditions	Customers already applied for bankruptcy with needed documents.
Success end condition	Bankruptcy application approved successfully.
Failed end condition	"Your application rejected" message will be displayed because User did not submit needed documents.
Primary, Secondary actors	Customer, loan officer and bank manager
Trigger	This Trigger use the Customers click the login page and see the his bankruptcy status approve or not

Description	Steps	Actions:
	1.	Customer opens the login Page.
	2.	Then select the bankruptcy option
	3.	Customers will fill the form with proper reason.
	4.	Customer will attach the needed documents
	5.	loan officer will verify customer's application and can approved or rejected
	6.	Customers can view the bankruptcy status
	7.	Status message will be sent to his registered Email ID.
	8.	All activities can viewed by bank manager

5. Non Functional Requirements

This section describes in detail all the non-functional requirements

5.1 Usability

9.1.1 The system shall allow the users to access the system from the Internet.

9.1.2 online help will be available for the system

9.1.3 The end users will be able to able to adapt to the system with a minimum training of 26 hours

9.1.4 Keyboard shortcuts will be available for all functions of the system

5.2 Security

5.2.1 Login requirements -

9.2.1.1 The customers, loan officer and loan collector will be provided access to the system after they are registered into the database.

9.2.1.2 While logging in the system for the first time, the customer, loan officer and collector will be provided an ID and a password.

9.2.1.3 On logging in, they can set a new password

5.2.2 Password requirements

9.2.2.1 Password will be case-sensitive.

9.2.2.2 Password must have at least 8 characters.

9.2.2.3 Password must have numbers and special characters

9.2.2.4 Password must be hashed and stored in the database

5.2.3 Inactivity timeouts

9.2.3.1 System should timeout when there is no activity for 15 minutes.

5.3 Performance

5.3.1 Response time

5.3.1.1 The response time will be less than 10 seconds for almost all the processes performed in the system.

5.4 Capacity

5.4.1 Storage

5.4.1.1 Hard disk space –

100 GB – Content

50 GB – Transaction Logs

5.5 Recovery

5.5.1 Recovery time scales

5.5.1.1 The system will be recovered within 1 hour from the down time

5.5.2 Backup Frequencies

5.5.2.1 Details of all the processes carried out by the customer, loan officer and collector will be stored in the back-up tapes.

5.5.2.2 All the back-up tapes will be provided to the bank database administrator..

5.5.2.3 The back-up data will be updated every 10 days.

5.6 Availability

5.6.1 Hours of operation

5.6.1.1 The system will be available on all days 24*7

5.7 Reliability

5.7.1 Mean Time between Failures

5.7.1.1 The mean time between failures for the system will be 90days

5.8 Maintainability

5.8.1 Mean Time to Recovery

5.8.1.1 The Mean Time To Recovery (MTTR) shall not exceed one day.

5.9 Portability

5.9.1 OS requirements

The system will run on windows XP / Vista / 7 / 8 / 8.1 / 10 and on MAC OS and on LINUX.

5.9.2 Browser requirements

The system will run on Internet Explorer, Internet Edge, Mozilla Firefox, Google Chrome, Safari and UC browser.

5.10 Privacy

5.10.1 Reports cannot be accessed by end users who are not given access to the reports

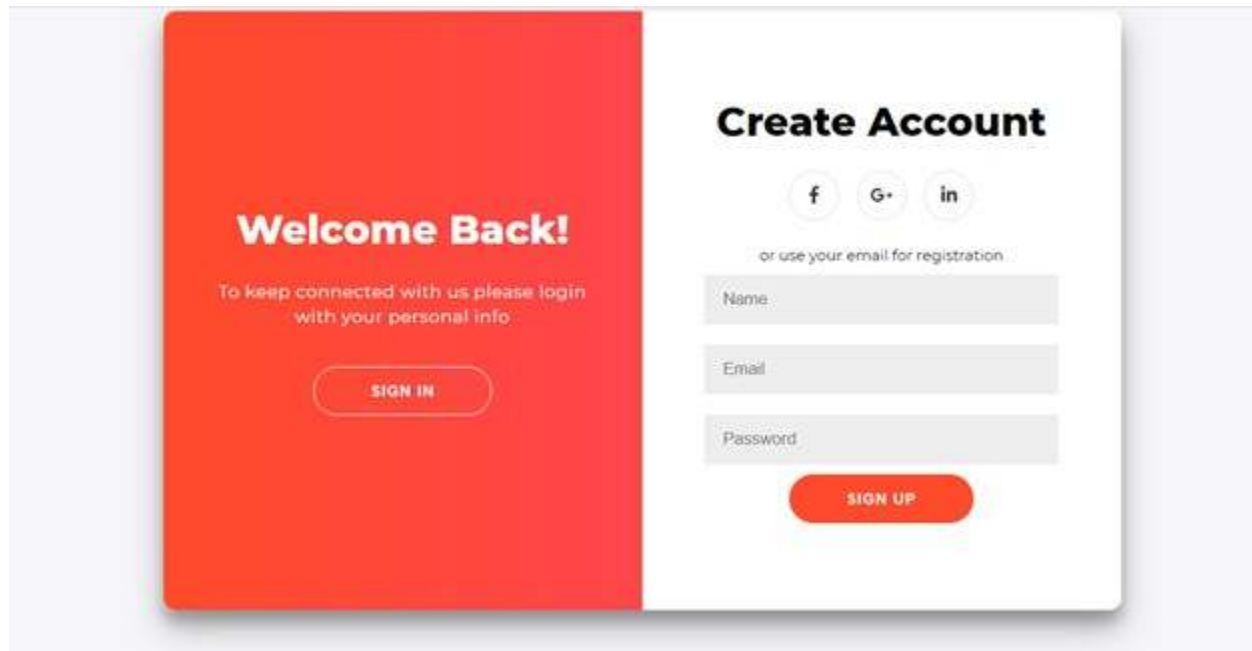
5.10.2 No two users would be able to see their problems.

5.10.3 Every problem registered by would have unique id.

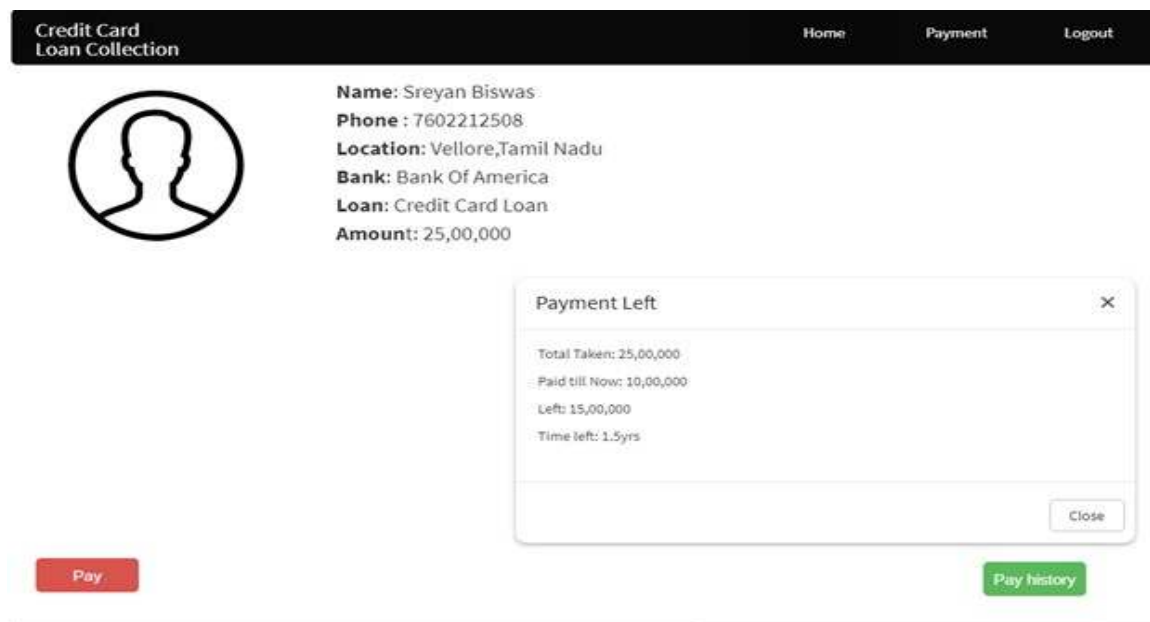
6. Prototypes

(18BCB0070-SREYAN BISWAS)

6.1 LOGIN PAGE







6.2 Installment Payment Page -




6.3 PAYMENTS PAGE - 18BCB0070-SREYAN BISWAS

Payment Details

CARD NUMBER



EXPIRATION DATE

MM / YY






CVV CODE

CVC

COUPON CODE

PAY

6.4 Payment Defaulters Page - 18BCB0070- SREYAN BISWAS

Credit Card Loan Collection			Logout
	Name: Sreyan Biswas Phone: 7602212508 Loan Due: 3 months	<input checked="" type="radio"/> In 2 months <input type="radio"/> Now	
	Name: Aditi Goel Phone: 7676217383 Loan Due: 6 months	<input type="radio"/> In 2 months <input checked="" type="radio"/> Now	
	Name: Jhankar Modak Phone: 8923456854 Loan Due: 3 months	<input type="radio"/> In 2 months <input checked="" type="radio"/> now	
	Name: Anupama Gautam Phone: 8734798099 Loan Due: 9 months	<input checked="" type="radio"/> In 2 months <input type="radio"/> now	
	Name: Shreya Pandey Phone: 8977769767 Loan Due: 9 months	<input type="radio"/> In 2 months <input checked="" type="radio"/> now	

6.5 18BCB0157 (AKILAN N) - APPLYING BANKRUPTCY)

BANKRUPTCY APPLY		Home	Logout
CUSTOMER ID	<input type="text"/>		
TYPE OF LOAN	<input type="text"/>		
LOAN ID	<input type="text"/>		
LOAN AMOUNT	<input type="text"/>		
BANKRUPTCY REASON	<input type="text"/>		
NEEDED DOCUMENTS	<div><div><input checked="" type="radio"/> AADHAR CARD</div><div><input checked="" type="radio"/> BANK PASSBOOK</div><div><input checked="" type="radio"/> LOAN DETAILS</div><div><input checked="" type="radio"/> VOTER ID</div></div>		
<div>APPLY</div>			

6.6 18BCB0157 (AKILAN N) - APPROVAL BANKRUPTCY

LOAN OFFICER APPROVAL		Home	Logout
LOAN ID	<input type="text"/>		
CUSTOMER ID	<input type="text"/>		
LOAN APPLY DETAILS	<input type="text"/>		
STATUS	<input type="text" value="CORRECT"/>		
CHECKING			
REASON	<input type="text"/>		
<div><div>VERIFIED</div><div>APPROVAL</div></div>			
Application move to the bank manager			

6.7 18BCB0157 AKILAN N (BANKRUPTCY STATUS)

BANKRUPTCY STATUS CHECKING

[Home](#)[Logout](#)

CUSTOMER ID

PASSWORD

☒ CheckBox

STATUS CHECKING

E Mail send to the customer

6.8 Deceased Credit Card Loans Main Page - 18BCB0056- SAARTHAK AGARWAL

Deceased Credit Card Loans

[Home](#)[Logout](#)

For Viewing Loan Borrower and their Decease Status ->

View Status

For Updating Borrower Details In case of demise of existing borrower ->

Update Borrower

For viewing status report of loan recovery ->

Deceased Loan Report


[<- Back](#)

6.9 Borrower Deceased Status - 18BCB0056 - SAARTHAK AGARWAL

Deceased Credit Card Loans		Home	Logout
Deceased Loan Repayment Status			
Total Installments	Borrower ID	Amount Borrowed	Amount Repaid
50	123	5000000	2000000
200	432	200000000	10000000
10	768	1000000	400000
12	988	1200000	300000

70

Auction
Home
Payment
Logout




22KT Gold Necklace Set
 Base Price:
2.5 Lakhs
 Current Bid Price:
3 Lakhs

Bid

Active Users

- Sara Porter
- Frank Riley
- Elizabeth Rivera
- Carol Murphy
- Kevin Jones
- Jessica Meyer
- Bobby Weber
- Harry Matthews
- Joe Garrett
- Julie Porter
- Amber Foster
- Larry Wallace
- Fionna Rivera
- Ronnie Guerrero
- Mary Ann Watk



44KT Gold Necklace Set
 Base Price:
5 Lakhs

6.12 Auction Notification- 18BCE2042 - PRANEETH

Auction Notification				Home	Logout
3/02/2020					
Title	Ref No.	End of bidder Reg	Bid Opening date		
10 Kg Gold bar	AF2	10/2/2020	12/2/2020		

6.13 Bidder Registration - 18BCE2042 - PRANEETH

6.14

6.15

Bidder Registration

Home Logout

First Name

Placeholder

Last Name

Placeholder

Email

Placeholder

Mobile No.

Placeholder

Pan Number

Placeholder

Aadhar Number

Placeholder

Address

Placeholder

Date of birth

Placeholder

SUBMIT

Reports/Statuses for Jewel & Credit

Home Customer Collector Relation Logout

Customer Collector Report

☐ View Individual collector report

☒ Collectors report for Jewel loan


☐ Collectors report for Credit loan

From : 06 Jan 2020

To : 07 Feb 2020

#	Total Amount Collected	Collector Name
1	100000	Akancha Agarwal
2	90000	Khushi
3	36000	Happy
4	1000	Anshman Agarwal
5	0	Devansh

6.16 Auction report for Jewel Loan - 18bce2313 vaibhav singh



User Type: Collector
User ID: 123LO

Auction System

Log out

HOME

CREDIT CARD
LOANS

JEWEL LOANS

Enter Date to view the report
DD/MM/YYYY

View Report

6.17 ANUPAMA GAUTAM (18BCB0019)

LOAN STATUS

BANKRUPTCY PERMISSION

NAME:

LOAN AMOUNT TO BE PAID:

LOAN ID:

MONTHLY INCOME:

ORIGINAL ADDRESS :

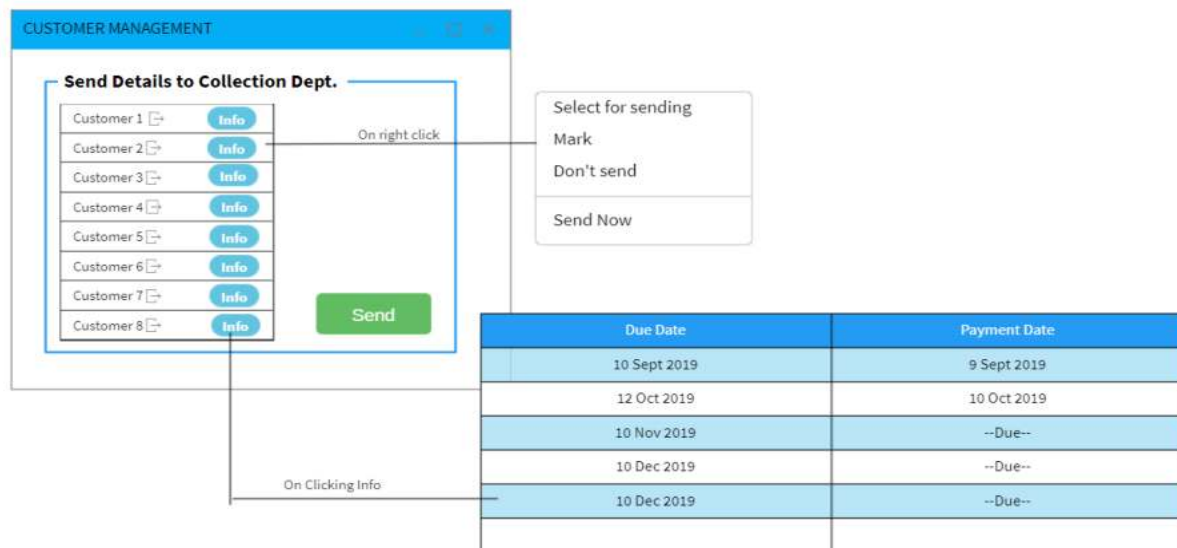
REASON FOR BANKRUPTCY:

AADHAR NO:

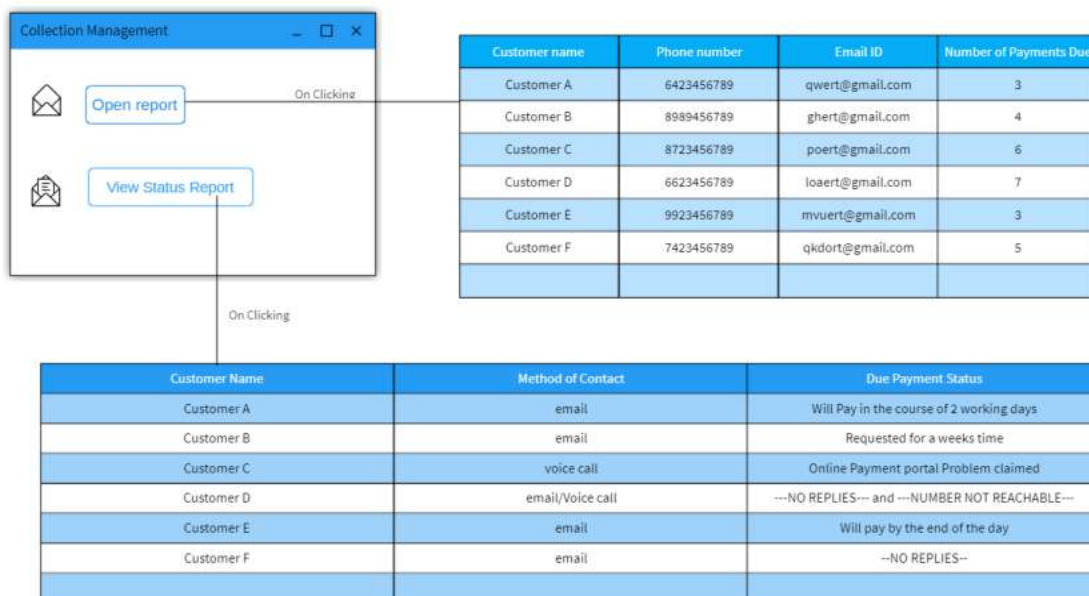
BANKRUPTCY LEGAL PROOF:

YOUR PERMISSION STATUS WILL BE SENT TO YOUR EMAIL WITHIN 30 WORKING DAYS.

6.19 CUSTOMER MANAGEMENT BY LOAN OFFICER (SIDDHARTHA MONDAL -18BCB0145)

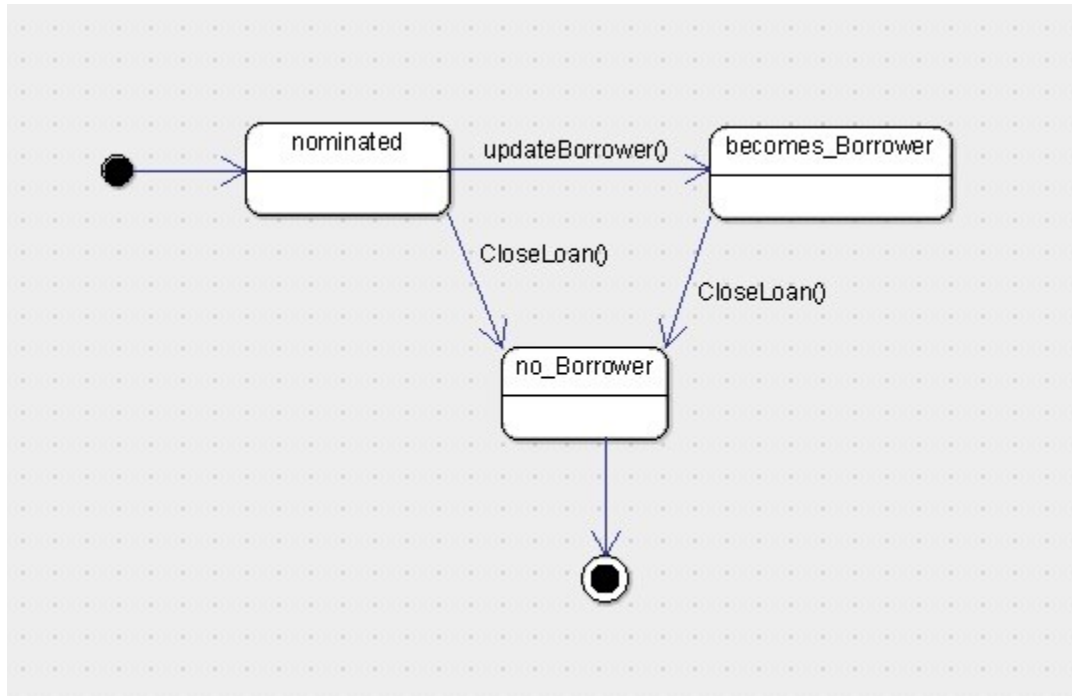


6.20 COLLECTION OFFICER MANAGEMENT WINDOW (SIDDHARTHA MONDAL -18BCB0145)

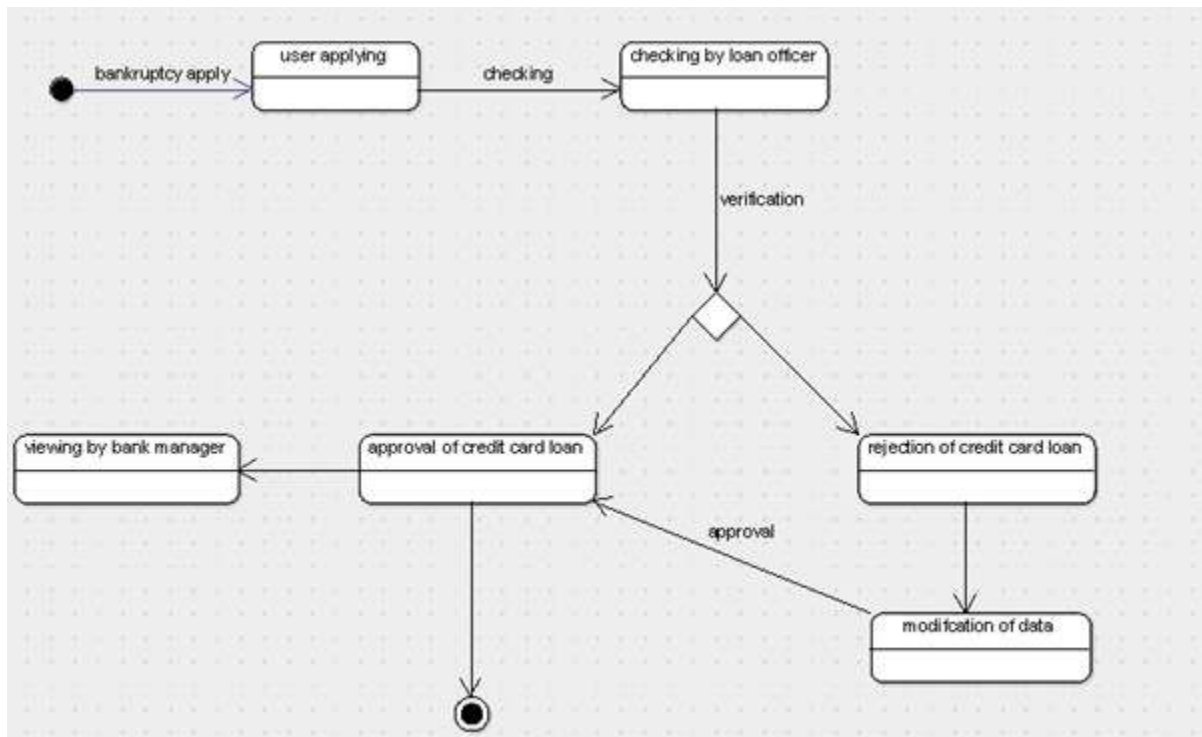


6. DESIGN CONTENTS

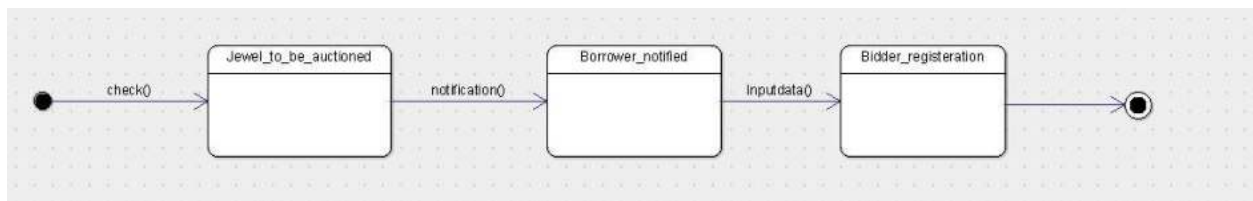
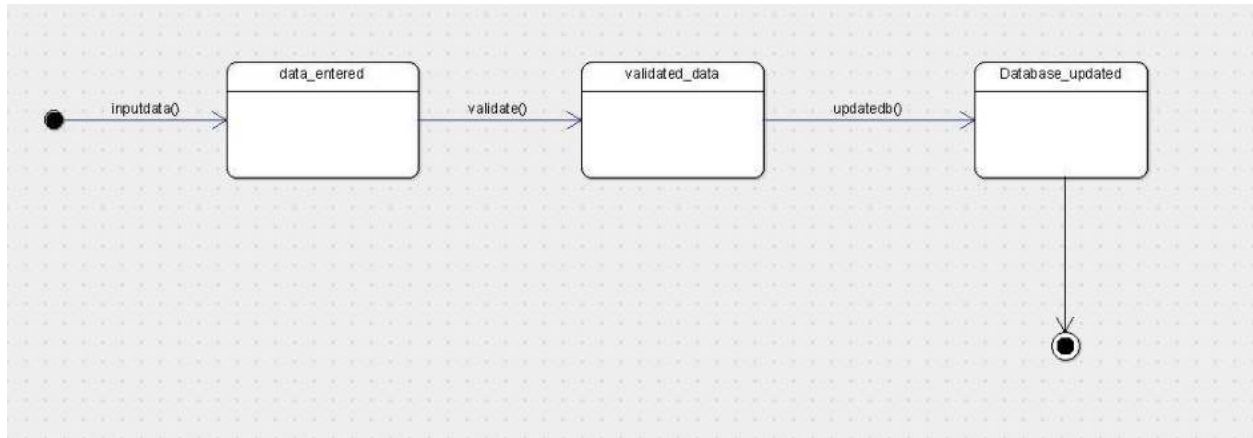




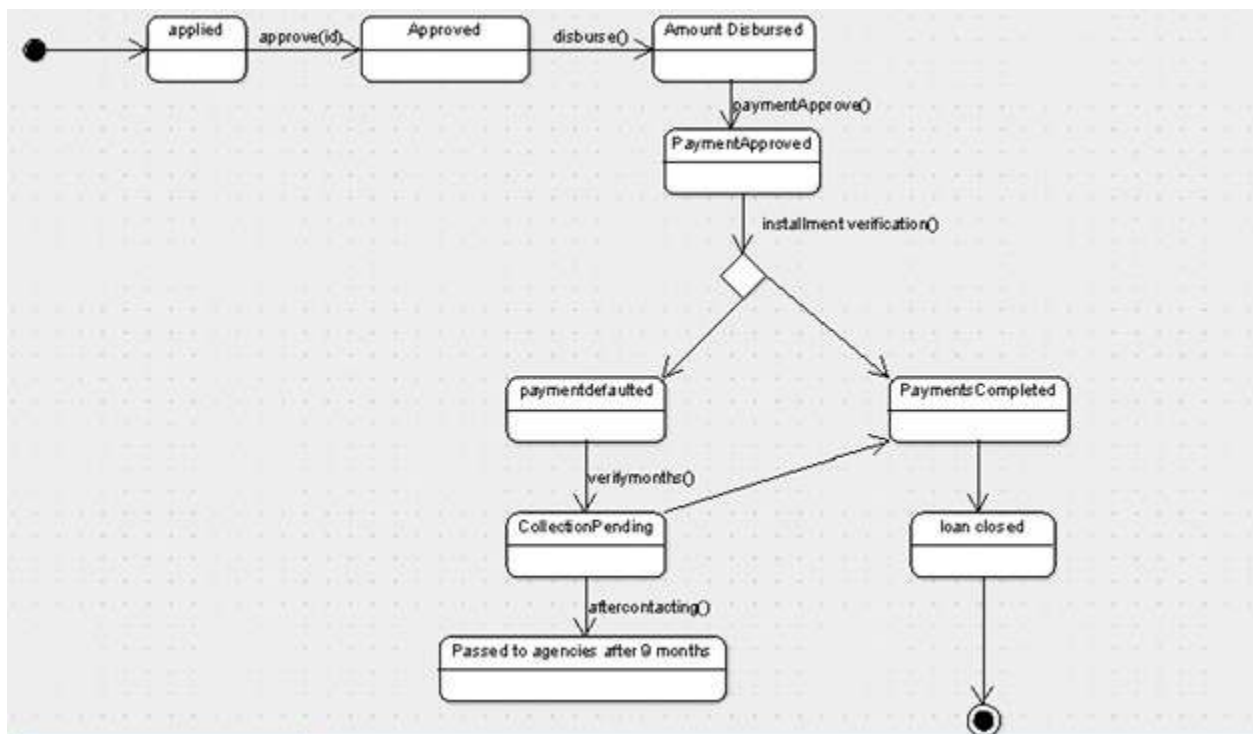
18BCB0019(ANUPAMA GAUTAM)



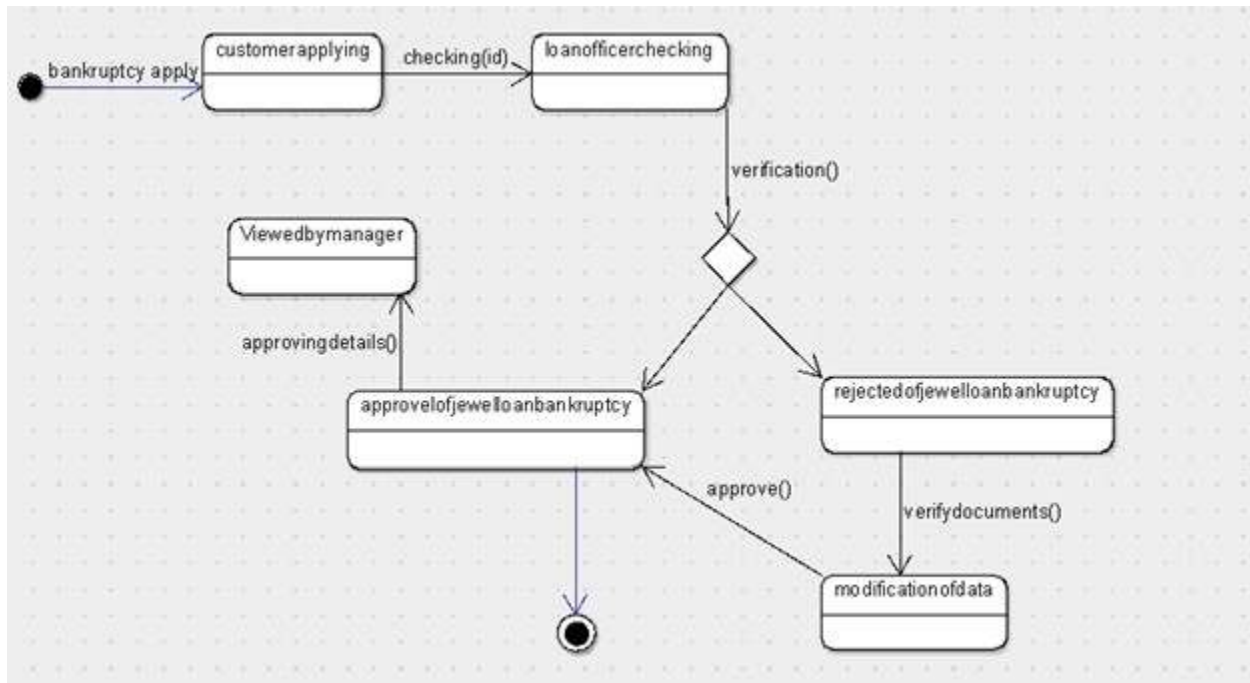
18BCE2042(PRANEETH)



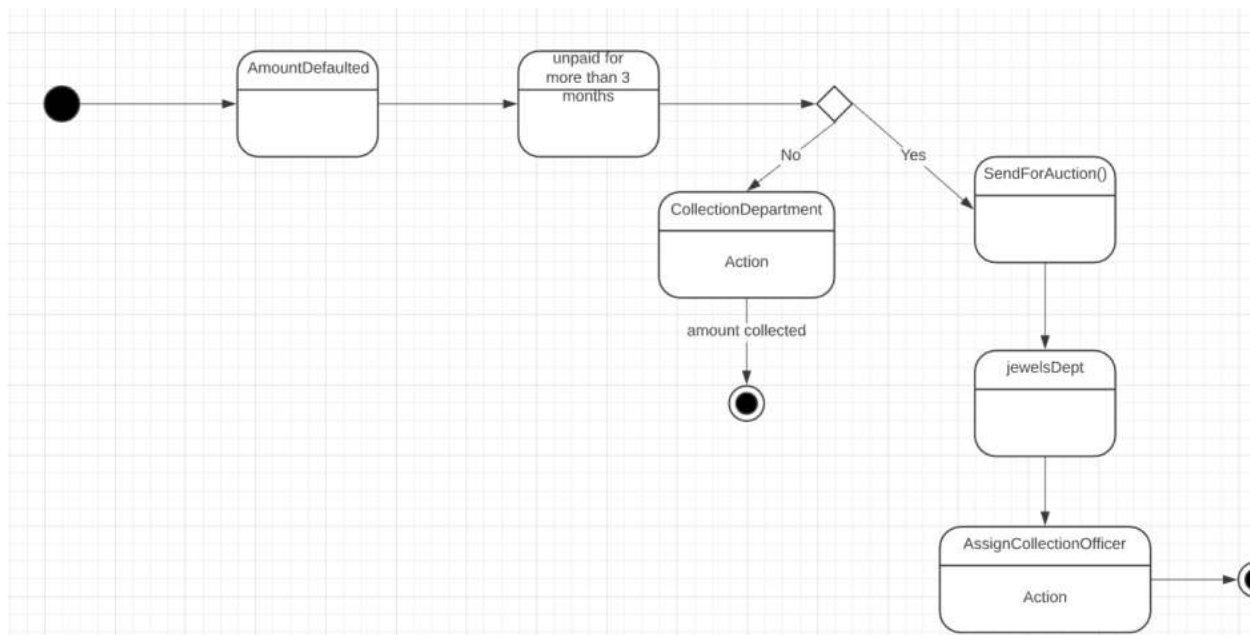
18BCB0070(SREYAN)



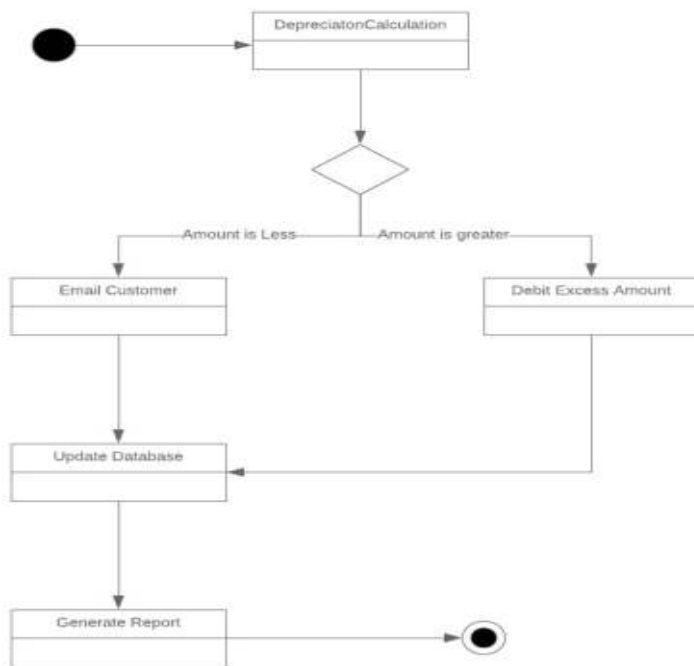
18BCB0157 (AKILAN N)



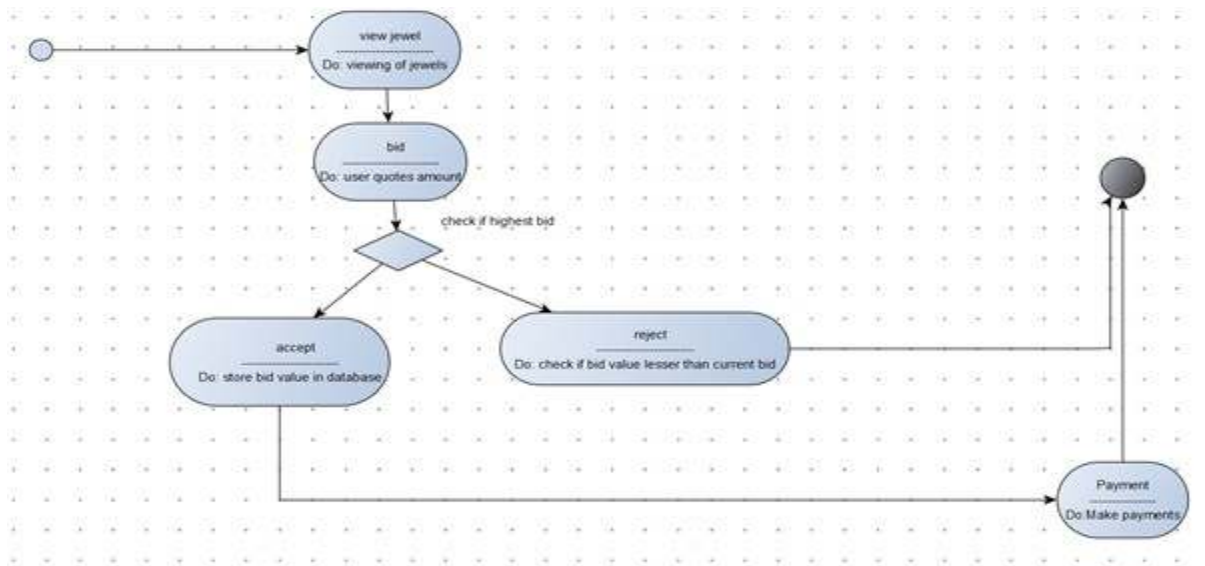
18BCB0145 (Siddhartha Mondal)



18BCE2313 - (VAIBHAV SINGH)

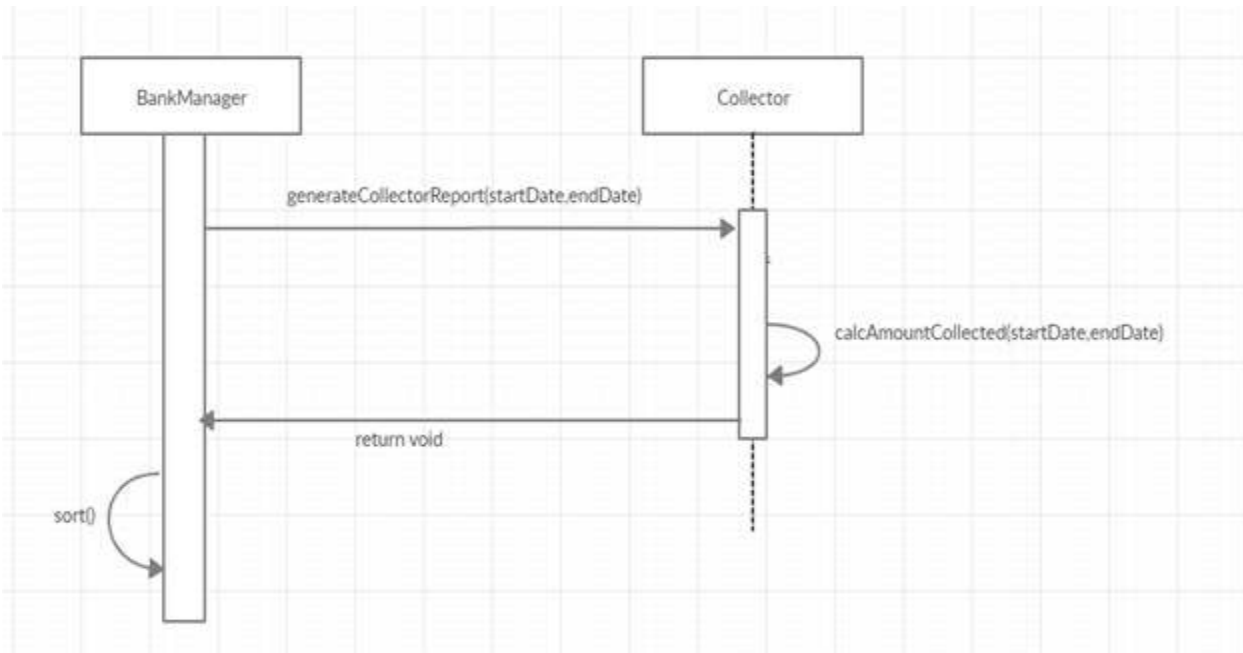


18BCE2190 - (VIDHI AGARWAL)

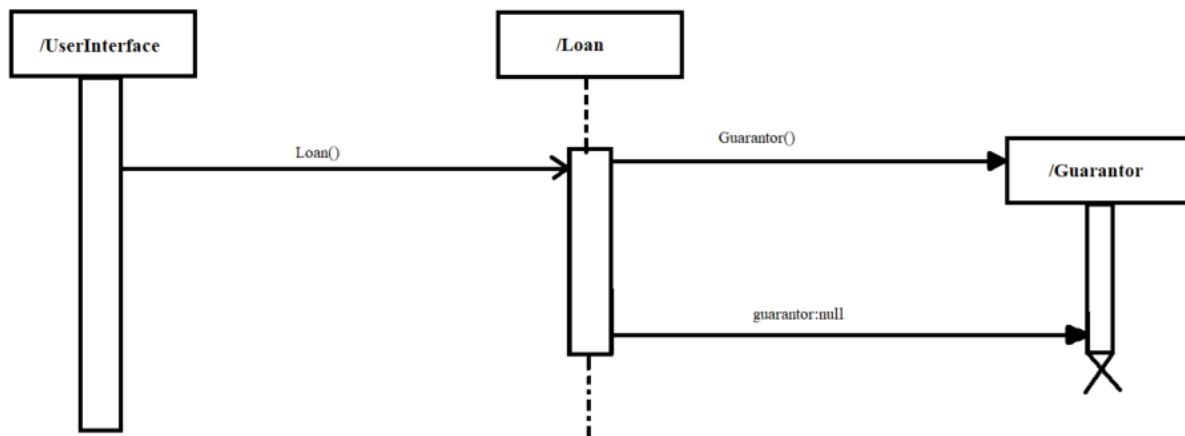


6.3 SEQUENCE DIAGRAM

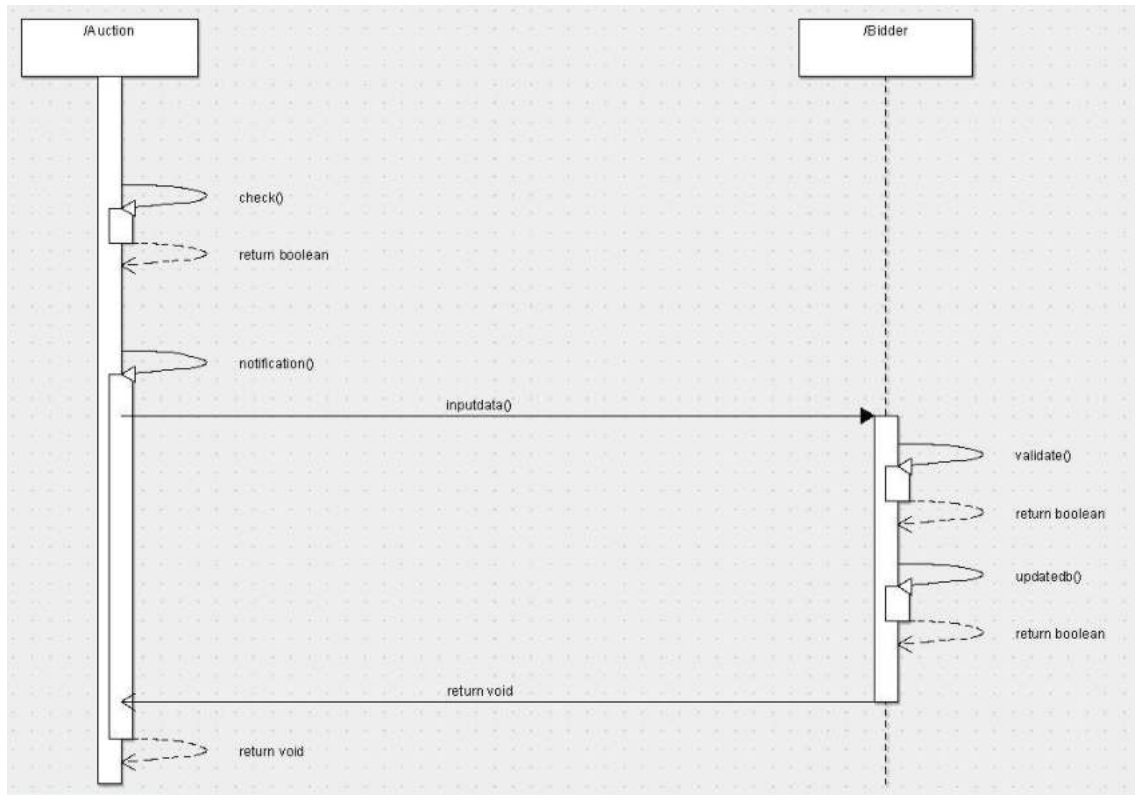
17BCB0016 (AKANCHHA AGARWAL)



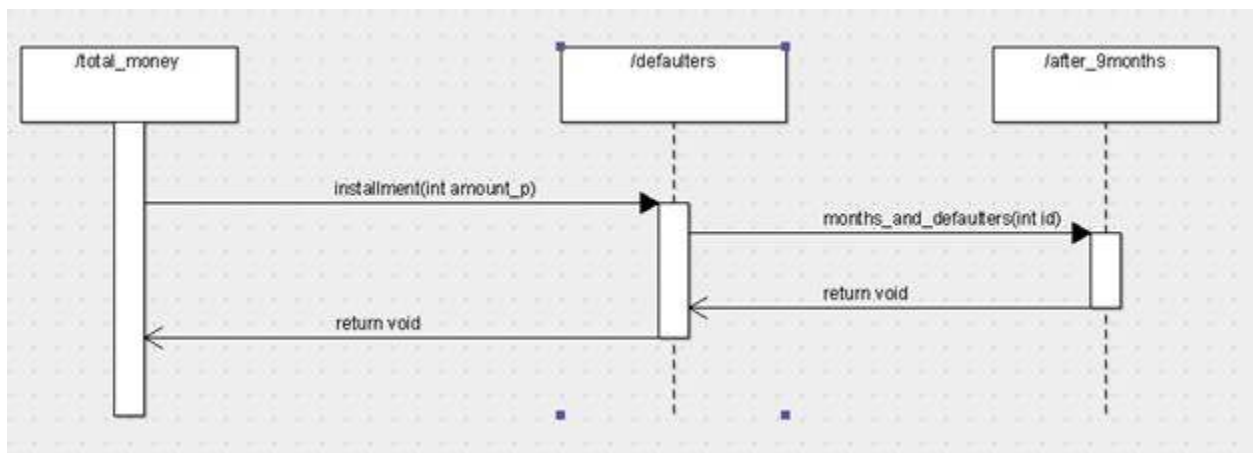
18BCB0056(SAARTHAK AGARWAL)



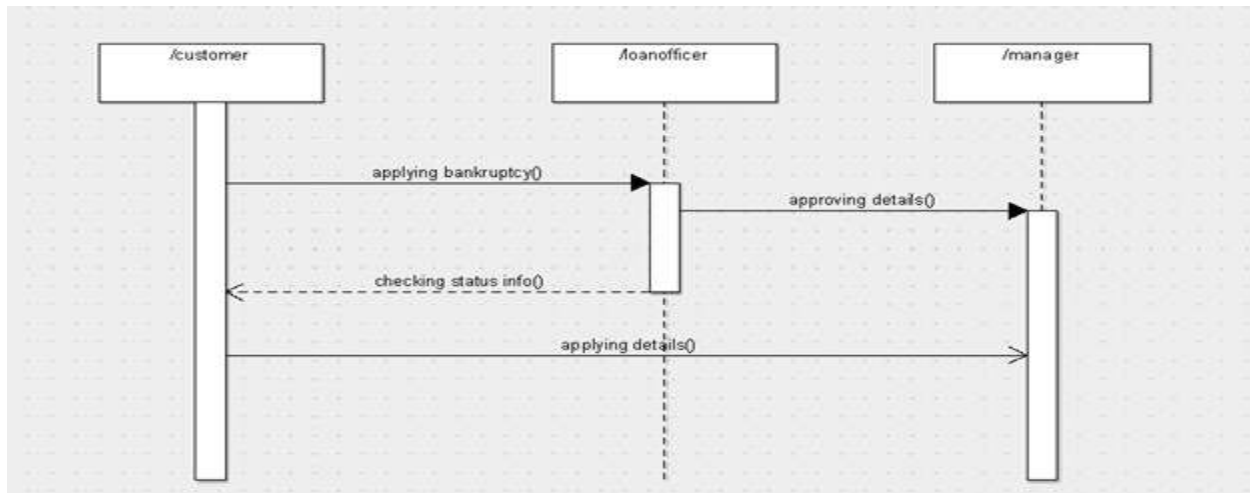
18BCE2042(PRANEETH)



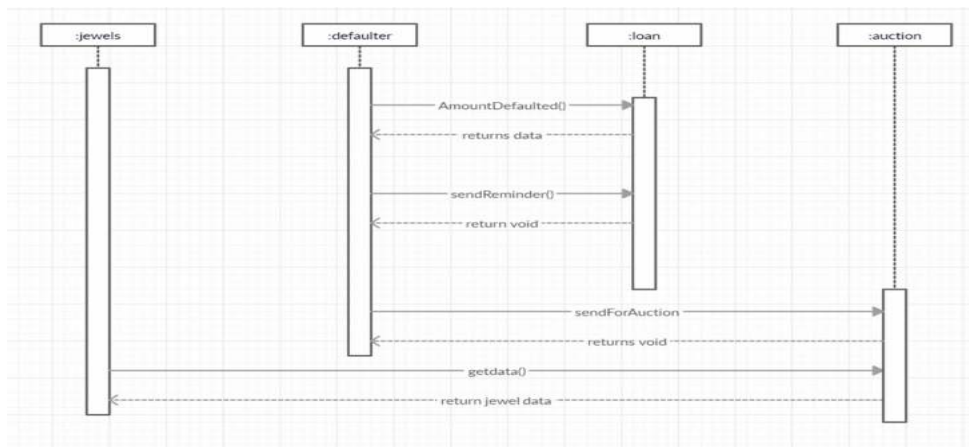
18BCB0070(SREYAN)



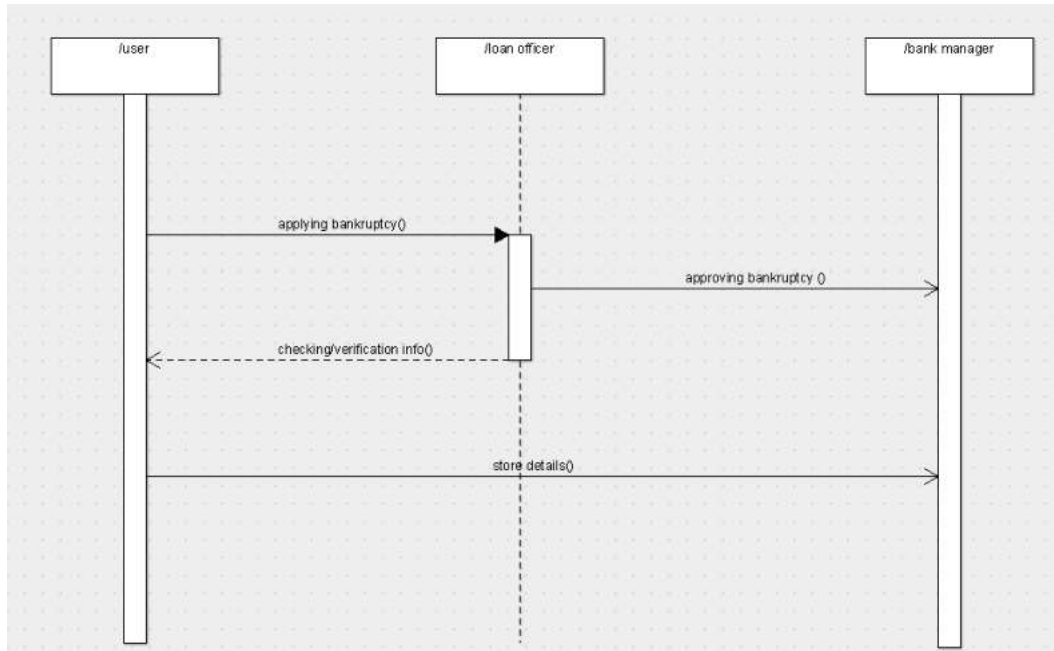
18BCB0157 (AKILAN N)



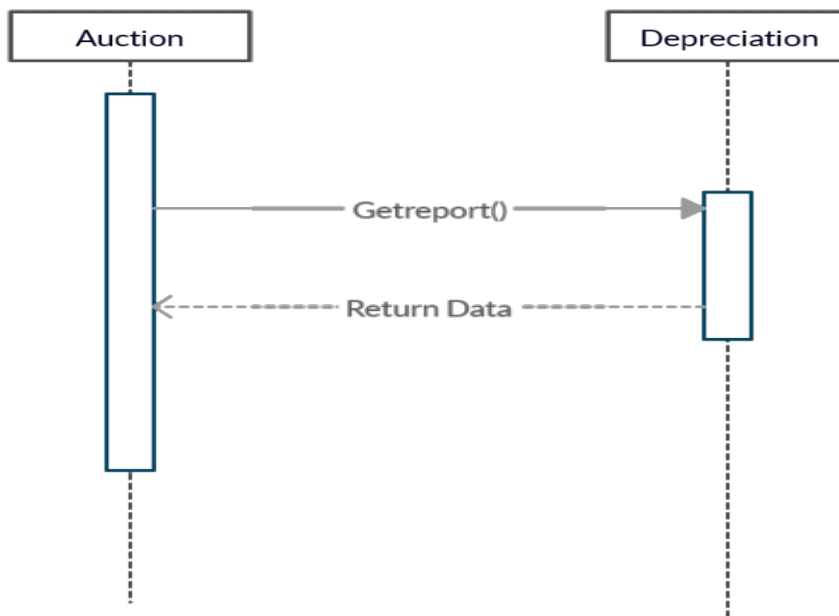
18BCB0145 (Siddhartha Mondal)



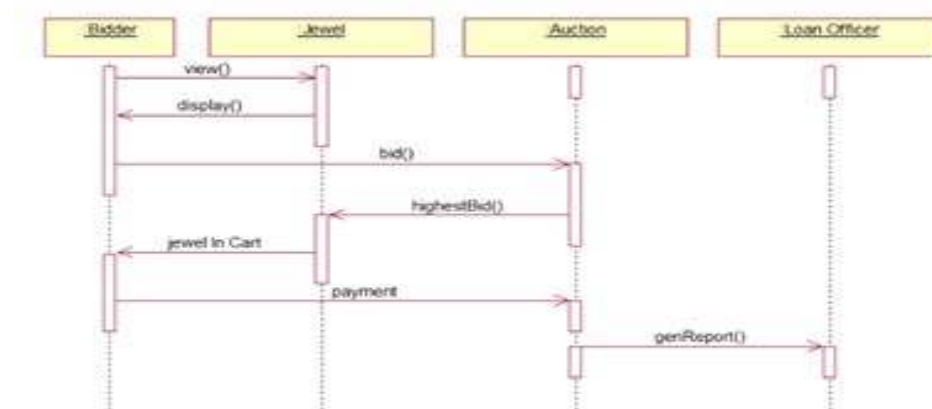
18BCB0019(ANUPAMA GAUTAM)



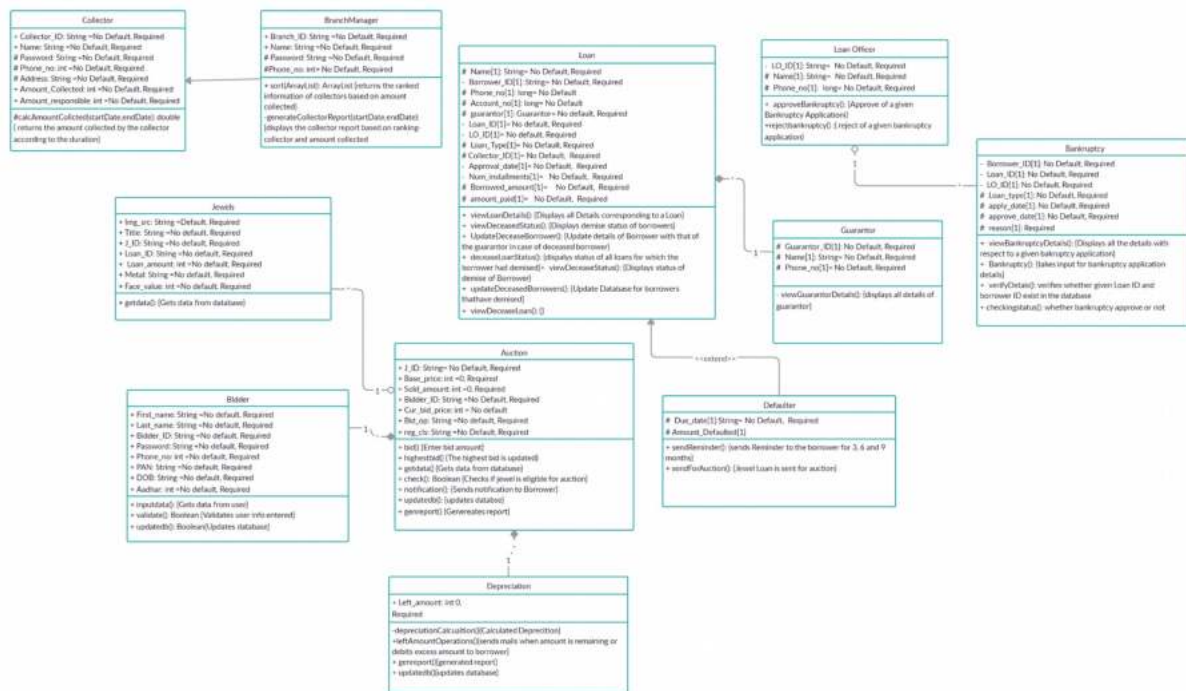
VAIBHAV Singh (18BCE2313)



VIDHI AGARWAL(18BCE2190)



6.4 CLASS DIAGRAM

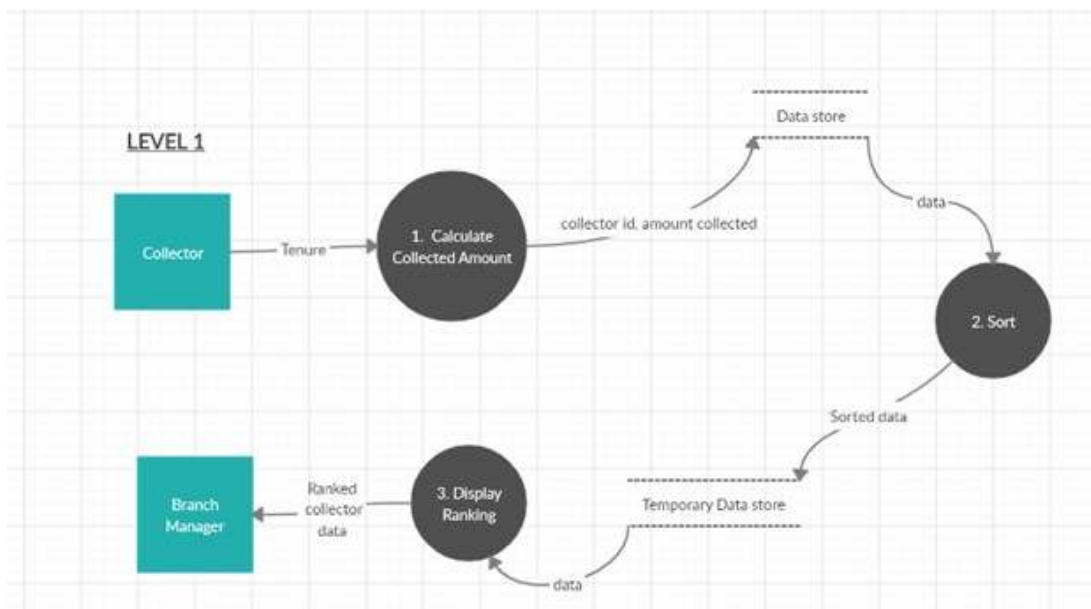


6.5 DATA FLOW DIAGRAM

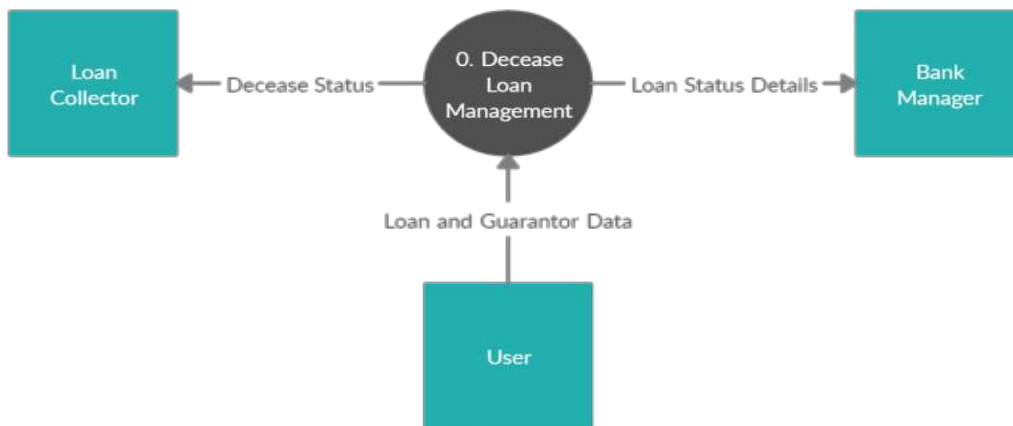
17BCB0016 (AKANCHA AGARWAL) - LEVEL 0



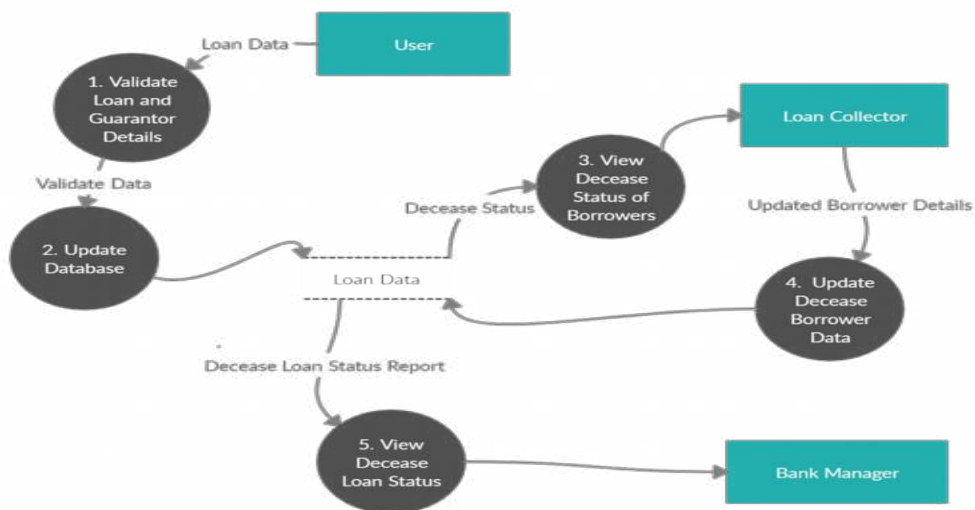
17BCB0016 (AKANCHA AGARWAL) - LEVEL 1



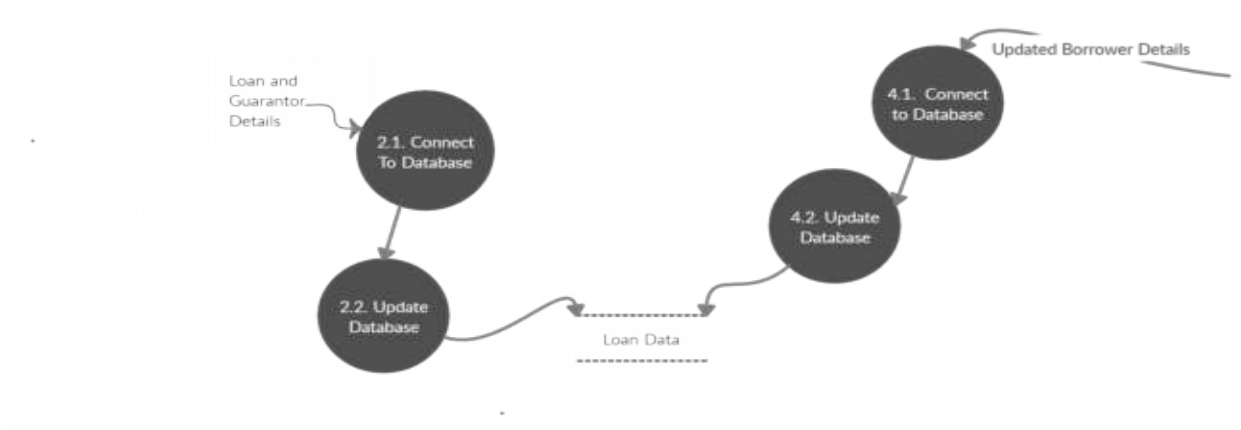
18BCB0056(SAARTHAK AGARWAL) - LEVEL 0



18BCB0056(SAARTHAK AGARWAL) - LEVEL 1



18BCB0056(SAARTHAK AGARWAL) - LEVEL 2

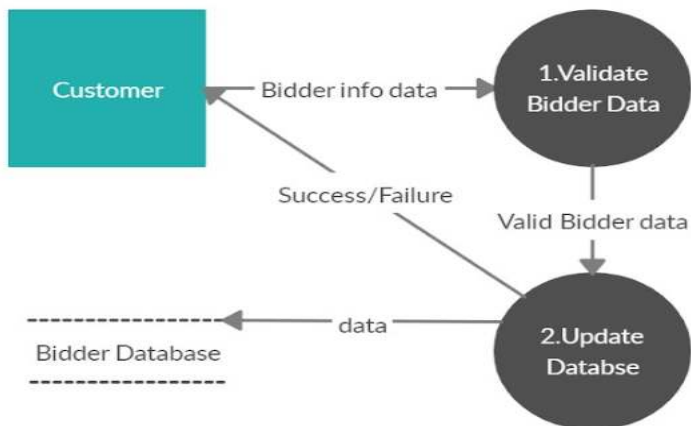


18BCE2042 - PRANEETH

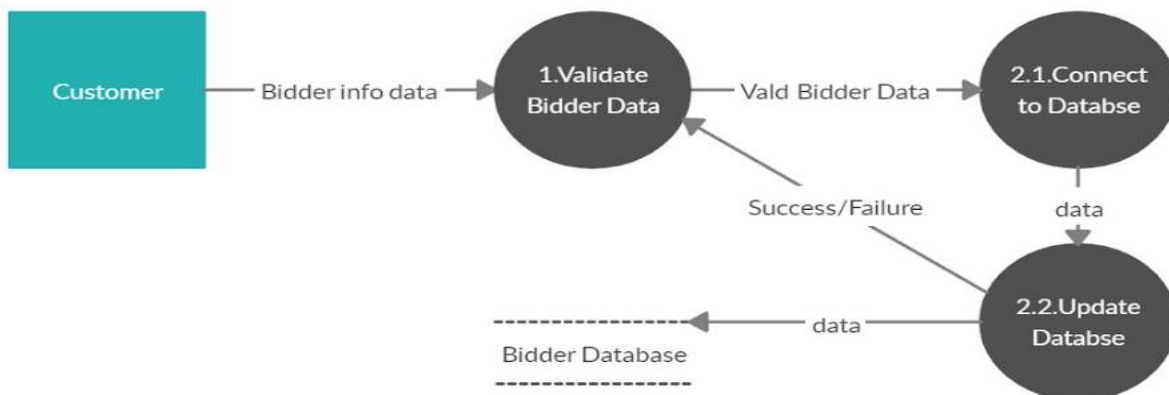
LEVEL - 0



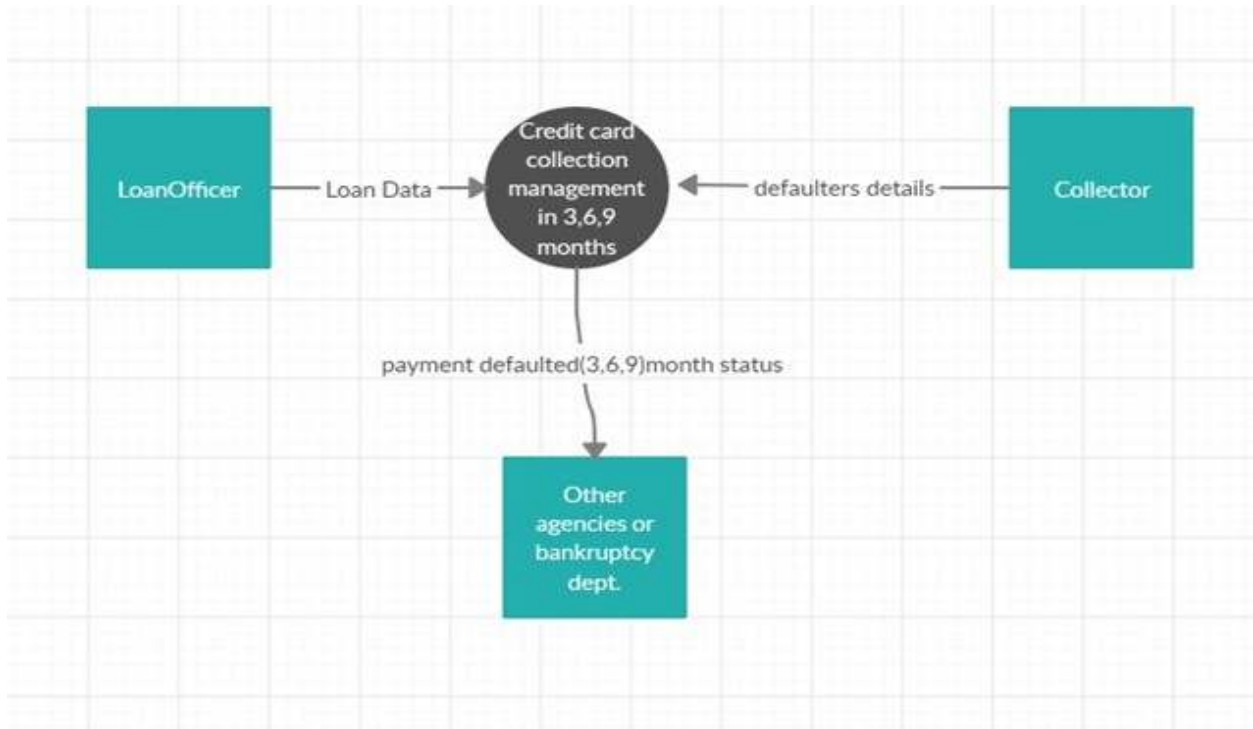
LEVEL - 1



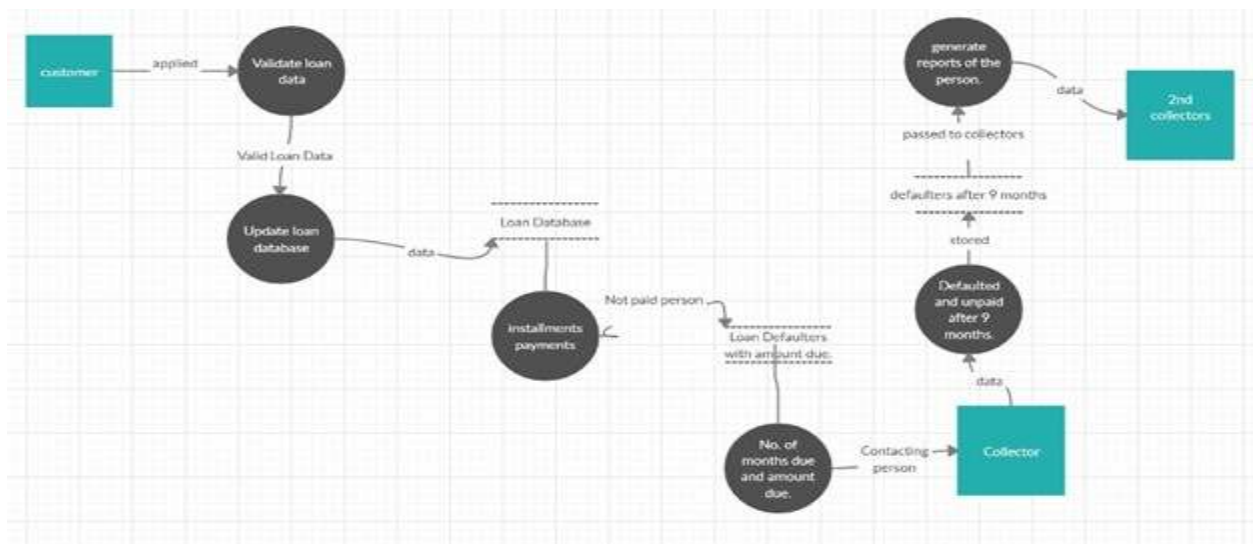
LEVEL - 2



LEVEL 0



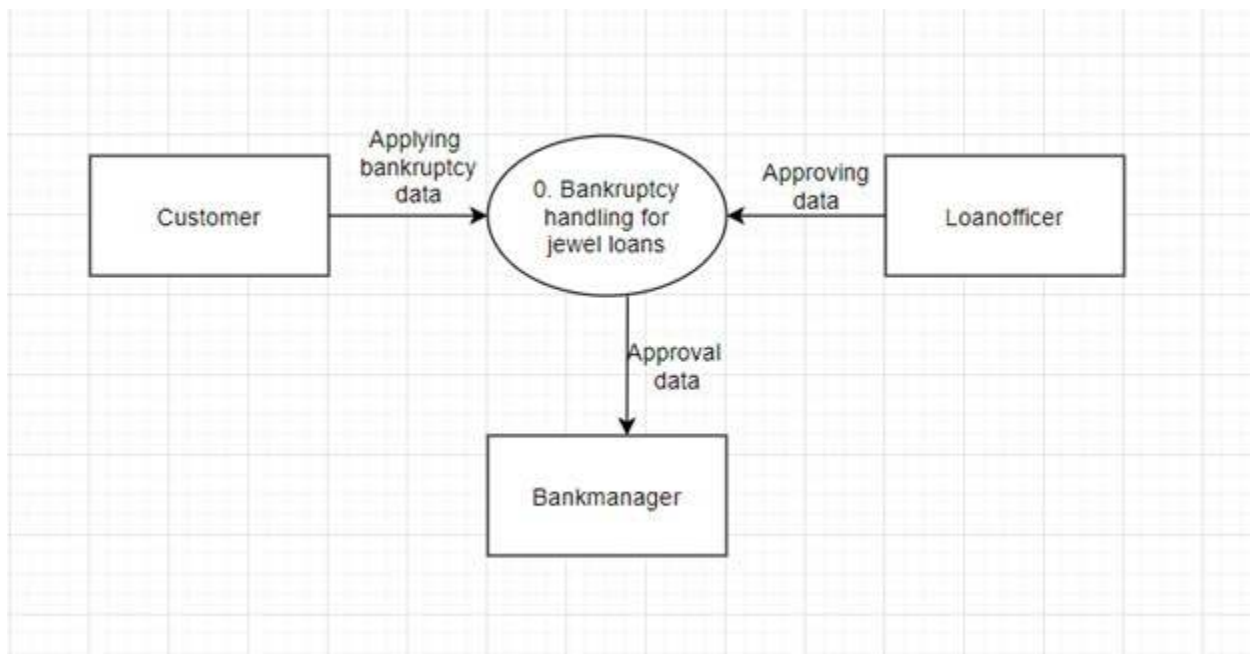
LEVEL 1



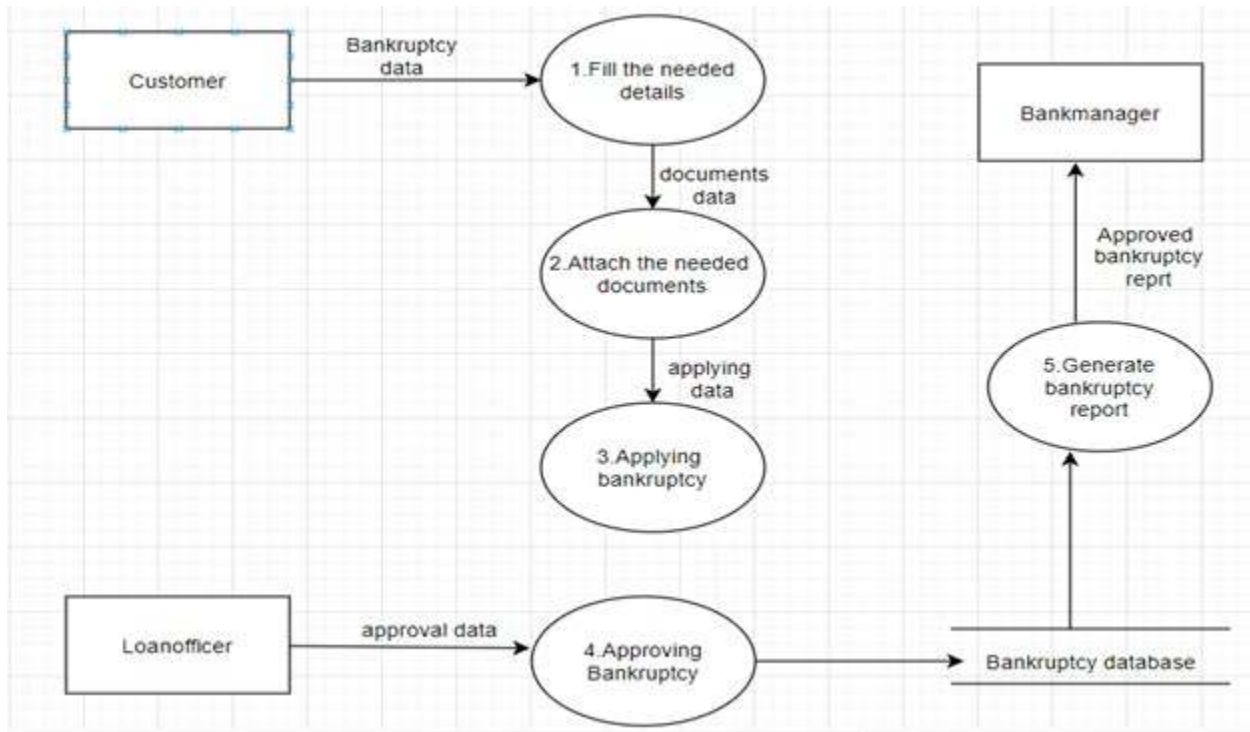
LEVEL 2



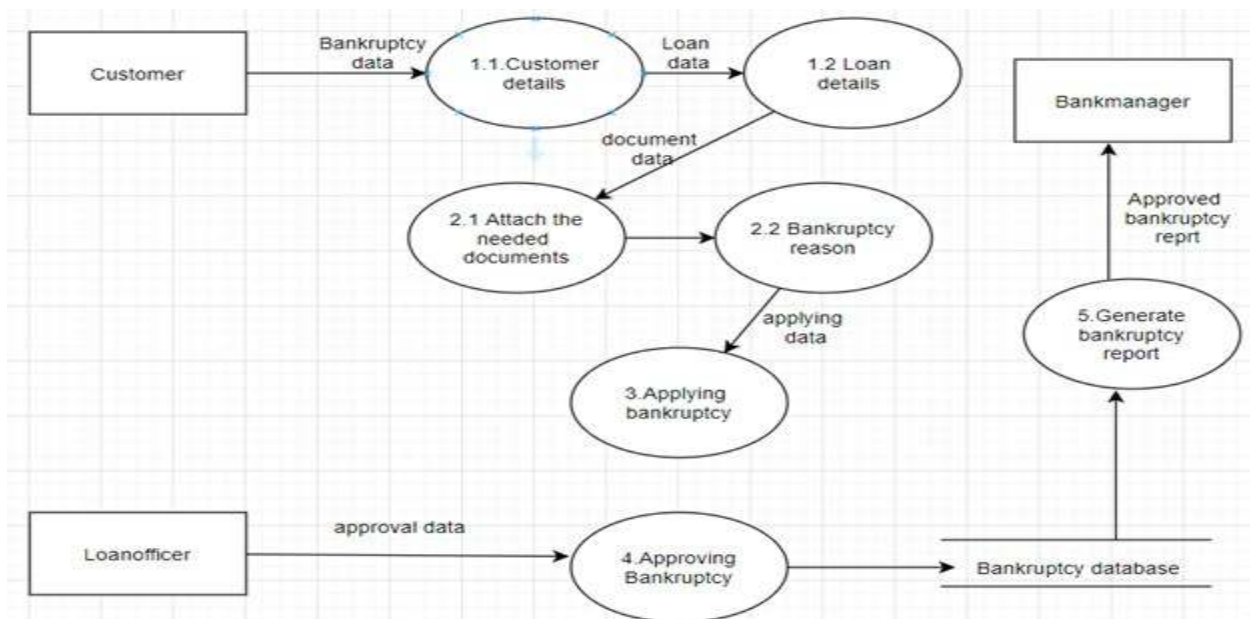
18BCB0157 (AKILAN N) DFD -LEVEL 0



18BCB0157 (AKILAN N) DFD- LEVEL1

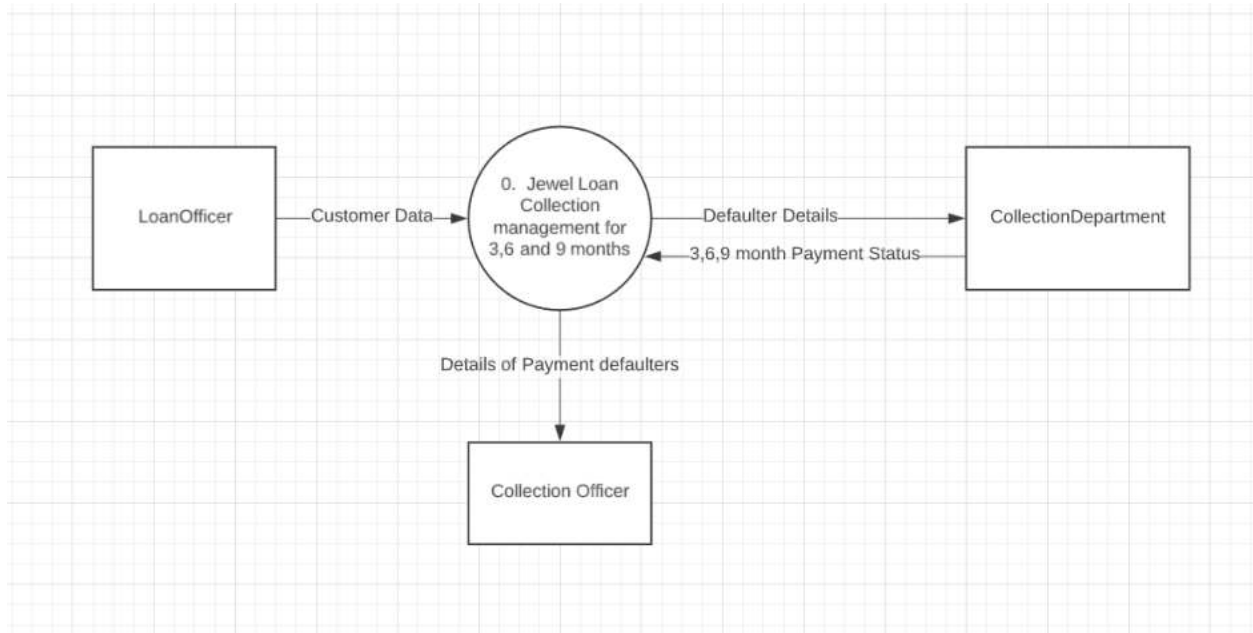


18BCB0157 (AKILAN N) DFD -LEVEL 2

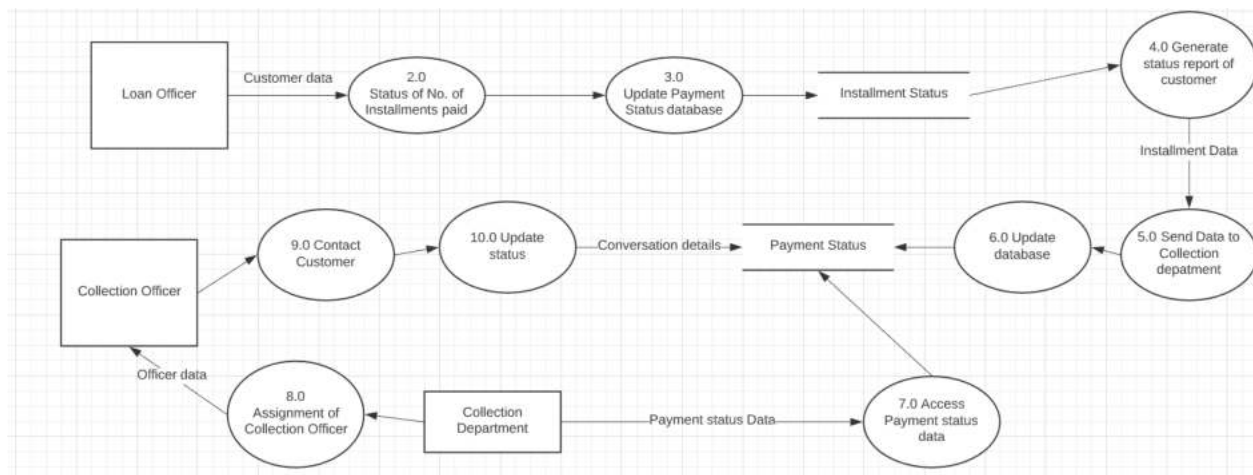


18BCB0145 (Siddhartha Mondal)

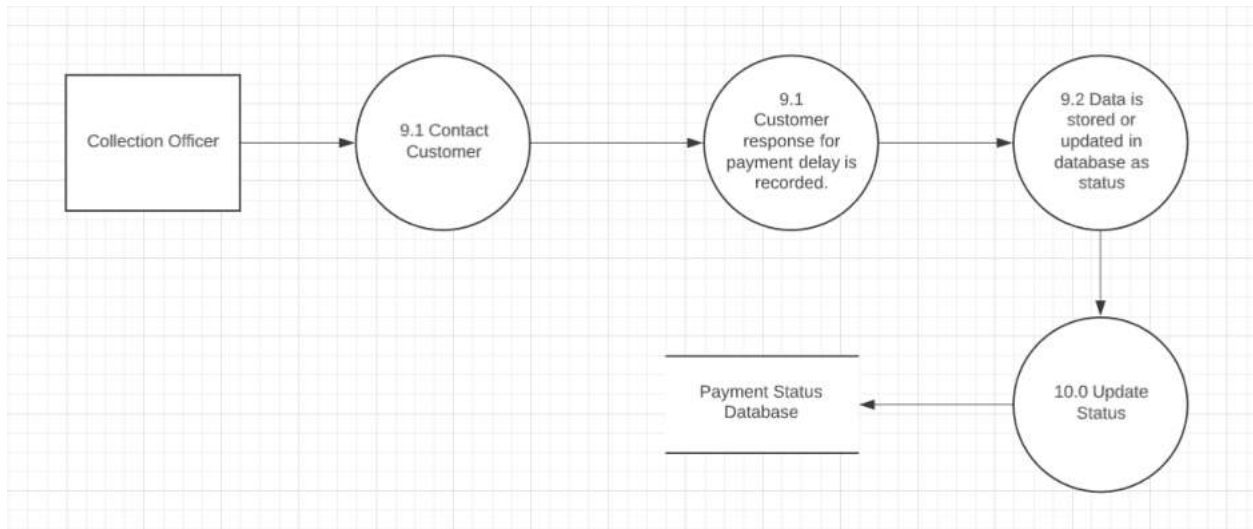
Level 0



Level 1

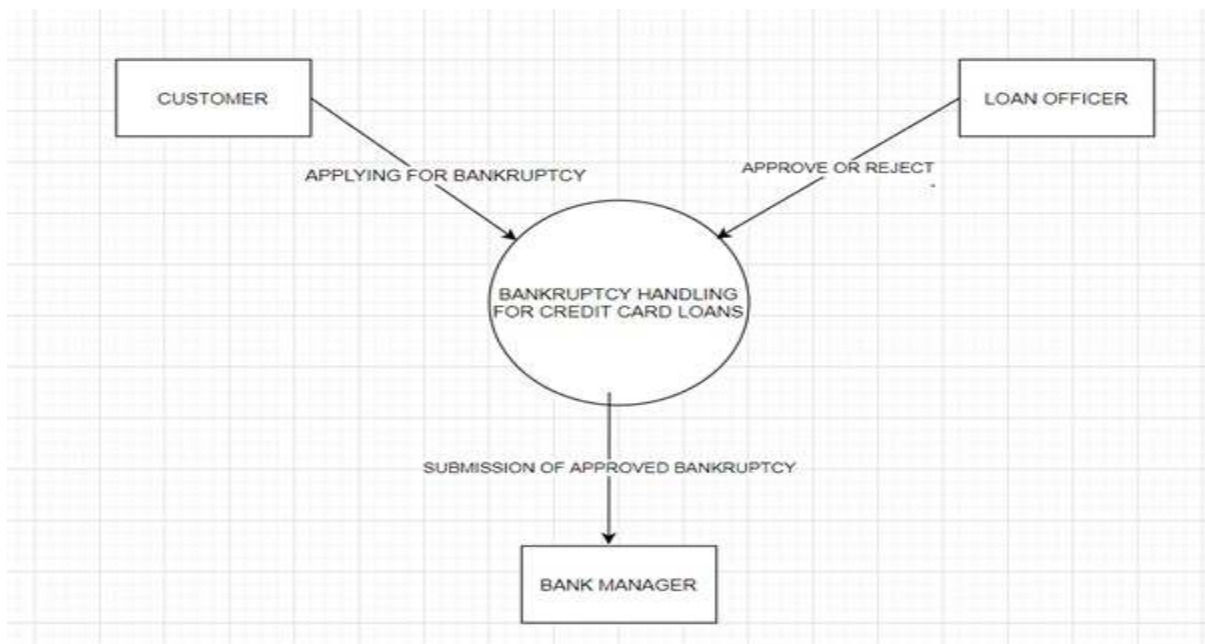


Level 2

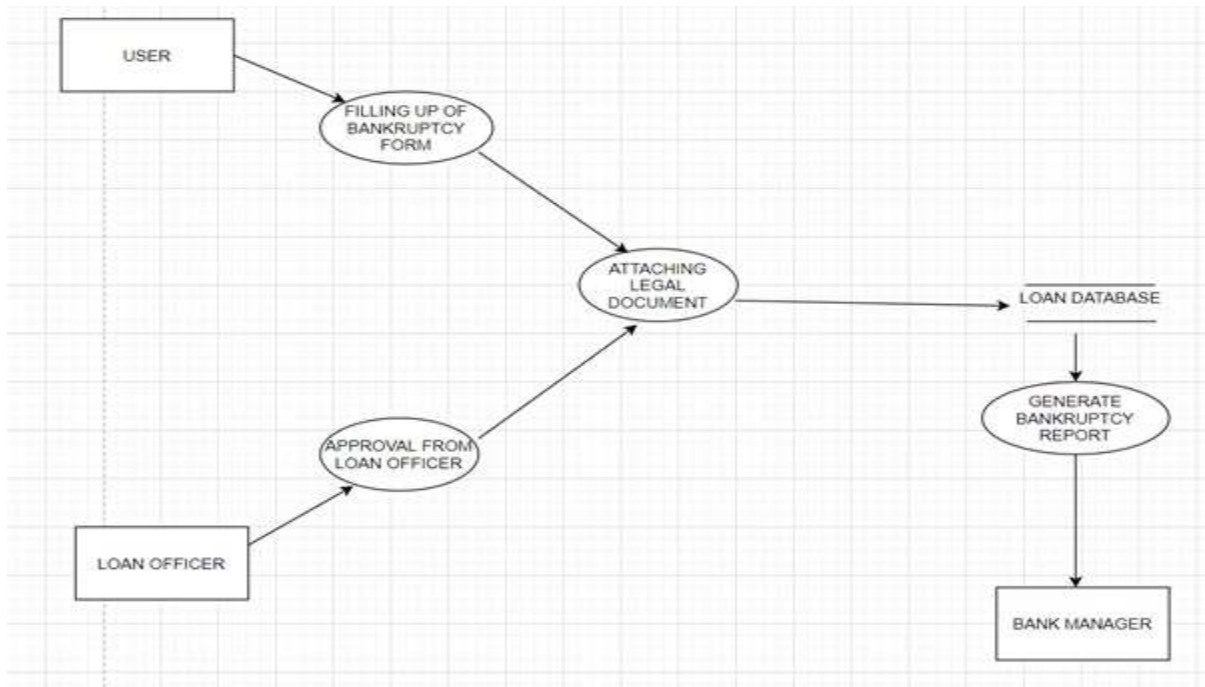


18BCB0019(ANUPAMA GAUTAM)

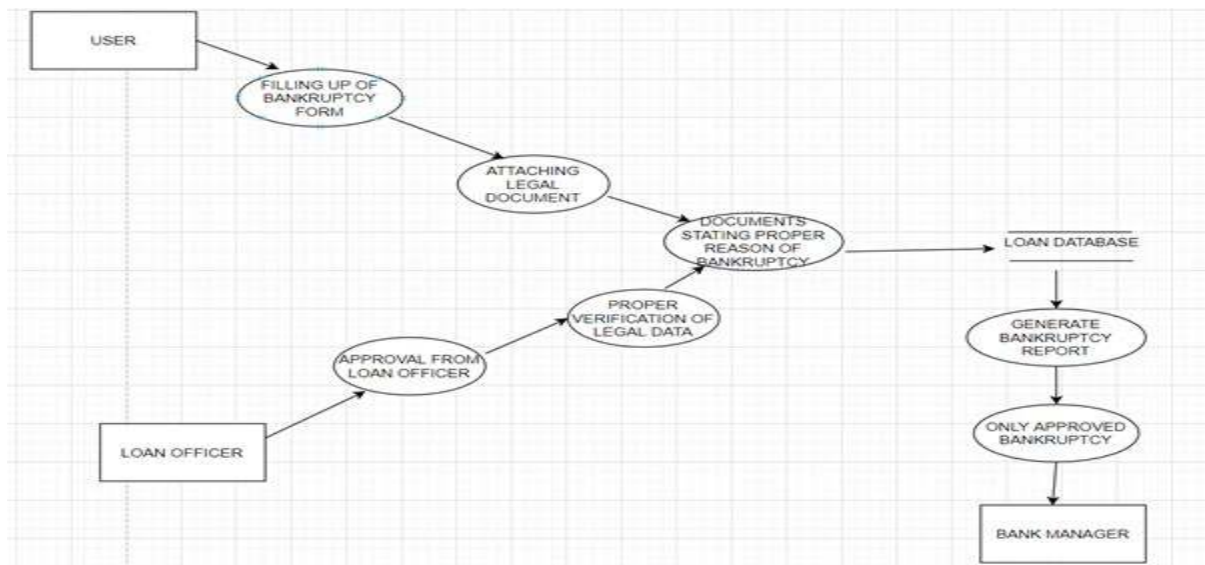
Level 0:



Level 1:

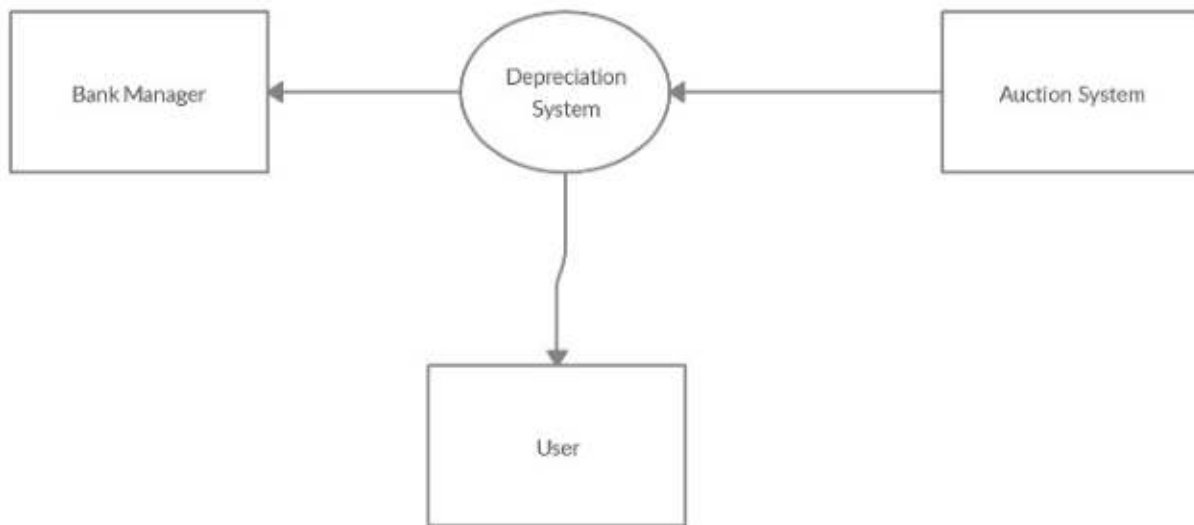


Level 2:

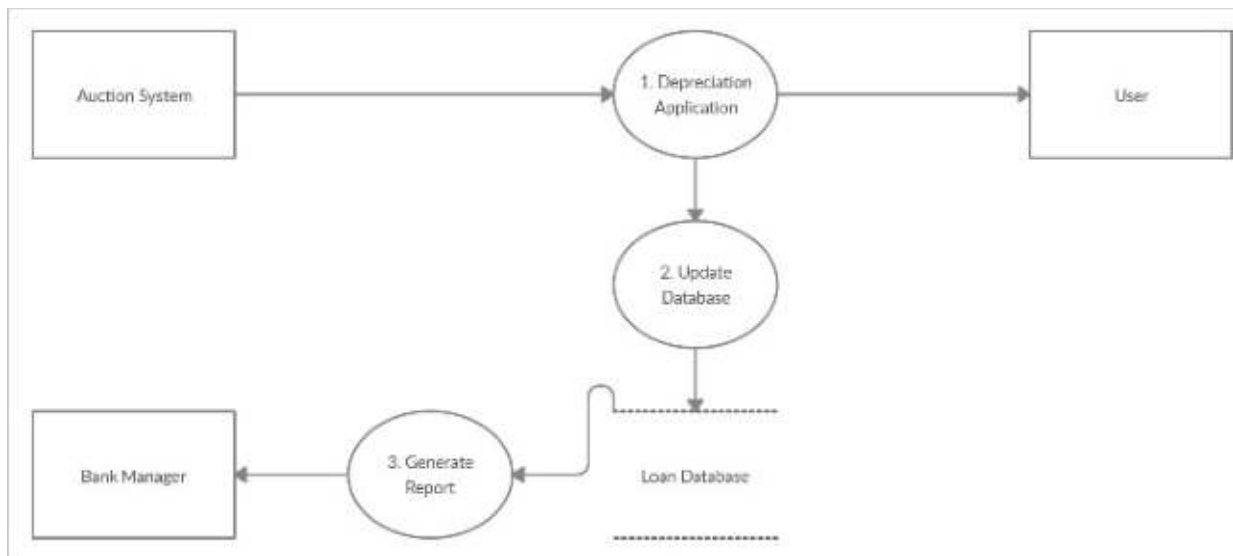


18BCE2313 :- VAIBHAV

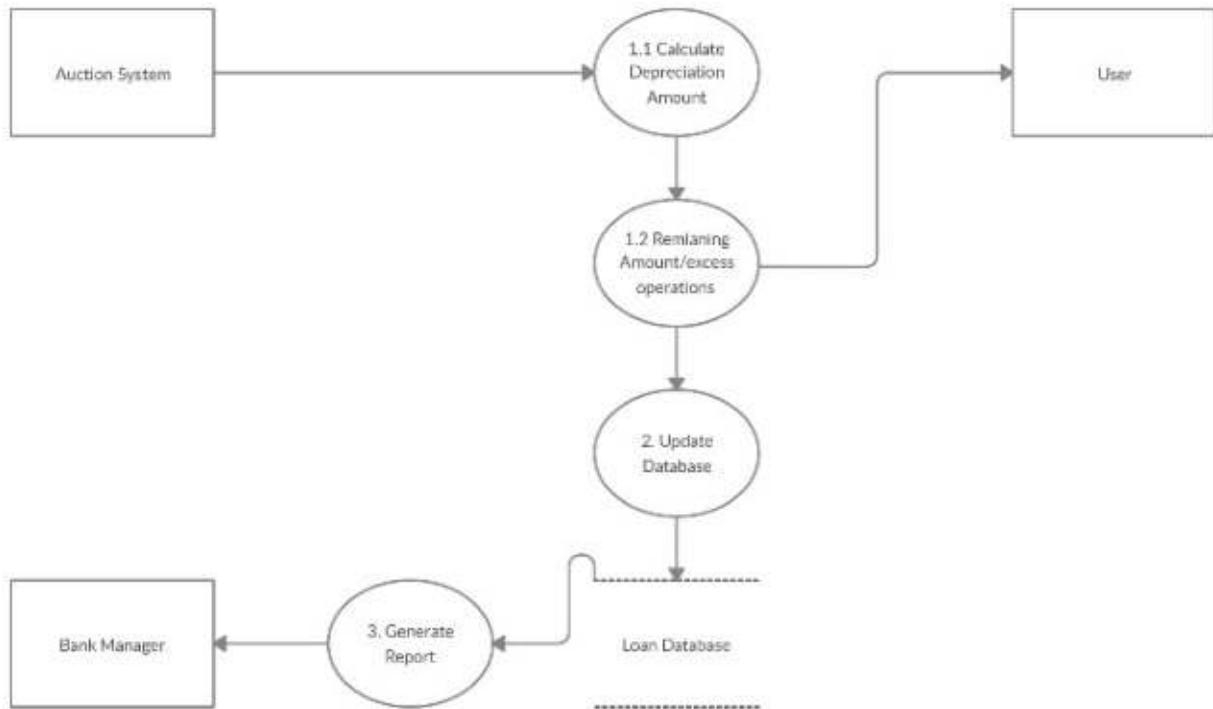
LEVEL 0 DFD



LEVEL 1 DFD

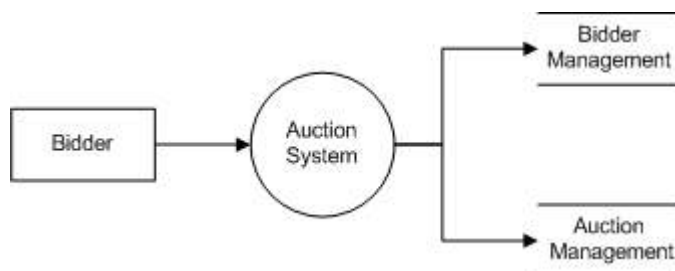


LEVEL 2 DFD

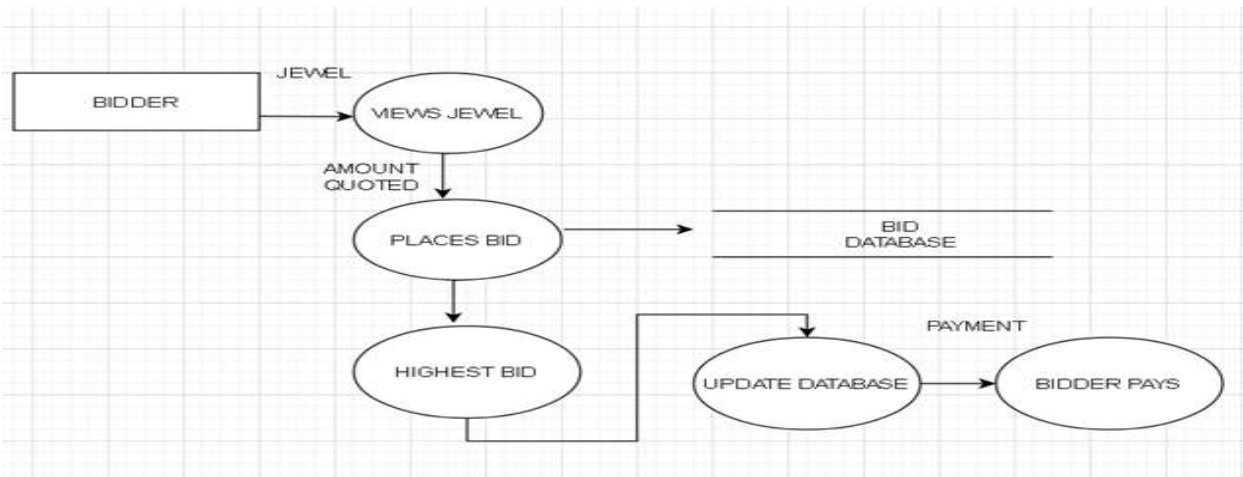


18BCE2190 :- VIDHI AGARWAL

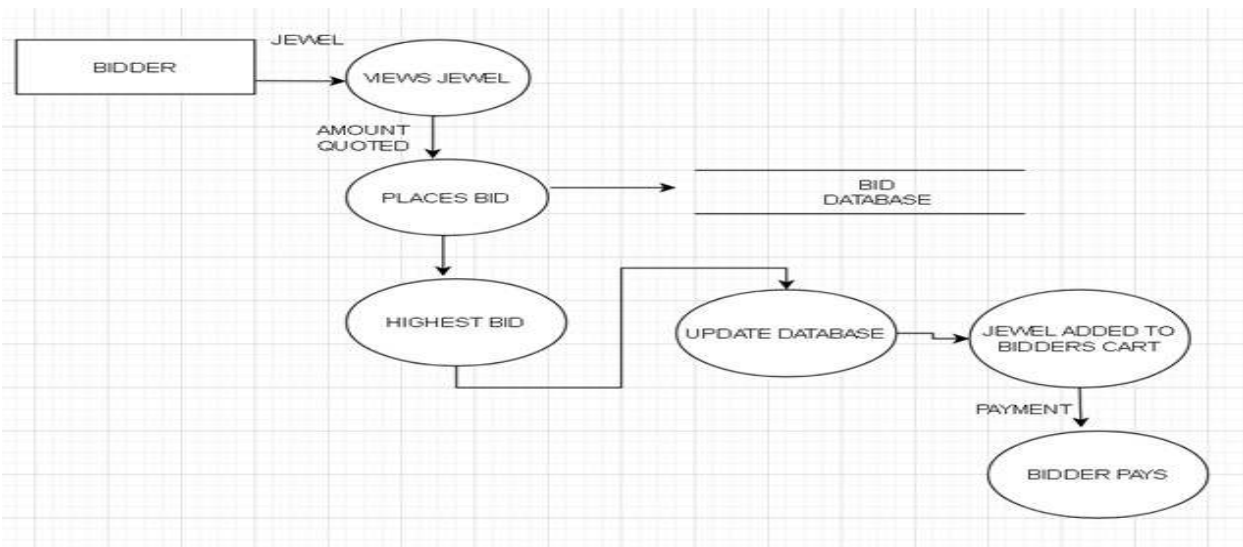
LEVEL 0 DFD



LEVEL 1 DFD



LEVEL 2 DFD



7. Implementation

17BCB0016

AKANCHA AGARWAL

customerCollectorReport.jsp

```
<%@include file="index.jsp" %>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Insert title here</title>

<style type="text/css">

    .jumbotron{

        min-height: 200 px;

        padding:1rem;

    }

</style>

</head>

<body>

    <%

        if(session.getAttribute("username")==null){

            response.sendRedirect("login.jsp");

        }

    %>

    <%

        response.setHeader("Cache-control","no-cache, no-store, must-revalidate");

        response.setHeader("Pragma","no-cache");

        response.setHeader("Expires","0");
```

```
%>

<div class="container">

    <br><br>

    <p class="font-weight-bolder text-left"><h4><u>Customer Collector Report</u></h4></p>

    <div class="container">

        <form action="customerCollectorReport.jsp" method="get">

            <div class="form-check form-check-inline col-md-4 float-right">

                <input class="form-check-input" type="radio" name="options"
id="inlineRadio1" value="1">

                <label class="form-check-label" for="inlineRadio1">View Individual
Collector Report</label>

            </div>

            <div class="form-check form-check-inline col-md-4 float-right">

                <input class="form-check-input" type="radio" name="options"
id="inlineRadio2" value="2">

                <label class="form-check-label" for="inlineRadio2">Collectors report for
Jewel loan</label>

            </div>

            <div class="form-check form-check-inline col-md-3 float-right">

                <input class="form-check-input" type="radio" name="options"
id="inlineRadio2" value="3">

                <label class="form-check-label" for="inlineRadio2">Collectors report for
Credit loan</label>

            </div>

            <br><br>

            <div class="form-row">

                <div class="col-md-4">

                    <div class="jumbotron">

                        <div class="container" >
```

```

class="form-control">
    <p class="lead">Collector Name</p>
    <hr class="my-4">
    <input type="text" name="cName"

    </div>
</div>
</div>
<div class="col-md-6">
    <div class="row">
        <div class="form-group col-md-1"></div>
        <div class="form-group col-md-4 float-right">
            From: <input type="date"
class="datepicker" data-date-format="mm/dd/yyyy" name="from">
        </div>
        <div class="form-group col-md-4 float-right">
            To: <input type="date"
class="datepicker" data-date-format="mm/dd/yyyy" name="to">
        </div>
    </div>
    <div class="form-row"></div>
    <div class="row">
        <div class="col-4"></div>
        <div class="form-row"></div>
        <button type="submit" class="btn btn-
dark">Generate Report</button>
    </div>
</div>
</div>

```

```

        </form>

    </div>

    <%

        String s=request.getParameter("options");

        if(s==null) System.out.println("Select some option");

        else{

            String url="jdbc:mysql://localhost:3306/bankproject";

            String userName="root";

            String pwd="xxxxxxx";

            String to=request.getParameter("to");

            String fromDate=request.getParameter("from");

            String cName=request.getParameter("cName");

            if(fromDate.equals("")) fromDate="1999-03-03";

            if(to.equals("")) to="2030-03-03";

            if(s.equals("1")){

                String sql="select borrowed_amount/no_of_installments as
amount_collected,date_of_payment,(select name from borrower where borrower.b_id=loans.b_id) as
customer,loans.b_id, collector_id, (select name from collector where
collector.collector_id=loans.collector_id) as collector_name from installments left join loans on
loans.loan_id=installments.loan_id where date_of_payment>\"'+fromDate+'\" and
date_of_payment<\"'+to+'\" having collector_name like '%"+cName+"%' order by amount_paid desc;";

            }

        }

    %>

    <table class="table">

        <thead class="thead-dark">

            <tr>

                <th scope="col">#</th>

                <th scope="col">Amount Collected</th>

                <th scope="col">Date</th>

```

```

        <th scope="col">Customer</th>

        <th scope="col">Customer ID</th>

        <th scope="col">Collector</th>

        <th scope="col">Collector ID</th>

    </tr>

</thead>

<tbody>

<%@page import="java.sql.*" %>

<%

try{

        Class.forName("com.mysql.jdbc.Driver");

        Connection con=DriverManager.getConnection(url,userName,pwd);

        PreparedStatement st=con.prepareStatement(sql);

        ResultSet rs=st.executeQuery();int i=1;

        while(rs.next()){

                out.println("<tr> <th scope=\"row\">"+i+"</th>");

                out.println("<td>"+rs.getDouble(1) + "</td>");

                out.println("<td>"+rs.getDate(2) + "</td>");

                out.println("<td>"+rs.getString(3) + "</td>");

                out.println("<td>"+rs.getString(4) + "</td>");

                out.println("<td>"+rs.getString(5) + "</td>");

                out.println("<td>"+rs.getString(6) + "</td>");

                out.println("</tr>");

                i++;

        }

        con.close();

```

```
    }catch(Exception e){System.out.println(e);}}
```

```
%>
```

```
</tbody>
```

```
</table>
```

```
<%
```

```
if(s.equals("2")){
```

```
        String sql="select sum(borrowed_amount/no_of_installments) as
amount_collected, collector_id, (select name from collector where
collector.collector_id=loans.collector_id) as collector_name from installments left join loans on
loans.loan_id=installments.loan_id where loan_type=\"jewel\" group by collector_id having collector_name
like \"%" + cName + "%\" order by amount_paid desc";
```

```
%>
```

```
<table class="table">
```

```
    <thead class="thead-dark">
```

```
        <tr>
```

```
            <th scope="col">#</th>
```

```
            <th scope="col">Total Amount Collected</th>
```

```
            <th scope="col">Collector Id</th>
```

```
            <th scope="col">Collector Name</th>
```

```
        </tr>
```

```
    </thead>
```

```
    <tbody>
```

```
<%@page import="java.sql.*" %>
```

```
<%    try{
```

```
        Class.forName("com.mysql.jdbc.Driver");
```

```
        Connection con=DriverManager.getConnection(url,userName,pwd);
```

```
        PreparedStatement st=con.prepareStatement(sql);
```

```
        ResultSet rs=st.executeQuery();int i=1;
```

```

        while(rs.next()){

            out.println("<tr> <th scope=\"row\">"+i+"</th>");

            out.println("<td>"+rs.getDouble(1) +"</td>");

            out.println("<td>"+rs.getString(2) +"</td>");

            out.println("<td>"+rs.getString(3) +"</td>");

            out.println("</tr>");

            i++;

        }

        con.close();

    }catch(Exception e){System.out.println(e);}}

    %>

</tbody>

</table>

<%

    if(s.equals("3")){

        String sql="select sum(borrowed_amount/no_of_installments) as
amount_collected, collector_id, (select name from collector where
collector.collector_id=loans.collector_id) as collector_name from installments left join loans on
loans.loan_id=installments.loan_id where loan_type=\"cc\" group by collector_id having collector_name
like \"'%\"+cName+\"%\"' order by amount_paid desc";

        %>

        <table class="table">

            <thead class="thead-dark">

                <tr>

                    <th scope="col">#</th>

                    <th scope="col">Total Amount Collected</th>

                    <th scope="col">Collector Id</th>

```



```

        <th scope="col">Collector Name</th>

    </tr>

</thead>

<tbody>

<%@page import="java.sql.*" %>

    <%    try{

            Class.forName("com.mysql.jdbc.Driver");

            Connection con=DriverManager.getConnection(url,userName,pwd);

            PreparedStatement st=con.prepareStatement(sql);

            ResultSet rs=st.executeQuery();int i=1;

            while(rs.next()){

                out.println("<tr> <th scope='row'>" +i+"</th>");

                out.println("<td>" +rs.getDouble(1) +"</td>");

                out.println("<td>" +rs.getString(2) +"</td>");

                out.println("<td>" +rs.getString(3) +"</td>");

                out.println("</tr>");

                i++;

            }

            con.close();

        }catch(Exception e){System.out.println(e);}

    %>

</tbody>

</table>

<% }} %>

</div>

</body>

```

</html>

OUTPUT:

The screenshot displays two web pages from a local development environment. The first page is a login form at `localhost:8080/BankProject/login.jsp`. It contains two input fields: "Enter username:" with the value "Akancha" and "Enter password:" with masked characters ".....". A "login" button is positioned below the password field. The second page is a report generation interface at `localhost:8080/BankProject/customerCollectorReport.jsp`. The page title is "Reports/Statuses for Jewel and Credit". The user is logged in as "Akancha". The main heading is "Customer Collector Report". There are three radio buttons for selection: "Collectors report for Credit loan" (selected), "Collectors report for Jewel loan", and "View Individual Collector Report". Below these, there is a "Collector Name" input field, two date pickers labeled "From:" and "To:" with the format "mm/dd/yyyy", and a "Generate Report" button. The Windows taskbar at the bottom shows the time as 2:43 AM on 6/6/2020.

localhost:8080/BankProject/login.jsp

Enter username: Akancha

Enter password:

login

localhost:8080/BankProject/customerCollectorReport.jsp

Reports/Statuses for Jewel and Credit Home Welcome Akancha

Customer Collector Report

☒ Collectors report for Credit loan ☐ Collectors report for Jewel loan ☐ View Individual Collector Report

Collector Name

From: mm/dd/yyyy To: mm/dd/yyyy

Generate Report

Activate Windows
Go to Settings to activate Windows.

Type here to search

2:43 AM
6/6/2020

cse1005 –Software Design and Development –J COMPONENT PROJECT WORK REPORT

← → ↻ localhost:8080/BankProject/customerCollectorReport.jsp?options=1&cName=Chutki&from=&to=

Reports/Statuses for Jewel and Credit Home Welcome Akancha

Customer Collector Report

☐ Collectors report for Credit loan ☐ Collectors report for Jewel loan ☒ View Individual Collector Report

Collector Name:

From: To:

#	Amount Collected	Date	Customer	Customer ID	Collector	Collector ID
1	25000.0	2019-05-03	ninja hathodi	B00000014	C00000014	Chutki
2	25000.0	2020-05-03	ninja hathodi	B00000014	C00000014	Chutki

Activate Windows
Go to Settings to activate Windows

Windows taskbar: Type here to search, 2:44 AM 6/6/2020

localhost:8080/BankProject/customerCollectorReport.jsp?options=1&cName=&from=&to=

Reports/Statuses for Jewel and Credit Home Welcome Akancha

Customer Collector Report

☐ Collectors report for Credit loan ☐ Collectors report for Jewel loan ☒ View Individual Collector Report

Collector Name:

From: To:

#	Amount Collected	Date	Customer	Customer ID	Collector	Collector ID
1	25000.0	2019-05-03	ninja hathodi	B00000014	C00000014	Chutki
2	25000.0	2020-05-03	ninja hathodi	B00000014	C00000014	Chutki
3	10000.0	2001-05-05	chota bheem	B000002126	C00000024	Doremon

Activate Windows
Go to Settings to activate Windows

Windows taskbar: Type here to search, 2:45 AM 6/6/2020

cse1005 –Software Design and Development –J COMPONENT PROJECT WORK REPORT

← → ↻ ⓘ localhost:8080/BankProject/customerCollectorReport.jsp?options=2&cName=&from=2018-03-03&to=2021-05-05 ☆ 🔊 🔌 🔋 ⌵

Reports/Statuses for Jewel and Credit Home · Welcome Akancha

Customer Collector Report

☐ Collectors report for Credit loan ☒ Collectors report for Jewel loan ☐ View Individual Collector Report

Collector Name

From:
03/03/2018

To:
05/05/2021

Generate Report

#	Total Amount Collected	Collector Id	Collector Name
1	10000.0	C00000024	Doremon

Activate Windows
Go to Settings to activate Windows.

Windows taskbar: Type here to search, Task View, File Explorer, Edge, Chrome, Word, Outlook, Settings, Network, Volume, 2:47 AM 6/6/2020

← → ↻ ⓘ localhost:8080/BankProject/customerCollectorReport.jsp?options=3&cName=&from=2018-03-03&to=2021-05-05 ☆ 🔊 🔌 🔋 ⌵

Reports/Statuses for Jewel and Credit Home · Welcome Akancha

Customer Collector Report

☒ Collectors report for Credit loan ☐ Collectors report for Jewel loan ☐ View Individual Collector Report

Collector Name

From:
mm/dd/yyyy

To:
mm/dd/yyyy

Generate Report

#	Total Amount Collected	Collector Id	Collector Name
1	50000.0	C00000014	Chutki

Activate Windows
Go to Settings to activate Windows.

Windows taskbar: Type here to search, Task View, File Explorer, Edge, Chrome, Word, Outlook, Settings, Network, Volume, 2:48 AM 6/6/2020

18BCB0056

SAARTHAK AGARWAL**deceaseloadsmenu.jsp**

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Loan Management System</title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>

</head>

<body>

<form action ="updateBorrower" method = "post">

<nav class="navbar navbar-inverse">

<div class="container-fluid">

<div class="navbar-header">

<a class="navbar-brand" href="#">Deceased Credit Card Loans</a>

</div>

<ul class="nav navbar-nav navbar-right">

<li class="active"><a href="#">Home</a></li>

<li><a href="#">Logout</a></li>

</ul>
```

```

</div>

</nav>

<br>

<br>

<div class = "container">

    <div class = "row">

        <div class = "col-md-1">

            <div class = "customDiv"> </div>

        </div>

        <div class = "col-md-11">

            <div class = "customDiv"><h3><b>For Viewing Borrower and their Deceased
status -></b></h3></div>

        </div>

    </div>

    <br>

    <div class = "row">

        <div class= "col-md-2">

            <div class = "customDiv"> </div>

        </div>

        <div class = "col-md-10">

            <a href="deceaseStatus.jsp"class = "btn btn-primary"> VIEW STATUS </a>

        </div>

    </div>

    <br>

    <div class = "row">

        <div class = "col-md-1">

```

```
<div class = "customDiv"> </div>

</div>

<div class = "col-md-11">

    <div class = "customDiv"><h3><b>For Updating Borrower details in case of
Demise of Existing Borrower -></b></h3></div>

</div>

</div>

<br>

<div class = "row">

    <div class= "col-md-2">

        <div class = "customDiv"> </div>

    </div>

    <div class = "col-md-10">

        <input type = "Submit" name ="action" class = "btn btn-primary" value =
"UPDATE BORROWER">

        </div>

    </div>

    <br>

    <div class = "row">

        <div class = "col-md-1">

            <div class = "customDiv"> </div>

        </div>

        <div class = "col-md-11">

            <div class = "customDiv"><h3><b>For Updating Viewing Status Report of Loan
Recovery -></b></h3></div>
```

```
</div>

</div>

<br>

<div class = "row">

    <div class= "col-md-2">

        <div class = "customDiv"> </div>

    </div>

    <div class = "col-md-10">

        <a href="loanStatus.jsp"class = "btn btn-primary"> DECEASED LOAN REPORT

</a>

    </div>

</div>

<br>

</div>

</form>

</body>

</html>
```

updateBorrower.java (Servlet)

```
import java.io.IOException;

import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
```



```

import javax.servlet.http.HttpServletResponse;

@WebServlet("/updateBorrower")
public class updateBorrower extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public updateBorrower() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        // TODO Auto-generated method stub
        response.getWriter().append("Served at: ").append(request.getContextPath());
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        final String dburl = "jdbc:mysql://localhost/Bank";
        final String dbuname = "root";
        final String dbpassword = "saarthak1238";
        final String dbdriver= "com.mysql.cj.jdbc.Driver";
        int updateCount = 0;

        Connection conn = null;
        Statement stmt = null;
        try{
            //STEP 2: Register JDBC driver
            Class.forName(dbdriver);

            //STEP 3: Open a connection
            conn = DriverManager.getConnection(dburl, dbuname, dbpassword);
            //STEP 4: Execute a query

            stmt = conn.createStatement();

            String sql = "Select G_ID, Name, ID_Proof, B_ID, Phone_no, Address from Borrower
where Date_of_demise IS NOT NULL AND G_ID is NOT NULL AND Loan_Type ='Credit Card' ";

```

```

        ResultSet rs = stmt.executeQuery(sql);
        //STEP 5: Extract data from result set
        Loan loan = new Loan();
        while(rs.next()){
            //Retrieve by column name
            loan.Borrower_ID = rs.getString(4);
            loan.guarantor.Guarantor_ID =rs.getString(1);
            loan.Phone_no = rs.getLong(5);
            loan.Address = rs.getString(6);
            loan.Name = rs.getString(2);
            loan.ID_proof = rs.getString(3);
            PreparedStatement pstmt = conn.prepareStatement("SELECT Name, ID_Proof,
Address, Phone_no FROM Bank.Guarantor WHERE G_ID = ?");
            pstmt.setString(1, loan.guarantor.Guarantor_ID);
            ResultSet rs1 = pstmt.executeQuery();
            //ResultSet rs1 = stmt.executeQuery("Select Name, ID_Proof, Address, Phone_no
from Guarantor where G_ID = ?");
            rs1.next();
            loan.guarantor.Name = rs1.getString("Name");
            loan.guarantor.ID_proof = rs1.getString("ID_Proof");
            loan.guarantor.Address = rs1.getString("Address");
            loan.guarantor.Phone_no = rs1.getLong("Phone_no");
            rs1.close();
            loan.Name = loan.guarantor.Name;
            loan.ID_proof = loan.guarantor.ID_proof;
            loan.Address = loan.guarantor.Address;
            loan.Phone_no = loan.guarantor.Phone_no;
            PreparedStatement psmt1 = conn.prepareStatement("Update Borrower set Name =
?"+"ID_Proof =? " +",Address =? "+" Phone_no =?"+", G_ID = NULL WHERE B_ID=?");
            psmt1.setString(1, loan.Name);
            psmt1.setString(2, loan.ID_proof);
            psmt1.setString(3, loan.Address);
            psmt1.setLong(4, loan.Phone_no);
            psmt1.setString(5, loan.Borrower_ID);
            psmt1.executeUpdate();
            PreparedStatement psmt2 = conn.prepareStatement("Delete from Guarantor where
G_ID = ?");

            psmt2.setString(1, loan.guarantor.Guarantor_ID);
            psmt2.executeUpdate();
            updateCount++;
        }
        rs.close();
    }catch(SQLException se){
        //Handle errors for JDBC
        se.printStackTrace();
    } catch (ClassNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

```

```

        PrintWriter out = response.getWriter();
        if(updateCount==0){
            out.println("<script type=\"text/javascript\">");
            out.println("alert('Records are upto date');");
            out.println("location='deceasealoansmenu.jsp';");
            out.println("</script>");
        }
        else{
            out.println("<script type=\"text/javascript\">");
            out.println("alert('"+Integer.toString(updateCount)+" records updated');");
            out.println("location='deceasealoansmenu.jsp';");
            out.println("</script>");
        }
    }
}

```

Loan.java (Class)

```

public class Loan {
    String Name;
    String Borrower_ID;
    long Phone_no;
    long Account_no;
    Guarantor guarantor;
    private String Loan_ID;
    private String LO_ID;
    String Loan_Type;
    String Collector_ID;
    String Approval_date;
    int Num_Installments;
    float Borrowed_Amount;
    float Amount_paid;
    String ID_proof;
    String Address;
    Loan()
    {
        guarantor = new Guarantor();
    }
}

```

Guarantor.java (Class)

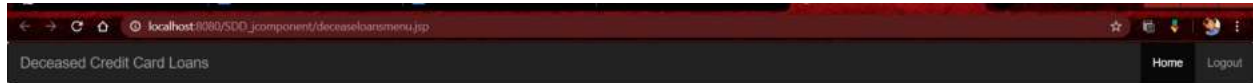
```

public class Guarantor {
    String Guarantor_ID;
    String Name;
    long Phone_no;
    String ID_proof;
}

```

```
String Address;  
}
```

OUTPUT SCREENSHOTS:



For Viewing Borrower and their Deceased status ->

[VIEW STATUS](#)

For Updating Borrower details in case of Demise of Existing Borrower ->

[UPDATE BORROWER](#)

For Updating Viewing Status Report of Loan Recovery ->

[DECEASED LOAN REPORT](#)



Borrower Decease Status

Borrower Name	Borrower ID	Amount Borrowed	Amount Repaid	Decease Status	Date of Demise	Guarantor's ID
Holly	b123	200000.0	0.0	Not Deceased	N/A	g123
Simran	b125	2002340.0	0.0	Not Deceased	N/A	g125
John Wick	b127	5.0E7	0.0	Borrower Transferred	8th December 2005	N/A
Alex	b122	50000.0	14500.0	Deceased	21st March 2017	g122

[<BACK](#)

Deceased Credit Card Loans

Home Logout

Decease Loan Re-payment Status

Loan ID	Borrower ID	Total Instalments	Amount Borrowed	Amount Repaid	Instalments expected till date	Pending Repayment Amount	Demise Date
1127	b127	55	5.0E7	0.0	12	5.0E7	8th December 2005

<-BACK



cse1005 –Software Design and Development –J COMPONENT PROJECT WORK REPORT



Deceased Credit Card Loans

Home Logout

Borrower Decease Status

Borrower Name	Borrower ID	Amount Borrowed	Amount Repaid	Decease Status	Date of Demise	Guarantor's ID
Holly	b123	200000.0	0.0	Not Deceased	N/A	g123
Simran	b125	2002340.0	0.0	Not Deceased	N/A	g125
John Wick	b127	5.0E7	0.0	Borrower Transferred	8th December 2005	N/A
Sam	b132	30000.0	14000.0	Borrower Transferred	21st March 2017	N/A

<-BACK

Deceased Credit Card Loans

Home Logout

Decease Loan Re-payment Status

Loan ID	Borrower ID	Total Installments	Amount Borrowed	Amount Repaid	Installments expected till date	Pending Repayment Amount	Demise Date
l127	b127	55	5.0E7	0.0	12	5.0E7	8th December 2005
l132	b132	30	30000.0	14000.0	2	16000.0	21st March 2017

<-BACK

18BCB0070(Sreyan Biswas)

Credit_debt.java

```
import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

@WebServlet("/loanleft")

public class status extends HttpServlet {

    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServletVersionUID = 1L;
```

```
et()

*/

public status() {

    super();

    // TODO Auto-generated constructor stub

}

/**

 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)

 */

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

    // TODO Auto-generated method stub

    response.getWriter().append("Served at: ").append(request.getContextPath());

}

/**

 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

 */

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

    final String dburl = "jdbc:mysql://localhost/Bank";

    final String dbuname = "root";

    final String dbpassword = "sreyan200";

    final String dbdriver= "com.mysql.cj.jdbc.Driver";

    int updateCount = 0;
```



```
Connection conn = null;

Statement stmt = null;

try{

    //STEP 2: Register JDBC driver

    Class.forName(dbdriver);

    //STEP 3: Open a connection

    conn = DriverManager.getConnection(dburl, dbuname, dbpassword);

    //STEP 4: Execute a query

    stmt = conn.createStatement();

    String sql = "Select ID, Name,Phone_no, Amount,month from Borrower where
Date_of_demise IS NOT NULL AND G_ID is NOT NULL";

    ResultSet rs = stmt.executeQuery(sql);

    //STEP 5: Extract data from result set

    Loan loan = new Loan();

    while(rs.next()){

        //Retrieve by column name

        loan.Borrower_ID = rs.getString(4);

        loan.Phone_no = rs.getLong(5);

        loan.Amount = rs.getString(6);

        loan.month = rs.getString(2);

        loan.name = rs.getString(3);

        PreparedStatement pstmt = conn.prepareStatement("SELECT Name, ID_Proof,
Address, Phone_no FROM Bank.Guarantor WHERE G_ID = ?");

        pstmt.setString(1, loan.guarantor.Guarantor_ID);
```

```

        ResultSet rs1 = pstmt.executeQuery();

        //ResultSet rs1 = stmt.executeQuery("Select Name, ID_Proof, Address, Phone_no
from Guarantor where G_ID = ?");

        rs1.next();

        loan.guarantor.Name = rs1.getString("Name");

        loan.guarantor.phone = rs1.getString("ID_Proof");

        loan.guarantor. = rs1.getString("Address");

        loan.guarantor.Phone_no = rs1.getLong("Phone_no");

        rs1.close();

        loan.Name = loan.guarantor.Name;

        loan.ID_proof = loan.guarantor.ID_proof;

        loan.Address = loan.guarantor.Address;

        loan.Phone_no = loan.guarantor.Phone_no;

        PreparedStatement psmt1 = conn.prepareStatement("Update Borrower set Name =
?"+" ,ID_Proof =? " +",Address =? " +", Phone_no =?"+" , G_ID = NULL WHERE B_ID=?");

        psmt1.setString(1, loan.Name);

        psmt1.setString(2, loan.ID_proof);

        psmt1.setString(3, loan.Address);

        psmt1.setLong(4, loan.Phone_no);

        psmt1.setString(5, loan.month);

        psmt1.executeUpdate();

        PreparedStatement psmt2 = conn.prepareStatement("Delete from Guarantor where
G_ID = ?");

        psmt2.setString(1, loan.guarantor.Guarantor_ID);

        psmt2.executeUpdate();

        updateCount++;

    }

```

```

        rs.close();

    }catch(SQLException se){

        //Handle errors for JDBC

        se.printStackTrace();

    } catch (ClassNotFoundException e) {

        // TODO Auto-generated catch block

        e.printStackTrace();

    }

    PrintWriter out = response.getWriter();

    if(updateCount==0){

        out.println("<script type=\"text/javascript\">");

        out.println("alert('Records are upto date');");

        out.println("location='ajax3.html';");

        out.println("</script>");

    }

    else{

        out.println("<script type=\"text/javascript\">");

        out.println("alert(\""+Integer.toString(updateCount)+" records updated");");

        out.println("location='ajax3.html';");

        out.println("</script>");

    }

}

}

```

ajax.html(loading users accounts)

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <meta http-equiv="X-UA-Compatible" content="ie=edge">

  <title>Ajax 3</title>

  <style>

    .user{

      display: flex;

      background:#f4f4f4;

      padding:10px;

      margin-bottom:10px;

    }

    .user ul{

      list-style: none;

    }

  </style>

</head>

<body>

  <button id="button">Loan Defaulters</button>

  <br><br>

  <h1>D</h1>

  <div id="users"></div>
```

```
<script>

document.getElementById('button').addEventListener('click', loadUsers);

function loadUsers(){

var xhr = new XMLHttpRequest();

xhr.open('GET', 'https://api.github.com/users', true);

xhr.onload = function(){

if(this.status == 200){

var users = JSON.parse(this.responseText);

var output = "";

for(var i in users){

output +=

'<div class="user">' +

'' +

'<ul>' +

'<li>ID: '+users[i].id+'</li>' +

'<li>Login: '+users[i].login+'</li>' +

'</ul>' +

'</div>';

}

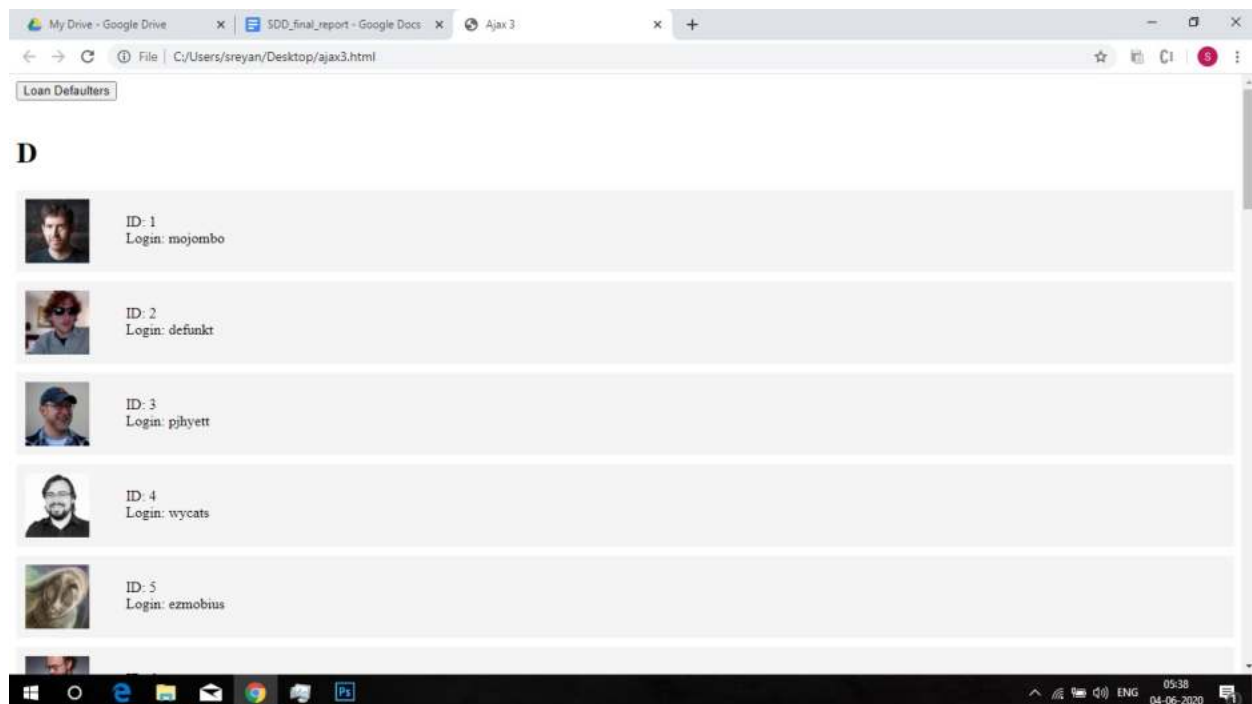
document.getElementById('users').innerHTML = output;

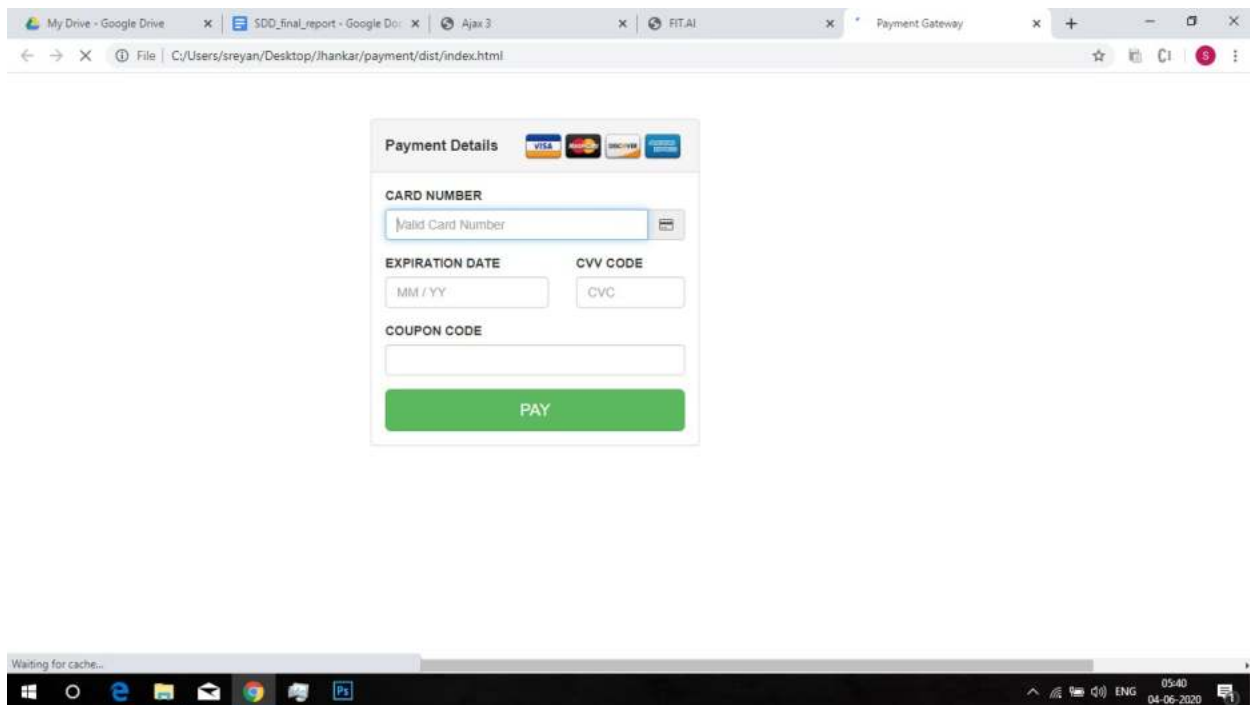
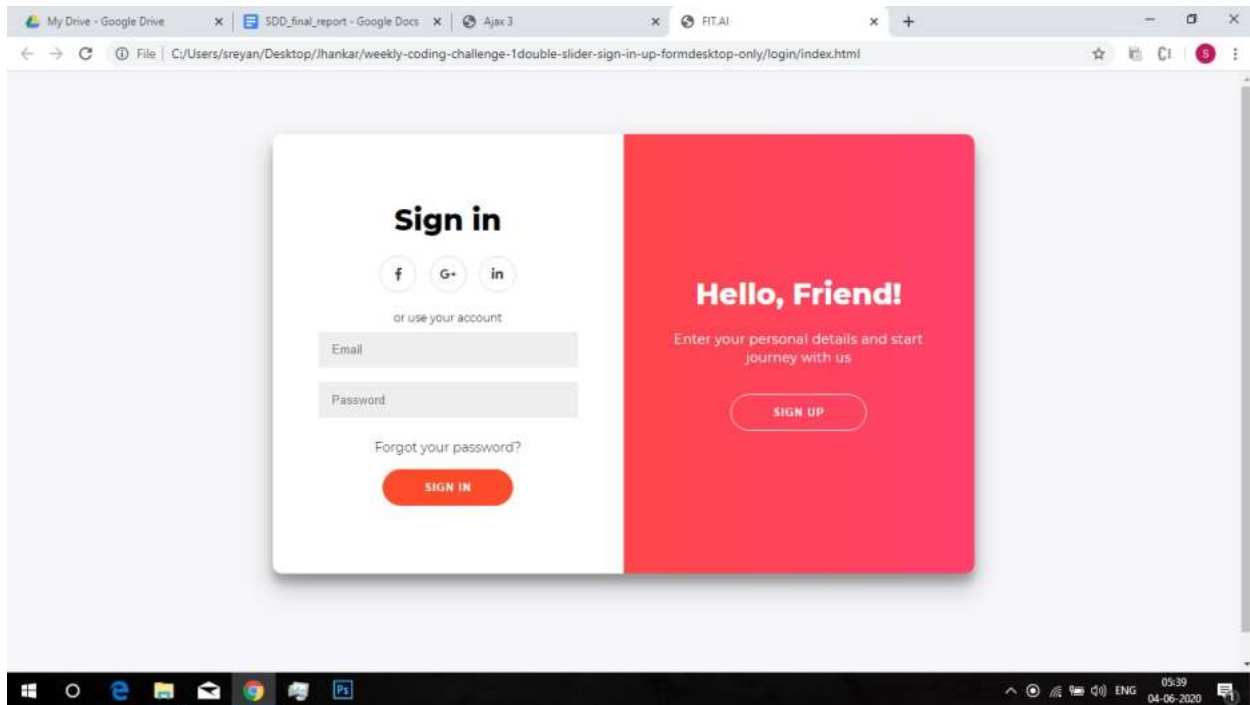
}

}
```

```
xhr.send();  
  
}  
  
</script>  
  
</body>  
  
</html>
```

Screenshot





Vidhi Agarwal (18BCE2190) :

index.jsp (main auction page with item details, active users and timer)

```
<%@page import="java.sql.*"%>

<%@page import="java.io.*"%>

<%@page import="javax.servlet.annotation.WebServlet"%>

<%@page import="javax.servlet.http.HttpServlet"%>

<%@page import="javax.servlet.http.HttpServletRequest"%>

<%@page import="javax.servlet.http.HttpServletResponse"%>

<%@page import="javax.servlet.*"%>

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1" />

<title>Auction Page</title>

<link

rel="stylesheet"

href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"

integrity="sha384-
9alt2nRrpC12Uk9gS9baDI411NQApFmC26EwAOH8WgZl5MYXxFc+NcPb1dKGj7Sk"

crossorigin="anonymous"

/>

</head>

<body>

<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
```



```

<a class="navbar-brand" href="#">Auction System</a>

<button
  class="navbar-toggler"
  type="button"
  data-toggle="collapse"
  data-target="#navbarNavAltMarkup"
  aria-controls="navbarNavAltMarkup"
  aria-expanded="false"
  aria-label="Toggle navigation"
>

  <span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNavAltMarkup">
  <div class="navbar-nav">
    <a class="nav-item nav-link active" href="index.jsp"
      >Home
      <span class="sr-only">(current)</span></a>
    >
    <a class="nav-item nav-link" href="pay.jsp">Payment</a>
    <a class="nav-item nav-link" href="#">Logout</a>
  </div>
</div>
</nav>

<div class="row">
  <div class="col-8">

```

```
<div class="card mb-3" style="max-width: 1080px;">
```

```
<div class="row no-gutters">
```

```
<div class="col-md-4">
```

```

```

```
</div>
```

```
<div class="col-md-8">
```

```
<div class="card-body text-dark">
```

```
<h5 class="card-title">
```

```
Jewel title
```

```
</h5>
```

```
<p class="card-text">
```

```
Base Price: <br />
```

```
Current Bid price:
```

```
<%
```

```
Connection conn=null;
```

```
Statement st=null;
```

```
ResultSet rs=null;
```

```
int maxID = 0;
```

```
try {
```

```
Class.forName( "com.mysql.jdbc.Driver");

conn=(Connection)DriverManager.getConnection("jdbc:mysql://localhost:3306/auction?
useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimez
one=UTC","root","");

//st=conn.createStatement();

//String qry="select * from auction where bid=MAX('bid')";

// String qry = "SELECT MAX(bid) FROM auction";

//rs=st.executeQuery(qry);

Statement s2 = conn.createStatement();

s2.execute("SELECT MAX(bid) FROM auction");

ResultSet rs2 = s2.getResultSet(); //

if (rs2.next()) {

    maxID = rs2.getInt(1);

}

}

catch(Exception ex){

    ex.printStackTrace(response.getWriter());

}

%>

<%= maxID %>
```

```

</p>

<p class="card-text">

  <medium class="text-muted"

    >Metal: <br />

    Weight:

  </medium>

</p>

<a

  href="form.jsp"

  class="btn btn-dark btn-lg"

  type="submit"

  >BID</a

  >

</div>

</div>

</div>

</div>

<div class="card mb-3" style="max-width: 1080px;">

  <div class="row no-gutters">

    <div class="col-md-4">

```

```
</div>
```

```
<div class="col-md-8">
```

```
<div class="card-body">
```

```
<h5 class="card-title">
```

```
Jewel title
```

```
</h5>
```

```
<p class="card-text">
```

```
Base Price: <br />
```

```
Current Bid price:
```

```
</p>
```

```
<p class="card-text">
```

```
<medium class="text-muted"
```

```
>Metal: <br />
```

```
Weight:
```

```
</medium>
```

```
</p>
```

```
<a
```

```
href="#"
```

```
class="btn btn-dark btn-lg"
```

```
type="submit"
```

```
>BID</a
```

```
>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<div class="card mb-3" style="max-width: 1080px;">
```

```
<div class="row no-gutters">
```

```
<div class="col-md-4">
```

```

```

```
</div>
```

```
<div class="col-md-8">
```

```
<div class="card-body">
```

```
<h5 class="card-title">
```

```
Jewel title
```

```
</h5>
```

```
<p class="card-text">
```

```
Base Price: <br />
```

```
Current Bid price:
```

```
</p>
```

```
<p class="card-text">
```

```
<medium class="text-muted"
```

```
>Metal: <br />
```

Weight:

```
</medium>
```

```
</p>
```

```
<a
```

```
href="#"
```

```
        class="btn btn-dark btn-lg"

        type="submit"

        >BID</a

    >

</div>

</div>

</div>

</div>

</div>

<div class="col-4">

    <div class="row row-cols-3 row-cols-md-2">

        <div class="col-sm-4">

            <div class="card">

                <div class="card-body">

                    <h5 class="card-title">

                        Active Users

                    </h5>

                    <p class="card-text">lore</p>

                    Time Left:

                    <p id="demo"></p>

                </div>

            </div>

        </div>

    </div>

</div>

<script >

var deadline = new Date("Jun 5, 2020 19:37:25").getTime();

var x = setInterval(function() {

var now = new Date().getTime();

var t = deadline - now;
```

```

var days = Math.floor(t / (1000 * 60 * 60 * 24));

var hours = Math.floor((t%(1000 * 60 * 60 * 24))/(1000 * 60 * 60));

var minutes = Math.floor((t % (1000 * 60 * 60)) / (1000 * 60));

var seconds = Math.floor((t % (1000 * 60)) / 1000);

document.getElementById("demo").innerHTML =

hours + "h " + minutes + "m " + seconds + "s ";

    if (t < 0) {

        clearInterval(x);

        document.getElementById("demo").innerHTML = "SESSION EXPIRED";

    }

}, 1000);

</script>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</body>

</html>

```

submit.jsp (to show successful submission and redirection)

```

<%@page import="java.sql.*"%>

<%@page import="java.io.*"%>

```



```
<%@page import="javax.servlet.annotation.WebServlet"%>

<%@page import="javax.servlet.http.HttpServlet"%>

<%@page import="javax.servlet.http.HttpServletRequest"%>

<%@page import="javax.servlet.http.HttpServletResponse"%>

<%@page import="javax.servlet.*"%>


<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

    <meta http-equiv="refresh" content="3;url=bids.jsp" />

<meta charset="ISO-8859-1">

<title>Submit</title>

</head>

<body>


<%

    Connection conn=null;

    Statement st=null;

    ResultSet rs=null;


    String name = request.getParameter("name");

    String phone = request.getParameter("phone");

    Integer amount = Integer.parseInt(request.getParameter("amount"));
```

```
try
{
    Class.forName( "com.mysql.jdbc.Driver");

    conn=(Connection)DriverManager.getConnection("jdbc:mysql://localhost:3306/auction?
useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimez
one=UTC","root","");

    st=conn.createStatement();

    int i = st.executeUpdate("insert into auction (`name`, `number`, `bid`)
values('"+name+"','"+phone+"','"+amount+"')");

    //st.executeUpdate(sql);
}
catch(Exception ex){
    ex.printStackTrace(response.getWriter());

    %>

    Some error here

    <%

    }

    %>
```

Successfully Submitted

Redirecting...

```
</body>
```

```
</html>
```

bids.jsp (to display bids by active users)

```
<%@page import="java.sql.*"%>
```

```
<%@page import="java.io.*"%>
```

```
<%@page import="javax.servlet.annotation.WebServlet"%>
```

```
<%@page import="javax.servlet.http.HttpServlet"%>
```

```
<%@page import="javax.servlet.http.HttpServletRequest"%>
```

```
<%@page import="javax.servlet.http.HttpServletResponse"%>
```

```
<%@page import="javax.servlet.*"%>
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
```

```
    pageEncoding="ISO-8859-1"%>
```

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
    <meta charset="UTF-8">
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
```

```
    <title>Jewel Evaluation Form</title>
```

```
    <!-- Font Icon -->
```

```
    <link rel="stylesheet" href="fonts/material-icon/css/material-design-iconic-font.min.css">
```

```
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
```

```
<!-- Main css -->
```

```
<link rel="stylesheet" href="css/style.css">
```

```
</head>
```

```
<body>
```

```
<div class="main">
```

```
<!-- <h1>Submit Details</h1>-->
```

```
<div class="container" style="width:400px">
```

```
<div class="sign-up-content">
```

```
<div class="tab">
```

```
<table border=1 style="width:200px">
```

```
<tr >
```

```
<th>Name</th>
```

```
<th>Bid</th>
```

```
</tr>
```

```
<%
```

```
Connection conn=null;
```

```
Statement st=null;
```

```
ResultSet rs=null;
```

```
try
```

```
{
```

```
    Class.forName( "com.mysql.jdbc.Driver");
```

```
    conn=(Connection)DriverManager.getConnection("jdbc:mysql://localhost:3306/auction?
useUnicode=true&useJDBCCompliantTimezoneShift=true&useLegacyDatetimeCode=false&serverTimez
one=UTC","root","");
```

```
    st=conn.createStatement();
```

```
    String qry="select * from auction order by bid DESC";
```

```
    rs=st.executeQuery(qry);
```

```
    while(rs.next())
```

```
    {
```

```
        %>
```

```
        <tr>
```

```
            <td><%=rs.getString(2) %></td>
```

```
            <td><%=rs.getString(4) %></td>
```

```
        </tr>
```

```
        <%
```

```
    }
```

```
}
```

```
        catch(Exception ex){

            ex.printStackTrace(response.getWriter());

        }

    %>

</table>

</div>

</div>

</div>

</div>

</div>

<center>

<a href = "form.jsp">

    <div class="form-textbox" style="width:100px;">

        <input type="submit" name="submit" id="submit" class="submit" value="Bid again" />

    </div>

</center>

<!-- JS -->

<script src="vendor/jquery/jquery.min.js"></script>

<script src="js/main.js"></script>

</body>

</html>
```


Screenshots - 18BCE2190

cse1005 –Software Design and Development –J COMPONENT PROJECT WORK REPORT


localhost / 127.0.0.1 / auction / ... Auction Page

localhost:8080/Auction_System/


Auction System Home Payment Logout



Diamond Necklace -001
Base Price: 10,00,000
Current Bid price: 1987
Metal: Diamond
Weight: 10gm
[BID](#)



Gold Necklace - 001
Base Price: 10,00,000
Current Bid price: Nil
Metal: Gold
Weight: 10gm
[BID](#)



Gold Necklace -002
Base Price: 10,00,000
Current Bid price: Nil
Metal: Gold

Active Users
Vidhi Agarwal
Sam Williams
Arjun Arora
Harry Potter

Time Left:
0h 13m 37s

localhost / 127.0.0.1 / auction / ... Jewel Evaluation Form

localhost:8080/Auction_System/bids.jsp

Name	Bid
sam williams	2000000
sam williams	1987
khushi	500
Arjun	100

[Bid again](#)

The screenshot displays a web browser window with the title 'Auction System'. The address bar shows 'localhost:5080/Auction_System/pay.jsp'. The main content area is titled 'Checkout form'. It contains three main sections: 'Billing address', 'Your cart', and 'Payment Options'. The 'Billing address' section has input fields for First name, Last name, Username, Email (Optional), Address, Address 2 (Optional), Country, State, and Zip. The 'Your cart' section shows a table with one item: 'Diamond Necklace - 001' with a 'Total price' of 'Rs 2000000'. The 'Payment Options' section has radio buttons for 'Credit card', 'Debit card', and 'Paypal'. Below these are input fields for 'Name on card' and 'Credit card number'. At the bottom of the form, there are checkboxes for 'Shipping address is the same as my billing address' and 'Save this information for next time'. The browser's taskbar at the bottom shows various application icons and the system clock indicating '02:19'.

VAIBHAV (18BCE2313)

dep_calculation.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" %>

<%
<sql:query var="stockresult" scope="request" dataSource="com.mysql.jdbc.Driver">
    SELECT base_value <!-- select the Base Value -->
    FROM Auction
    WHERE productID = $productID
</sql:query>

<!-- calculate the new value from the first row of the query result -->

<c:set var="newstocklevel" value="${Auction_value.rows[0].price - base_value}"/>

<sql:update var="newstock" scope="request" dataSource="com.mysql.jdbc.Driver">
    UPDATE Auction
    SET stock=$Dep_value
    WHERE productID = $productID
</sql:query>
%>

<%
```



```

if(request.getParameter("a")!=null){ // this is to avoid null pointer exception
try{
// create a java mysql database connection
String myDriver = "com.mysql.jdbc.Driver";
String myUrl = "jdbc:mysql://localhost:3306/studentEnrollment";
Class.forName(myDriver);
Connection conn = DriverManager.getConnection(myUrl, "root", "12");

// create the java mysql update preparedstatement
String query = "update etudiants set dv_o_math = ? where first_name = ?";
PreparedStatement preparedStmt = conn.prepareStatement(query);
preparedStmt.setInt(1, 15);
preparedStmt.setString(2,request.getParameter("bt2"));

// execute the java preparedstatement
preparedStmt.executeUpdate();

conn.close();
}
}catch(Exception e ){
e.getMessage();
}
%>

```

Display_database.jsp

```

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Depriciation report page</title>

<style>
    .back{
        text-align: right;
        color: white;
        padding: 30px;
    }
    h1{
        color: white;
    }
</style>
</head>
<body>

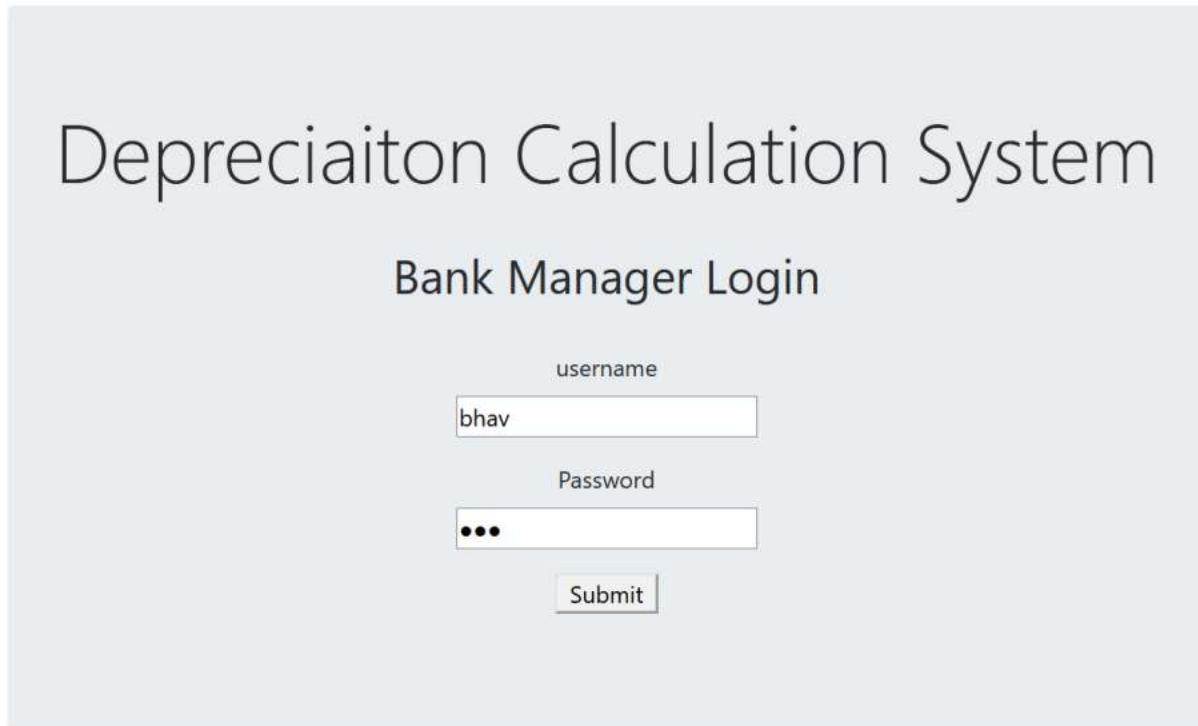
```

```

<h1>Retrieve data from database in jsp</h1>
<table border="1">
<tr>
<td>Loan borrower</td>
<td>Loan id</td>
<td>Buyer name</td>
<td>Date of auction</td>
<td>Base price</td>
<td>Fianl Price</td>
<td>Description</td>
</tr>
<%
try{
connection = DriverManager.getConnection(connectionUrl+database, userid, password);
statement=connection.createStatement();
String sql ="select * from users";
resultSet = statement.executeQuery(sql);
while(resultSet.next()){
%>
<tr>
<td><%=resultSet.getString("id") %></td>
<td><%=resultSet.getString("first_name") %></td>
<td><%=resultSet.getString("Buyer_name") %></td>
<td><%=resultSet.getString("Date_of_auction") %></td>
<td><%=resultSet.getString("Base_price") %></td>
<td><%=resultSet.getString("Final_price") %></td>
<td><%=resultSet.getString("Description") %></td><td><a href="update.jsp?id=<
%=resultSet.getString("id")%>">update</a></td>
</tr>
<%
}
connection.close();
} catch (Exception e) {
e.printStackTrace();
}
%>
</body>
</html>

```

Screenshots (18BCE2313):



The screenshot shows a web interface for a 'Depreciation Calculation System'. The main heading is 'Depreciation Calculation System' in a large, dark font. Below it, the sub-heading is 'Bank Manager Login'. There are two input fields: 'username' with the text 'bhav' and 'Password' with three dots indicating a masked password. A 'Submit' button is located below the password field.

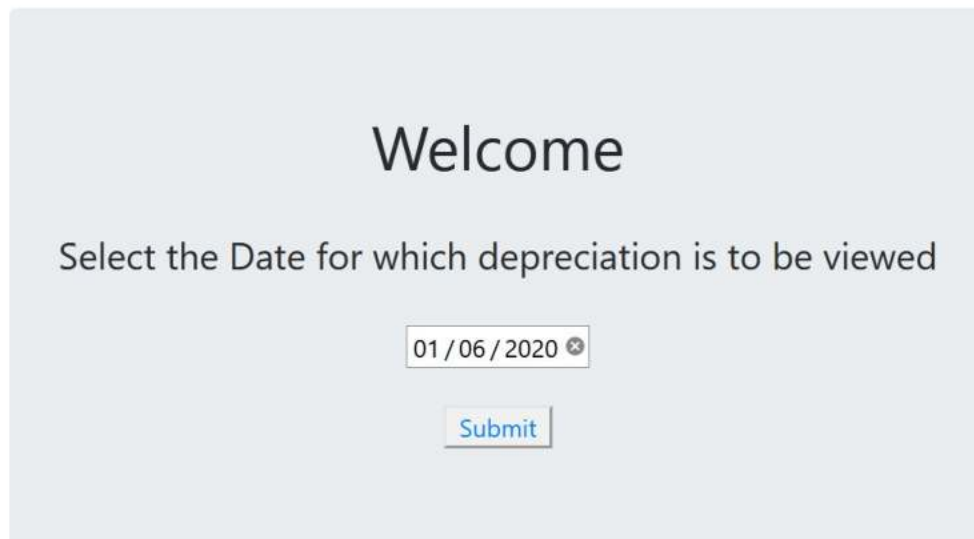
Depreciation Calculation System

Bank Manager Login

username

Password

Submit



The screenshot shows a 'Welcome' message. Below it, the text 'Select the Date for which depreciation is to be viewed' is displayed. There is a date input field showing '01 / 06 / 2020' with a calendar icon. A 'Submit' button is located below the date field.

Welcome

Select the Date for which depreciation is to be viewed

Submit

Depriciation Report

IOAN BORROWER NAME	IOAN ID	BUYER NAME	DATE OF AUCTION	BASE PRICE	SOLD PRICE	DEPRICIATON VALUE	CURRENT STATUS
Akansha	SBI1313	Ram	01-06-2020	10000	9000	-1000	Borrower updated back to active list with 1000 remaining
Vaibhav	SBI1413	Ram	01-06-2020	90000	82000	-8000	Borrower updated back to active list with 8000 remaining
Gopal	SBI1513	Ram	01-06-2020	50000	55000	+5000	5000 returned to borrower
Vaibhav	SBI1413	Ram	01-06-2020	90000	82000	-8000	Borrower updated back to active list with 8000 remaining
Gopal	SBI1513	Ram	01-06-2020	50000	55000	+5000	5000 returned to borrower
Vaibhav	SBI1413	Ram	01-06-2020	90000	82000	-8000	Borrower updated back to active list with 8000 remaining
Gopal	SBI1513	Ram	01-06-2020	50000	55000	+5000	5000 returned to borrower
Vaibhav	SBI1413	Ram	01-06-2020	90000	82000	-8000	Borrower updated back to active list with 8000 remaining
Gopal	SBI1513	Ram	01-06-2020	50000	55000	+5000	5000 returned to borrower
Vaibhav	SBI1413	Ram	01-06-2020	90000	82000	-8000	Borrower updated back to active list with 8000 remaining
Gopal	SBI1513	Ram	01-06-2020	50000	55000	+5000	5000 returned to borrower
Vaibhav	SBI1413	Ram	01-06-2020	90000	82000	-8000	Borrower updated back to active list with 8000 remaining
Gopal	SBI1513	Ram	01-06-2020	50000	55000	+5000	5000 returned to borrower

[LOGOUT](#)

PRANEETH(18BCE2042)

Auction.java(class)

```
/*
```

```
* Author:      Praneeth
```

```
* Reg. No.:    18BCE2042
```

```
* Name:        Auction.java
```

```
* Purpose:     Java class for Auction
```

```
* Class Used by:  auctionservlet.java; AuctionDAO.java; auction_notification.jsp;
```

```
* Classes Used:  -
```

```
*/
```

```
package trial_2;
```

```
public class Auction {

    private String J_ID;

    private String Bid_op;

    private String reg_cls;

    private String bid_reg;

    private String Base_price;


    public Auction(String bid_reg) {

        super();

        this.bid_reg = bid_reg;

    }

}
```

```
//Constructors
```

```
public Auction(String j_ID, String bid_op, String reg_cls) {  
    super();  
    J_ID = j_ID;  
    Bid_op = bid_op;  
    this.reg_cls = reg_cls;  
}
```

```
//Getters
```

```
public String getJ_ID() {
```

```
        return J_ID;
    }

    public String getBid_op() {
        return Bid_op;
    }

    public String getReg_cls() {
        return reg_cls;
    }

    public String getBid_reg() {
        return bid_reg;
    }

    public String getBase_price() {
        return Base_price;
    }
}
```

AuctionDAO.java

```
/*
 * Author:      Praneeth
 * Reg. No.:    18BCE2042
 * Name:        AuctionDAO.java
 * Purpose:     Gets data from SQL database (Jewels and Auction database) and forwards to
                auctionservlet.java in
```

* Hashmap format

* Class Used by: Auction.java, Jewels.java;

* Classes Used: auctionservlet.java;

*/

```
package trial_2;

import java.sql.*;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.Date;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import trial_2.Auction;

import trial_2.Jewels;

public class AuctionDAO {

    private String jdbcURL = "jdbc:mysql://localhost/trial_test";

    private String jdbcUsername = "root";

    private String jdbcPassword = "root";
```

```
public ResultSet date;

public Connection connection = null;

public Statement stmt1 = null;

public Statement stmt2 =null;

static DateFormat df = new SimpleDateFormat("yyyy-MM-dd");

static Date dateobj = new Date();

static String cur_date = (df.format(dateobj));

private static String SELECT_DETAILS = "SELECT auction.J_ID,Title,bid_op,reg_cls,bid_reg FROM
trial_test.auction, trial_test.jewels WHERE auction.J_ID = jewels.J_ID AND bid_reg ="";

private static final String SELECT_DATES = "SELECT DISTINCT bid_reg FROM trial_test.auction
order by bid_reg DESC;";

private static String SELECT_MODAL = "SELECT auction.J_ID,Title,bid_op,reg_cls,bid_reg FROM
auction,jewels WHERE auction.J_ID = jewels.J_ID AND bid_reg ="" + cur_date + "",";

public AuctionDAO() {}

/*
    * Author:      Praneeth
    * Reg. No:      18BCE2042
    * Function Name: getConnection()
    * Purpose:      Connects to SQL database using pre-defined variables
    * Arguments:     -
    * Return:        Connection
    */

protected Connection getConnection() {

    Connection connection = null;

    try {

        Class.forName("com.mysql.jdbc.Driver");
```



```
connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);

stmt1 = connection.createStatement();

} catch (SQLException e) {

    // TODO Auto-generated catch block

    e.printStackTrace();

} catch (ClassNotFoundException e) {

    // TODO Auto-generated catch block

    e.printStackTrace();

}

return connection;

/*

    * Author:      Praneeth

    * Reg. No:     18BCE2042

    * Function Name: selectpresent()

    * Purpose:      Gets data from SQL database using a pre-defined string variable which contains
a query and

    *               sends the data to auctionservlet.java in Hashmap format. Used
for POP-UP/MODAL in auction_notification.jsp

    * Arguments:    -

    * Return:       Map<String, Map<Jewels,Auction>>

    */

public Map<String, Map<Jewels,Auction>> selectpresent(){

    Map<String, Map<Jewels,Auction>> sm = new HashMap<String, Map<Jewels,Auction>>();

    // Step 1: Establishing a Connection

    try { Connection connection = getConnection();

        stmt2=connection.createStatement();
```

```

        System.out.println(SELECT_MODAL);

// Step 2: Execute the query or update query
ResultSet rs_sm = stmt2.executeQuery(SELECT_MODAL);

// Step 3: Process the ResultSet object.
Map<Jewels,Auction> tmp1 = new HashMap<Jewels,Auction>();

while (rs_sm.next()) {

    String j_id = rs_sm.getString("J_ID");

    String bid_op = rs_sm.getString("bid_op");

    String reg_cls = rs_sm.getString("reg_cls");

    String title = rs_sm.getString("Title");

    Auction users = new Auction(j_id, bid_op, reg_cls);

    Jewels users1 = new Jewels(title);

    tmp1.put(users1,users);

}

boolean ch = tmp1.isEmpty();

if (!ch) {

    sm.put(cur_date,tmp1);}

} catch (SQLException e) {

    printSQLException(e);

}

System.out.println("final_sm"); fe

    System.out.println(sm);

return sm;

}

/*

```

* Author: Praneeth

* Reg. No: 18BCE2042

* Function Name: selectAllUsers()

* Purpose: Gets data from SQL database using a pre-defined string variable which contains a query and

* sends the data to auctionservlet.java in Hashmap format. Used for Table in auction_notification.jsp

* Arguments: -

* Return: Map<String, Map<Jewels,Auction>>

*/

```
public Map<String, Map<Jewels,Auction>> selectAllItems() {  
  
    try {  
  
        Connection connection = getConnection();  
  
        stmt1=connection.createStatement();  
  
        System.out.println(SELECT_DATES);  
  
        this.date = stmt1.executeQuery(SELECT_DATES);  
  
        }  
  
    catch (SQLException e) {  
  
        printSQLException(e);  
  
    }  
  
    // using try-with-resources to avoid closing resources (boiler plate code)  
  
    Map<String, Map<Jewels,Auction>> hm = new HashMap<String, Map<Jewels,Auction>>();  
  
    // Step 1: Establishing a Connection  
  
    try { Connection connection = getConnection();  
  
        // Step 2:Create a statement using connection object  
  
        while(date.next()) {  
  
            String SELECT_DETAILS_2 = SELECT_DETAILS + date.getString("bid_reg") + " ORDER BY  
J_ID;";
```

```
PreparedStatement preparedStatemen = connection.prepareStatement(SELECT_DETAILS_2);

System.out.println(preparedStatemen);

// Step 3: Execute the query or update query

ResultSet rs = preparedStatemen.executeQuery();

// Step 4: Process the ResultSet object.

Map<Jewels,Auction> tmp = new HashMap<Jewels,Auction>();

while (rs.next()) {

    /*

    * users.clear(); users1.clear();

System.out.println("before_users");

    * System.out.println(users); System.out.println("before_users1");

    * System.out.println(users1);

    */

    String j_id = rs.getString("J_ID");

    String bid_op = rs.getString("bid_op");

    String reg_cls = rs.getString("reg_cls");

    String title = rs.getString("Title");

    Auction users = new Auction(j_id, bid_op, reg_cls);

    Jewels users1 = new Jewels(title);

    tmp.put(users1,users);

}

hm.put(date.getString("bid_reg"),tmp);}

} catch (SQLException e) {

    printSQLException(e);

}

System.out.println("final_hm");
```

```

        System.out.println(hm);

    return hm;

}

/*
    * Author:    Praneeth
    * Reg. No:    18BCE2042
    * Function Name: printSQLException()
    * Purpose:    Prints errors
    * Arguments:  SQLException
    * Return:     -
    */

private void printSQLException(SQLException ex) {
    for (Throwable e: ex) {
        if (e instanceof SQLException) {
            e.printStackTrace(System.err);

            System.err.println("SQLState: " + ((SQLException) e).getSQLState());
            System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
            System.err.println("Message: " + e.getMessage());

            Throwable t = ex.getCause();

            while (t != null) {
                System.out.println("Cause: " + t);
                t = t.getCause();
            }
        }
    }
}

```

```
}  
  
}
```

auctionservlet.java

```
/*  
  
 * Author:      Praneeth  
  
 * Reg. No.:    18BCE2042  
  
 * Name:        auctionservlet.java  
  
 * Purpose:     Sending Data from Jewel and Auction Database via AuctionDAO.java to  
auction_notification.jsp  
  
 * Class Used by: auction_notification.jsp;  
  
 * Classes Used: AuctionDAO.java; Auction.java; Jewels.java;  
  
 */
```

```
package trial_2;  
  
import java.io.IOException;  
  
import java.sql.SQLException;  
  
import java.util.HashMap;  
  
import java.util.List;  
  
import java.util.Map;  
  
import javax.servlet.RequestDispatcher;  
  
import javax.servlet.ServletException;  
  
import javax.servlet.annotation.WebServlet;  
  
import javax.servlet.http.HttpServlet;  
  
import javax.servlet.http.HttpServletRequest;
```

```

import javax.servlet.http.HttpServletResponse;

import trial_2.Auction;

import trial_2.Jewels;

import trial_2.AuctionDAO;

/**
 * Servlet implementation class auctionservlet
 */
@WebServlet("/auction")

public class auctionservlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    private AuctionDAO auctionDAO;

    public void init() {

        auctionDAO = new AuctionDAO();

    }

    /**
     * @see HttpServlet#HttpServlet()
     */
    public auctionservlet() {

        super();

        // TODO Auto-generated constructor stub

    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */

```

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// TODO Auto-generated method stub

try {

listUser(request, response);

} catch (SQLException ex) {

throw new ServletException(ex);

}

}

/*

* Author: Praneeth

* Reg. No: 18BCE2042

* Function Name: listUser()

* Purpose: Gets data in hashmap format from AuctionDAO.java and dispatches it to auction_notification.jsp

* Arguments: HttpServletRequest; HttpServletResponse;

* Return: -

*/

private void listUser(HttpServletRequest request, HttpServletResponse response)

throws SQLException, IOException, ServletException {

Map<String, Map<Jewels,Auction>> listitems = auctionDAO.selectAllitems();

Map<String, Map<Jewels,Auction>> listModal = auctionDAO.selectpresent();

Map<String,Map<String, Map<Jewels,Auction>>> listTotal = new
HashMap<String,Map<String, Map<Jewels,Auction>>>();

listTotal.put("listModal",listModal);


```
listTotal.put("listUser",listitems);

System.out.println("Listtotal");

System.out.println(listTotal);

request.setAttribute("listTotal", listTotal);

RequestDispatcher dispatcher =
request.getRequestDispatcher("auction_notification.jsp");

dispatcher.forward(request, response);

}

/**

 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)

 */

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

    // TODO Auto-generated method stub

    doGet(request, response);

}

}
```

bidder.java

```
/*

 * Author:      Praneeth

 * Reg. No.:    18BCE2042

 * Name:        bidder.java

 * Purpose:     Java class for bidder

 * Class Used by: bidderservlet.java; bidder_sql.java;

 * Classes Used: -

 */
```

```
package trial_2;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.Date;

public class bidder {

    private String Fname;

    private String Lname;

    private String Bidder_ID;

    private String Address;

    private String Phone_no;

    private String PAN;

    private String DOB;

    private String Aadhar;

    private String Reg_bidder;

    //Constructors

    public bidder(String fname, String lname, String bidder_ID, String address, String phone_no,
String pAN,

        String dOB, String aadhar) {

        super();

        Fname = fname;

        Lname = lname;

        Bidder_ID = bidder_ID;

        Address = address;

        Phone_no = phone_no;

        PAN = pAN;

        DOB = dOB;
```

```
        Aadhar = aadhar;
    }

    //Getters

    public String getFname() {

        return Fname;

    }

    public String getLname() {

        return Lname;

    }

    public String getBidder_ID() {

        return Bidder_ID;

    }

    public String getAddress() {

        return Address;

    }

    public String getPhone_no() {

        return Phone_no;

    }

    public String getPAN() {

        return PAN;

    }

    public String getDOB() {

        return DOB;

    }

    public String getAadhar() {

        return Aadhar;
```

```
}  
  
/*  
  
    * Author:      Praneeth  
  
    * Reg. No:     18BCE2042  
  
    * Function Name: validate()  
  
    * Purpose:      Compares I/P date with current date and returns a int value.  
  
    * Arguments:    Date  
  
    * Return:       int  
  
    */  
  
public int validate(Date d1) {  
  
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");  
  
    Date d2 = new Date();  
  
    //java.util.Date d1 = sdf.parse("2019-04-15")  
  
    System.out.println("The date 1 is: " + sdf.format(d1));  
  
    System.out.println("The date 2 is: " + sdf.format(d2));  
  
    if(d1.compareTo(d2) > 0) {  
  
        return -1; //Date 1 occurs after Date 2  
  
    } else if(d1.compareTo(d2) < 0) {  
  
        return 1; // Date 1 occurs before Date 2  
  
    } else if(d1.compareTo(d2) == 0) {  
  
        return 0; //equal dates  
  
    }  
  
    return -2; //error in I/P  
  
}  
  
}
```

Bidder_sql.java

```
/*
* Author:      Praneeth
* Reg. No.:    18BCE2042
* Name:        bidder.java
* Purpose:      Java class for bidder
* Class Used by: bidderservlet.java; bidder_sql.java;
* Classes Used: -
*/

package trial_2;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;

public class bidder {

    private String Fname;

    private String Lname;

    private String Bidder_ID;

    private String Address;

    private String Phone_no;

    private String PAN;

    private String DOB;

    private String Aadhar;

    private String Reg_bidder;

    //Constructors
```

```
public bidder(String fname, String lname, String bidder_ID, String address, String phone_no,
String pAN,
                String dOB, String aadhar) {
    super();
    FName = fname;
    Lname = lname;
    Bidder_ID = bidder_ID;
    Address = address;
    Phone_no = phone_no;
    PAN = pAN;
    DOB = dOB;
    Aadhar = aadhar;
}

//Getters
public String getFname() {
    return FName;
}

public String getLname() {
    return Lname;
}

public String getBidder_ID() {
    return Bidder_ID;
}

public String getAddress() {
    return Address;
}
```

```
public String getPhone_no() {  
    return Phone_no;  
}  
  
public String getPAN() {  
    return PAN;  
}  
  
public String getDOB() {  
    return DOB;  
}  
  
public String getAadhar() {  
    return Aadhar;  
}  
  
/*  
    * Author:    Praneeth  
    * Reg. No:   18BCE2042  
    * Function Name: validate()  
    * Purpose:    Compares I/P date with current date and returns a int value.  
    * Arguments:  Date  
    * Return:     int  
    */  
  
public int validate(Date d1) {  
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");  
    Date d2 = new Date();  
    //java.util.Date d1 = sdf.parse("2019-04-15")
```

```
        System.out.println("The date 1 is: " + sdf.format(d1));

        System.out.println("The date 2 is: " + sdf.format(d2));

        if(d1.compareTo(d2) > 0) {

            return -1; //Date 1 occurs after Date 2

        } else if(d1.compareTo(d2) < 0) {

            return 1; // Date 1 occurs before Date 2

        } else if(d1.compareTo(d2) == 0) {

            return 0; //equal dates

        }

        return -2; //error in I/P

    }

}
```

bidderservlet.java

```
/*

* Author:      Praneeth

* Reg. No.:    18BCE2042

* Name:        bidderservlet.java

* Purpose:     Receives data from bidder_reg.jsp and sends data to bidder_sql.java

* Class Used by: bidder.java; bidder_sql.java; bidder_reg.jsp; suc.jsp; fail.jsp

* Classes Used: -

*/

package trial_2;

import java.io.*;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;
```



```

import javax.servlet.http.HttpServletResponse;

import javax.servlet.*;

/**
 * Servlet implementation class bidderservlet
 */
@SuppressWarnings("serial")
@WebServlet("/bidderservlet")
public class bidderservlet extends HttpServlet {

    /**
     * @see HttpServlet#HttpServlet()
     */
    Bidder_sql y = new Bidder_sql();

    /**
     * Author:      Praneeth
     * Reg. No:     18BCE2042
     * Function Name: service()
     * Purpose:      Gets data from bidder_reg.jsp and sends it to bidder_sql.java to update sql
     database. Redirects page(suc.jsp, fail.jsp)
     * Arguments:    HttpServletRequest; HttpServletResponse;
     * Return:       -
     */
    public void service(HttpServletRequest req, HttpServletResponse res)
    {
        bidder x = new bidder(req.getParameter("first name"),req.getParameter("last
name"),req.getParameter("email"),req.getParameter("address"),req.getParameter("phone"),req.getParam
eter("pan"),req.getParameter("dob"),req.getParameter("aadhar"));

        System.out.println("First name:" + x.getFname() + " Last Name:" + x.getLname());
    }
}

```

```
        boolean check = y.SendData(x);

        //Redirecting page

        if (check){

            res.setContentType("text/html");

            String site = new String("suc.jsp");

            res.setStatus(res.SC_MOVED_TEMPORARILY);

            res.setHeader("Location", site);

        }

        else {

            res.setContentType("text/html");

            String site = new String("fail.jsp");

            res.setStatus(res.SC_MOVED_TEMPORARILY);

            res.setHeader("Location", site);

        }

    }

}
```

Jewels.java(class)

```
/*

* Author:      Praneeth

* Reg. No.:    18BCE2042

* Name:        Jewels.java

* Purpose:     Java class for Jewels

* Class Used by: auction_notification.jsp; AuctionDAO.java

* Classes Used: -

*/
```

```
package trial_2;
```

```
public class Jewels {  
    private String J_ID;  
    private String Title;  
    private String img_src;  
    //Constructors  
    public Jewels(String title) {  
        super();  
        Title = title;  
    }  
    //Getters  
    public String getJ_ID() {  
        return J_ID;  
    }  
    public String getTitle() {  
        return Title;  
    }  
    public String getImg_src() {  
        return img_src;  
    }  
}
```

Screenshots(18BCE2042):

Bidder registration

Bidder Registration

Auction Notification Home Logout

Bidder Registration

First Name:

Last Name:

Email:

Mobile No.:

PAN Number:

Aadhar:

Address:

Date of Birth:

Auction Notification

Auction Notification

June 4, 2020			
Title	Reference no.	End of Bidder Registration	Bid opening date
Diamond	1	10/4/2015	10/3/2015
July 22, 2019			
Title	Reference no.	End of Bidder Registration	Bid opening date
Gold Nugget	3	20/7/2017	20/4/2016
Silver Chain	5	13/7/2019	14/5/2019
June 21, 2019			
Title	Reference no.	End of Bidder Registration	Bid opening date
Gold Bar	4	14/6/2017	13/5/2016
Gold Necklace	2	10/6/2015	10/4/2015

AKILAN N 18BCB0157 :

ApplyCheck1.java

```
import java.io.IOException;

import java.util.Scanner;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;
```

```
import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class ApplyCheck1
 */
@WebServlet("/ApplyCheck1")
public class ApplyCheck1 extends HttpServlet {

    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public ApplyCheck1() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        // TODO Auto-generated method stub
        response.getWriter().append("Served at: ").append(request.getContextPath());
    }

    /**
```

```
* @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
*/

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

    // TODO Auto-generated method stub

    String Custid=request.getParameter("Custid");

    String Type=request.getParameter("Type");

    String Loanid=request.getParameter("Loanid");

    String Amount=request.getParameter("Amount");

    String Reason=request.getParameter("Reason");

    String Aadharno=request.getParameter("Aadharno");

    String Accountno=request.getParameter("Accountno");

    Member member=new Member(Custid,Type,Loanid,Amount,Reason,Aadharno,Accountno);

    register reg=new register();

    String result=reg.insert(member);

        response.getWriter().print(result);

    }

}
```

register.java

```
import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

public class register

{
```

```
private String dburl="jdbc:mysql://localhost:3306/userdb";

private String dbuname="akil";

private String dbpassword="akilan2001";

private String dbdriver="com.mysql.jdbc.Driver";

public void loadDriver(String dbDriver)

{

    try {

        Class.forName(dbDriver);

    } catch (ClassNotFoundException e) {

        // TODO Auto-generated catch block

        e.printStackTrace();

    }

}

public Connection getConnection()

{

    Connection con=null;

    try {

        con=DriverManager.getConnection(dburl,dbuname,dbpassword);

    } catch (SQLException e) {

        // TODO Auto-generated catch block

        e.printStackTrace();

    }

    return con;

}

public String insert(Member member)

{

    loadDriver(dbdriver);

    Connection con=getConnection();
```

```
String result="data entered successfully";

String sql="insert into userdb.member values(?,?,?,?,?,?,?)";

try {

    PreparedStatement ps=con.prepareStatement(sql);

    ps.setString(1, member.getCustid());

    ps.setString(2, member.getLoanid());

    ps.setString(3, member.getAmount());

    ps.setString(4, member.getType());

    ps.setString(5, member.getReason());

    ps.setString(6, member.getAadharno());

    ps.setString(7, member.getAccountno());

    ps.executeUpdate();

} catch (SQLException e) {

    // TODO Auto-generated catch block

    e.printStackTrace();

    result="data not entered";

return result;

}

}
```

member.java

```
public class Member

{

    private String custid,type,loanid,amount,reason,Aadharno,Accountno;

    public Member() {
```



```
super();  
}
```

```
public Member(String custid, String type, String loanid, String amount, String reason, String  
Aadharno, String Accountno) {
```

```
    super();  
  
    this.custid = custid;  
  
    this.type = type;  
  
    this.loanid = loanid;  
  
    this.amount = amount;  
  
    this.reason = reason;  
  
    this.Aadharno = Aadharno;  
  
    this.Accountno=Accountno;  
  
}
```

```
public String getCustid() {  
    return custid;  
}
```

```
public void setCustid(String custid) {  
    this.custid = custid;  
}
```

```
public String getType() {  
    return type;  
}
```

```
public void setType(String type) {  
    this.type = type;
```

```
}
```

```
public String getLoanid() {  
    return loanid;  
}
```

```
public void setLoanid(String loanid) {  
    this.loanid = loanid;  
}
```

```
public String getAmount() {  
    return amount;  
}
```

```
public void setAmount(String amount) {  
    this.amount = amount;  
}
```

```
public String getReason() {  
    return reason;  
}
```

```
public void setReason(String reason) {  
    this.reason = reason;  
}
```

```
public String getAadharno() {
```

```
        return Aadharno;
    }

    public void setAadharno(String aadharno) {
        Aadharno = aadharno;
    }

    public String getAccountno() {
        return Accountno;
    }

    public void setAccountno(String accountno) {
        Accountno = accountno;
    }
}
```

ApplyCheck.java

```
import java.io.IOException;
import java.util.Scanner;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class ApplyCheck
 */
```

```

*/

@WebServlet("/ApplyCheck")

public class ApplyCheck extends HttpServlet {

    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public ApplyCheck() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        // TODO Auto-generated method stub
        response.getWriter().append("Served at: ").append(request.getContextPath());
    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        // TODO Auto-generated method stub

```

```

        String customerid=request.getParameter("customerid");

        String loan=request.getParameter("loan");

        String id=request.getParameter("id");

        String amount=request.getParameter("amount");

        String reason=request.getParameter("reason");

        String doc=request.getParameter("doc");

        if(customerid.isEmpty()||loan.isEmpty()||id.isEmpty()||amount.isEmpty()||
reason.isEmpty()||doc.isEmpty())
        {

            response.sendRedirect("error.jsp");

        }
        else
        {

            int a=0;

            String str1=Integer.toString(a);

            if(!(customerid.equals(str1)||id.equals(str1)||amount.equals(str1)))

            {

                response.sendRedirect("Approval.jsp");

            }
            else
            {

                response.sendRedirect("verify.jsp");

            }

        }

    }
}

```

Approvalcheck.java

```
import java.io.IOException;
```

```
import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;


/**
 * Servlet implementation class Approval
 */
@WebServlet("/Approvalcheck")
public class Approvalcheck extends HttpServlet {

    private static final long serialVersionUID = 1L;


    /**
     * @see HttpServlet#HttpServlet()
     */
    public Approvalcheck() {
        super();
        // TODO Auto-generated constructor stub
    }


    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

        // TODO Auto-generated method stub

        response.getWriter().append("Served at: ").append(request.getContextPath());
    }
}
```

```

    }

    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

        // TODO Auto-generated method stub

        String id=request.getParameter("id");

        String customerid=request.getParameter("customerid");

        String loan=request.getParameter("loan");

        String reason=request.getParameter("reason");

        String choose=request.getParameter("choose");

        if(id.isEmpty()||customerid.isEmpty()||loan.isEmpty()||reason.isEmpty()||
choose.isEmpty())

        {

            response.sendRedirect("error.jsp");

        }
        else

        {

            int a=0;

            String str1=Integer.toString(a);

            if(!(id.equals(str1)||customerid.equals(str1)))

            {

                response.sendRedirect("verify.jsp");

            }
            else

            {

                System.out.println("Invalid value");

```

Or

```
Response.sendRedirect("error.jsp");
```

```
}
```

```
}
```

```
}
```

```
}
```

error.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
```

```
    pageEncoding="ISO-8859-1"%>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<meta charset="ISO-8859-1">
```

```
<title>Insert title here</title>
```

```
</head>
```

```
<body>
```

Please Fill the Details

```
</body>
```

```
</html>
```

verify.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
```

```
    pageEncoding="ISO-8859-1"%>
```

```
<!DOCTYPE html>
```

```
<html>
```



```
<head>
```

```
<meta charset="ISO-8859-1">
```

```
<title>Insert title here</title>
```

```
</head>
```

```
<body>
```

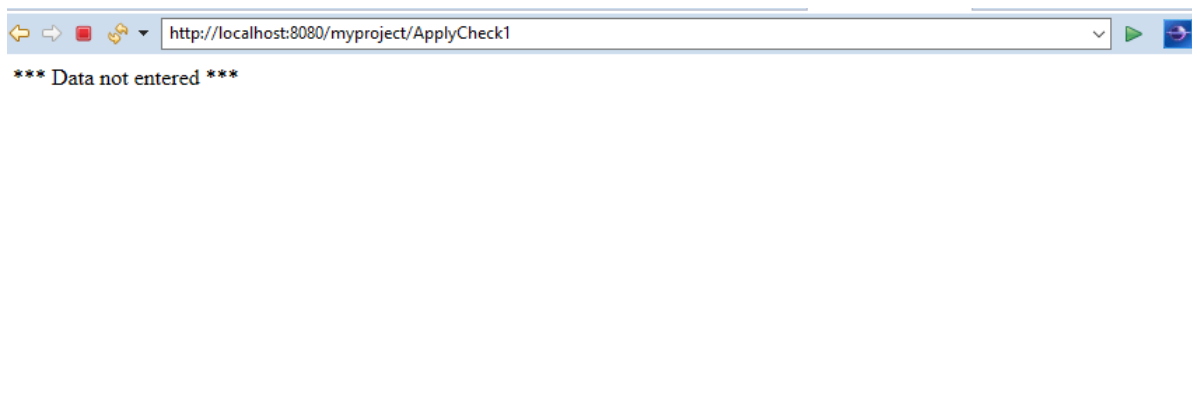
Application is Approved By Bank Manager

```
</body>
```

```
</html>
```

OUTPUTS :

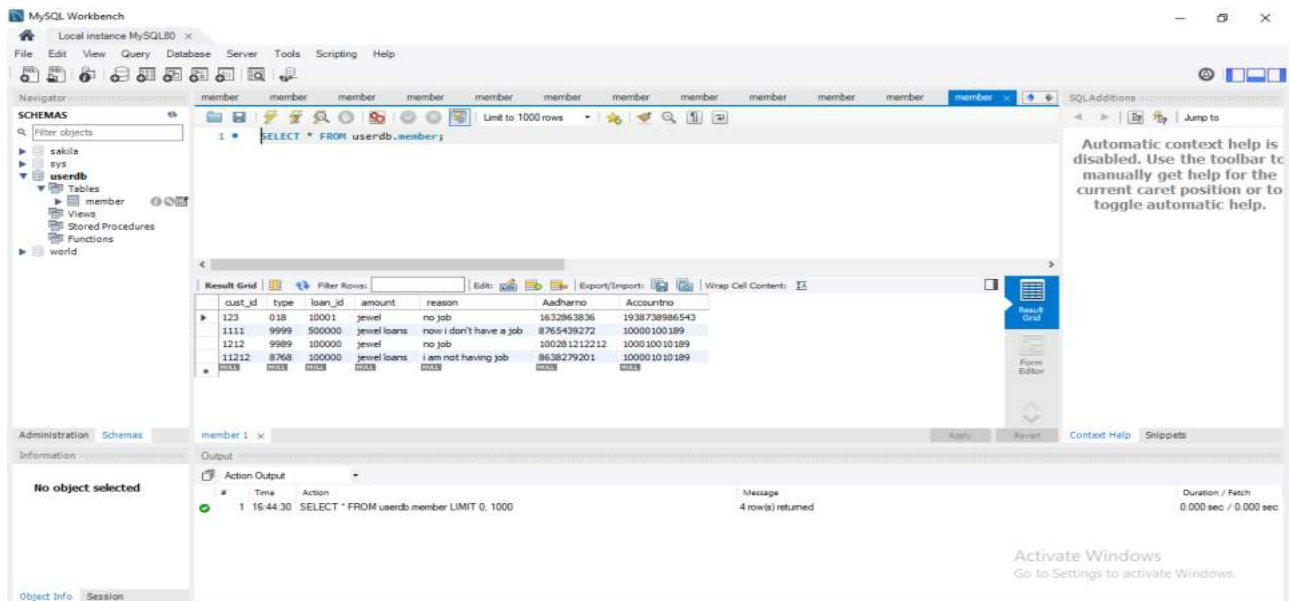
BANKRUPTCY APPLY		HOME Logout
CUSTOMER ID	<input type="text"/>	
TYPE OF LOAN	<input type="text"/>	
LOAN ID	<input type="text"/>	
LOAN AMOUNT	<input type="text"/>	
BANKRUPTCY REASON	<input type="text"/>	
AADHAR NO	<input type="text"/>	
ACCOUNT NO	<input type="text"/>	
<input type="button" value="Apply"/>		





***** Bankruptcy applied successfully*****

LOAN OFFICER APPROVAL	
LOAN OFFICER ID	<input type="text"/>
CUSTOMER ID	<input type="text"/>
BANKRUPTCY APPLY DETAILS	<div style="border: 1px solid black; height: 40px;"></div>
STATUS CHECKING	<div style="border: 1px solid black; padding: 2px;">CORRECT ▾</div>
REASON	<div style="border: 1px solid black; height: 40px;"></div>
<div style="background-color: green; color: white; padding: 2px;">APPROVE</div>	<div style="background-color: blue; color: white; padding: 2px;">DENY</div>



18BCB0145 Siddhartha Mondal**Collector_login.jsp**

```

<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Collection Department</title>
<link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">

<!-- jQuery library -->

<script

    src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

<!-- Latest compiled JavaScript -->
<script
    src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
</head>
<body>
    <div class="container">
        <button type="button" class="btn btn-info btn" data-target="#mymodal"
            data-toggle="modal">Collection Officer Login</button>
        <div id="mymodal" class="modal fade" role="dialog">
            <div class="modal-dialog">
                <div class="modal-content">
                    <div class="modal-header">
                        <button type="button" class="close" data-
dismiss="modal">&times;</button>

                        <h4 class="modal-title">Login</h4>
                    </div>
                    <div class="modal-body">
                        <form class="form-horizontal">
                            <div class="form-group">
                                <label class="control-label"
for="email">Email:</label><br>

                                <div class="col-sm-10">
                                    <input type="email"
class="form-control" id="email"

                                    placeholder="Enter
email">

```

```

</div>
</div>
<div class="form-group">
    <label class="control-label"
for="pwd">Password:</label><br>
    <div class="col-sm-10">
        <input type="password"
class="form-control" id="pwd"
placeholder="Enter
password">
    </div>
</div>
<div class="form-group">
    <div class="col-sm-offset-2">
        <a
href="Display__data.jsp">login&nbsp;&nbsp;<span
class="glyphicon
glyphicon-log-in alert-info"></span></a>
    </div>
</div>
</form>
<div class="modal-footer">
    <div class="col-sm-10">
        <button class="btn btn-alert "
type="button"
data-dismiss="modal">Close</button>
    </div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</div>
</body>
</html>
Display_data.jsp

<%@page import="java.sql.DriverManager"%>

<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.Connection"%>
<%
    String id = request.getParameter("userid");

```

```

String driver = "com.mysql.jdbc.Driver";
String connectionUrl = "jdbc:mysql://localhost:3306/";
String database = "try1";
String userid = "root";
String password = "root";
try {
    Class.forName(driver);
} catch (ClassNotFoundException e) {
    e.printStackTrace();
}
Connection connection = null;
Statement statement = null;
ResultSet resultSet = null;
%>

<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">

<!-- jQuery library -->
<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

<!-- Latest compiled JavaScript -->
<script
    src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
</head>
<body>
    <h3>Customers</h3>
    <table
        style="width: 80%; border: 1px solid blue; cell-padding: 1px; align: center"
        class="table table-bordered table-hover table-responsive">
        <tr>
            <td>C_id</td>
            <td>C_name</td>
            <td>Email</td>
            <td>Phone</td>
            <td>Payments_due</td>
        </tr>
    <%
        try {
            connection = DriverManager.getConnection(connectionUrl + database,
userid, password);

            statement = connection.createStatement();
            String sql = "select * customer";
            resultSet = statement.executeQuery(sql);
            while (resultSet.next())

```

```

        {
    %>
    <tr>
        <td><%=resultSet.getInt("C_id")%></td>
        <td><%=resultSet.getString("C_name")%></td>
        <td><%=resultSet.getString("Email")%></td>
        <td><%=resultSet.getInt("Phone")%></td>
        <td><%=resultSet.getInt("Payments_due")%></td>
    </tr>
    <%
        }
        connection.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
    %>
</table>
<button type="button" class="btn btn-large btn-success"
    data-dismiss="modal">Select for contacting</button>
</body>
</html>

```

View_report.jsp

```

<%@page import="java.sql.DriverManager"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.Statement"%>
<%@page import="java.sql.Connection"%>
<%
    String id = request.getParameter("userid");
    String driver = "com.mysql.jdbc.Driver";
    String connectionUrl = "jdbc:mysql://localhost:3306/";
    String database = "try1";
    String userid = "root";
    String password = "root";
    try {
        Class.forName(driver);
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
    Connection connection = null;
    Statement statement = null;
    ResultSet resultSet = null;
    %>

<!DOCTYPE html>
<html>
<head>

```

```

<link rel="stylesheet"
      href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">

<!-- jQuery library -->
<script
      src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

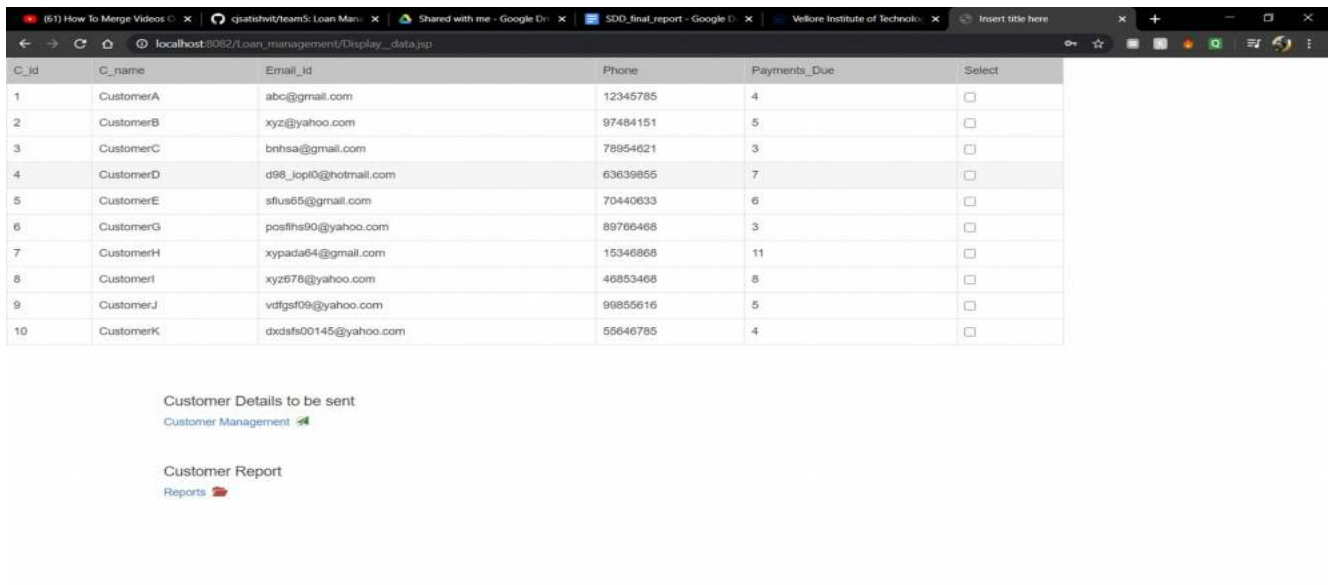
<!-- Latest compiled JavaScript -->
<script
      src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
</head>
<body>
    <h3>Customers</h3>
    <table style="width: 80%; border: 1px solid blue; cell-padding: 1px; align: center"
          class="table table-bordered table-hover table-responsive">
        <tr>
            <td>C_id</td>
            <td>C_name</td>
            <td>Method of Contact</td>
            <td>DuePaymentStatus</td>
        </tr>
        <%
        try {
            connection = DriverManager.getConnection(connectionUrl + database,
userid, password);

            statement = connection.createStatement();
            String sql = "select * viewreport2";
            resultSet = statement.executeQuery(sql);
            while (resultSet.next()) {
                <tr>
                    <td><%=resultSet.getInt("C_id")%></td>
                    <td><%=resultSet.getString("C_name")%></td>
                    <td><%=resultSet.getString("MethodOfContact ")%></td>
                    <td><%=resultSet.getInt("DuePaymentStatus")%></td>
                </tr>
                <%
            }
            connection.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
        %>
    </table>

```

</body>
</html>

Screenshots(18BCB0145)

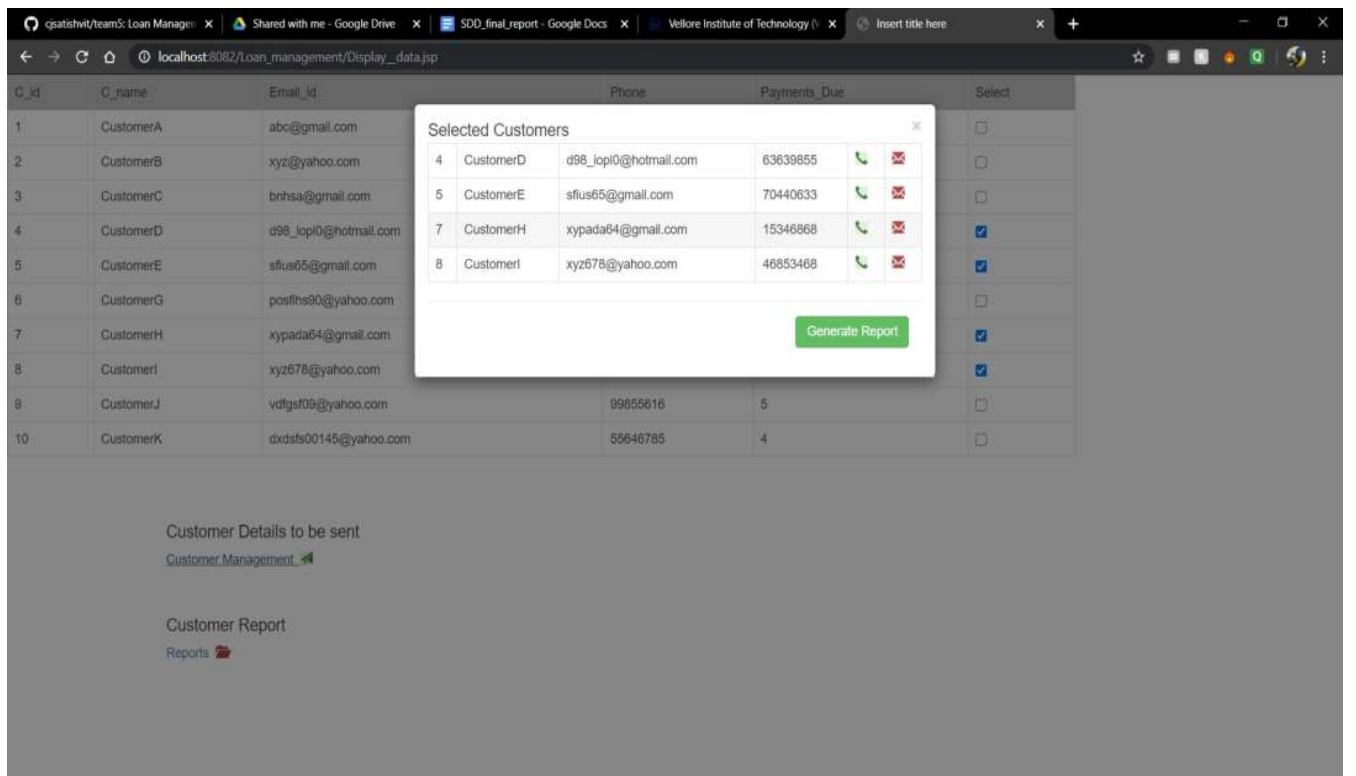
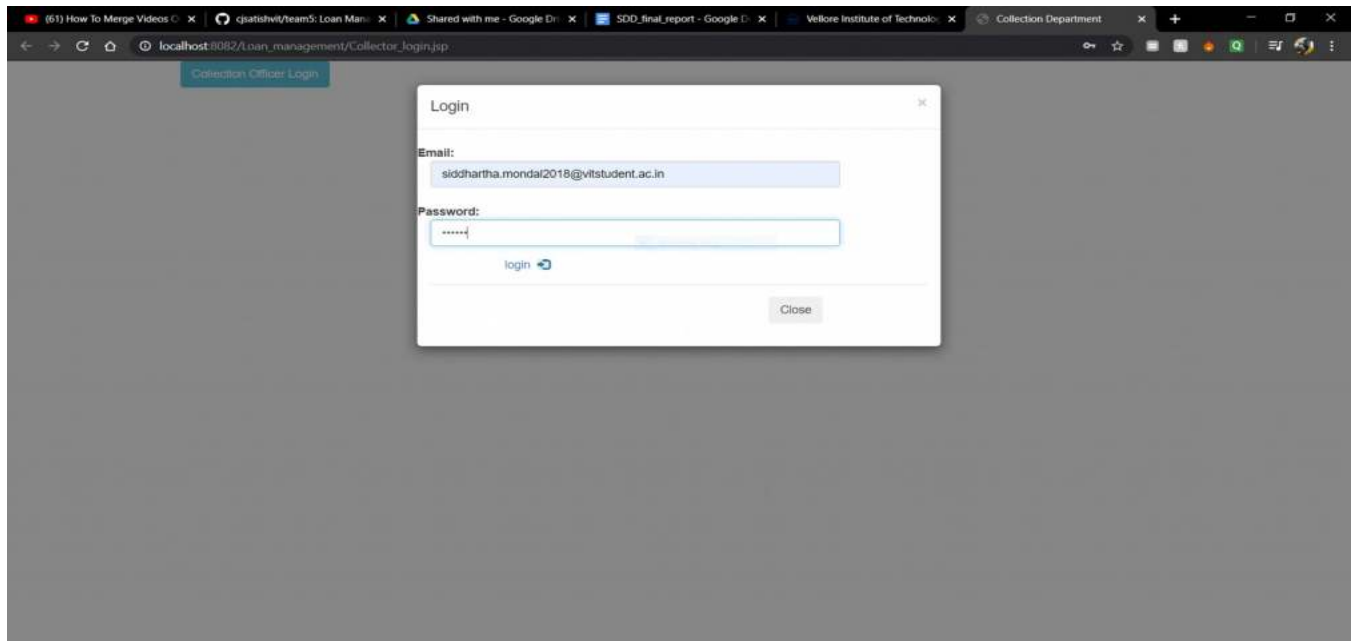


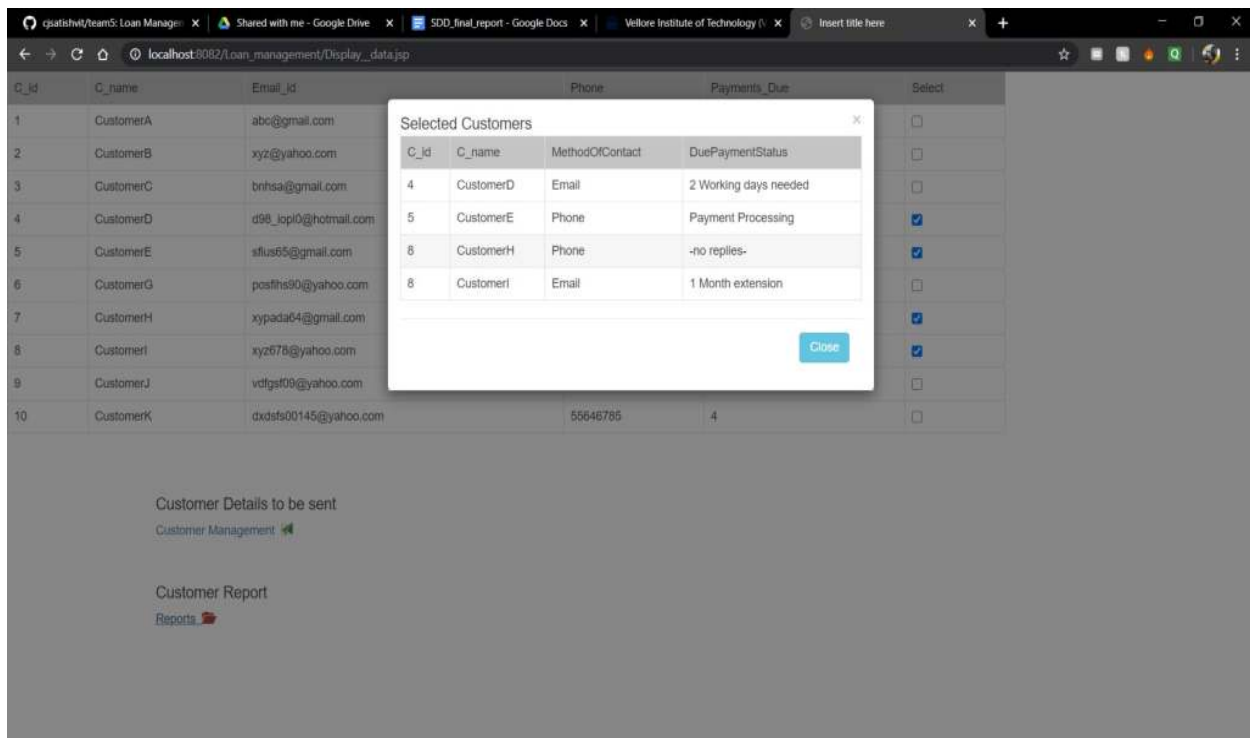
C_id	C_name	Email_id	Phone	Payments_Due	Select
1	CustomerA	abc@gmail.com	12345785	4	<input type="checkbox"/>
2	CustomerB	xyz@yahoo.com	97484151	5	<input type="checkbox"/>
3	CustomerC	bnhsa@gmail.com	78954621	3	<input type="checkbox"/>
4	CustomerD	d98_lopl0@hotmail.com	63639855	7	<input type="checkbox"/>
5	CustomerE	stfus65@gmail.com	70440633	6	<input type="checkbox"/>
6	CustomerG	postfhs90@yahoo.com	89766468	3	<input type="checkbox"/>
7	CustomerH	xypada64@gmail.com	15346868	11	<input type="checkbox"/>
8	CustomerI	xyz676@yahoo.com	46853468	8	<input type="checkbox"/>
9	CustomerJ	vdfgsf09@yahoo.com	99855616	5	<input type="checkbox"/>
10	CustomerK	dxdsfs00145@yahoo.com	55646785	4	<input type="checkbox"/>

Customer Details to be sent
[Customer Management](#) 🟢

Customer Report
[Reports](#) 🔴

cse1005 –Software Design and Development –J COMPONENT PROJECT WORK REPORT





ANUPAMA GAUTAM(18BCB0019) :

Register.java

```
import java.io.IOException;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/**
 * Servlet implementation class register
 */

@WebServlet("/Register")

public class Register extends HttpServlet {
```

```

private static final long serialVersionUID = 1L;

/**
 * @see HttpServlet#HttpServlet()
 */
public Register() {
    super();

    // TODO Auto-generated constructor stub
}

/**
 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
 */
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    // TODO Auto-generated method stub

    response.getWriter().append("Served at: ").append(request.getContextPath());
}

/**
 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
 */
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    // TODO Auto-generated method stub

    String name=request.getParameter("name");

    String customer=request.getParameter("customer");

    String amount=request.getParameter("amount");

    String id=request.getParameter("id");

    String income=request.getParameter("income");

```

```
String address=request.getParameter("address");

String reason=request.getParameter("reason");

String aadhar=request.getParameter("aadhar");

String proof=request.getParameter("proof");

Member member=new Member(name, customer, amount, id, income, address, reason,
aadhar, proof);

RegisterDao rDao=new RegisterDao();

String result=rDao.insert(member);

response.getWriter().print(result);

    }
}
```

RegisterDao.java

```
import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.SQLException;

public class RegisterDao

{

    private String dburl="jdbc:mysql://localhost:3306/banruptcy";

    private String dbuname="anu";

    private String dbpassword="Anu@2000";

    private String dbdriver="com.mysql.jdbc.Driver";

    public void loadDriver(String dbDriver)

    {

        try {

            Class.forName(dbDriver);

        } catch (ClassNotFoundException e) {
```

```

        e.printStackTrace();
    }
}

public Connection getConnection()
{
    Connection con=null;
    try {
        con=DriverManager.getConnection(dburl, dbuname, dbpassword);
    } catch (SQLException e) {

        e.printStackTrace();
    }
    return con;
}

public String insert(Member member)
{
    loadDriver(dbdriver);
    Connection con=getConnection();

    String result="Data entered successfully. Your status will be mailed to you in 30 working
days";

    String sql="insert into banruptcy.member values(?,?,?,?,?,?,?,?)";
    try {
        PreparedStatement ps=con.prepareStatement(sql);
        ps.setString(1, member.getName());
    }
}

```

```
ps.setString(2, member.getCustomer());  
ps.setString(3, member.getAmount());  
ps.setString(4, member.getId());  
ps.setString(5, member.getIncome());  
ps.setString(6, member.getAddress());  
ps.setString(7, member.getReason());  
ps.setString(8, member.getAadhar());  
ps.setString(9, member.getProof());  
ps.executeUpdate();
```

```
    } catch (SQLException e) {  
  
        e.printStackTrace();  
        result="Data not entered";  
    }
```

```
    return result;
```

```
}
```

```
}
```

Member.java

```
public class Member
```

```
{
```

```
    private String name,customer,amount,id,income,address,reason,aadhar,proof;
```

```
public Member() {  
    super();  
}  
  
    public Member(String name, String customer, String amount, String id, String income, String  
address, String reason,  
        String aadhar, String proof) {  
        super();  
        this.name = name;  
        this.customer = customer;  
        this.amount = amount;  
        this.id = id;  
        this.income = income;  
        this.address = address;  
        this.reason = reason;  
        this.aadhar = aadhar;  
        this.proof = proof;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name =name;  
    }  
  
    public String getCustomer() {  
        return customer;  
    }  
  
    public void setCustomer(String customer) {  
        this.customer =customer;  
    }  
  
    public String getAmount() {  
        return amount;
```

```
}

public void setAmount(String amount) {

    this.amount =amount;

}

public String getId() {

    return id;

}

public void setId(String id) {

    this.id =id;

}

public String getIncome() {

    return income;

}

public void setIncome(String income) {

    this.income =income;

}

public String getAddress() {

    return address;

}

public void setAddress(String address) {

    this.address =address;

}

public String getReason() {

    return reason;

}

public void setReason(String reason) {
```



```
        this.reason =reason;
    }

    public String getAadhar() {
        return aadhar;
    }

    public void setAadhar(String aadhar) {
        this.aadhar =aadhar;
    }

    public String getProof() {
        return proof;
    }

    public void setProof(String proof) {
        this.proof =proof;
    }
}
```

Approval.java

```
import java.sql.DriverManager;

import java.sql.Connection;

import java.sql.Statement;

import java.sql.ResultSet;


public class approval {


    public static void main(String[] args) {

        // TODO Auto-generated method stub

        try
```

```
{  
  
    Class.forName("com.mysql.jdbc.Driver");  
  
    Connection con =  
DriverManager.getConnection("jdbc:mysql://localhost:3306/userdb","anu","Anu@2000");  
  
    Statement stat = con.createStatement();  
  
    ResultSet rs = stat.executeQuery("select * from banruptcy.member");  
  
    while(rs.next())  
  
        System.out.println(rs.getString(1)+" "+rs.getString(2)+" "+rs.getString(3)+"  
"+rs.getString(4)+" "+rs.getString(5)+" "+rs.getInt(6)+" "+rs.getInt(7)+rs.getString(8)+" "+rs.getString(9));  
  
    con.close();  
  
}  
  
catch(Exception e)  
  
{  
  
    System.out.println(e);  
  
}  
  
}
```

OUTPUT SCREENSHOTS (18BCB0019): ANUPAMA GAUTAM

LOAN STATUS
BANKRUPTCY PERMISSION
CREDIT CARD LOANS

HOME | Logout

NAME	<input type="text" value="anujama"/>
CUSTOMER ID	<input type="text" value="8888"/>
LOAN AMOUNT TO BE PAID	<input type="text" value="500000"/>
LOAN ID	<input type="text" value="8787"/>
MONTHLY INCOME	<input type="text" value="90000"/>
ORIGINAL ADDRESS	<input type="text" value="Acikate"/>
REASON FOR BANKRUPTCY	<input type="text" value="Business Loan"/>
AADHAR NO.	<input type="text" value="88888"/>
BANKRUPTCY LEGAL PROOF	<input type="text" value="Business document stat"/>

Your permission status will be sent to your email within 30 working days

http://localhost:8082/bankrupt/Register

Data entered successfully. Your status will be mailed to you in 30 working days

APPROVAL OF BANKRUPTCY
CREDIT CARD LOANS

HOME | Logout

NAME	<input type="text" value="anujama"/>
LOAN ID	<input type="text" value="8088"/>
CUSTOMER ID	<input type="text" value="8888"/>
LOAN DETAILS	<input type="text" value="500000"/>
VERIFICATION OF LEGAL DOCUMENTS	<input type="text" value="CORRECT ✓"/>
REASON OF APPROVAL/REJECTION	<input type="text" value="verified correct legal documents and police documents"/>