

## **E-BANKING**

*An industrial oriented mini project report*

Submitted By

**ANEK PORWAL**  
**(Regd. No: 13W91A0509)**

Under the Esteemed Guidance of

**Mr. D. KALYAN KUMAR**  
Assistant Professor, CSE  
*To*

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY**  
**HYDERABAD**

In partial fulfillment of the requirements for award of degree of

**BACHELOR OF TECHNOLOGY**  
**IN**  
**COMPUTER SCIENCE & ENGINEERING**  
2013 – 2017



**MALLA REDDY**  
INSTITUTE OF  
ENGINEERING AND TECHNOLOGY

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**  
**MALLA REDDY INSTITUTE OF ENGINEERING AND**  
**TECHNOLOGY (MRET - W9)**

(Sponsored by Malla Reddy Educational society)  
(Affiliated to JNTUH, Accredited by NBA)

## **DECLARATION**

We hereby declare that the project entitled “**E-BANKING**” submitted to Malla Reddy Institute of Engineering and Technology (MRET-W9), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of Bachelor of Technology in Computer Science & Engineering is a result of original industrial oriented project done by us.

It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

1.     **Anek Porwal - (13W91A0509).**
2.     **Manish Ambati - (13W91A0508).**
3.     **Aishwarya Devi - (13A91A0525).**



**MALLA REDDY**  
INSTITUTE OF  
ENGINEERING AND TECHNOLOGY

## **MALLA REDDY INSTITUTE OF ENGINEERING & TECHNOLOGY**

(Sponsored by Malla Reddy Educational Society)  
Affiliated to JNTUH, Accredited By NBA  
Maisammaguda, Dhulapally (post via Hakimpet), Sec'Bad-500 014.  
Phone: 040-65969674, Cell: 9348161223

### Department of Computer Science and Engineering

### PROJECT CERTIFICATE

This is to certify that this is the bonafide record of the project titled “**E-BANKING**” is submitted by **ANEK PORWAL (13W91A0509)** , **MANISH AMBATI (13W91A0508)**, **AISHWARYA DEVI (13W91A0525)** of B.Tech in the partial fulfillment of the requirements for the degree of **Bachelor of Technology in Computer Science and Engineering**, Dept. of Computer Science & Engineering and this has not been submitted for the award of any other degree of this institution.

**INTERNAL GUIDE**

**PRINCIPAL**

**PROJECT COORDINATOR**

**HOD**

**INTERNAL EXAMINER**

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

First and foremost, I am grateful to the Principal **Dr. K. E. Balachandrudhu**, for providing me with all the resources in the college to make my project a success. I thank him for his valuable suggestions at the time of seminars which encouraged me to give my best in the project.

I would like to express my gratitude to **Mr. M. Sreenivas**, Dean Academics for his support and valuable suggestions during the dissertation work.

I would like to express my gratitude to **Mr. M. Sreenivas**, Head of the Department, Department of Computer Science and Engineering for his support and valuable suggestions during the dissertation work.

I offer my sincere gratitude to my project coordinator **Mrs. B. Bhavya** and internal guide **Mr. D. Kalyan Kumar** Assistant Professor of Computer Science and Engineering department who has supported me throughout this project with their patience and valuable suggestions.

I would also like to thank all the supporting staff of the Dept. of CSE and all other departments who have been helpful directly or indirectly in making the project a success.

I am extremely grateful to my parents for their blessings and prayers for my completion of project that gave me strength to do my project.

**ANEK PORWAL.**

**MANISH AMBATI.**

**AISHWARYA DEVI.**

## INDEX

Abstract .....	i
List of Figures.....	ii
List of Screens.....	iii
Acronyms.....	iv

CONTENT	PAGE NO.
<b>1. INTRODUCTION</b>	<b>1</b>
1.1 Motivation	2
1.2 Problem Definition	4
1.3 Objective of the Project	4
1.4 Limitations of the Project	5
1.5 Organization of Documentation	5
<b>2. LITERATURE SURVEY</b>	<b>7</b>
2.1 Introduction	8
2.2 Existing System	8
2.3 Disadvantages of Existing system	9
2.4 Proposed System	10
2.5 Conclusion	10
<b>3. ANALYSIS</b>	<b>11</b>
3.1 Introduction	12
3.2 Software Requirement Specification	35
3.2.1 User requirement	35
3.2.2 Software requirement	37
3.2.3 Hardware requirement	37
3.3 Content of the Project	37
3.4 Conclusion	38

<b>4. DESIGN</b>	<b>39</b>
4.1 Introduction	40
4.2 UML diagrams	47
4.3 Module design and organization	67
4.4 Conclusion	68
<b>5. IMPLEMENTATION &amp; RESULTS</b>	<b>69</b>
5.1 Introduction	70
5.2 Explanation of Key functions	70
5.3 Method of Implementation	71
5.3.1 Sample Code	71
5.3.2 Output Screens	76
5.3.3 Result Analysis	82
5.4 Conclusion	82
<b>6. TESTING &amp; VALIDATION</b>	<b>83</b>
6.1 Introduction	84
6.2 Design of test cases and scenarios	84
6.3 Validation	86
6.4 Conclusion	94
<b>7. CONCLUSION</b>	<b>95</b>
7.1 Project Conclusion	96
7.2 Future Enhancement	96
<b>8. REFERENCES</b>	<b>97</b>
8.1 Referred Book(s)	98
8.2 Referred Application(s)	98

## **ABSTRACT**

This project aims at creation of a secure Internet banking system. This will be accessible to all customers who have a valid User Id and Password. This is an approach to provide an opportunity to the customers to have some important transactions to be done from where they are at present without moving to bank. In this project we are going to deal the existing facts in the bank i.e.; the transactions which takes place between customer and bank. We provide a real time environment for the existing system in the bank. We deal in the method transaction in the bank can be made faster and easier that is our project is an internet based computerized approach towards banking.

Our Project aims on improving the overall security of the Internet Banking System by providing Multi-factor Authentication and implementing the online system with Hyper-Text-Transfer-Protocol along with Secure-Socket-layer-Protocol. We are also providing the safe environment for user by implementing One-Way-Authentication.

## LIST OF FIGURES

<b>S NO.</b>	<b>FIG NO.</b>	<b>DESRPTION</b>	<b>PAGE NO.</b>
<b>1</b>	<b>3.1</b>	<b>Java Architecture</b>	<b>17</b>
<b>2</b>	<b>3.2</b>	<b>System Architecture</b>	<b>38</b>
<b>3</b>	<b>4.1</b>	<b>Requirements Gathering Stage</b>	<b>41</b>
<b>4</b>	<b>4.2</b>	<b>Analysis Stage</b>	<b>42</b>
<b>5</b>	<b>4.3</b>	<b>Design Stage</b>	<b>43</b>
<b>6</b>	<b>4.4</b>	<b>Developing Stage</b>	<b>44</b>
<b>7</b>	<b>4.5</b>	<b>Integration &amp; Testing Stage</b>	<b>45</b>
<b>8</b>	<b>4.6</b>	<b>Installation &amp; Acceptance Stage</b>	<b>46</b>
<b>9</b>	<b>4.7</b>	<b>Use Case Diagram</b>	<b>50</b>
<b>10</b>	<b>4.8</b>	<b>Class Diagram</b>	<b>53</b>
<b>11</b>	<b>4.9</b>	<b>Activity Diagram for Transaction</b>	<b>56</b>
<b>12</b>	<b>4.10</b>	<b>Activity Diagram for Login</b>	<b>57</b>
<b>13</b>	<b>4.11</b>	<b>Sequence Diagram for Transaction</b>	<b>60</b>
<b>14</b>	<b>4.12</b>	<b>Sequence Diagram for Registration</b>	<b>61</b>
<b>15</b>	<b>4.13</b>	<b>E-R Diagram</b>	<b>66</b>
<b>16</b>	<b>6.1</b>	<b>Validation &amp; WorkFlow</b>	<b>86</b>



**LIST OF SCREENSHOTS**

<b>SCREEN SHOT NO.</b>	<b>DESCRIPTION</b>	<b>PAGE NO.</b>
<b>1</b>	<b>Home Page</b>	<b>76</b>
<b>2</b>	<b>Developer Page</b>	<b>77</b>
<b>3</b>	<b>Login Page</b>	<b>78</b>
<b>4</b>	<b>Profile Edit Page</b>	<b>79</b>
<b>5</b>	<b>Transaction Page</b>	<b>80</b>
<b>6</b>	<b>Registration Page</b>	<b>81</b>
<b>7</b>	<b>Password Not Entered Test Case</b>	<b>88</b>
<b>8</b>	<b>Username Not Entered Test Case</b>	<b>89</b>
<b>9</b>	<b>Incorrect Password Test Case</b>	<b>90</b>
<b>10</b>	<b>Password Not Matching Test Case</b>	<b>91</b>
<b>11</b>	<b>Invalid Account Test Case</b>	<b>92</b>
<b>12</b>	<b>Invalid Transaction Password Test Case</b>	<b>93</b>

**ACRONYMNS**

<b>SNO.</b>	<b>Acronym</b>	<b>Description</b>
<b>1</b>	<b>JDBC</b>	<b>Java Driver Bridge Connection.</b>
<b>2</b>	<b>HTML</b>	<b>Hyper Text Markup Language.</b>
<b>3</b>	<b>JS</b>	<b>JavaScript.</b>
<b>4</b>	<b>IBM</b>	<b>International Business Machine.</b>
<b>5</b>	<b>HTTP</b>	<b>Hyper Text Transfer Protocol.</b>
<b>6</b>	<b>JSP</b>	<b>Java Server Pages.</b>
<b>7</b>	<b>API</b>	<b>Application Package Interface.</b>
<b>8</b>	<b>JVM</b>	<b>Java Virtual Machine.</b>
<b>9</b>	<b>GUI</b>	<b>Graphical User Interface.</b>
<b>10</b>	<b>CGI</b>	<b>Common Gateway Interface.</b>



# INTRODUCTION



## **1. INTRODUCTION.**

### **1.1. MOTIVATION.**

The advent of E-Business accompanied with technological innovations and globalization is constantly propelling the businesses organization to redefine their business operations in terms of value chain reengineering and restructuring business models. Likely, the financial sector is metamorphosing under the impact of competitive, regulatory and technological forces . Financial institutions especially the banking sector is currently in a transition phase. The banks have put themselves in the World Wide Web to take advantage of the internet's power and reach, to cope with the accelerating pace of change of business environment. The famous quote by Bill Gates that banking is vital to a healthy economy, but banks themselves are not, highlights the crucial nature of the electronic forces that are affecting banks more than any other financial service provider group. This transition of business operations by banks have created new mode of operation called E-Banking. E-banking is the banking of new era. The term Internet Banking or E-Banking Internet both are used as supplement. Making banking products and other services available to wholesale and retail customers, through an electronic distribution channel is called e-banking. In other words E-Banking refers to the banking operations, which is done over World Wide Web. E-banking is the outcome of technological innovations and competition. In fact, banks have been using electronic and telecommunication networks for delivering a wide range of value added products and services. The devices have been telephone, personal computers including Automated Teller Machines (ATM). The delivery channels have been direct dial up connections, private and public networks. To this newer edition of e-banking are being added e.g. Internet banking and mobile banking. The use of ATM's lead to the concept of 'anywhere' and 'anytime' banking. Through the use of ATM cards, one can operate his bank account to withdraw money from any of bank's ATM installed or available at the nearest site. This had broken down the time and space barriers. The new banks are providing some of the services exclusively through ATM's.

The growing popularity of personal computers, easy access to internet and world wide web (WWW), has increased the use of internet by banks as a channel for receiving

instructions and also delivering their products and services to the customers,. This is generally referred to as 'Internet banking' or I-banking or Net Banking. This is one of the newer forms e-banking which is gaining popularity. The other popular name for e-banking is on line banking. A. Indian Scene: internet banking has gained wide acceptance internationally. In India also the things are changing fast. India is on the threshold of a major banking revolution with the introduction of 'Net-banking'. In year 2002, only a dozen or so banks are providing services at different levels. However, almost double those numbers are ready to make entry. Others may be compelled to follow. B. Future Scene in India: The Indian banks lag far behind the international banks in providing online banking. In fact, this is not possible without creating sufficient infrastructure or presence of sufficient number of users. The experience of ICICI Bank Ltd. and HDFC Bank Ltd. shows that the number of transactions carried out on the Net are very limited. No matter where online banking comes from or where it is today, it is most certainly here to stay. As a tool of modern living and as a lifestyle aid, it is absolutely indispensable. The fact is that many services that are now being offered with online banking are almost impossible to avail of in regular banking. This holds even truer for developments that the future of online banking will bring. Judging by its current popularity and rate of implementation, online banking will increase in scope and user base in the future. Individuals and businesses that have refused to adopt it as a commercial tool before now will not really be left with much choice. As things stand now, using conventional methods of business inquiry, confirmation, order-placement and bill payment is retrograde at best. The speed (and reduced manpower) with which the same activities can be done online leaves the traditional methods completely in the shade. The same holds true for normal vs. online business-related banking. In a domestic setting, too, shopping has been and will continue to be revolutionized by the Internet option. Payments for goods bought online will almost always be done via a bank-related credit card or direct deduction from a bank account. However, traditional biases against online banking are not going to be overcome entirely by hard facts and figures in its favor. The rate at which this sector of banking will grow will depend on how many user-friendly facilities are incorporated the additional facilities that will be added and the way the concept is packaged for the general public. It is unfortunately a fact that banks and their customers have rarely agreed on what is a useful facility and what isn't. A lot of market research and customer polling will be required before the gap between what is needed in banking and what is available is filled. When that finally happens, the final frontier of online banking will indeed be explored.

## **1.2. PROBLEM DEFINITION:**

The major concern for an Internet -banking is the 'security'. There are many remote customers accessing the system and placing various requests/queries to get the required information or to make transactions with the bank at the time demanded. There are various aspects that are needed to address in this application. There should be a report generating Balance Enquiry the system need to guarantee the funds transfer to another account of the same bank. The system should provide assistance for request for

- Cheque book
- Change of address
- Stop payment of cheque's

The system must generate various reports for the customers to view monthly and annual statements.

## **1.3. OBJECTIVES OF THE PROJECT:**

- To enable the user to register, i.e., opens an account in the bank by sitting anywhere by using Internet.
- To enable the user to login and deposit the amount in to bank.
- To allow the user to login and with draws the amount from the bank.

To allow the user to login and see the balance and also the see the transactions he performed with the bank date wise in a tabular form.

### **Need for Computerization**

Computerization is absolutely necessary to facilitate or automate various procedures and several transactions. Some salient features of computerization are:

- Reduction in processing time
- Data security.
- Reduced redundancy & inconsistency.

## **1.4. LIMITATIONS OF PROJECT**

- Withdrawal is not available through E-Banking.



- Services Related to Credit-Card Not available.
- Pass-Book Printing Services Not Available.
- Inter-Bank Transaction Not Available.

### **1.5. ORGANIZATION OF DOCUMENT:**

Online banking, also known as internet banking, e-banking or virtual banking, is an electronic payment system that enables customers of a bank or other financial institution to conduct a range of financial transactions through the financial institution's website. The online banking system will typically connect to or be part of the core banking system operated by a bank and is in contrast to branch banking which was the traditional way customers accessed banking services.

To access a financial institution's online banking facility, a customer with internet access would need to register with the institution for the service, and set up a password and other credentials for customer verification. The credentials for online banking is normally not the same as for telephone or mobile banking. Financial institutions now routinely allocate customers numbers, whether or not customers have indicated an intention to access their online banking facility. Customer numbers are normally not the same as account numbers, because a number of customer accounts can be linked to the one customer number. Technically, the customer number can be linked to any account with the financial institution that the customer controls, though the financial institution may limit the range of accounts that may be accessed to, say, cheque, savings, loan, credit card and similar accounts.

The customer visits the financial institution's secure website, and enters the online banking facility using the customer number and credentials previously set up. The types of financial transactions which a customer may transact through online banking are determined by the financial institution, but usually includes obtaining account balances, a list of the recent transactions, electronic bill payments and funds transfers between a customer's or another's accounts. Most banks also enable a customer to download copies of bank statements, which can be printed at the customer's premises (some banks charge a fee for mailing hard copies of bank statements). Some banks also enable customers to download transactions directly into the customer's accounting software.

The largest bank, and the oldest still in existence, is the State Bank of India (S.B.I). It originated as the Bank of Calcutta in June 1806. In 1809, it was renamed as the Bank of Bengal. This was one of the three banks funded by a presidency government, the other two were the Bank of Bombay and the Bank of Madras. The three banks were merged in 1921 to form the Imperial Bank of India, which upon India's independence, became the State Bank of India in 1955. For many years the presidency banks had acted as quasi-central banks, as did their successors, until the Reserve Bank of India was established in 1935, under the Reserve Bank of India Act, 1934.<sup>[5][6]</sup>



# **LITERATURE SURVEY**

## **2. LITERATURE SURVEY**

### **2.1. INTRODUCTION:**

During the analysis phase the existing system was studied. The data flow in the existing system was studied. As part of the analysis; various documents for account opening, money transferred issuing, Cash withdrawing, customer information reports, and transaction reports were all collected. These were used in later stages to design the computerized forms used the existing system was determined. The deliverable for this stage was documentation on the existing system.

The system study is the first phase in the system life cycle. It involves studying the ways an organization currently retrieves and process data to produce information with the goal of determining how to make it better. For this, system analyst should develop alternative system and evaluate each terms of cost, benefit and feasibility. The term analysis, design and development are used in sequence, because in practice this sequence of steps used to construct computer based information system. System analysis includes the investigation and possible changes to the existing system. Analysis is used to gain an understanding of the existing system description and set of requirements for a new system. If there is no existing system, then analysis only defines the requirements. Development begins by defining a model of the new system and continues this model to a working system. The module of the system shows what the system must do to satisfy these' requirements. Finally, the data models are to a database and processed to user procedures and computer programs.

### **2.2. EXISTING SYSTEM**

In current situation if any person wants to go for banking they can do all transactions either manually or through online. But in both situations customer must be interact with bank people and they must be interact with database and also administrator (bank staff) can view all customers reports but customers can't view their own individual transaction reports. This is the main disadvantage in online banking.

The existing system involves the following activities:

- The present system consists of networking environment wherein regular activities are automated.

- Readability of the records, which are maintained manually, is also constrained in the present system.
- Since record are kept on a paper registers, again is also a problem.
- Further retrieving information from such records for a period is tedious, as the storage place restricts, old records will be kept off the disk.
- Also report generation of the various areas is done manually using great amount of manpower and time.
- Erroneous records may lead to misleading information, which is more likely in manual system.
- The great limitation to the existing system is that the service to the customers is limited to the bank hours only. The online banking facility provides 24 hours service to the customer.

### **2.3. DISADVANTAGES OF EXISTING SYSTEM:**

The following are the Disadvantages of the Existing System.

- Leads to tedious manual work.
- Enormous amount of time consumption for recording all transactions.
- Error can occur during the manipulation of several records.
- Lack of technical background towards the system.
- Lack of Data Security.
- No Proper Implementation of Encryption & Decryption.

### **2.4. PROPOSED SYSTEM:**

This proposed system aims at creation of a secure Internet -Banking system. This will be accessible to all customers who have a valid 'user id' and 'Password' the system provides .The following important functionalities Balance Enquiry, Funds Transfer to another account in the same bank, Request for Cheque book\change of address\stop payment of Cheque, Viewing monthly and annual statement.

- Balance Enquiry .

- Funds Transfer to another account in the same bank .
- Request for Cheque book\change of address\stop payment of Cheque.
- Viewing monthly and annual statements.
- Protection from SQL Attacks.
- One-Way Encryption.
- Multi-Factor Authentication for Money Transaction.

In order to overcome the drawbacks in the existing system database is created which is:

- Integrated
- Accessibility
- Reliable
- Consistent
- Flexible
- Secure

The present database Helps in speedy information retrieval, Extract information from tables using menus, Offers options of the online updating and in main ting up to date information.

## **2.5 CONCLUSION:**

By using this Application, the user is able to track nearby locations and the products available and their prices. This makes the Application more reliable and flexible when compared to other applications in the market. And, due to the usage of IBM's Work - light, it is easier to deploy the Application onto different platforms, and hence expands the user count leading to an increase in profit.

# ANALYSIS



## **3. ANALYSIS**

### **3.1. INTRODUCTION**

#### **3.1.1 Study of the system**

To provide flexibility to the users, the interfaces have been developed that are accessible through a browser. The GUI'S at the top level have been categorized as

- Administrative user interface
- The operational or generic user interface

The 'administrative user interface' concentrates on the consistent information that is practically, part of the organizational activities and which needs proper authentication for the data collection. These interfaces help the administrators with all the transactional states like Data insertion, Data deletion and Date updation along with the extensive data search capabilities.

The 'operational or generic user interface' helps the end users of the system in transactions through the existing data and required services. The operational user interface also helps the ordinary users in managing their own information in a customized manner as per the included flexibilities

#### **3.1.2 Input & Output Representation**

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

### **INPUT STAGES:**

The main input stages can be listed as below:

- Data recording
- Data transcription

- Data conversion
- Data verification
- Data control
- Data transmission
- Data validation
- Data correction

**INPUT TYPES:**

It is necessary to determine the various types of inputs. Inputs can be categorized as follows:

- External inputs, which are prime inputs for the system.
- Internal inputs, which are user communications with the system.
- Operational, which are computer department's communications to the system?
- Interactive, which are inputs entered during a dialogue.

**INPUT MEDIA:**

At this stage choice has to be made about the input media. To conclude about the input media consideration has to be given to;

- Type of input
- Flexibility of format
- Speed
- Accuracy
- Verification methods
- Rejection rates
- Ease of correction
- Storage and handling requirements
- Security

- Easy to use
- Portability

Keeping in view the above description of the input types and input media, it can be said that most of the inputs are of the form of internal and interactive. As Input data is to be directly keyed in by the user, the keyboard can be considered to be the most suitable input device.

## **OUTPUT DESIGN:**

In general are:

- External Outputs whose destination is outside the organization.
- Internal Outputs whose destination is within organization and they are the User's main interface with the computer. Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly with the system.

## **OUTPUT DEFINITION**

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

For Example

- Will decimal points need to be inserted
- Should leading zeros be suppressed.

## **OUTPUT MEDIA:**

In the next stage it is to be decided that which medium is the most appropriate for the output. The main considerations when deciding about the output media are:

- The suitability for the device to the particular application.
- The need for a hard copy.
- The response time required.
- The location of the users
- The software and hardware available.

Keeping in view the above description the project is to have outputs mainly coming under the category of internal outputs. The main outputs desired according to the requirement specification are:

The outputs were needed to be generated as a hard copy and as well as queries to be viewed on the screen. Keeping in view these outputs, the format for the output is taken from the outputs, which are currently being obtained after manual processing. The standard printer is to be used as output media for hard copies.

## **About Java:**

Initially the language was called as “oak” but it was renamed as “java” in 1995. The primary motivation of this language was the need for a platform-independent (i.e. architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

- Java is a programmer’s language
- Java is cohesive and consistent
- Except for those constraint imposed by the Internet environment. Java gives the programmer, full control.

- Finally Java is to Internet Programming where c was to System Programming.
- Importance of Java to the Internet

Java has had a profound effect on the Internet. This is because; java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the server and the personal computer. They are passive information and Dynamic active programs. in the areas of Security and probability. But Java addresses these concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

Applications and applets. An application is a program that runs on our Computer under the operating system of that computer. It is more or less like one creating using C or C++ .Java's ability to create Applets makes it important. An Applet is an application, designed to be transmitted over the Internet and executed by a Java-compatible web browser. An applet is actually a tiny Java program, dynamically downloaded across the network, just like an image. But the difference is, it is an intelligent program, not just a media file. It can be react to the user input and dynamically change.

## **Java Architecture**

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

## **Compilation of code**

When you compile the code, the Java compiler creates machine code (called byte code)for a hypothetical machine called Java Virtual Machine(JVM). The JVM is supposed to execute the byte code. The JVM is created for the overcoming the issue of probability. The code is written and compiled for one machine and interpreted on all machines .This machine is called Java Virtual Machine.

## **Compiling and interpreting java source code.**

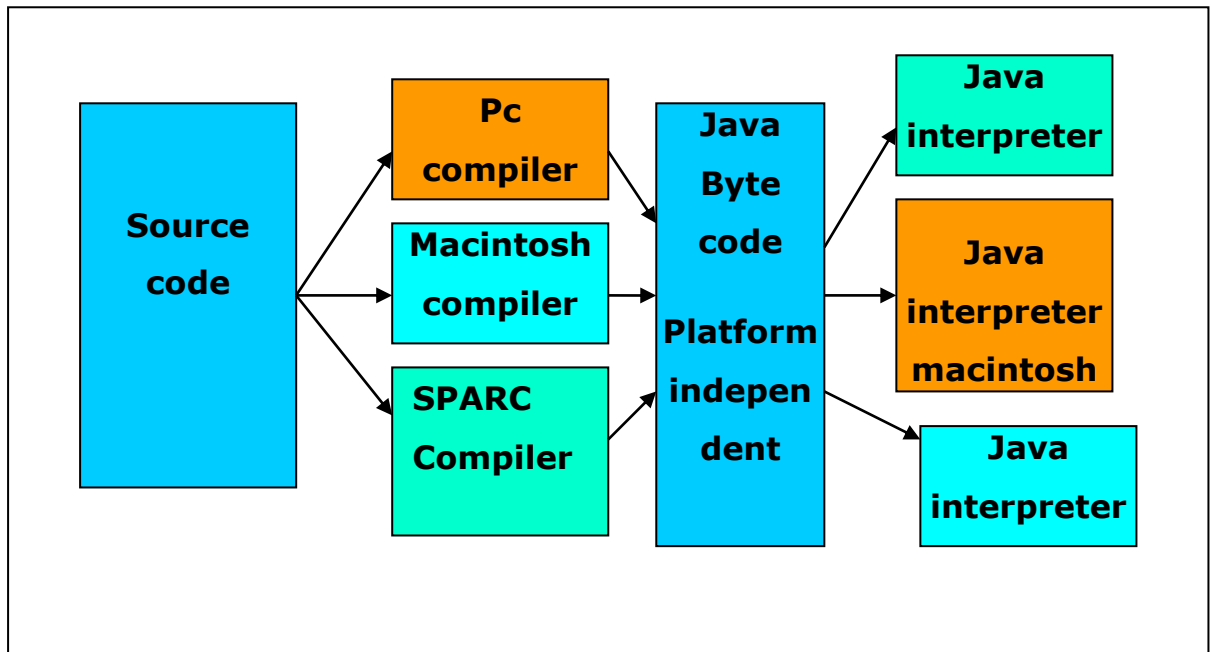


Figure 3. 1 Java Architecture

running on a Java Virtual Machine. In reality this could be an Intel Pentium windows 95 or sun SPARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through internet and run the Applets.

### **Simple:**

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ Programmer. Learning Java will oriented features of C++ . Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

### **Object oriented:**

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank state. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

### **Robust:**

The multi-platform environment of the web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create

robust programs. Was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and runtime.

Java virtually eliminates the problems of memory management and deal location, which is completely automatic. In a well-written Java program, all run-time errors can and should be managed by your program.

## **Servlets/JSP:**

### **Introduction**

A Servlet Is a generic server extension. a Java class that can be loaded dynamically to expand the functionality of a server. Servlets are commonly used with web servers. Where they can take the place CGI scripts.

A servlet is similar to proprietary server extension, except that it runs inside a Java Virtual Machine (JVM) on the server, so it is safe and portable

Servlets operate solely within the domain of the server.

Unlike CGI and Fast CGI, which use multiple processes to handle separate program or separate requests, separate threads within web server process handle all servlets. This means that servlets are all efficient and scalable.

Servlets are portable; both across operating systems and also across web servers. Java Servlets offer the best possible platform for web application development.

Servlets are used as replacement for CGI scripts on a web server, they can extend any sort of server such as a mail server that allows servlets extend its functionality perhaps by performing a virus scan on all attached documents or handling mail filtering tasks.

Servlets provide a Java-based solution used to address the problems currently associated with doing server-side programming including inextensible scripting solutions platform-specific API's and incomplete interface.

Servlets are objects that conform to a specific interface that can be plugged into a Java-based server. Servlets are to the server-side what applets are to the client-side what applets are to the client-side-object byte codes that can be dynamically loaded off the net. They differ from applets in that they are faceless objects(without graphics or a GUI component).They serve as platform independent, dynamically loadable, pluggable helper

byte code objects on the server side that can be used to dynamically extend server-side functionality.

For example an HTTP servlet can be used to generate dynamic HTML content when you use servlets to do dynamic content you get the following advantages:

They're faster and cleaner than CGI scripts

They use a standard API( the servlet API)

They provide all the advantages of Java (run on a variety of servers without needing to be rewritten).

There are many features of servlets that make them easy and attractive to use these include:

- Easily configure using the GUI-based Admin tool]
- Can be Loaded and Invoked from a local disk or remotely across the network.
- Can be linked together or chained, so that one servlet can call another servlet, or several servlets in sequence.
- Can be called dynamically from within HTML, pages using server-side include-tags.
- Are secure-even when downloading across the network, the servlet security model and servlet and box protect your system from unfriendly behavior.,

### **Advantages of the servlet API**

One of the great advantages of the servlet API is protocol independent. It assumes nothing about:

The protocol being used to transmit on the net

How it is loaded

The server environment it will be running in

These quantities are important, because it allows the Servlet API to be embedded in many different kinds of servers. There are other advantages to the servlet API as well These include:

It's extensible-you can inherit all your functionality from the base classes made available to you



It's simple, small, and easy to use.

### **Features of Servlets:**

- Servlets are persistent. Servlets are loaded only by the web server and can maintain services between requests.
- Servlets are fast. Since servlets only need to be loaded once, they offer much better performance over their CGI counterparts.
- Servlets are platform independent.
- Servlets are extensible. Java is a robust, object-oriented programming language, which easily can be extended to suit your needs.
- Servlets are secure.
- Servlets are used with a variety of clients.

Servlets are classes and interfaces from two packages, `java.servlet` and `javax.servlet.http`. The `java.servlet` package contains classes to support generic, protocol-independent servlets. The classes in the `javax.servlet.http` package to add HTTP specific functionality extend these classes.

Unlike a Java program, a servlet does not have a `main()` method. Instead, the server in the process of handling requests invokes certain methods of a servlet. Each time the server dispatches a request to a servlet, it invokes the servlet's `service()` method.

A generic servlet should override its `service()` method to handle requests as appropriate for the servlet. The `service()` accepts two parameters: a request object and a response object. The request object tells the servlet about the request, while the response object is used to return a response.

In contrast, an HTTP servlet usually does not override the `service()` method. Instead, it overrides `doGet()` to handle GET requests and `doPost()` to handle POST requests. An HTTP servlet can override either or both of these methods. The `service()` method of `HttpServlet` handles the setup and dispatching to all the `doXXX()` methods, which is why it usually should not be overridden.

The remainders in the `javax.servlet` and `javax.servlet.http` packages are largely support classes. The `ServletRequest` and `ServletResponse` classes in `javax.servlet` provide access to generic server requests and responses, while `HttpServletRequest` and

HttpServletResponse classes in javax.servlet provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse in javax.servlet.http provide access to HTTP requests and responses. The javax.servlet.http package contains an HttpSession class that provides built-in session tracking functionality and Cookie class that allows quick setup and processing of HttpCookies.

### **Loading Servlets:**

Servlets can be loaded from their places. From a directory that is on the CLASSPATH. The CLASSPATH of the Java Webserver includes service root/classes/, which is where the system classes reside.

From the <SERVICE\_ROOT>/servlets/directory. This is not in the server's class path. A class loader is used to create servlets from this directory. New servlets can be added-existing servlets can be recompiled and the server will notice these changes. From a remote location. For this a code base like <http://nine.eng/classes/foo/> is required in addition to the servlet's class name. Refer to the admin GUI docs on servlet section to see how to set this up.

- Loading Remote Servlets
- Remote servlets can be loaded by:
- Configuring the admin Tool to setup automatic loading of remote servlets.
- Selection up server side include tags in .html files
- Defining a filter chain Configuration

### **Invoking Servlets:**

A servlet invoker is a servlet that invokes the “server” method on a named servlet. If the servlet is not loaded in the server, then the invoker first loads the servlet (either from local disk or from the network) and then invokes the “service” method. Also like applets, local servlets in the server can be identified by just the class name. In other words, if a servlet name is not absolute, it is treated as local.

### **A Client can Invoke Servlets in the Following Ways:**

- The client can ask for a document that is served by the servlet.

- The client(browser) can invoke the servlet directly using a URL, once it has been mapped using the SERVLET ALIASES Section of the admin GUI
- The servlet can be invoked through server side include tags.
- The servlet can be invoked by placing it in the servlets/directory
- The servlet can be invoked by using it in a filter chain

### **The Servlet Life Cycle:-**

The Servlet life cycle is one of the most exciting features of Servlets. This life cycle is a powerful hybrid of the life cycles used in CGI programming and lower-level NSAPI and ISAPI programming.

The servlet life cycle allows servlet engines to address both the performance and resource problems of CGI and the security concepts of low level server API programming.

Servlet life cycle is highly flexible Servers have significant leeway in how they choose to support servlets. The only hard and fast rule is that a servlet engine must conform to the following life cycle contract:

- Create and initialize the servlets
- Handle zero or more service from clients
- Destroy the servlet and then garbage Collects it.

It's perfectly legal for a servlet to be loaded, created and initialized in its own JVM, only to be destroyed and garbage collected without handling any client request or after handling just one request.

The most common and most sensible life cycle implementations for HTTP servlets are:

Single java virtual machine and stateless persistence.

### **Init and Destroy:-**

Just like Applets servlets can define Init() and destroy() methods, A servlets init(ServiceConfig) method is called by the server immediately after the server constructs the servlet's instance. Depending on the server and its configuration, this can be at any of these times.

- When the server starts.

- When the servlet is first requested, just before the service() method is invoked.
- At the request of the server administrator.

In any case, init() is guaranteed to be called before the servlet handles its first request.

The init() method is typically used to perform servlet initialization creating or loading objects that are used by the servlet in handling of its request. In order to providing a new servlet any information about itself and its environment, a server has to call a servlets init() method and pass an object that implement the ServletConfig interface.

This ServletConfig object supplies a servlet with information about its initialization parameters. These parameters are given to the servlets and are not associated with any single request. They can specify initial values, such as where a counter should begin counting, or default values, perhaps a template to use when not specified by the request,

The server calls a servlet's destroy() method when the servlet is about to be unloaded. In the destroy() method, a servlet should free any resources it has acquired that will not be garbage collected. The destroy() method also gives a servlet a chance to write out its unsaved, cached information or any persistent information that should be read during the next call to init().

### **Session Tracking:**

HTTP is a stateless protocol, it provides no way for a server to recognize that a sequence of requests is all from the same client. This causes a problem for application such as shopping cart applications. Even in chat application server can't know exactly who's making a request of several clients.

The solution for this is for client to introduce itself as it makes each request, Each clients needs to provide a unique identifier that lets the server identify it, or it needs to give some information that the server can use to properly handle the request, There are several ways to send this introductory information with each request Such as:

### **USER AUTHORIZATION:**

One way to perform session tracking is to leverage the information that comes with User authorization. When a web server restricts access to some of its resources to only those clients that log in using a recognized username and password. After the client logs in, the username is available to a servlet through getRemoteUser().

When use the username to track the session. Once a user has logged in, the browser remembers her username and resends the name and password as the user views new pages on the site. A servlet can identify the user through her username and they're by Track her session.

The biggest advantage of using user authorization to perform session tracking is that it's easy to implement. Simply tell the protect a set of pages, and use `getRemoteUser()` to identify each client. Another advantage is that the technique works even when the user accesses your site form or exists her browser before coming back.

The biggest disadvantage of user authorization is that it requires each user to register for an account and then log in in each time the starts visiting your site. Most users will tolerate registering and lagging in as a necessary evil when they are accessing sensitive information, but it's all overkill for simple session tracking. Other problem with user authorization is that a user cannot simultaneously maintain more than one session at the same site.

### **Hidden Form Fields:**

One way to support anonymous session tracking is to use hidden from fields. As the name implies, these are fields added to an HTML, form that are not displayed in the client's browser, They are sent back to the server when the form that contains them is submitted.

In a sense, hidden form fields define constant variables for a form. To a servlet receiving a submitted form, there is no difference between a hidden fields and a visible filed.

As more and more information is associated with a client's session . It can become burdensome to pass it all using hidden form fields. In these situations it's possible to pass on just a unique session ID that identifies as particular clients session.

That session ID can be associated with complete information about its session that is stored on the server.

The advantage of hidden form fields is their ubiquity and support for anonymity. Hidden fields are supported in all the popular browsers, they demand on special server requirements, and they can be used with clients that haven't registered or logged in.

The major disadvantage with this technique, however is that works only for a sequence of dynamically generated forms, The technique breaks down immediately with static documents, emailed documents book marked documents and browser shutdowns.

## **URL Rewriting:**

URL rewriting is another way to support anonymous session tracking. With URL rewriting every local URL the user might click on is dynamically modified, or rewritten, to include extra information. The extra information can be in the form of extra path information, added parameters, or some custom, server-specific URL change. Due to the limited space available in rewriting a URL, the extra information is usually limited to a unique session.

Each rewriting technique has its own advantage and disadvantage

Using extra path information works on all servers, and it works as a target for forms that use both the Get and Post methods. It does not work well if the servlet has to use the extra path information as true path information

The advantages and disadvantages of URL rewriting closely match those of hidden form fields. The major difference is that URL rewriting works for all dynamically created documents, such as the Help servlet, not just forms. With the right server support, custom URL rewriting can even work for static documents.

## **Persistent Cookies:**

A fourth technique to perform session tracking involves persistent cookies. A cookie is a bit of information, sent by a web server to a browser that can later be read back from that browser. When a browser receives a cookie, it saves the cookie and thereafter sends the cookie back to the server each time it accesses a page on that server, subject to certain rules. Because a cookie's value can uniquely identify a client, cookies are often used for session tracking.

Persistent cookies offer an elegant, efficient easy way to implement session tracking. Cookies provide as automatic an introduction for each request as we could hope for. For each request, a cookie can automatically provide a client's session ID or perhaps a list of clients performance. The ability to customize cookies gives them extra power and versatility.

The biggest problem with cookies is that browsers don't always accept cookies sometimes this is because the browser doesn't support cookies. More often it's because

The browser doesn't support cookies. More often it's because the user has specifically configured the browser to refuse cookies.

### **The power of serves:**

The power of servlets is nothing but the advantages of servlets over other approaches, which include portability, power, efficiency, endurance, safety elegance, integration, extensibility and flexibility.

### **Portability:**

As servlets are written in java and conform to a well-defined and widely accepted API they are highly portable across operating systems and across server implementation.

We can develop a servlet on a windows NT machine running the java web server and later deploy it effortlessly on a high-end Unix server running apache. With servlets we can really "write once, serve everywhere".

Servlet portability is not the stumbling block it so often is with applets, for two reasons

First, Servlet portability is not mandatory i.e. servlets has to work only on server machines that we are using for development and deployment

Second, servlets avoid the most error-prone and inconstancy implemented portion of the java languages.

### **Power:**

Servlets can harness the full power of the core java. API's: such as Networking and Url access, multithreading, image manipulation, data compression, data base connectivity, internationalization, remote method invocation(RMI) CORBA connectivity, and object serialization, among others.

### **Efficiency And Endurance:**

Servlet invocation is highly efficient, Once a servlet is loaded it generally remains in the server's memory as a single object instance, There after the server invokes the servlet to handle a request using a simple, light weighted method invocation .Unlike the CGI, there's no process to spawn or interpreter to invoke, so the servlet can begin handling the request almost immediately, Multiple, concurrent requests are handled the request almost

immediately. Multiple, concurrent requests are handled by separate threads, so servlets are highly scalable.

Servlets in general are enduring objects. Because a servlet stays in the server's memory as a single object instance, it automatically maintains its state and can hold onto external resources, such as database connections.

**Safety:**

Servlets support safe programming practices on a number of levels.

As they are written in java, servlets inherit the strong type safety of the java language. In addition the servlet API is implemented to be type safe. Java's automatic garbage collection and lack of pointers mean that servlets are generally safe from memory management problems like dangling pointers invalid pointer references and memory leaks.

Servlets can handle errors safely, due to java's exception – handling mechanism. If a servlet divides by zero or performs some illegal operations, it throws an exception that can be safely caught and handled by the server.

A server can further protect itself from servlets through the use of java security manager. A server can execute its servlets under the watch of a strict security manager.

**Elegance:**

The elegance of the servlet code is striking. Servlet code is clean, object oriented modular and amazingly simple one reason for this simplicity is the served API itself. Which includes methods and classes to handle many of the routine chores of servlet development. Even advanced to operations like cookie handling and session tracking are abstracted in convenient classes.

**Integration:**

Servlets are tightly integrated with the server. This integration allows a servlet to cooperate with the server in two ways. for e.g.: a servlet can use the server to translate file paths, perform logging, check authorization, perform MIME type mapping and in some cases even add users to the server's user database.



## **Extensibility and Flexibility:**

The servlet API is designed to be easily extensible. As it stands today the API includes classes that are optimized for HTTP servlets. But later it can be extended and optimized for another type of servlets. It is also possible that its support for HTTP servlets could be further enhanced.

Servlets are also quite flexible, Sun also introduced java server pages. which offer a way to write snippets of servlet code directly with in a static HTML page using syntax similar to Microsoft's Active server pages(ASP)

## **JDBC**

### **What is JDBC?**

any relational database. One can write a single program using the JDBC API, and the JDBC is a Java API for executing SQL ,Statements(As a point of interest JDBC is trademarked name and is not an acronym; nevertheless, Jdbc is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java Programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API

Using JDBC, it is easy to send SQL statements to virtually program will be able to send SQL .statements to the appropriate database. The Combination of Java and JDBC lets a programmer writes it once and run it anywhere.

### **What Does JDBC Do?**

Simply put ,JDBC makes it possible to do three things

- Establish a connection with a database
- Send SQL statements
- Process the results

## **JDBC Driver Types**

The JDBC drivers that we are aware of this time fit into one of four categories

- JDBC-ODBC Bridge plus ODBC driver

- Native-API party-java driver
- JDBC-Net pure java driver
- Native-protocol pure Java driver

An individual database system is accessed via a specific JDBC driver that implements the `java.sql.Driver` interface. Drivers exist for nearly all-popular RDBMS systems, through few are available for free. Sun bundles a free JDBC-ODBC bridge driver with the JDK to allow access to a standard ODBC, data sources, such as a Microsoft Access database, Sun advises against using the bridge driver for anything other than development and very limited development.

JDBC drivers are available for most database platforms, from a number of vendors and in a number of different flavors. There are four driver categories

### **Type 01-JDBC-ODBC Bridge Driver**

Type 01 drivers use a bridge technology to connect a java client to an ODBC database service. Sun's JDBC-ODBC bridge is the most common type 01 driver. These drivers implemented using native code.

### **Type 02-Native-API party-java Driver**

Type 02 drivers wrap a thin layer of java around database-specific native code libraries for MYSQL databases, the native code libraries might be based on the OCI(MYSQL call Interface) libraries, which were originally designed for C/C++ programmers, Because type-02 drivers are implemented using native code. in some cases they have better performance than their all-java counter parts.

### **Type 03-Net-Protocol All-Java Driver**

Type 03 drivers communicate via a generic network protocol to a piece of custom middleware. The middleware component might use any type of driver to provide the actual database access.

### **Type-04-Native-protocol All-java Driver**

Type 04 drivers are the most direct of the lot. Written entirely in java, Type 04 drivers understand database-specific networking. protocols and can access the database directly without any additional software

## **JDBC-ODBC Bridge**

If possible use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge(that is, the Bridge native library, the ODBC driver manager library, library, the ODBC driver library, and the database client library)

## **What is JDBC-ODBE Bridge ?**

The JDBC-ODBC Bridge is a Jdbc driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge is implemented as the sun.jdbc.odbc Java package and contains a native library used to access ODBC. The Bridge is joint development of Intervolve and Java Software

## **MYSQL**

MYSQL is a relational database management system, which organizes data in the form of tables. MYSQL is one of many database servers based on RDBMS model, which manages a seer of data that attends three specific things-data structures, data integrity and data manipulation.

With MYSQL cooperative server technology we can realize the benefits of open, relational systems for all the applications. MYSQL makes efficient use of all systems resources, on all hardware architecture; to deliver unmatched performance, price performance and scalability. Any DBMS to be called as RDBMS has to satisfy Dr.E.F.Codd's rules.

## **Features of MYSQL:**

### **Portable**

The MYSQL RDBMS is available on wide range of platforms ranging from PCs to super computers and as a multi user loadable module for Novel NetWare,.

### **Compatible**

MYSQL commands can be used for communicating with IBM DB2 mainframe RDBMS that is different from MYSQL, which is MYSQL compatible with DB2. MYSQL

RDBMS is a high performance fault tolerant DBMS, which is specially designed for online transaction processing and for handling large database applications.

### **Multithreaded Server Architecture**

MYSQL adaptable multithreaded server architecture delivers scalable high performance for very large number of users on all hardware architecture including symmetric multiprocessors (sumps) and loosely coupled multiprocessors. Performance is achieved by eliminating CPU, I/O, memory and operating system bottlenecks and by optimizing the MYSQL DBMS server code to eliminate all internal bottlenecks.

- MYSQL has become the most popular RDBMS in the market because of its ease of use
- Client/server architecture.
- Data independence.
- Ensuring data integrity and data security.
- Managing data concurrency.
- Parallel processing support for speed up data entry and online transaction processing used for applications.
- DB procedures, functions and packages.

### **Dr.E.F.Codd's Rules:**

These rules are used for valuating a product to be called as relational database management systems. Out of 12 rules, a RDBMS product should satisfy at least 8 rules + rule called rule 0 that must be satisfied.

#### **RULE 0: Foundation Rule**

For any system to be advertised as, or claimed to be relational DBMS should manage database with in itself, without using an external language.

#### **RULE 1: Information Rule**

All information in relational database is represented at logical level in only one way as values in tables.

#### **RULE 2: Guaranteed Access**

Each and every data in a relational database is guaranteed to be logically accessibility by using to a combination of table name, primary key value and column name.

### **RULE 3: Systematic Treatment of Null Values**

Null values are supported for representing missing information and inapplicable information. They must be handled in systematic way, independent of data types.

### **RULE 4: Dynamic Online Catalog based Relation Model**

The database description is represented at the logical level in the same way as ordinary data so that authorized users can apply the same relational language to its interrogation as they do to the regular data.

### **RULE 5: Comprehensive Data Sub Language**

A relational system may support several languages and various models of terminal use. However there must be one language whose statement can express all of the following:

Data Definitions, View Definitions, Data Manipulations, Integrity, Constraints, Authorization and transaction boundaries.

### **RULE 6: View Updating**

Any view that is theoretical can be updatable if changes can be made to the tables that effect the desired changes in the view.

### **RULE 7: High level Update, Insert and Delete**

The capability of handling a base relational or derived relational as a single operand applies not only retrieval of data also to its insertion, updating, and deletion.

### **RULE 8: Physical Data Independence**

Application program and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access method.

### **RULE 9: Logical Data Independence**

Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access methods.

### **RULE 10: Integrity Independence**

Integrity constraints specific to particular database must be definable in the relational data stored in the catalog, not in application program.

### **RULE 11: Distributed Independence**

Whether or not a system supports database distribution, it must have a data sub-language that can support distributed databases without changing the application program.

### **RULE 12: Non Sub-Version**

If a relational system has low level language, that low language cannot use to subversion or by pass the integrity rules and constraints expressed in the higher level relational language.

### **MYSQL supports the following Codd's Rules**

- Rule 1: Information Rule (Representation of information)-YES.
- Rule 2: Guaranteed Access-YES.
- Rule 3: Systematic treatment of Null values-YES.
- Rule 4: Dynamic on-line catalog-based Relational Model-YES.
- Rule 5: Comprehensive data sub language-YES.
- Rule 6: View Updating-PARTIAL.
- Rule 7: High-level Update, Insert and Delete-YES.
- Rule 8: Physical data Independence-PARTIAL.
- Rule 9: Logical data Independence-PARTIAL.
- Rule 10: Integrity Independence-PARTIAL.
- Rule 11: Distributed Independence-YES.
- Rule 12: Non-subversion-YES.

### **HTML**

Hypertext Markup Language(HTML), the languages of the world wide web(WWW), allows users to produces web pages that included text, graphics and pointer to other web pages (Hyperlinks).

HTML is not a programming language but it is an application of ISO Standard 8879,SGML(Standard Generalized Markup Language),but

Specialized to hypertext and adapted to the Web. The idea behind Hypertext one point to another point. We can navigate through the information based on our interest and preference. A markup language is simply a series of items enclosed within the elements should be displayed.

Hyperlinks are underlined or emphasized words that lead to other documents or some portions of the same document.

HTML can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop. HTML provides tags(special codes) to make the document look attractive. HTML provides are not case-sensitive. Using graphics, fonts, different sizes, color, etc., can enhance the presentation of the document. Anything

### **Basic Html Tags:**

<code>&lt;!-- --&gt;</code>	Specific Comments.
<code>&lt;A&gt;.....&lt;/A&gt;</code>	Creates Hypertext links.
<code>&lt;B&gt;.....&lt;/B&gt;</code>	Creates hypertext links.
<code>&lt;Big&gt;.....&lt;/Big&gt;</code>	Formats text in large-font
<code>&lt;Body&gt;.....&lt;/Body&gt;</code>	contains all tags and text in the Html-document
<code>&lt;Center&gt;.....&lt;/Center&gt;</code>	Creates Text
<code>&lt;DD&gt;.....&lt;/DD&gt;</code>	Definition of a term.
<code>&lt;TABLE&gt;.....&lt;/TABLE&gt;</code>	creates table
<code>&lt;Td&gt;.....&lt;/Td&gt;</code>	indicates table data in a table.
<code>&lt;Tr&gt;.....&lt;/Tr&gt;</code>	designates a table row
<code>&lt;Th&gt;.....&lt;/Th&gt;</code>	creates a heading in a table.

### **ADVANTAGES:**

A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.

- HTML is platform independent
- HTML tags are not case-sensitive.

## **JAVA SCRIPT**

JavaScript is a compact, object-based scripting language for developing client and server internet applications. Netscape Navigator 2.0 interprets JavaScript statements embedded directly in an HTML page. And Livewire enables you to create server-based applications similar to common gateway interface (cgi) programs.

In a client application for Navigator, JavaScript statements embedded in an HTML Page can recognize and respond to user events such as mouse clicks form Input, and page navigation.

For example, you can write a JavaScript function to verify that users enter valid information into a form requesting a telephone number or zip code . Without any network transmission, an Html page with embedded Java Script can interpret the entered text and alert the user with a message dialog if the input is invalid or you can use JavaScript to perform an action (such as play an audio file, execute an applet, or communicate with a plug-in) in response to the user opening or exiting a page.

## **3.2 SOFTWARE REQUIREMENTS SPECIFICATION:**

### **3.2.1 USER REQUIREMENTS:**

User Requirements of the Project are as follows:

- Balance enquiry
- Funds Transfer to another account in the same bank
- Request for Cheque book/change of address/stop payment of
- Viewing Monthly and annual statements.
- System help.

As the application of project is regarding internet banking ,we used “JAVA” a simple, object-oriented, network – savvy , interpreted, robust, secure, architecture neutral, portable, high performance, multithreaded dynamic language.



## 1. Admin Module:

This module is belongs to bank staff. By using this module Administrator can add type of accounts (saving and current etc.), minimum deposit amount in that particular account, interest rate and period of time.

## 2. User Module:

This module is all about customers of a bank. By using this module customers can go for different operations like they can apply for a new account in a bank, they can apply for different loans based on their requirement and also they can view their personal information, modify both personal and login information etc. Following operations can be done by any customer.

- **New Account:** Customer can open a new account.
- **Loan Services:** Customer can apply for a new Loan.
- **Transaction Details:** Customer can transfer his funds to other accounts and also they can pay tax to govt.
- **Funds Transfer:** By using this functionality customer can send money from one account to another account.
- **Requests:** By using this functionality customer can go for different type of transactions on drafts and Cheque(s).

## 3. Reports Module:

In this module administrator will get different types of reports regarding customers like Number of customers of this portal etc. And this module is controlled by administrator only.

- **Accounts:** By using this functionality customers can view individual account holder details of this portal.
- **Customers:** By using this functionality customers can view other existing customer details that are having accounts in the same bank.
- **Funds Transfer:** By using this functionality customers can view individual reports of funds transfer.

## 4. One Way Encryption

In computer science, a **one-way function** is a function that is easy to compute on every input, but hard to invert given the image of a random input. Here, "easy" and "hard" are to be understood in the sense of computational complexity theory, specifically the theory of polynomial time problems. Not being one-to-one is not considered sufficient of a function for it to be called one-way.

### 3.2.2 Software Requirements:

- OS : Windows XP/7/8.1/10 OR Linux
- Web Technologies : JSP, Java Beans, HTML, CSS, JS, Bootstrap Framework.
- Server : Tomcat 8.0
- Browser : Internet Edge/Google Chrome/Mozilla
- Database : MYSQL.

### 3.2.3 Hardware Requirement:

- Processor : Intel Core-i3 and above.
- Hard Disk : 80GB and above.
- RAM : 4GB and above.

## 3.3 CONTENT DIAGRAM OF PROJECT:

### 3.3.1 System Architecture.

Below architecture diagram represents mainly flow of requests from users to database through servers. In this scenario overall system is designed in three tiers separately using three layers called presentation layer, business logic layer and data link layer. This project was developed using 3-tier architecture.

URL pattern represents how the requests are flowing through one layer to another layer and how the responses are getting by other layers to presentation layer through server in architecture diagram.

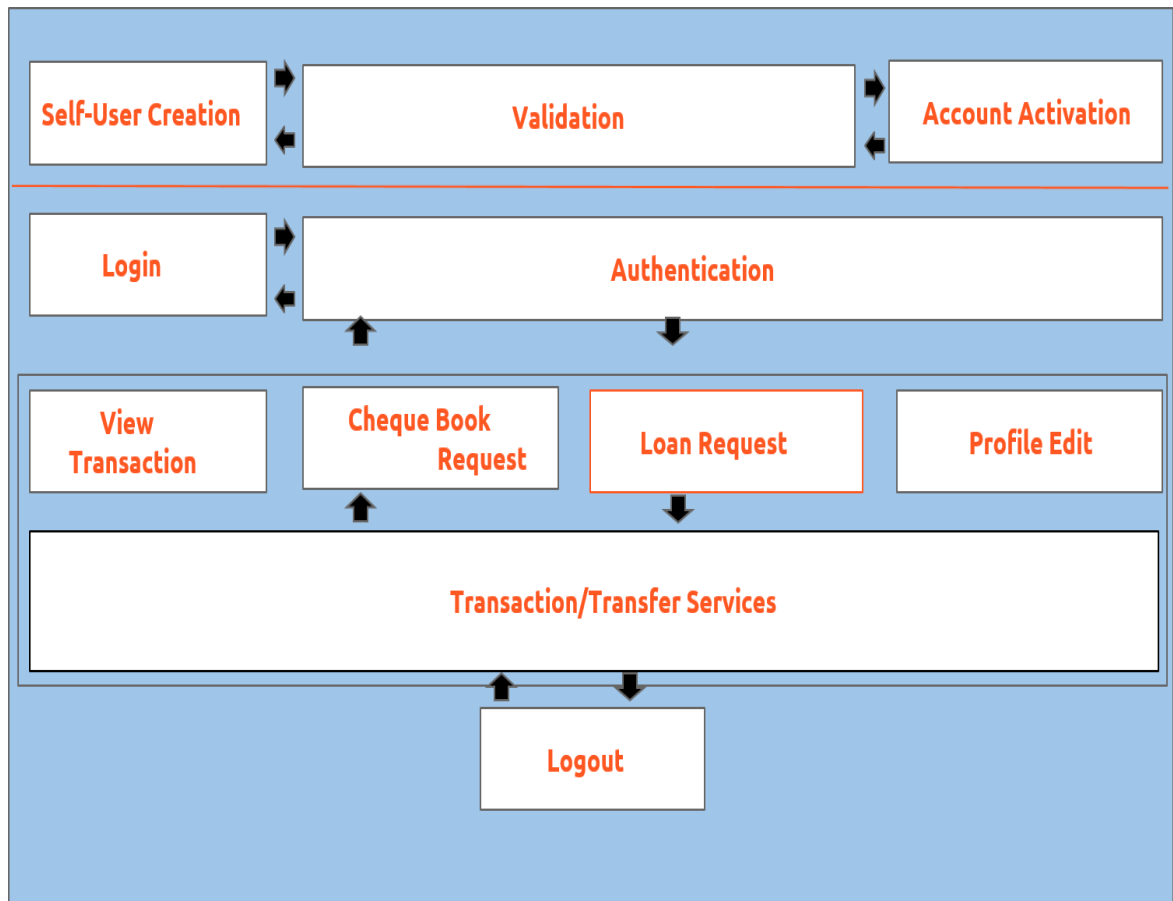


Figure 3. 2 System Architecture

### 3.4 CONCLUSION:

Technical feasibility understands the different technologies involved in the proposed system. Before commencing to the project be very clear about what are the technologies that are to be required for the development of the new system. As the technical requirements of proposed system have been identified, the system is technically feasible. This application was developed by MYSQL and ECLIPSE.

# DESIGN

## 4. DESIGN

### 4.1. INTRODUCTION:

#### PROCESS MODEL USED WITH JUSTIFICATION

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

#### Stages in SDLC:

- Requirement Gathering
- Analysis
- Designing
- Coding
- Testing
- Maintenance

#### Requirements Gathering stage:

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and Textual description.

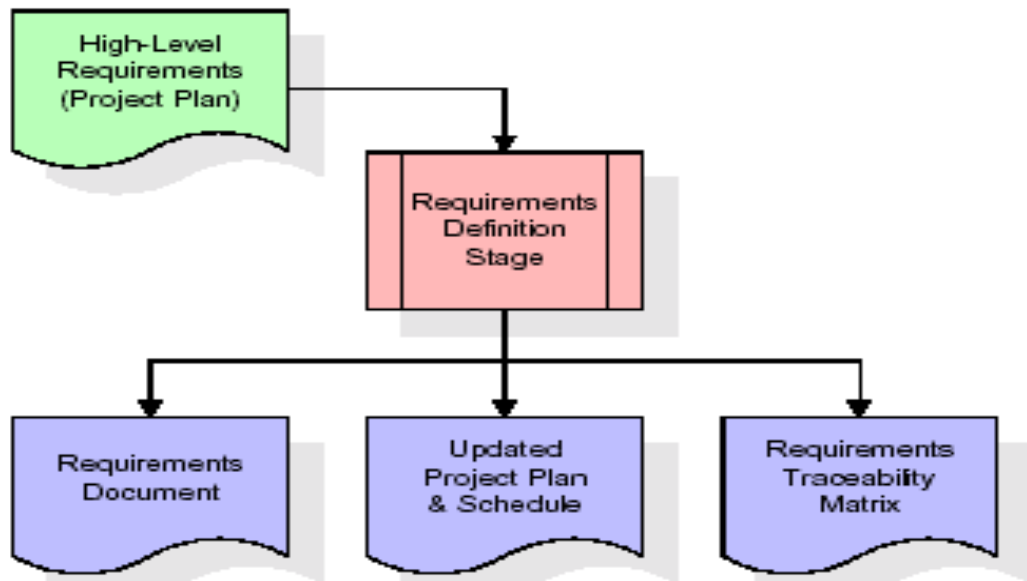


Figure 4. 1 Requirements Gathering Stage

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term **requirements traceability**.

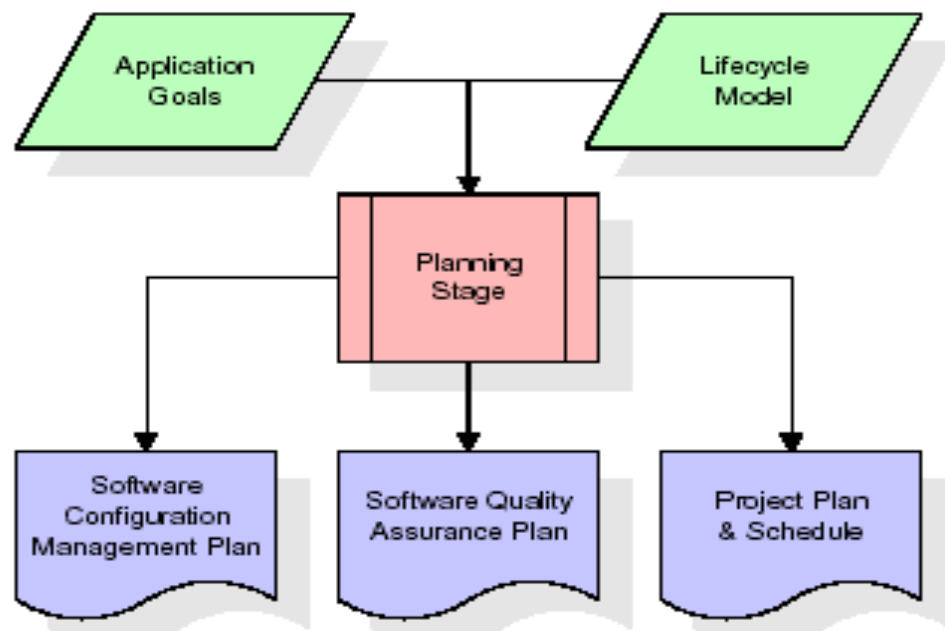
The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

- Feasibility study is all about identification of problems in a project.

- No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
- Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

### Analysis Stage:

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



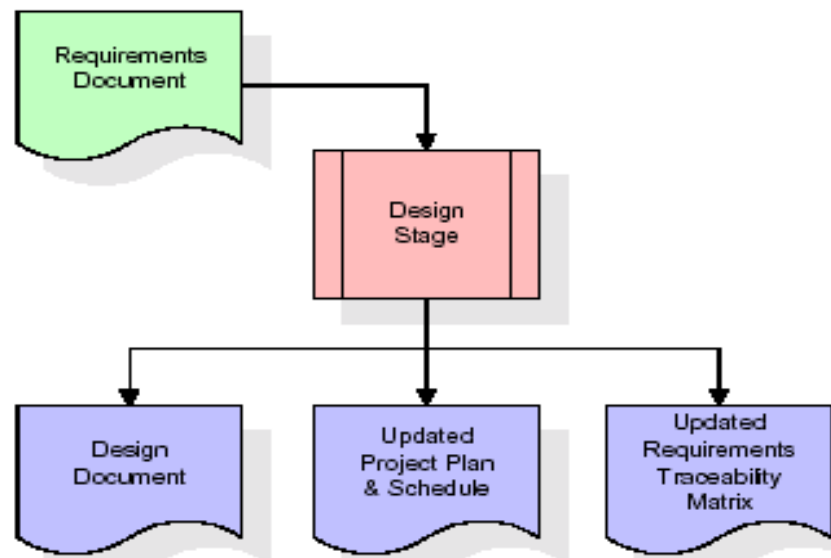
**Figure 4. 2 Analysis Stage**

The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of

scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

### **Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.



**Figure 4. 3 Designing Stage**

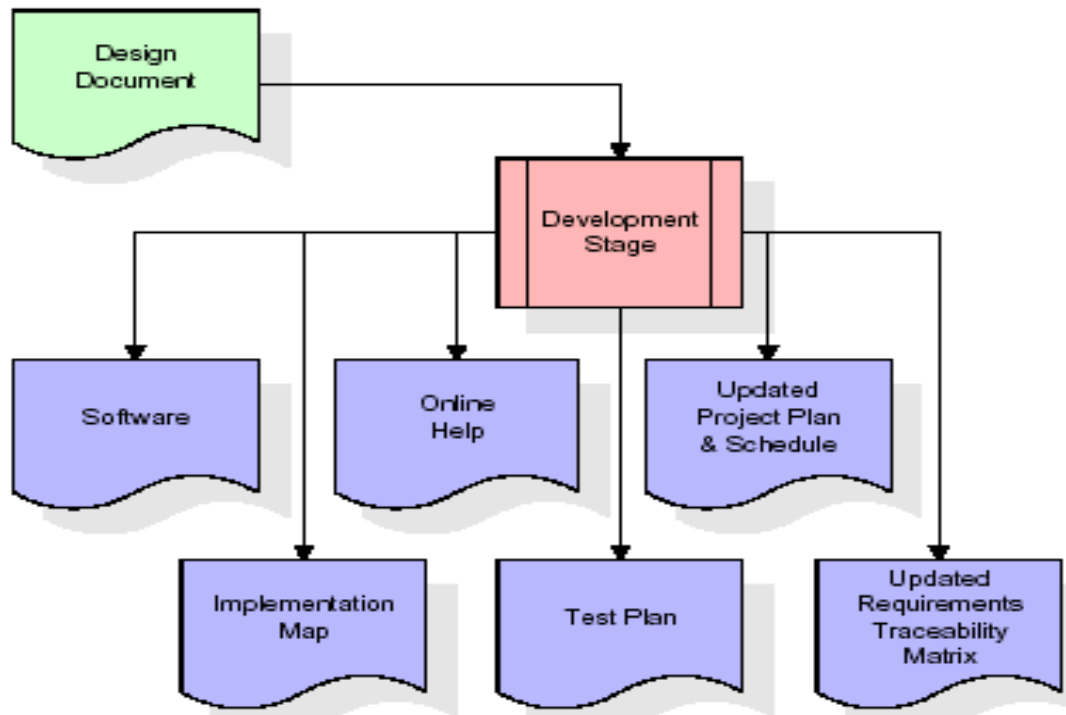
When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

### **Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs,



data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their Sequences with the software.



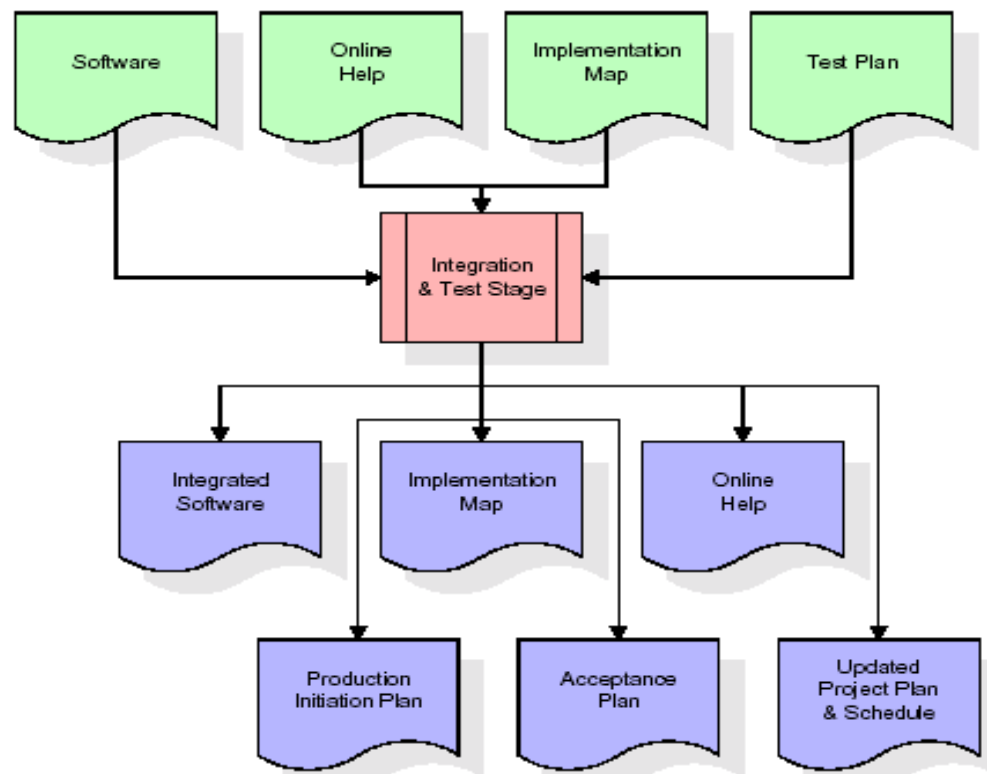
**Figure 4. 4 Development Stage**

The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

#### **Integration & Test Stage:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point,

all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user .



**Figure 4. 5 Integration & Testing Stage**

The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

### **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.

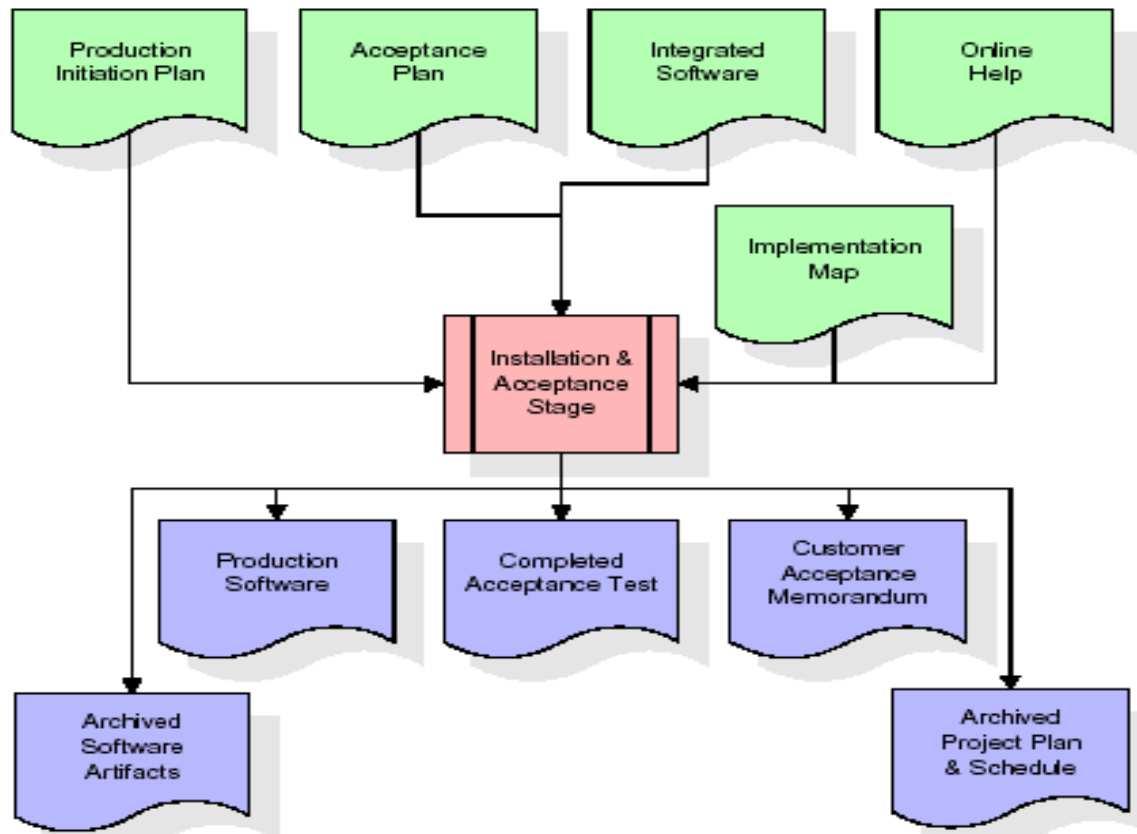


Figure 4. 6 Installation & Acceptance Stage

The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

### Maintenance:

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category.

For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

## **4.2 UML DIAGRAMS:**

### **Unified Modeling Language:**

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

#### **User Model View**

This view represents the system from the users perspective.

The analysis representation describes a usage scenario from the end-users perspective.

#### **Structural model view**

In this model the data and functionality are arrived from inside the system.

This model view models the static structures.

#### **Behavioral Model View**

It represents the dynamic of behavioral as parts of the system, depicting the Sequences of collection between various structural elements described in the user model and structural model view.

#### **Implementation Model View**

In this the structural and behavioral as parts of the system are represented as they are to be built.

#### **Environmental Model View**

In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

UML is specifically constructed through two different domains they are:

- UML Analysis modeling, this focuses on the user model and structural model views of the system.
- UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

Use case Diagrams represent the functionality of the system from a user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view. Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer ...etc., or another system like central database.

### **Use Case Diagram:**

To model a system the most important aspect is to capture the dynamic behaviour. To clarify a bit in details, *dynamic behaviour* means the behaviour of the system when it is running /operating.

So only static behaviour is not sufficient to model a system rather dynamic behaviour is more important than static behaviour. In UML there are five diagrams available to model dynamic nature and use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature there should be some internal or external factors for making the Sequence.

These internal and external agents are known as actors. So use case diagrams are consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system.

So to model the entire system numbers of use case diagrams are used.

### **Purpose**

The purpose of use case diagram is to capture the dynamic aspect of a system. But this definition is too generic to describe the purpose.

Because other four diagrams (activity, sequence, collaboration and Statechart) are also having the same purpose. So we will look into some specific purpose which will distinguish it from other four diagrams.

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

Now when the initial task is complete use case diagrams are modelled to present the outside view.

So in brief, the purposes of use case diagrams can be as follows:

- Used to gather requirements of a system.
- Used to get an outside view of a system.

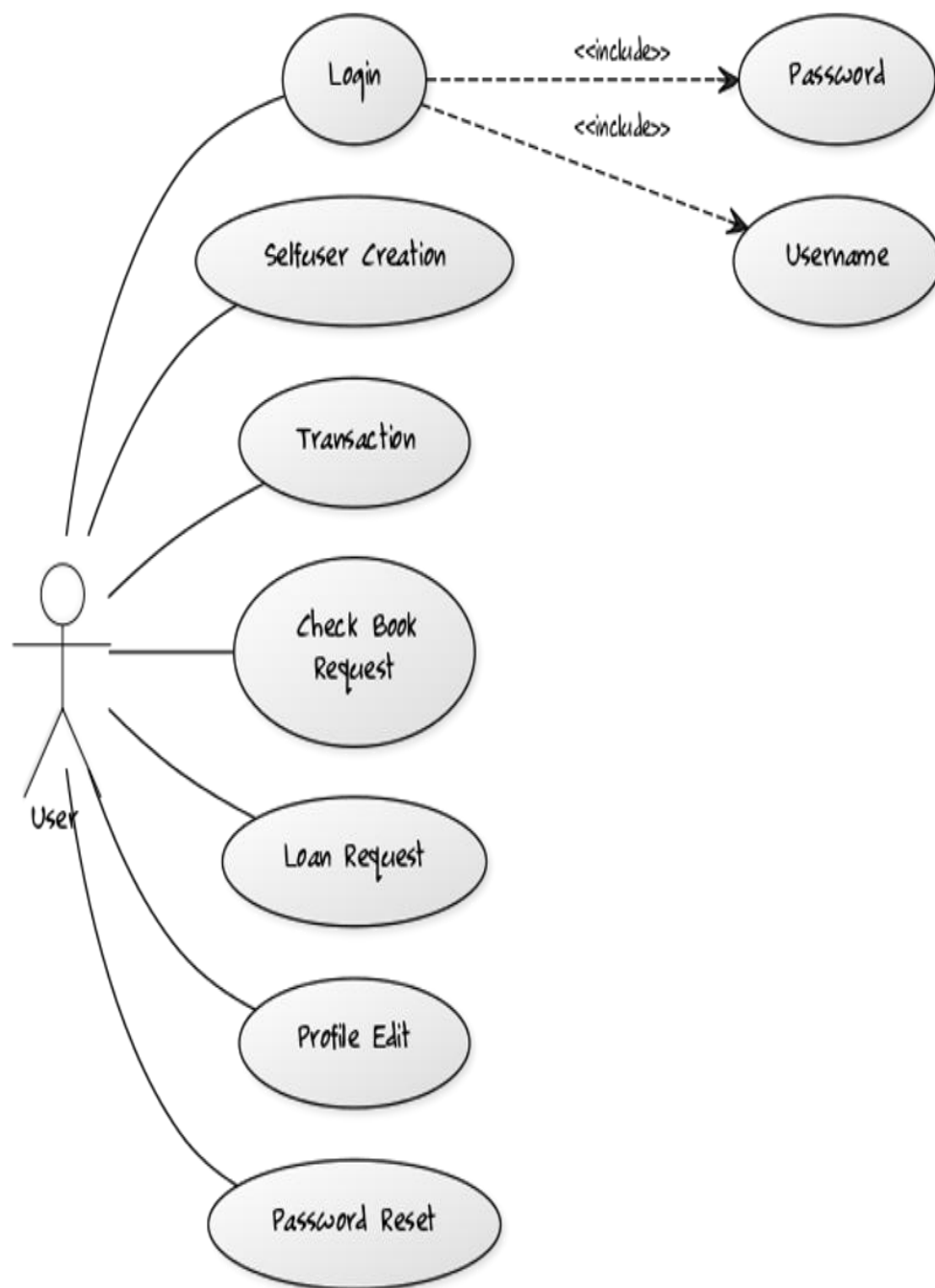


Figure 4. 7 Use Case

**Class Diagram:**

The class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application.

The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams which can be mapped directly with object oriented languages.

The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as a *structural diagram*.

**Purpose:**

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

The UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application but class diagram is a bit different. So it is the most popular UML diagram in the coder community.

So the purpose of the class diagram can be summarized as:

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

**How to draw Class Diagram?**

Class diagrams are the most popular UML diagrams used for construction of software applications. So it is very important to learn the drawing procedure of class diagram.



Class diagrams have lot of properties to consider while drawing but here the diagram will be considered from a top level view.

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. So a collection of class diagrams represent the whole system.

The following points should be remembered while drawing a class diagram:

- The name of the class diagram should be meaningful to describe the aspect of the system.
- Each element and their relationships should be identified in advance.
- Responsibility (attributes and methods) of each class should be clearly identified.
- For each class minimum number of properties should be specified. Because unnecessary properties will make the diagram complicated.
- Use notes whenever required to describe some aspect of the diagram. Because at the end of the drawing it should be understandable to the developer/coder.

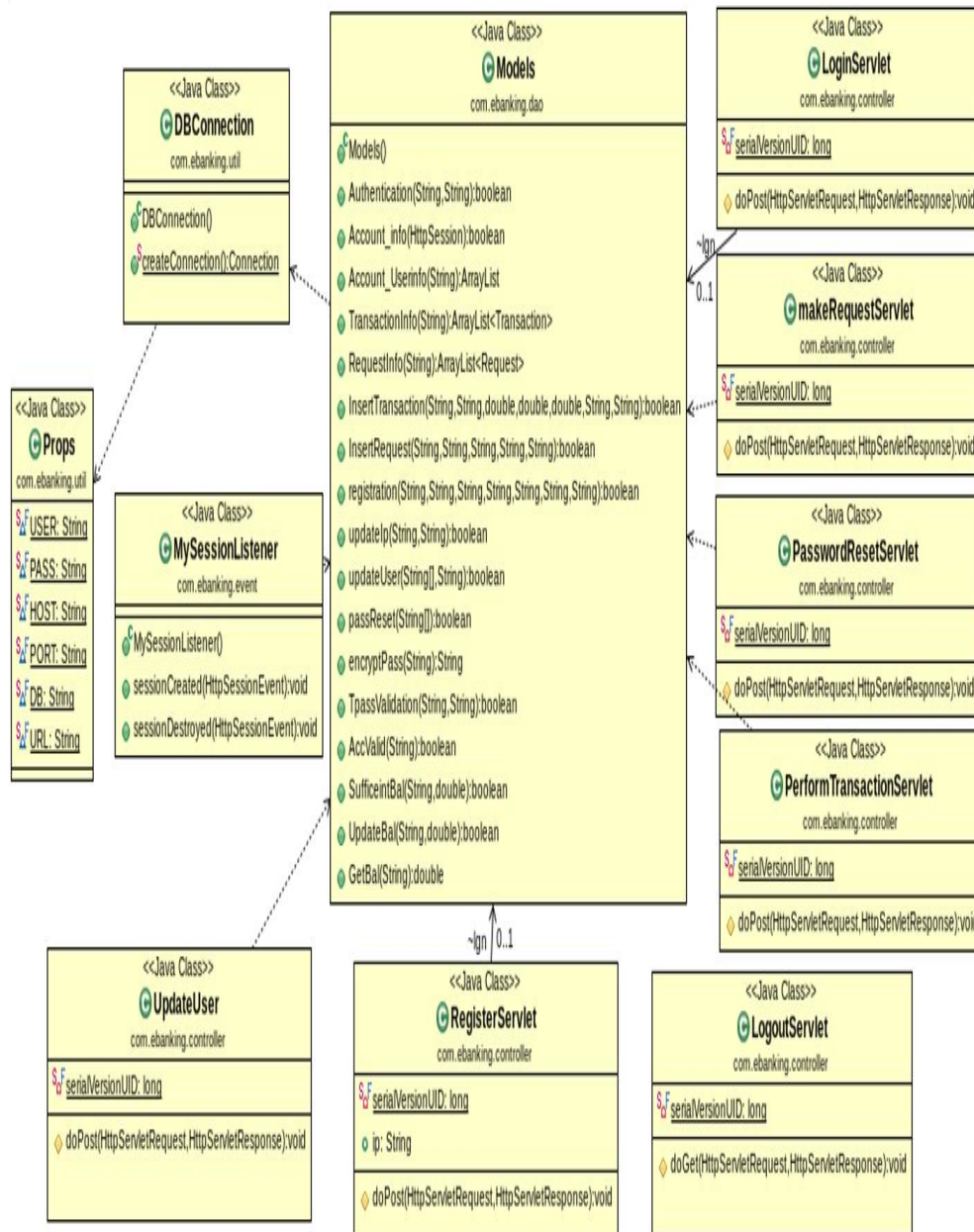


Figure 4. 8 Class Diagram

**Activity Diagram:**

Activity diagram is another important diagram in UML to describe dynamic aspects of the system.

Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deal with all types of flow control by using different elements like fork, join etc.

**Purpose:**

The basic purposes of activity diagrams are similar to other four diagrams. It captures the dynamic behaviour of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

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

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flow chart. Although the diagram looks like a flow chart but it is not. It shows different flows like parallel, branched, concurrent and single.

So the purposes can be described as:

Draw the activity flow of a system.

Describe the sequence from one activity to another.

Describe the parallel, branched and concurrent flow of the system.

**How to draw Activity Diagram?**

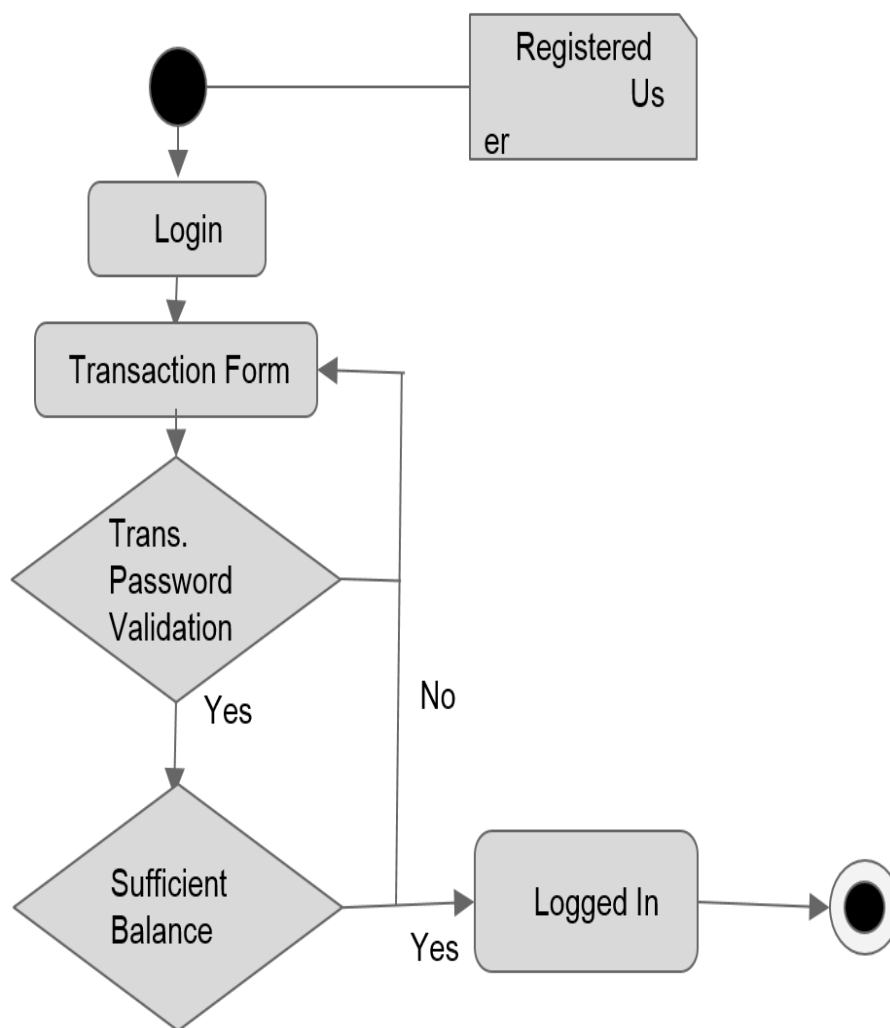
Activity diagrams are mainly used as a flow chart consists of activities performed by the system. But activity diagram are not exactly a flow chart as they have some additional capabilities. These additional capabilities include branching, parallel flow, swim lane etc.

Before drawing an activity diagram we must have a clear understanding about the elements used in activity diagram. The main element of an activity diagram is the activity itself. An activity is a function performed by the system. After identifying the activities we need to understand how they are associated with constraints and conditions.

So before drawing an activity diagram we should identify the following elements:

- Activities
- Association
- Conditions
- Constraints

Once the above mentioned parameters are identified we need to make a mental layout of the entire flow. This mental layout is then transformed into an activity diagram.



**Figure 4. 9 Activity Diagram for User Transaction**

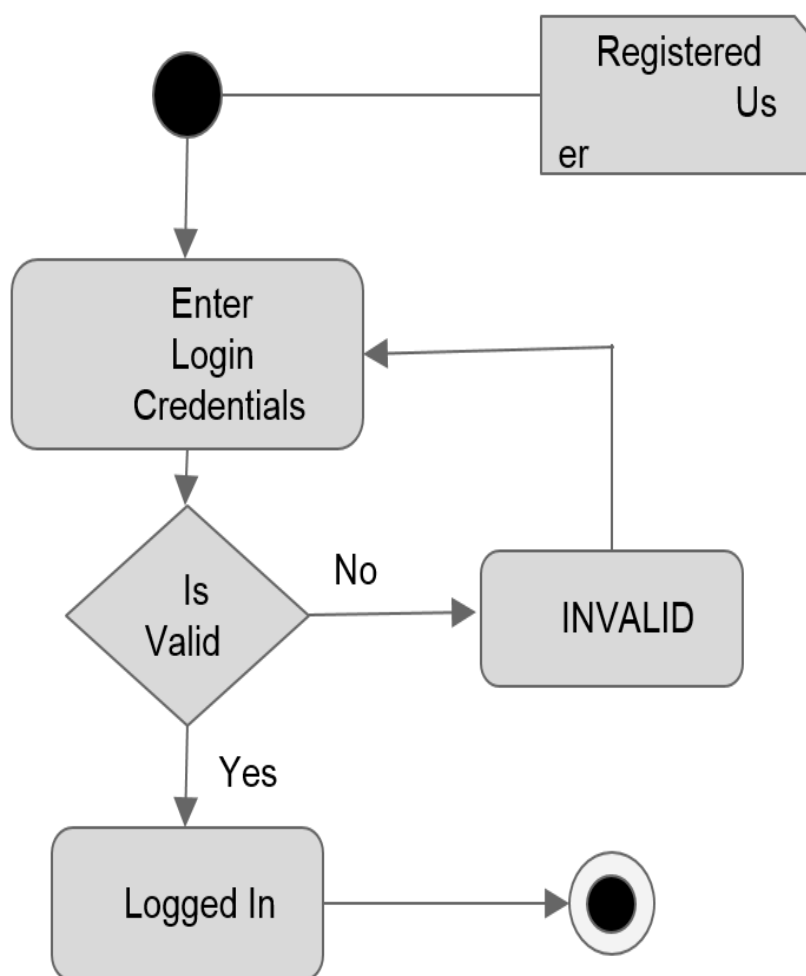


Figure 4. 10 Activity Diagram for Login

**Sequence Diagram:**

From the name *Sequence* it is clear that the diagram is used to describe some type of Sequences among the different elements in the model. So this Sequence is a part of dynamic behaviour of the system.

Sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive messages.

**Purpose:**

The purposes of Sequence diagrams are to visualize the interactive behaviour of the system. Now visualizing Sequence is a difficult task. So the solution is to use different types of models to capture the different aspects of the Sequence.

That is why sequence and collaboration diagrams are used to capture dynamic nature but from a different angle.

So the purposes of Sequence diagram can be describes as:

- To capture dynamic behaviour of a system.
- To describe the message flow in the system.
- To describe structural organization of the objects.
- To describe Sequence among objects.

**How to draw Sequence Diagram?**

As we have already discussed that the purpose of Sequence diagrams are to capture the dynamic aspect of a system. So to capture the dynamic aspect we need to understand what a dynamic aspect is and how it is visualized. Dynamic aspect can be defined as the snap shot of the running system at a particular moment.

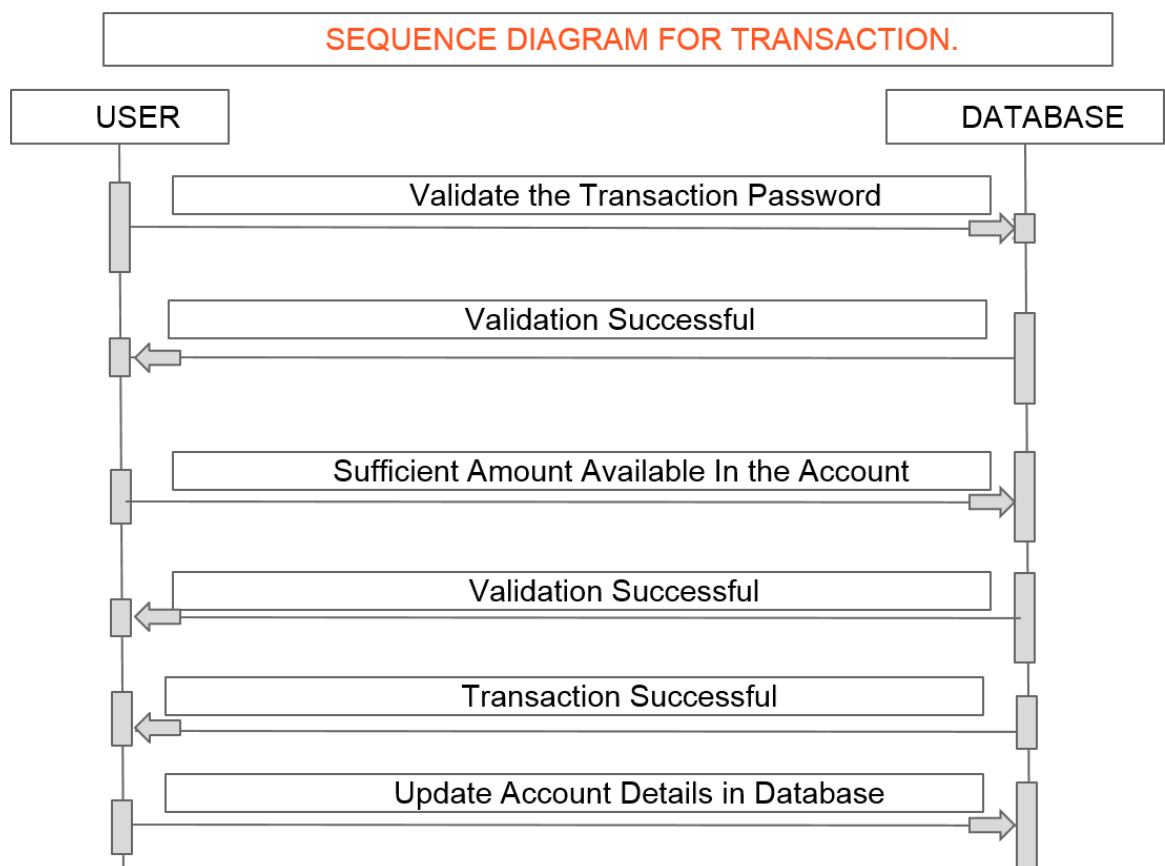
We have two types of Sequence diagrams in UML. One is sequence diagram and the other is a collaboration diagram. The sequence diagram captures the time sequence of message flow from one object to another and the collaboration diagram describes the organization of objects in a system taking part in the message flow.

So the following things are to identified clearly before drawing the Sequence diagram:

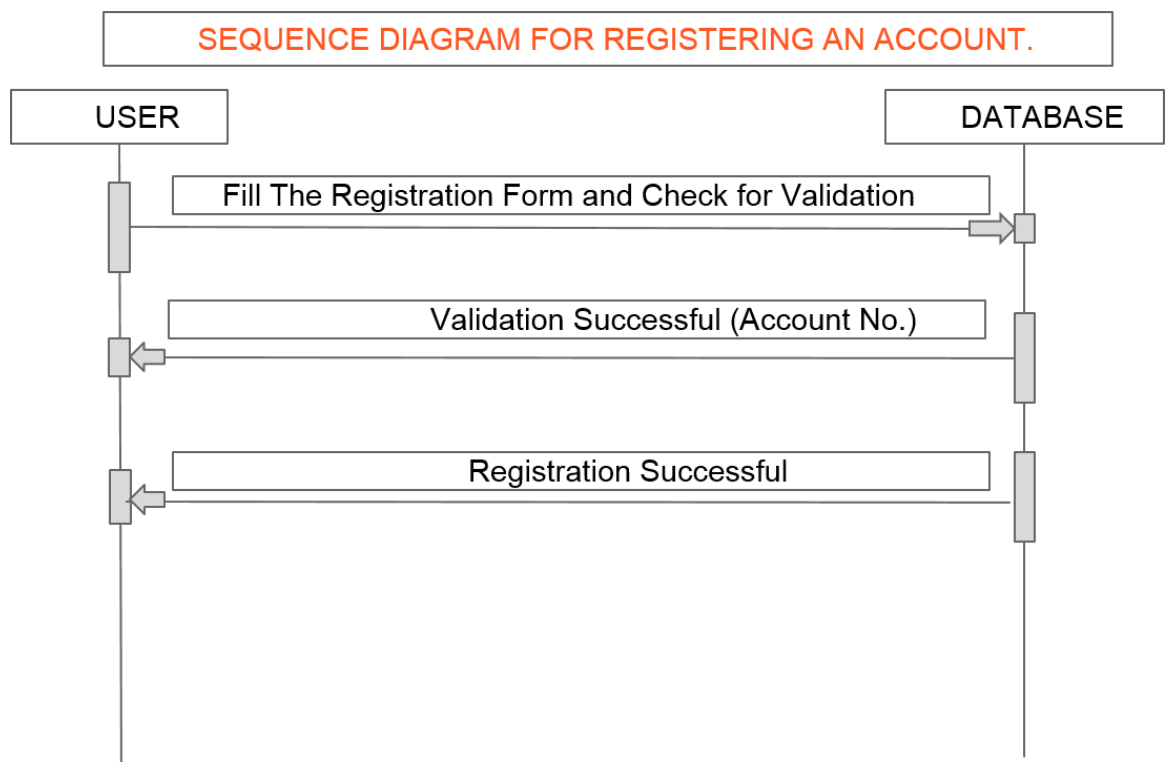
- Objects taking part in the Sequence.
- Message flows among the objects.
- The sequence in which the messages are flowing.
- Object organization.

Following are two Sequence diagrams modeling order management system. The first diagram is a sequence diagram and the second is a collaboration diagram.





**Figure 4. 11 Sequence Diagram for Transaction**



**Figure 4. 12 Sequence Diagram for Registration**

**E-R Diagram:**

The ER model defines the conceptual view of a database. It works around real-world entities and the associations among them. At view level, the ER model is considered a good option for designing databases.

**Entity**

An entity can be a real-world object, either animate or inanimate, that can be easily identifiable. For example, in a school database, students, teachers, classes, and courses offered can be considered as entities. All these entities have some attributes or properties that give them their identity.

An entity set is a collection of similar types of entities. An entity set may contain entities with attribute sharing similar values. For example, a Students set may contain all the students of a school; likewise a Teachers set may contain all the teachers of a school from all faculties. Entity sets need not be disjoint.

**Attributes**

Entities are represented by means of their properties, called **attributes**. All attributes have values. For example, a student entity may have name, class, and age as attributes.

There exists a domain or range of values that can be assigned to attributes. For example, a student's name cannot be a numeric value. It has to be alphabetic. A student's age cannot be negative, etc.

**Types of Attributes**

**Simple attribute** – Simple attributes are atomic values, which cannot be divided further. For example, a student's phone number is an atomic value of 10 digits.

**Composite attribute** – Composite attributes are made of more than one simple attribute. For example, a student's complete name may have first\_name and last\_name.

**Derived attribute** – Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the

database. For example, average\_salary in a department should not be saved directly in the database, instead it can be derived. For another example, age can be derived from data\_of\_birth.

**Single-value attribute** – Single-value attributes contain single value. For example – Social\_Security\_Number.

**Multi-value attribute** – Multi-value attributes may contain more than one values. For example, a person can have more than one phone number, email\_address, etc.

These attribute types can come together in a way like –

simple single-valued attributes

simple multi-valued attributes

composite single-valued attributes

composite multi-valued attributes

### **Entity-Set and Keys**

Key is an attribute or collection of attributes that uniquely identifies an entity among entity set.

For example, the roll\_number of a student makes him/her identifiable among students.

**Super Key** – A set of attributes (one or more) that collectively identifies an entity in an entity set.

**Candidate Key** – A minimal super key is called a candidate key. An entity set may have more than one candidate key.

**Primary Key** – A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

### **Relationship**

The association among entities is called a relationship. For example, an employee **works at** a department, a student **enrolls** in a course. Here, Works at and Enrolls are called relationships.

### Relationship Set

A set of relationships of similar type is called a relationship set. Like entities, a relationship too can have attributes. These attributes are called **descriptive attributes**.

### Degree of Relationship

The number of participating entities in a relationship defines the degree of the relationship.

Binary = degree 2

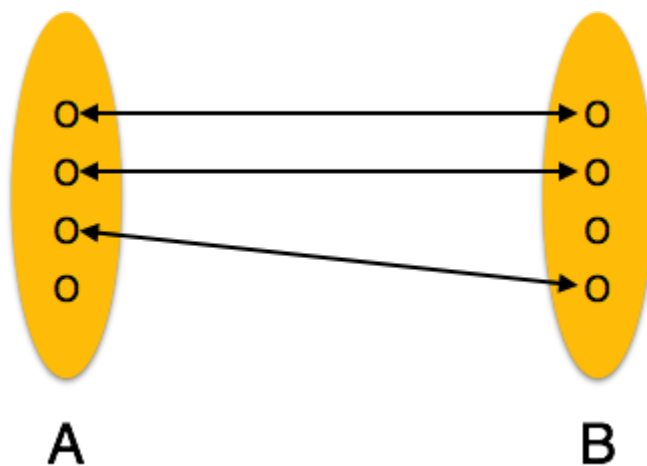
Ternary = degree 3

n-ary = degree

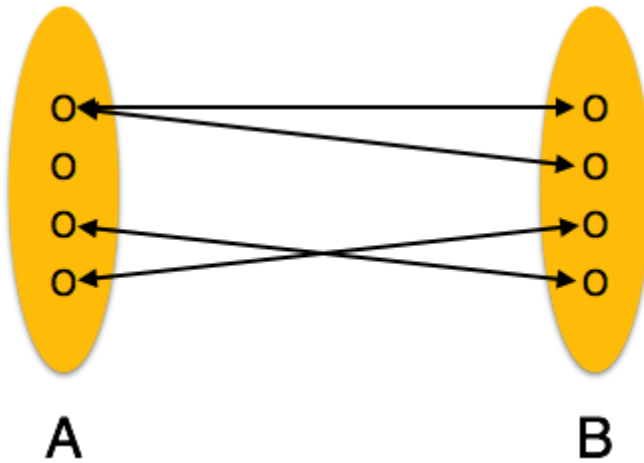
### Mapping Cardinalities

**Cardinality** defines the number of entities in one entity set, which can be associated with the number of entities of other set via relationship set.

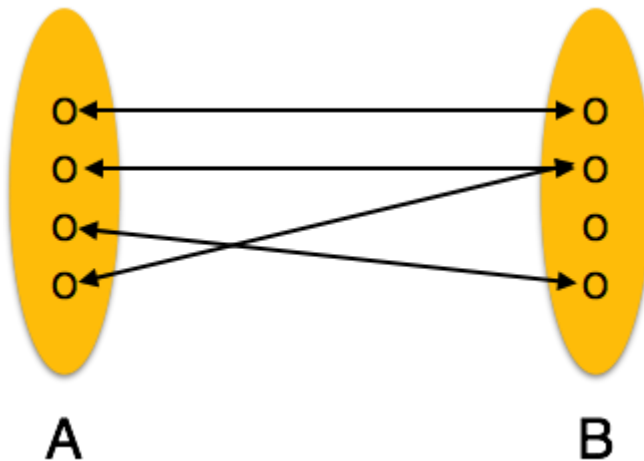
**One-to-one** – One entity from entity set A can be associated with at most one entity of entity set B and vice versa.



**One-to-many** – One entity from entity set A can be associated with more than one entities of entity set B however an entity from entity set B, can be associated with at most one entity.



**Many-to-one** – More than one entities from entity set A can be associated with at most one entity of entity set B, however an entity from entity set B can be associated with more than one entity from entity set A.



**Many-to-many** – One entity from A can be associated with more than one entity from B and vice versa.

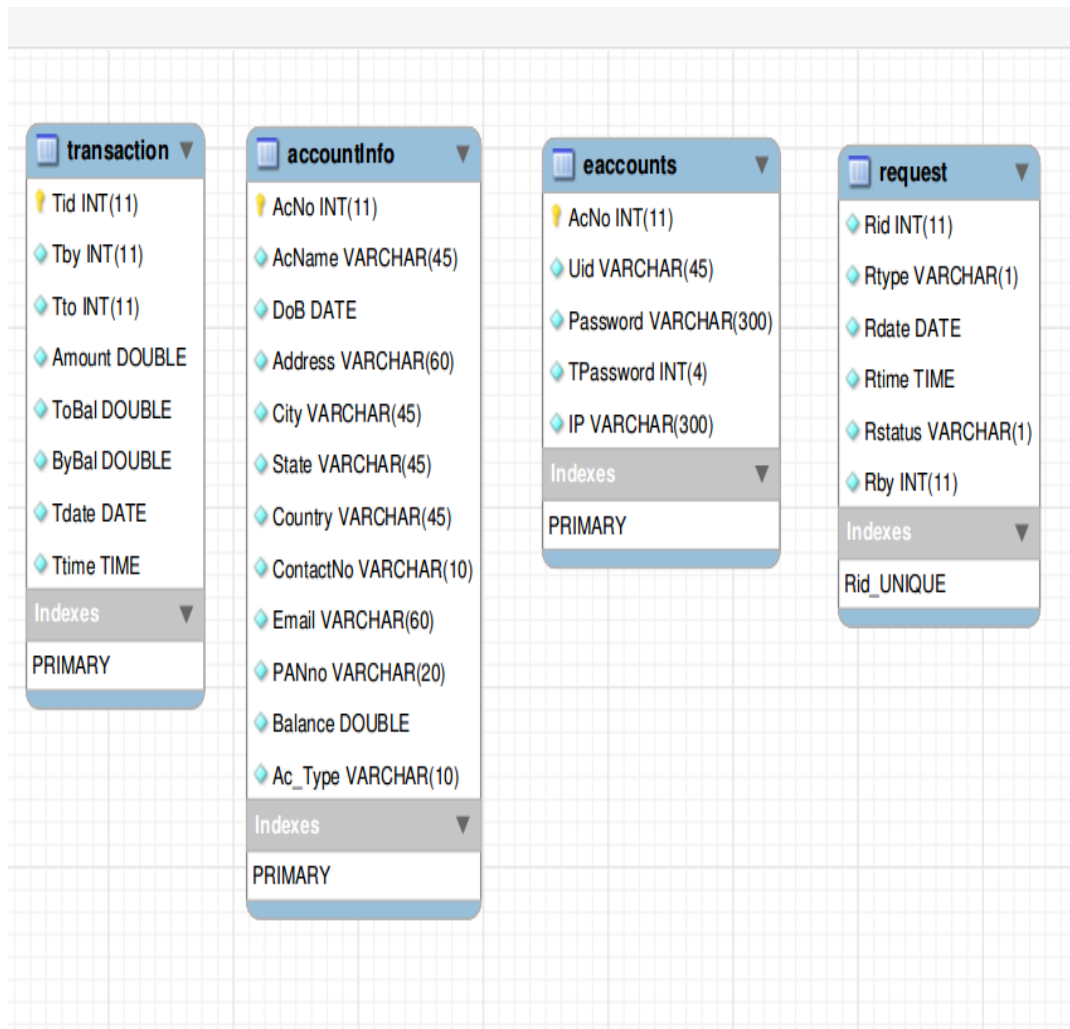


Figure 4. 13 E-R Diagram

### 4.3. MODULE DESIGN AND ORGANIZATION:

This application consists following modules:

- Admin module
- User module
- Reports Module

#### **Admin Module:**

This module is belongs to bank staff. By using this module Administrator can add type of accounts (saving and current etc.), minimum deposit amount in that particular account, interest rate and period of time.

#### **User Module:**

This module is all about customers of a bank. By using this module customers can go for different operations like they can apply for a new account in a bank, they can apply for different loans based on their requirement and also they can view their personal information, modify both personal and login information etc. Following operations can be done by any customer.

- **New Account:** Customer can open a new account.
- **Transaction Details:** Customer can transfer his funds to other accounts and also they can pay tax to govt.
- **Funds Transfer:** By using this functionality customer can send money from one account to another account.
- **Requests:** By using this functionality customer can go for different type of transactions on drafts and cheque(s).
- **Check Recorder:** By using this functionality customers can do cheque transactions through their accounts.
- **New Loan Request:** By using this functionality customer can apply for new Loan card.



- **Modify Customer Information:** By using this functionality customers can update their personal information by providing account number.

#### **Reports Module:**

In this module administrator will get different types of reports regarding customers like Number of customers of this portal etc. And this module is controlled by administrator only.

- **Accounts:** By using this functionality customers can view individual account holder details of this portal.
- **Card Transactions:** By using this functionality customers can view individual card transaction details.
- **Cheque reorder:** By using this functionality customers can view individual cheque transaction details.
- **Customers:** By using this functionality customers can view other existing customer details that are having accounts in the same bank.
- **Funds Transfer:** By using this functionality customers can view individual reports of funds transfer.

#### **4.4. CONCLUSION:**

Thus from above analysis & design it is very clear that “E-Banking” is a very useful application by which users can transfer money from one account to another at very ease at their fingertips with just few clicks. Also it enables the User to save time from going to bank to apply for cheque book or edit profile user information.

# **IMPLEMENTATION AND RESULTS**

## **5. IMPLEMENTATION & RESULTS**

### **5.1 INTRODUCTION:**

The Implementation and Results show the overall outcome of the application with the help of different output screenshots and explanation.

### **5.2 EXPLANATION OF KEY FUNCTION:**

The Key Functions of our Project include following

- Perform-Transaction.
- Make-Request.
- Register User.
- Profile-Edit.
- One Way Encryption.

#### **Perform-Transaction:**

This function provides all the services to user like performing transaction of money from one account to another account. A User can transfer to another account provided by account no. of the another person account and the transaction password which is one way for providing extra security to the user.

#### **Make – Request:**

This function provides all the service related to perform request to bank for the example an user can request bank to issue a new cheque book at their door step by just a click of button. In this function also user can request for loan but only an intimation is send to bank help station and further bank will ask to contract for getting further details on loan .

**Register-User:**

In this Register-User function a User can create one online E-bank account if he already has an account in the Bank. There is a prerequisite that the user can only have an E-bank account in the Bank if he has already account in the Bank.

**Profile-Edit:**

In this function Profile-Edit for the change of any information related to user the user no need to visit all the formalities and paper work for the change of any information, the user can go to Profile-Edit and change the information related to him at just clicks. This saves the user's precious time by not visiting the bank every-time he buys a new sim card or change any information.

**One-Way Encryption:**

In this function SHA-256 Algorithm is used to secure the user password from getting vulnerable to be known by anyone. This enhances and takes the security to next level. If even some is anyhow able to access and read the database he won't be able to get the user password because it is encrypted.

**5.3 Method of Implementation****5.3.1. Forms:**

The following forms are included in the project.

- **Register-Form:**
- **Login-Form:**
- **Transaction-Form:**
- **Profile-Edit Form:**
- **Password-Reset Form:**

## Register Form:

- Code:

```

<form name="reg" class="navbar-form navbar-right" role="search">
    <div class="form-group">
        <input type="text" class="form-control" placeholder="Search" />
    </div>
    <button type="submit" class="btn btn-danger"><span class="glyphicon
glyphicon-search" aria-hidden="true"></span></button>
</form>
</div>
</div><!-- /.navbar-collapse -->
</div><!-- /.container -->
</nav><!-- /.navbar -->
<!--REGISTER-FORM-->
<div id="form-container" class="col-lg-12 col-md-6 col-xs-4">
    <form name="reg" onsubmit="return myFunction();"role="form" class="col-lg-
4 col-sm-4 col-xs-12" id="login-form" method="POST" action="register"
autocomplete="off">
        <h3>Self-User Creation</h3>
        <div class="form-group">
            <input placeholder="Account No." type="text" class="form-control"
id="accno" name="accno" required>
        </div>
        <div class="form-group">
            <input placeholder="Username" type="text" class="form-control" id="uid"
name="uid" required>
        </div>
        <hr style="border-color:#000;">
        <div class="form-group">
            <input placeholder="DoB" type="date" class="form-control" id="date"
name="date" required>
        </div>
        <div class="form-group">
            <input placeholder="PAN no." type="text" class="form-control" id="panno"
name="panno" required>
        </div>
        <div class="form-group">
            <input placeholder="Password" type="password" class="form-control"
id="pwd" name="pwd" required>
        </div>
        <div class="form-group">
            <input placeholder="Confirm-Password" type="password" class="form-
control" id="cnpwd" name="cnpwd" onkeyup="checkPass(); return false;"
required>
        </div>
        <div class="form-group">

```

```

        <input placeholder="Transaction-Password" type="password" class="form-
control" id="Tpwd" name="Tpwd" required>
    </div>
    <input type="submit" class="btn btn-primary btn-lg" value="Submit"
style="border-color: #000;"></input>
    <button type="reset" class="btn btn-default btn-lg" style="border-color:
#000;">Clear</button>
</form>
</div>

```

## Login-Form:

- Code

```

<form role="form" class="col-lg-4 col-sm-4 col-xs-12" id="login-form"
action="login" method="POST" autocomplete="off" >
    <h3>Login</h3>
    <div class="form-group">
        <input placeholder="Username" type="text" class="form-control"
id="uname" name="uname" required="required" >
    </div>
    <div class="form-group">
        <input placeholder="Password" type="password" class="form-control"
id="upass" name="upass" required="required" >
    </div>
    <div id="" class="text-danger">
    </div>
    <a class="text-danger" href="passwordReset.jsp" style="font-weight:
700;">Forgot Password ?</a>
    <br>
    <button type="submit" class="btn btn-primary btn-lg">Login</button>
    <a type="submit" class="btn btn-default btn-lg" href="register.jsp">Register</a>
</form>

```

## Transaction-Form:

- Code:

```

<form name="Tform" role="form" class="col-lg-4 col-sm-4 col-xs-12"
id="transfer-form" method="POST" action="PerformTransaction"
autocomplete="off">
    <h3>Transfer Form</h3>
    <hr style="border-color: #000;">
    <div class="form-group">
        <input placeholder="Account No" type="text"
class="form-control readonly" id="Tby" name="Tby"
value="<%=session.getAttribute("acc")%>" disabled>

```

```

</div>
<div class="form-group">
  <input placeholder="To Account" type="text" class="form-control"
    id="Tto" name="Tto" required>
</div>
<div class="form-group">
  <input placeholder="Amount" type="number" class="form-control"
    id="Tamount" name="Tamount" required>
</div>
<div class="form-group">
  <input placeholder="Transaction Password" type="password"
    class="form-control" id="Tpass" name="Tpass" required>
</div>
<small class="text-muted">* Mandatory, All Fields will get
Updated. Check Again the Details.
</small>
<br>
<input type="submit"
class="btn btn-danger btn-lg" value="Transfer"
style="border-color: #000;"></input>
<button type="reset" class="btn btn-default btn-lg"
style="border-color: #000;">Clear</button>
</form>

```

## Profile-Edit Form:

- Code:

```

<form name="pedit" role="form" class="col-lg-4 col-sm-4 col-xs-12"
  id="profileEdit-form" method="POST" action="update" autocomplete="off">
  <h3>Profile Edit Form</h3>
  <hr style="border-color: #000;">
  <div class="form-group">
    <input placeholder="Account Name" type="text" class="form-control"
      id="AcName" name="AcName" required>
  </div>
  <div class="form-group">
    <input placeholder="DoB" type="date" class="form-control" id="DoB"
      name="DoB" required>
  </div>
  <div class="form-group">
    <input placeholder="Address" type="text" class="form-control"
      id="Address" name="Address" required>
  </div>
  <div class="form-group">
    <input placeholder="City" type="text" class="form-control" id="City"
      name="City" required>
  </div>

```

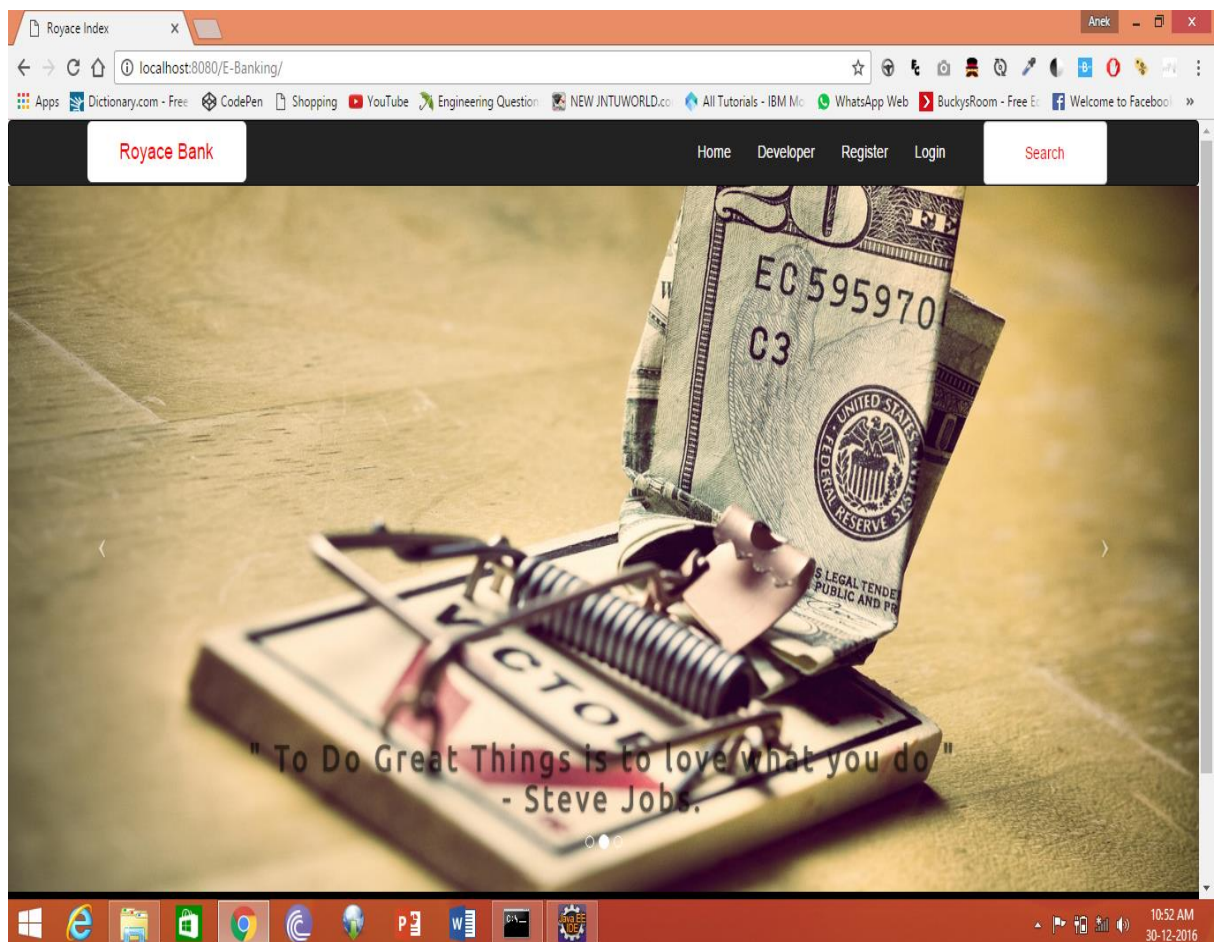
```

<div class="form-group">
  <input placeholder="State" type="text" class="form-control"
    id="State" name="State" required>
</div>
<div class="form-group">
  <input placeholder="Country" type="text" class="form-control"
    id="Country" name="Country" required>
</div>
<div class="form-group">
  <input placeholder="Contact No" type="tel" class="form-control"
    id="ContactNo" name="ContactNo" required>
</div>
<div class="form-group">
  <input placeholder="Email" type="email" class="form-control"
    id="Email" name="Email" required>
</div>
<small class="text-muted">* Mandatory, All Fields will get
Updated</small>
<br>
<div class="text-danger">
  ${emsg}
</div>
<input type="submit" class="btn btn-primary btn-lg"
value="Update" style="border-color: #000;"></input>
<button type="reset" class="btn btn-default btn-lg"
style="border-color: #000;">Clear</button>
</form>

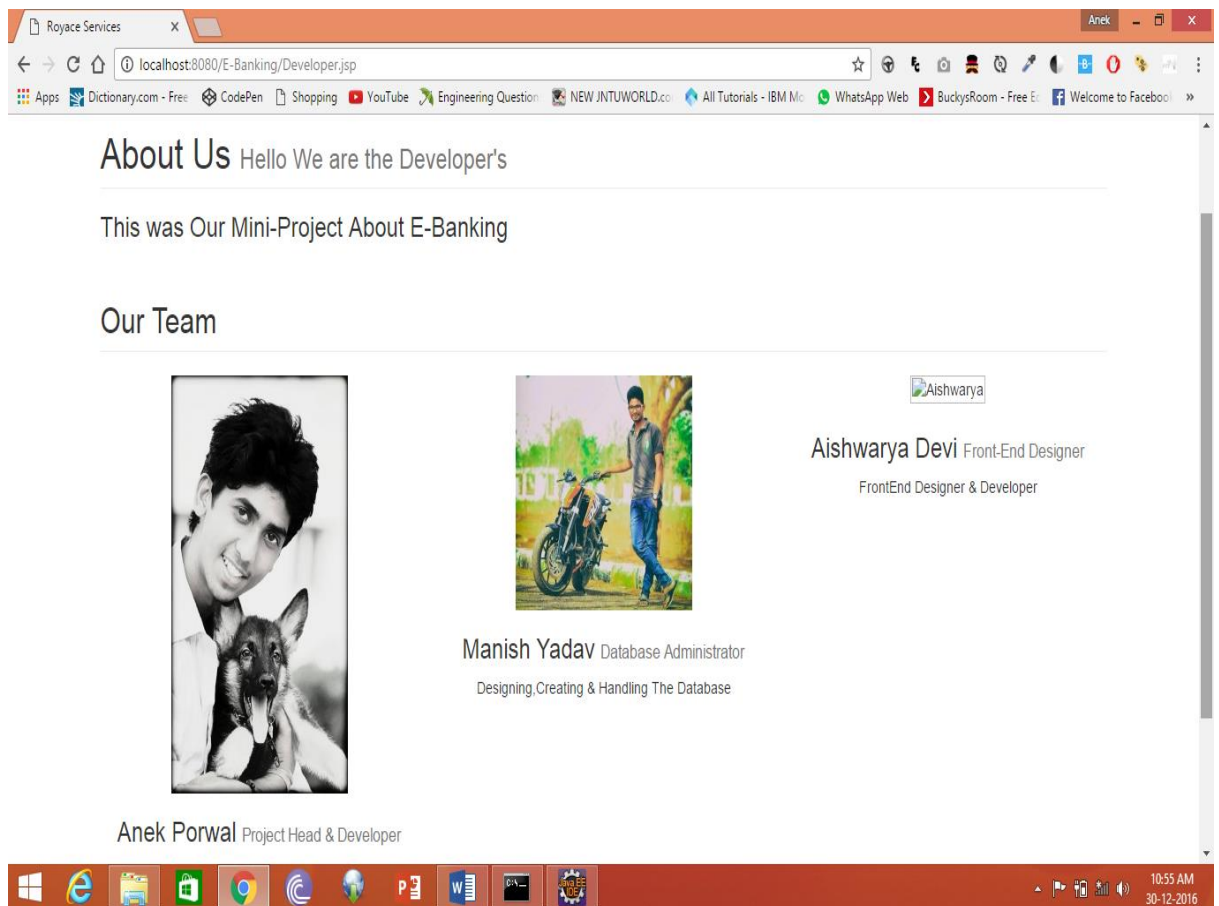
```



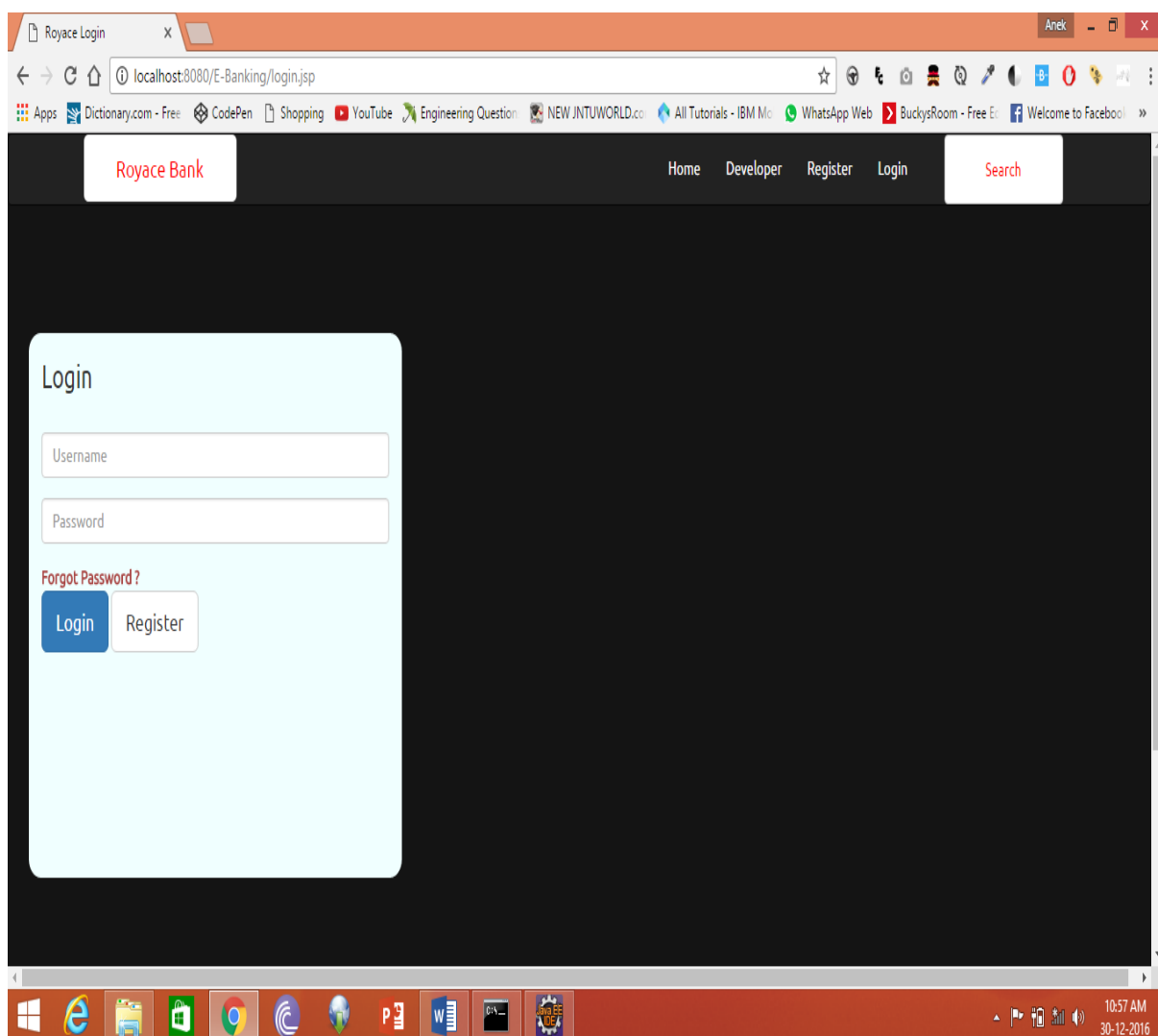
### 5.3.2. Output Screens:



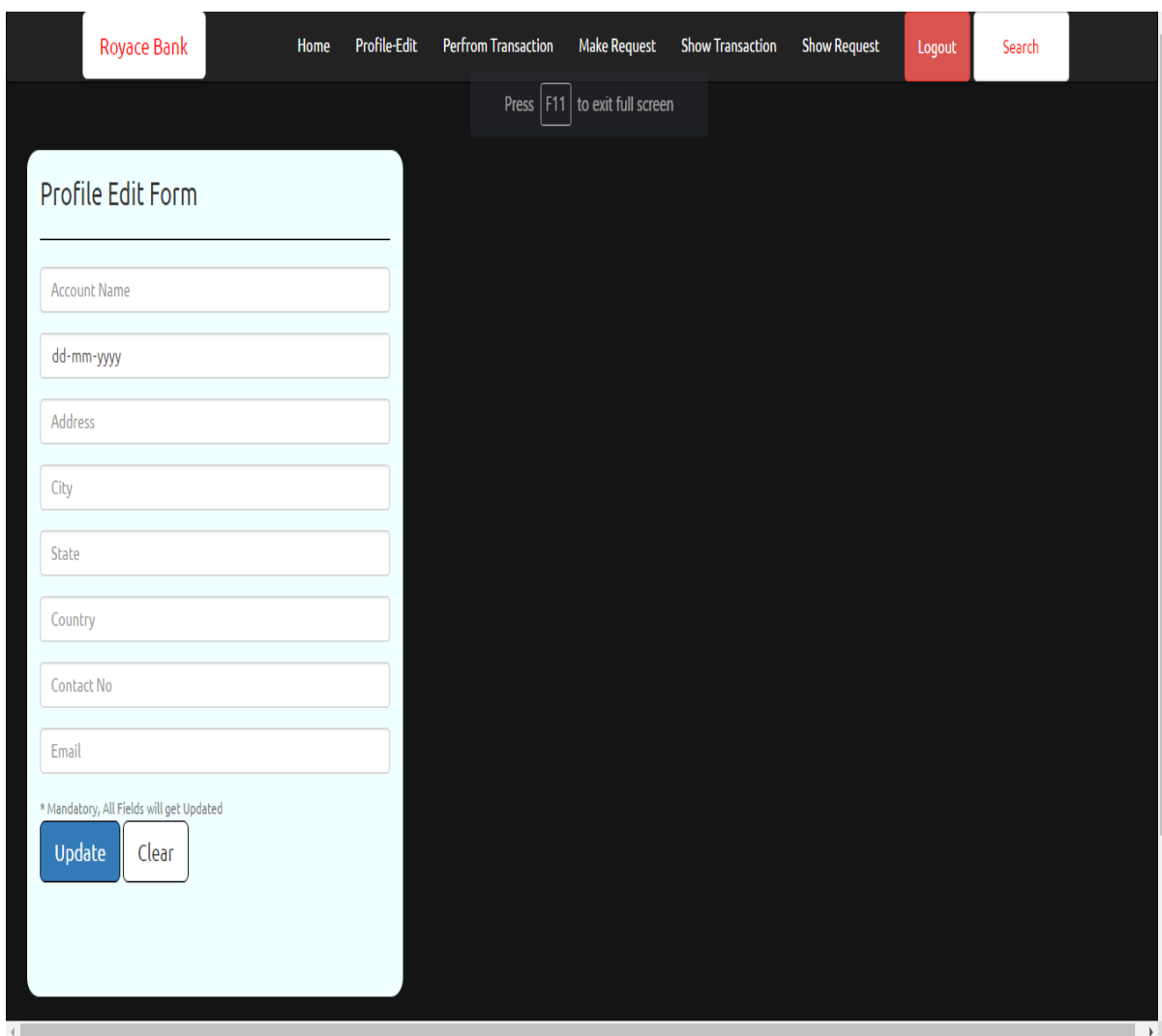
#### 5. 1 Home Page



## 5. 2 Developer Page



### 5. 3 Login Page



The screenshot displays the 'Profile Edit Form' on the Royace Bank website. The form is a light blue rounded rectangle on a dark background. It contains several input fields: 'Account Name', a date field with the placeholder 'dd-mm-yyyy', 'Address', 'City', 'State', 'Country', 'Contact No', and 'Email'. Below these fields is a note: '\* Mandatory, All Fields will get Updated'. At the bottom of the form are two buttons: a blue 'Update' button and a white 'Clear' button. The website's navigation bar at the top includes links for Home, Profile-Edit, Perform Transaction, Make Request, Show Transaction, Show Request, Logout, and Search. A tooltip above the form indicates 'Press F11 to exit full screen'.

Royace Bank

Home Profile-Edit Perform Transaction Make Request Show Transaction Show Request Logout Search

Press F11 to exit full screen

### Profile Edit Form

Account Name

dd-mm-yyyy

Address

City

State

Country

Contact No

Email

\* Mandatory, All Fields will get Updated

Update Clear

#### 5. 4 Profile-Edit Page

Transaction Information : 12345

T-Id	From	To	Amount	Date
7	12345	54321	5500	2016-12-30
8	12345	45211	3000	2016-12-30
9	12345	54963	49000	2016-12-30
10	12345	22565	36500	2016-12-30
11	12345	45211	5260	2016-12-30

All © Rights Reserved 2016. Royace Bank Inc.

### 5. 5 Transaction Successful Page

localhost:8080/E-Banking/showRequest.jsp

Royace Bank

Home Profile-Edit Perfrom Transaction Make Request Show Transaction Show Request Logout Search

## Request Information : 12345

Type: C-Check-Book & L-Loan

R-Id	Type(C/L)	Date	Time	Status
1	C	2016-12-30	23:37:11	P
2	L	2016-12-30	23:37:18	P
3	C	2016-12-30	23:40:07	P
4	L	2016-12-30	23:41:49	P

All © Rights Reserved 2016. Royace Bank Inc.

Windows taskbar: 11:42 PM, 30-12-2016

### 5. 6 Request Page

**5.3.3 Result Analysis:**

The analysis of the results based on the expected and obtained outputs can be stated to be valid according to the requirements of the application and all the features that are necessary for the functionality of the application.

**5.4 CONCLUSION:**

Based on the obtained outputs, we check the requirements and compare it with the result. As the result is satisfactory, we conclude that the application is functioning properly.

# **TESTING AND VALIDATION**



## **6. TESTING AND VALIDATION**

### **6.1 INTRODUCTION:**

Once source code has been generated, software must be tested to uncover (and correct) as many errors as possible before delivery to customer. Our goal is to design a series of test cases that have a high likelihood of finding errors. To uncover the errors software techniques are used. These techniques provide systematic guidance for designing test that (1) Exercise the internal logic of software components, and (2) Exercise the input and output domains of the program to uncover errors. In program function, behavior and performance.

### **6.2 TEST CASES:**

**Steps:** Software is tested from two different perspectives:

- (1) Internal program logic is exercised using —White Box test case design Techniques.
- (2) Software requirements are exercised using —Block Box test case. Design techniques.

In both cases, the intent is to find the maximum number of errors with the Minimum amount of effort and time.

### **Testing Methodologies:**

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. A strategy must provide guidance for the practitioner and a set of milestones for the manager. Because the steps of the test strategy occur at a time when deadline pressure begins to rise, progress must be measurable and problems must surface as early as possible. Following testing techniques are well known and the same strategy is adopted during this project testing.

**Unit Testing:**

Unit testing focuses verification effort on the smallest unit of software design- the software component or module. The unit test is white-box oriented. The unit testing implemented in Notebook module of E-Banking Application. By giving correct manual input to the system, the data's are stored in database and retrieved. If you want required module to access input or get the output from the End user. Any error will accrued the time will provide handler to show what type of error will accrued.

**System Testing:**

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Below we have described the two types of testing which have been taken for this project. It is to check all modules worked on input basis .if you want change any values or inputs will change all information. So specified input is must.

**Performance Testing:**

Performance Testing Performance testing is designed to test the run-time performance of software within the context of an integrated system. Performance testing occurs throughout all steps in the testing process. Even at the unit level, the performance of an individual module may be assessed as white-box tests are conducted.

This project reduces Notebook table code. It will generate quick reports and will not have extra time or waiting of results. Entered data will show result few millisecond. Just used only low memory of our system. Automatically do not getting access at another software. Get user permission and access to other applications.

**Test Cases:**

Test case is an object for execution for other modules in the architecture does not represent any interaction by itself. A test case is a set of sequential steps to execute a test operating on a set of predefined inputs to produce certain expected outputs. There are two types of test cases:-manual and automated. A manual test case is executed manually while

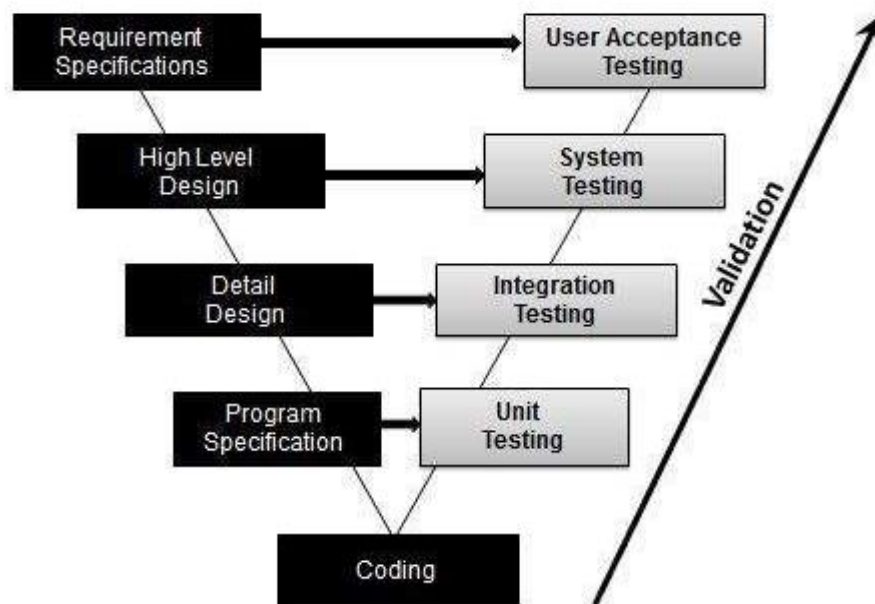
an automated test case is executed using automation. In system testing, test data should cover the possible values of each parameter based on the requirements. Since testing every value is impractical, a few values should be chosen from each equivalence class. An equivalence class is a set of values that should all be treated the same. Ideally, test cases that check error conditions are written separately from the functional test cases and should have steps to verify the error messages and logs. Realistically, if functional test cases are not yet written, it is ok for testers to check for error conditions when performing normal functional test cases. It should be clear which test data, if any is expected to trigger errors.

### 6.3. VALIDATION:

Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

#### VALIDATION TESTING-WORKFLOW:

Validation testing can be best demonstrated using V-Model. The Software/product under test is evaluated during this type of testing.



#### 6.1 Validation & Workflow

**ACTIVITIES:**

- Unit Testing
- Integration Testing
- System Testing
- User Acceptance Testing

**UNIT TESTING:**

- Unit testing, a testing technique using which individual modules are tested to determine if there are any issues by the developer himself. It is concerned with functional correctness of the standalone modules.
- The main aim is to isolate each unit of the system to identify, analyze and fix the defects.

**INTEGRATION TESTING:**

- Upon completion of unit testing, the units or modules are to be integrated which gives rise to integration testing. The purpose of integration testing is to verify the functional, performance, and reliability between the modules that are integrated.

**SYSTEM TESTING:**

- System Testing (ST) is a black box testing technique performed to evaluate the complete system the system's compliance against specified requirements. In System testing, the functionalities of the system are tested from an end-to-end perspective.

**TESTCASE SCREENSHOTS:**

The screenshot displays the login interface of the Royace Bank website. The header includes the bank's name and navigation links. The login form contains fields for username and password, with a 'Forgot Password?' link and 'Login' and 'Register' buttons. A validation message is shown for the password field.

Royace Bank

Home Developer Register Login Search

Login

jainanek

Password

Forgot Password?

Login Register

Please fill out this field.

**6. 2 Password Not Entered Test Case**

The screenshot displays the login interface of the Royace Bank. The header includes the bank's name and navigation links. The login form contains fields for username and password, with a 'Forgot Password?' link and 'Login' and 'Register' buttons. A validation message is shown for the password field.

Royace Bank

Home Developer Register Login Search

### Login

Username

\*\*\*\*\*

Please fill out this field.

Forgot Password?

Login Register

### 6. 3 Username Not Entered Test Case

**Error** Occured, Please Try Again!!.

- 1.May be Wrong User Credentials
- 2.May be Your Account is Not Validated
- 3.Something went Wrong.

[HOME](#) [LOGIN](#) [REGISTER](#)

#### 6. 4 Incorrect Password Test Case

The screenshot displays the 'Self-User Creation' form on the Royace Bank website. The form is set against a dark background. At the top, a navigation bar includes the 'Royace Bank' logo, links for 'Home', 'Developer', 'Register', 'Login', and a 'Search' button. The form itself is a light blue rounded rectangle containing several input fields: a text field with '12345', a text field with 'Anek Jain', a date field with '25-05-1996', a text field with 'PAN no.', a password field with four asterisks, a green confirmation password field with four asterisks, and another password field with four asterisks. A red error message box with an exclamation mark icon and the text 'Please fill out this field.' is positioned over the first password field. At the bottom of the form are 'Submit' and 'Clear' buttons.

### 6. 5 Password Not Matching in Registration Test Case



The screenshot displays the 'Transfer Form' on the Royace Bank website. The form is set against a dark background. It includes a header with the bank's name and navigation links. The form fields are: a source account number (12345), a 'To Account' field, an 'Amount' field with a red error message 'Please fill out this field.', and a 'Transaction Password' field. At the bottom, there are 'Transfer' and 'Clear' buttons, along with a mandatory field notice.

Royace Bank

Home Profile-Edit Perfrom Transaction Make Request Show Transaction Show Request Logout Search

### Transfer Form

12345

To Account

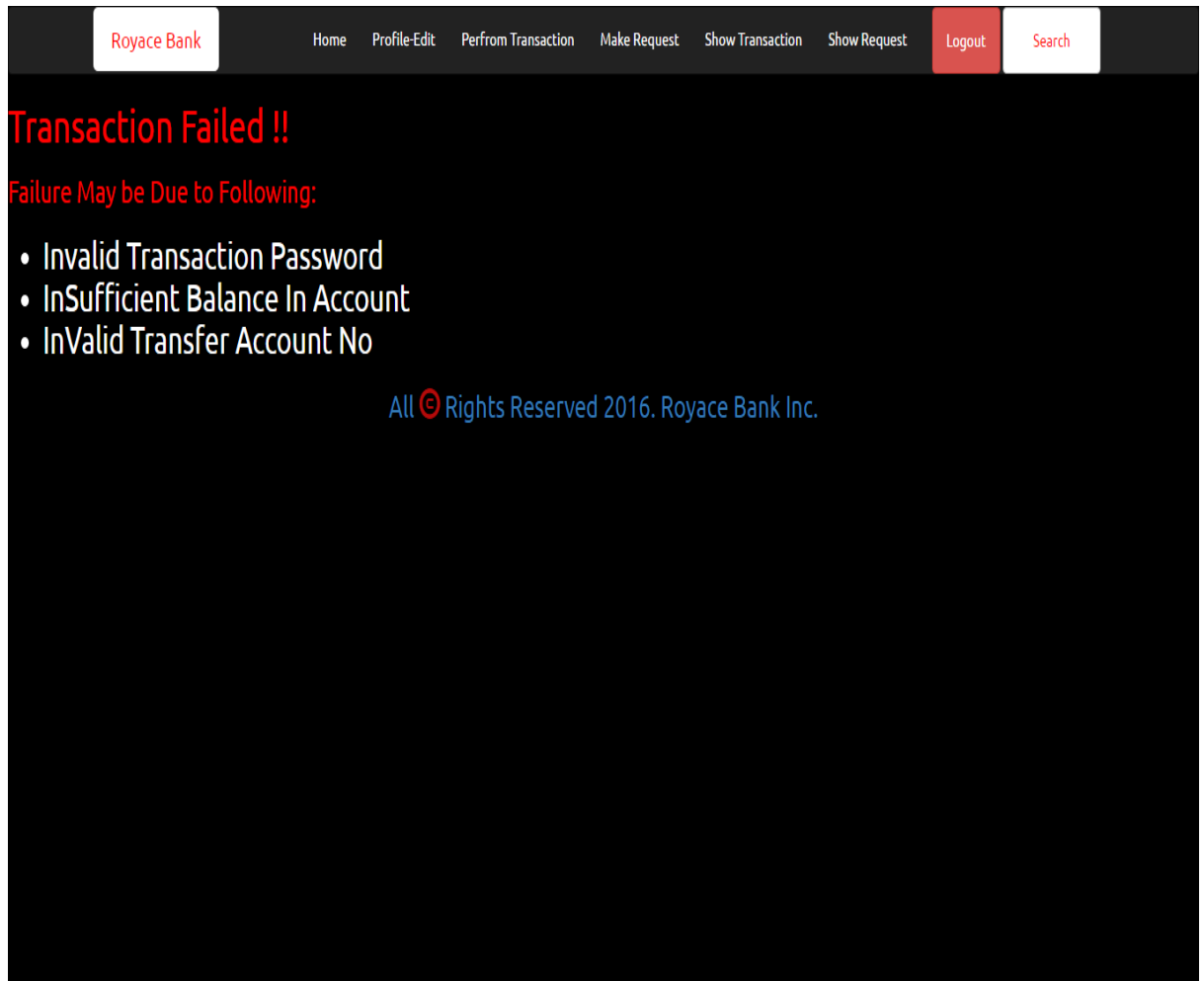
Amount ! Please fill out this field.

Transaction Password

\* Mandatory, All Fields will get Updated. Check Again the Details.

Transfer Clear

## 6. 6 Invalid Account During Transaction Test Case



#### 6. 7 Invalid Transaction Password Test Case

## **6.4 CONCLUSION:**

critically important verification method that takes up a very large portion of a project's resources, including schedule, budget, staffing, and facilities. Unlike the many constructive activities of systems engineering, testing is relatively unique because it is inherently destructive. Its primary purpose is to force the system or its components to fail so that the defects that caused the failure can be uncovered and then fixed.

# CONCLUSION

## **7. CONCLUSION**

### **7.1. PROJECT CONCLUSION:**

To conclude, Project E-Banking works like a component which can access all the databases and picks up different functions. Easy implementation Environment Generate report flexibly. This project provides an secure way from transferring amount from one account to another at user clicks.

### **7.2. FUTURE ENCHANCEMENT:**

The project has a very vast scope in future. The project can be implemented on intranet in future. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion. With the proposed software of database Space Manager ready and fully functional the client is now able to manage and hence run the entire work in a much better, accurate and error free manner. The following are the future scope for the project.

Provision of additional location options for the user to choose.

Credit-Card and Debit-Card Services to be availed to User.

Enhanced Notification to User about transaction & login access Details for confirmation to his Mobile No.

Enhanced Security by providing One Time Password to User's Mobile No. for Transaction.

# REFERENCES

## **8. REFERENCES**

### **8.1 Referred Book(s):**

- 1. IBM Career Education- Enterprise Mobile Application Development Using IBM WorkLight (Part [01] and Part [02]).**
- 2. Pure JSP: Java Server Pages by James Goodwill (Sams, 2000).**
- 3. Web Development with Java Server Pages by Duane K. Fields and Mark A. Kolb (Manning Publications, 2000).**
- 4.Design Patterns By Enrich Gamma.**

### **8.2 Referred Link(s):**

<https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>.

<https://en.wikipedia.org/wiki/SHA-2>.

[https://en.wikipedia.org/wiki/Cryptographic\\_hash\\_function](https://en.wikipedia.org/wiki/Cryptographic_hash_function)

<https://getbootstrap.com/getting-started/>

<https://docs.oracle.com/javase/7/docs/api/>

# MRIET

## B TECH 2013-17 BATCH MINI PROJECT REPORT

**Name:** \_\_\_\_\_

**Father Name:** \_\_\_\_\_

**Roll No:** \_\_\_\_\_

**Contact No:** \_\_\_\_\_

**Project Title:** \_\_\_\_\_

**Company Name:** \_\_\_\_\_

**Contact No:** \_\_\_\_\_

**Final Records Submission Date:** \_\_\_\_\_

**Soft Copy**

**Hard Copy**

**HOD**

**PRINCIPAL**

**Verified By**

**Verified By**