2020-21

Shivajirao Kadam Institute of Technology & Management, Indore

Submitted To:

Prof. Swapnil Waghela

(Asst. Prof.,CSE)

Department of Computer Science & Engineering

Software Engineering

Submitted By:

Name of Student: Kartik Kulshreshtha

Enrollment No. : 0875CS191048

Class/Year/Sem : CS-A/2nd / 4th

[LAB ASSIGNMENT SE (CS-403)]

To understand the software engineering approach. The purpose of this assignment is to cover the underlying concepts and methods used in Software Engineering.

Program Outcome (PO)

The Engineering graduate of this institute will demonstrate:

a. **Apply** knowledge of mathematics, science, computing and engineering fundamentals to computer science engineering problems.
b. Able to **identify, formulate,** and demonstrate with excellent programming, and problem solving skills.
c. **Design solutions** for engineering problems including design of experiment and processes to meet desired needs **within reasonable constraints** of manufacturability, sustainability, ecological, intellectual and health and safety considerations.
d. Propose and develop effective **investigational** solution of complex problems using research methodology; including design of experiment, analysis and interpretation of data, and combination of information to provide suitable conclusion. synthesis
e. Ability to create, select and use the **modern techniques** and various **tools** to solve engineering problems and to evaluate solutions with an understanding of the limitations.
f. Ability to acquire knowledge of **contemporary issues** to assess societal, health and safety, legal and cultural issues.
g. Ability to evaluate the **impact** of engineering solutions on individual as well as organization in a societal and environmental context, and recognize sustainable development, and will be aware of emerging technologies and current professional issues.
h. Capability to possess leadership and managerial skills, and understand and commit to professional **ethics** and responsibilities.
i. Ability to demonstrate the team work and **function** effectively as an individual, with an ability to design, develop, test and debug the project, and will be able to work with a multi-disciplinary team.
j. Ability to **communicate effectively** on engineering problems with the community, such as being able to write effective reports and design documentation.
k. Flexibility to feel the recognition of the need for, and have the ability to engage in independent and **life-long learning by** professional development and quality enhancement programs in context of technological change.
A **practice** of engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and entrepreneurship.

About the Manual

This manual is intended for 2nd Year 4th Sem B.Tech. Computer Science & Engineering students in the subject of Software Engineering. This manual typically contains Lab Sessions and experiments related to various Software development models and techniques and its applications related to our real life problems (like requirement gathering, documenting requirements, analysis, designing, implementing the design and testing). Although the University syllabus prescribes only the applications of these designing approaches, but we have tried to relate them with different real life problem to better understanding. In this section we also try to give the concept that which type of approaches will be used to solve them efficiently.

Introduction

In this section we discussed about various SDLC approaches based on some basic criteria, Problem Solved by using various methods existed much before the computer was born. Software engineering is defined as a process of analyzing user requirements and then designing, building, and testing software application which will satisfy those requirements. Software engineering is a process of analyzing user requirements and then designing, building, and testing software application which will satisfy that requirements. Important reasons for using software engineering are: 1) Large software, 2) Scalability 3) Adaptability 4) Cost and 5) Dynamic Nature. In late 1960s many software becomes over budget. Therefore it offers unreliable software which is expensive to maintain. The late 1970s saw the widespread uses of software engineering principles. Software engineering concept 1) Computer Science 2) Management Science 3) System engineering and 4) Economics. Increased market demands for fast turn around time is the biggest challenges of software engineering field. 1)Maintainability, 2) Dependability, 3) Efficiency and, 4) Usability are the most important attributes of software products. Three most important characteristics of good software are 1) Operational 2)Transitional 3)Maintenance.

Since most of the software problems do not have a unique solution, we are always interested in finding the better solution. A better solution is judged based on its performance. Some of the performance measures include the time taken by the solution, the quality of the solution, the simplicity of the solution, etc. Improving the performance of a solution can be done by improving the algorithm design, database design, and transaction design and by paying attention to the end-user psychology. Also continuous improvements in hardware and communication infrastructure aid in improving the performance of a solution. This document is an instructor's manual to accompany Introduction to Algorithms. It is intended for use in a course on algorithms. You might also and some of the material here in to be useful for course in data structures. It is organized around the undergraduate algorithms course for the UG students.

We have chosen to organize the manual for students according to chapters of the text. That is, for most chapters we have provided a set of lecture notes and a set of exercise and problem solutions pertaining to the chapter. This department allows you to decide how to best use the concept in problem solving in your own course. We have not included lecture notes and solutions for every chapter, nor have we included solutions for every exercise and problem within the chapters that we have selected. We have also omitted the chapters that are not covered in the courses that we teach. Future editions of this manual may include some of these chapters. There are two reasons that we have not included solutions to all exercises and problems in the selected chapters. First, writing up all these solutions would take a long time, and we felt it more important to release this manual in as timely a fashion as possible. Second, if we were to include all solutions, this manual would be longer than the text itself.

Software Engineering (CS403)

Course Outcomes (CO):

1. CO1: Graduates must know of Development of Software's as a team using different S/W Models.
2. CO2: Graduates formulate real life problems by Software Requirement Specification.
3. CO3: Graduates design architecture of real life problems with UML Modeling Techniques.
4. CO4: Students validate the quality of designed product using different testing strategies.
5. CO5: Graduates maintain the delivered product for higher customer satisfaction.

Course Learning Objectives (CLO):

1. CLO1: Be able to function effectively as a team member.
2. CLO2: Be able to elicit, analyze and specify software requirements through a productive working relationship with various stakeholders of a software development project.
3. CLO3: Be able to use Unified Modeling Language in software specification documents.
4. CLO4: Be able to identify, formulate, and solve software engineering problems, including the specification, design, implementation, and testing of software systems that meet specification, performance, maintenance and quality requirements.
5. CLO5: Be able to evaluate the impact of potential solutions to software engineering problems in a global society, using the knowledge of contemporary issues and emerging software engineering trends, models, tools, and techniques.

Mapping of CO with PO

Program OutCome--->		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
Course Outcomes	CO1	√	√									√	
	CO2		√	√									
	CO3			√	√								
	CO4									√			
	CO5									√			√

Mapping of CLO with CO

Program OutCome--->		CO 1	CO 2	CO 3	CO 4	CO 5
Course Learning Objective	CLO1	√				
	CLO2		√			
	CLO3			√		√
	CLO4		√	√	√	
	CLO5		√			√

Shivaji Rao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-01
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	B2
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	**Software Engineering**				**CO No.**
Name of Student	Kartik Kulshreshtha	Enrolment No		0875CS191048	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

- **Objective :** To get the basics of S/W Engg. and Phases of SDLC.
- **Outcome :** Must be aware about phases and models of S/W Engineering.
- **Hardware & Software Specification :**
  Operating System – Ubuntu 18.04.3 LTS
  Processor – Intel Core i3 7th Gen.
  RAM – 4GB
- **Theory :**
- *Software Engineering* :- Software Engineering is a systematic approach to the design, development, operation, and maintenance of a software system.

Or

Software engineering is defined as a process of analyzing user requirements and then designing, building, and testing software application which will satisfy those requirements.

(a). Software Engg. has wide variety of scope & necessity. Some of 'em are:

I. Software Engineering Tools.
II. S/W Development Process.
III. S/W Testing.
IV. S/W Design.

V. S/W Maintenance.
VI. S/W Quality.
VII. S/W Development.
VIII. S/W Configuration Management.
IX. S/W Engineering Management.

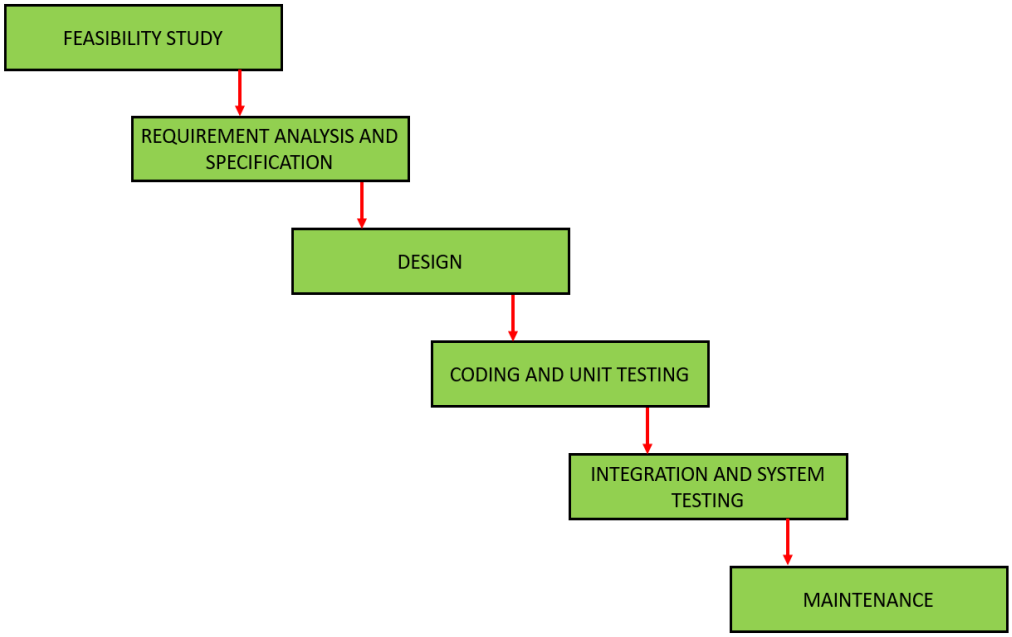
(b). A piece of program is just a set of instructions or commands to perform a one particular task. Whereas S/W product is a program or set of programs containing instructions which provide desired functionality & apart from this it is the total summation, from the creation of software to it's maintenance.

(c). The phase development of s/w is often referred as s/w development life cycle(SDLC) or s/w life cycle. The main objective behind the phases is to develop methods for large systems, which help developers obtaining high-quality s/w in min time & at low cost. These phases follow a top-to-bottom approach, implying that the phases take inputs from the previous phases, add features, and then produce outputs.

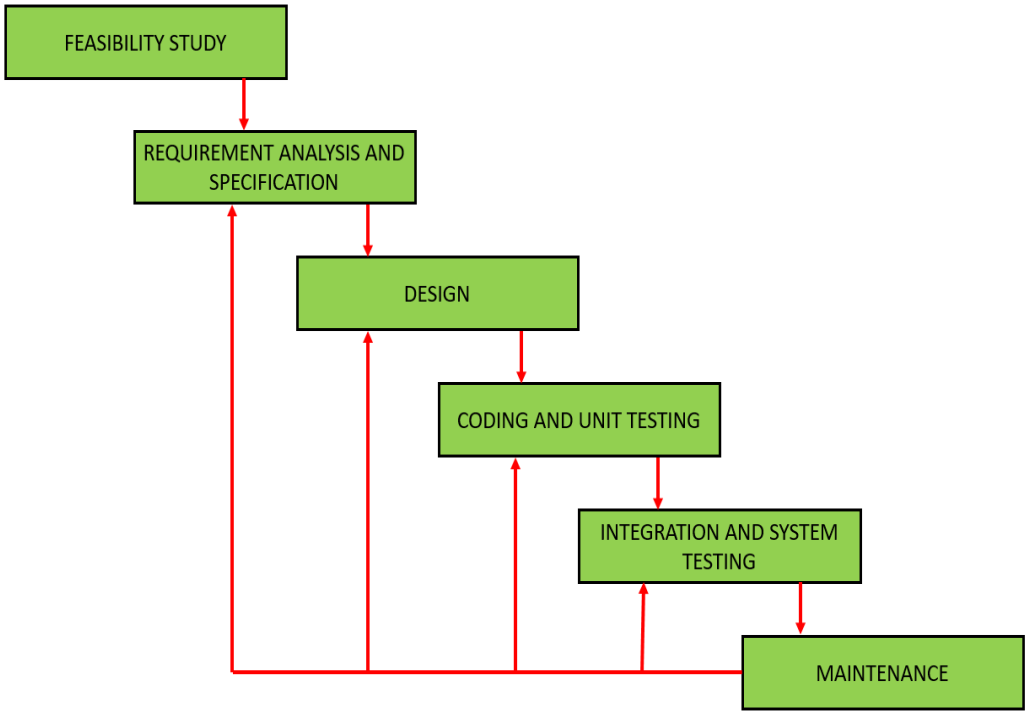
I. Requirement gathering – The aim of requirement gathering and analysis phase is to understand the exact requirement & document them properly.
II. Design – The goal of design phase is to transform the specified requirement in SRS into a structure suitable for implementation in some programming language.
III. Coding – The purpose of coding & unit testing phase is to translate the s/w designs into source code. Each component of the design is implemented as a program module . The end product of this phase is set of individual modules that has been tested individually.
IV. Integration Testing – During the integration & System testing, modules are integrated in a planned manner. During each integration step the partially integrated system is tested & set of previously developed modules are added to it. Finally when all the modules have been successfully integrated the system testing is carried out.
V. Deployment phase – When the s/w product is deploy to the actual customer provided the user manual.
VI. Maintenance phase – In this phase following activities are performed :-

a. . Correcting errors that was not developed during development or testing phase.
b. . The improving the implementation of system & enhancing the functionality according to customer requirement.
c. . Porting a s/w to work in new requirement.

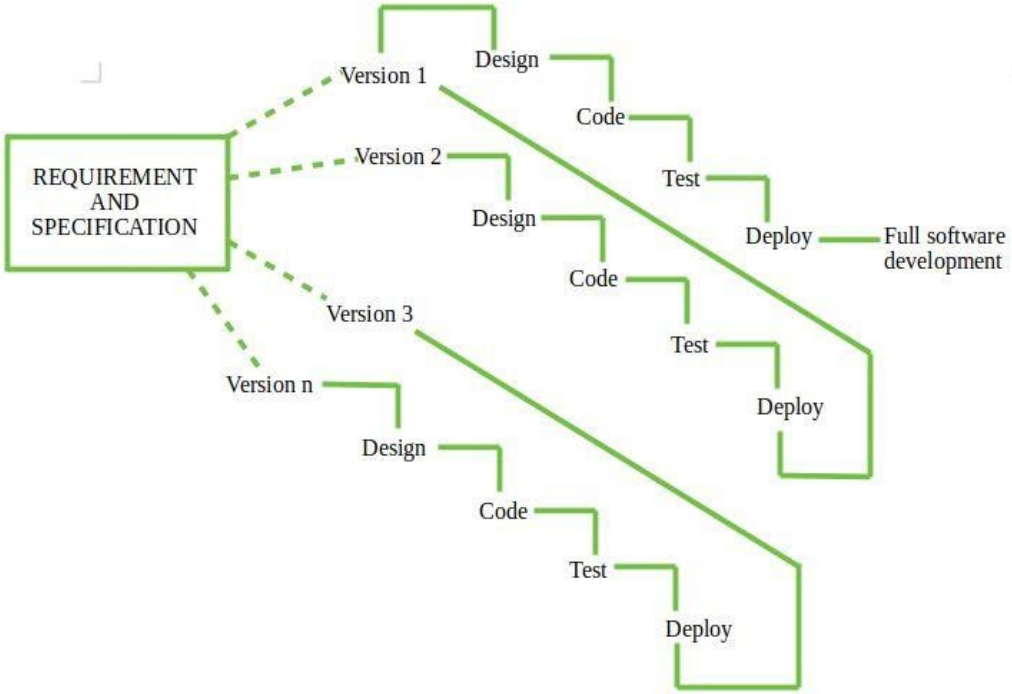
1. **Classical waterfall model -** is the basic software development life cycle model. It is very simple but idealistic. Earlier this model was very popular but nowadays it is not used. But it is very important because all the other software development life cycle models are based on the classical waterfall model. Classical waterfall model divides the life cycle into a set of phases. This model considers that one phase can be started after completion of the previous phase. That is the output of one phase will be the input to the next phase. Thus the development process can be considered as a sequential flow in the waterfall.



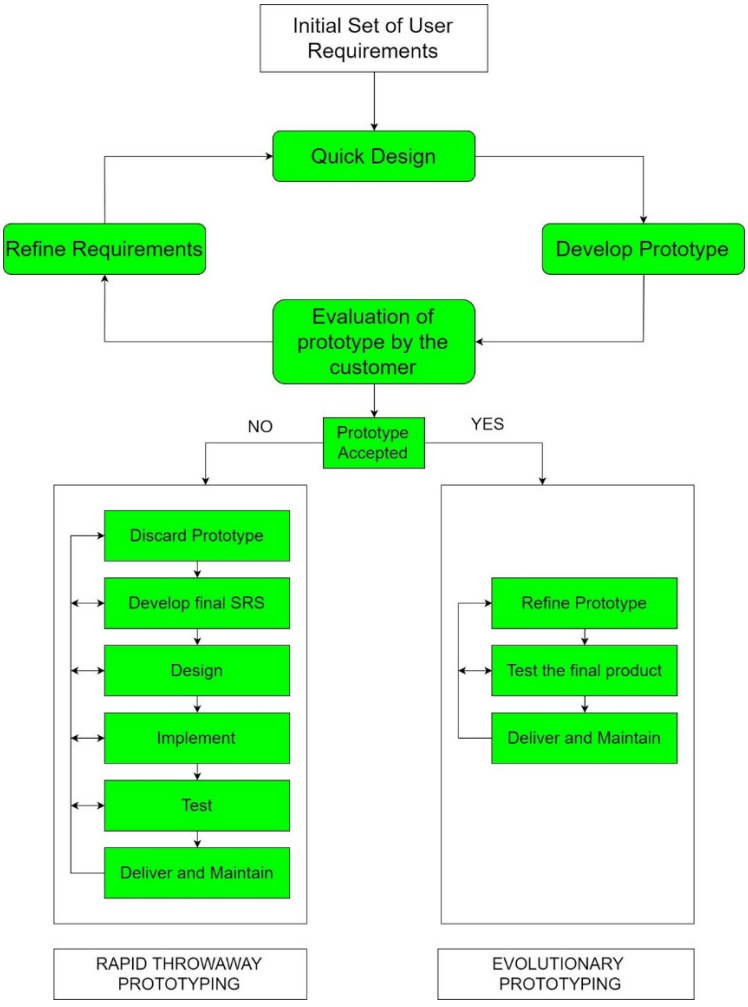
2. <u>Iterative Waterfall Model</u> - The iterative waterfall model provides feedback paths from every phase to its preceding phases, which is the main difference from the classical waterfall model.



3. <u>Incremental process model</u> - Incremental process model is also know as Successive version model. As each successive version of the software is constructed and delivered, now the feedback of the Customer is to be taken and these were then incorporated in the next version. Each version of the software have more additional features over the previous ones.



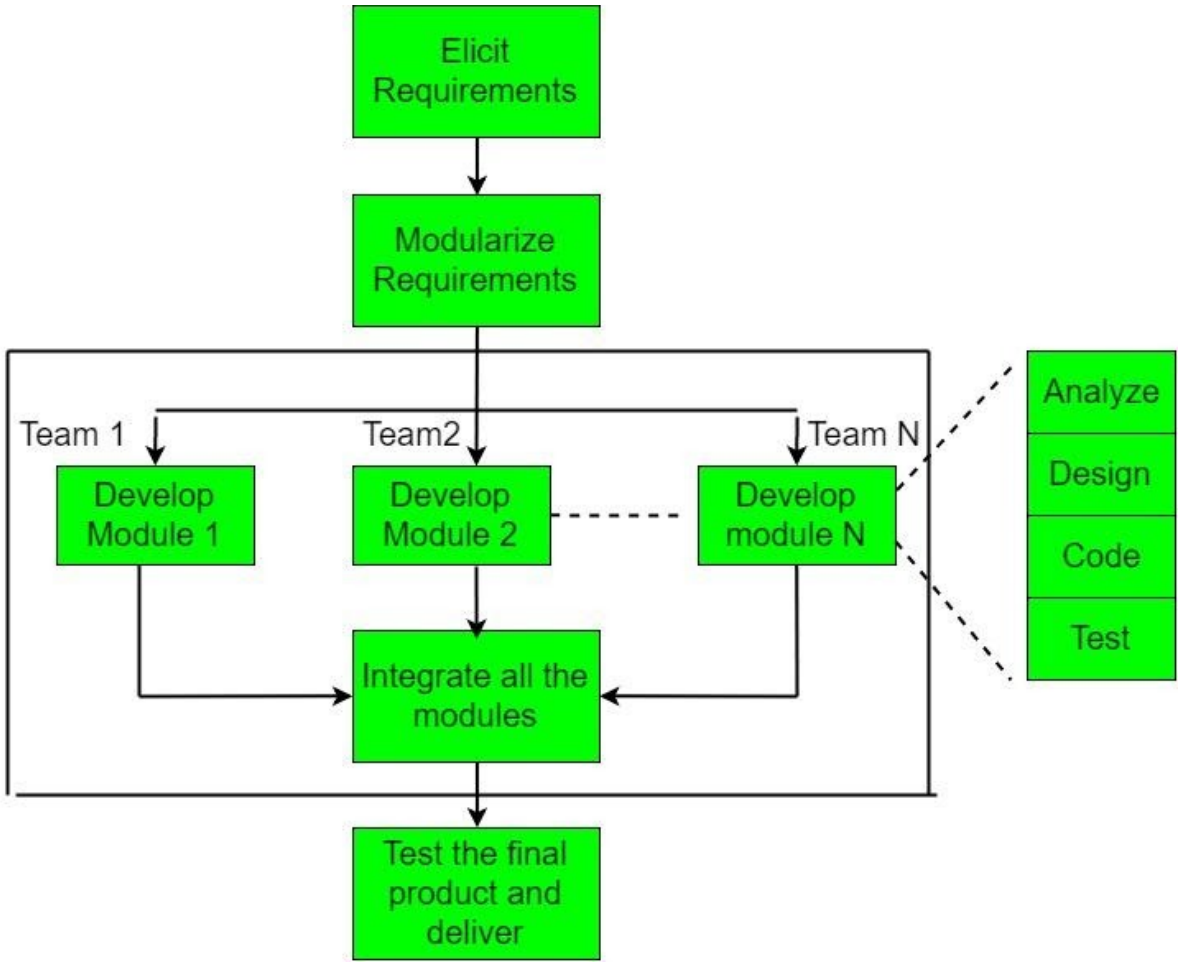
4. **<u>Prototyping Model</u>** - Prototype methodology is defined as a Software Development model in which a prototype is built, test, and then reworked when needed until an acceptable prototype is achieved. It also creates a base to produce the final system.



5. **<u>Rapid Application Development</u>** - RAD or Rapid Application Development process is an adoption of the waterfall model; it targets at developing software in a short span of time.

It has the following phases – 1. Business Modelling 2.Data Modelling 3.Process Modelling 4.Application Generation 5.Testing & Turnover

1. Business Modelling - On basis of the flow of information and distribution between various business channels, the product is designed.
2. Data Modelling -The information collected from business modelling is refined into a set of data objects that are significant for the business.
3. Process Modelling - The data object that is declared in the data modelling phase is transformed to achieve the information flow necessary to implement a business function.
4. Application Generation - Automated tools are used for the construction of the software, to convert process and data models into prototypes.
5. Testing and Turnover - As prototypes are individually tested during every iteration, the overall testing time is reduced in RAD.

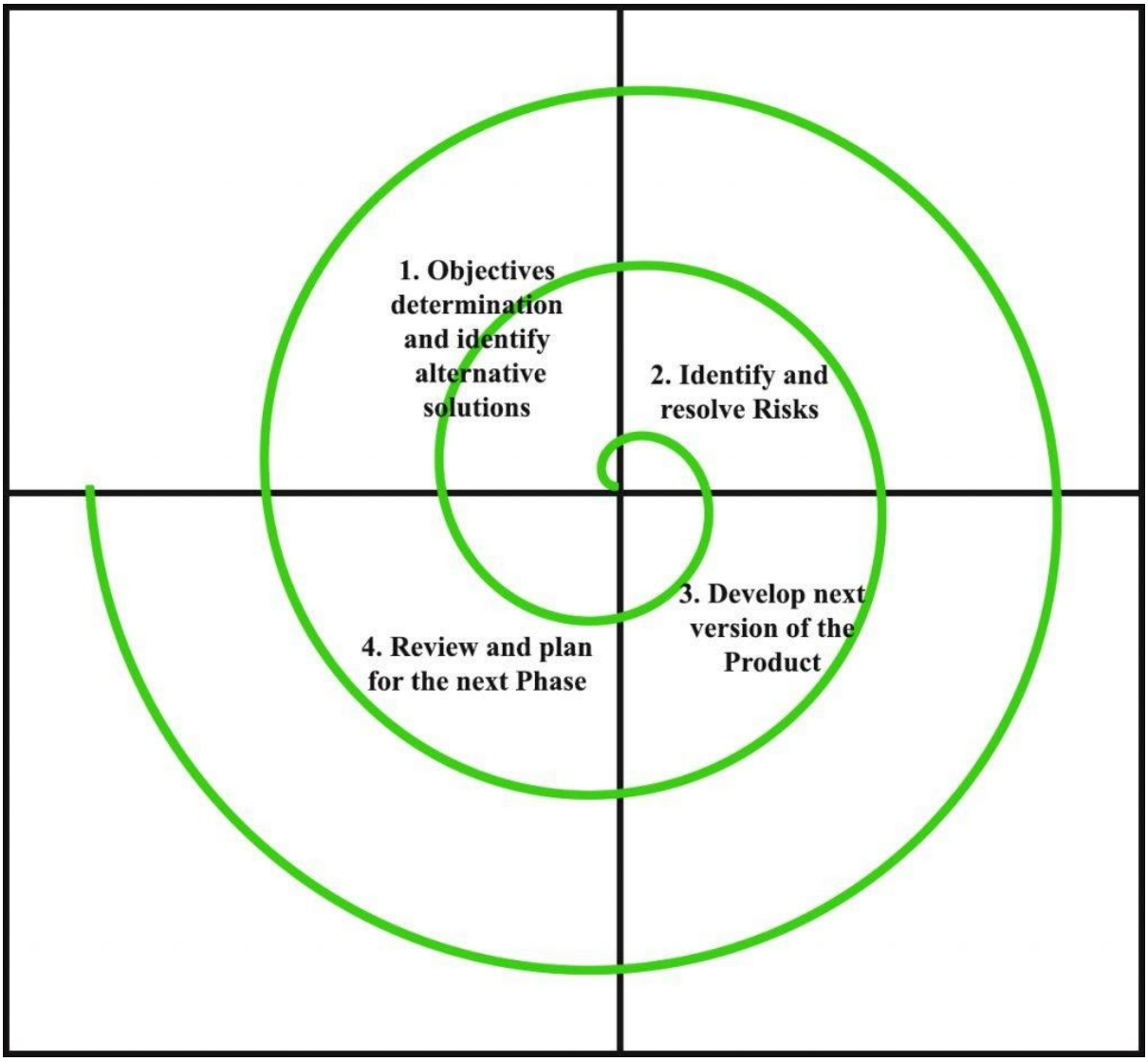


6. **<u>Spiral Model</u>** - Spiral model is one of the most important Software Development Life Cycle models, which provides support for Risk Handling. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called a Phase of the software development process. The Radius of the spiral at any point represents the expenses(cost) of the project so far, and the angular dimension represents the progress made so far in the current phase.

Each phase of Spiral Model is divided into four quadrants as shown in the above figure. The functions of these four quadrants are discussed below-

1. Objectives determination and identify alternative solutions.
2. Identify and resolve Risks.

3. Develop next version of the Product.
4. Review and plan for the next Phase.



7.<u>Agile Development Models</u> - In the Agile model, the requirements are decomposed into many small parts that can be incrementally developed. The Agile model adopts Iterative development. Each incremental part is developed over an iteration. Each iteration is intended to be small and easily manageable and that can be completed within a couple of weeks only. At a time one iteration is planned, developed and deployed to the customers. Long-term plans are not made.

Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-02
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	**Rational Rose**				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

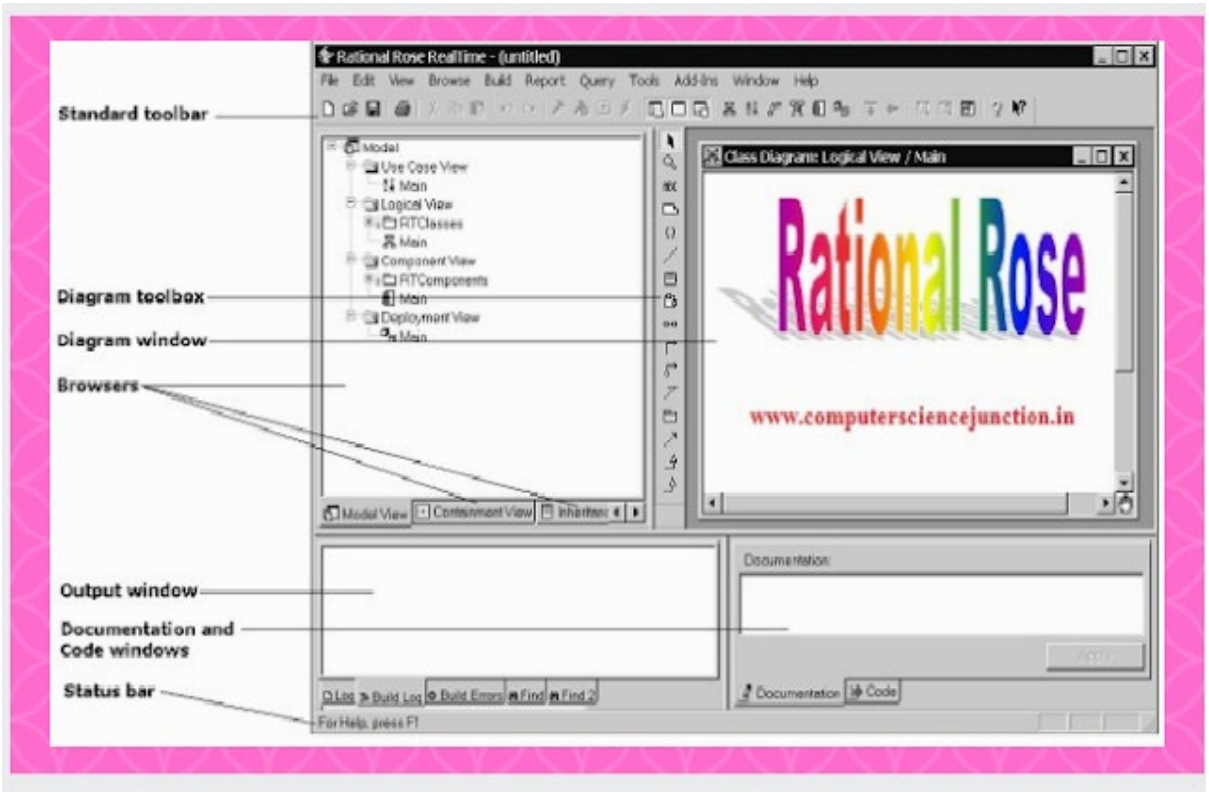
Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

- **Objective :** Awareness of make use of Rational Software.
- **Outcome :** Must be able to make use of Software Design tool.
- **Hardware & Software Specification :**
  Operating System – Ubuntu 18.04.3 LTS
  Processor – Intel Core i3 7th Gen.
  RAM – 4GB
  Software – Rational Rose.
- **Theory :** ROSE stands for Rational Object Oriented Software Engineering. Rational Rose Software is a set of visual modeling tools for development of object-oriented Modeling. Object oriented modeling is the process of graphically depicting the software system. This rational rose software tutorial focus on applications and features of rational rose software.
- **Applications of Rational Rose Software:** Rational rose software is basically used to draw UML diagram. It is a professional and widely used in industry . In academics rational rose software helps in making the diagram during the design phase of software development life cycle.
  - Modeling can be useful at any point in the application.
  - In Software Development process during design phase.

- Initial Design Work (Requirement Analysis and Definition).

- **Features of Rational Rose Graphical User Interface :** The main features of the Rational Rose Real Time user interface are as follow:
    1. The Standard Toolbar remains the same for all views and diagrams. It contains standard Windows functions as well as those specific to Rational Rose Real Time.
    2. The Diagram Toolbox is used for adding elements to the model by drawing them on a diagram. The toolbox elements change depending on the active diagram. For example, the Use-Case Diagram has a tool for adding actors, but the Component Diagram does not have this tool.
    3. Browsers are hierarchical. When you start Rational Rose Real Time, the Model View, the Containment View, and the Inheritance View browsers are on the left side of the interface in a stacked format. They can be set to visible/invisible, docked, or floating. To activate a specific browser,
    4. Select the appropriate tab located at the bottom of the interface. There are two additional browsers, also referred to as editors, that performs specific tasks: the Structure/State Diagram Browser/Editor, and the Run Time System (RTS) Browser/Editor.
    5. Rational Rose Real Time offers four main views located on the Model View browser. Each view is corresponds to a software life cycle phase, and the diagrams are artifacts of those phases.



Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-03
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	**Requirement Engineering**				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

- **Objective :** To Understand Process of Requirement Engineering with its types.
- **Outcome :** Must be able to gather requirement on any software system.
- **Hardware & Software Specification :**
  Operating System – Ubuntu 18.04.3 LTS
  Processor – Intel Core i3 7th Gen.
  RAM – 4GB
- **Theory :**
- **Requirements Engineering Process -** Requirement Engineering is the process of defining, documenting and maintaining the requirements. It is a process of gathering and defining service provided by the system.
  Requirements Engineering Process consists of the following main activities:
  I. **Requirements elicitation** - It is related to the various ways used to gain knowledge about the project domain and requirements. The various sources of domain knowledge include customers, business manuals, the existing software of same type, standards and other stakeholders of the project.
  II. **Requirements specification** - This activity is used to produce formal software requirement models. All the requirements including the functional as well as the non-functional requirements and the constraints are specified by these models in totality. During specification, more knowledge about the problem may be required which can again trigger the elicitation process.
  III. **Requirements verification and validation -** *Verification:* It refers to the set of tasks that ensures that the software correctly implements a specific function.
  Validation: It refers to a different set of tasks that ensures that the software that has been built is traceable to customer requirements.

IV. **Requirements management -** It is the process of analyzing, documenting, tracking, prioritizing and agreeing on the requirement and controlling the communication to relevant stakeholders. This stage takes care of the changing nature of requirements.

- **Non-Functional Requirement of library management system** - These are those that specify some criteria that can be used to evaluate the performance of a system in some particular conditions.

1. **Efficiency Requirement -** Through this system, the students or teachers and the librarian gets a way to ease their work. Through this system, the student can search and get the book issued easily.
   Also, less time will be needed to spend by the librarian to handle this. Therefore the throughput is faster processing of library management system.

2. Reliability Requirement - The system does its work with more accuracy like user registration to the system, user validation and authorization, book search and issue operation, return status, and updating the database by synchronizing between database and application.

3. Usability Requirement - The proposed library management system provides a user-friendly environment to the users so that the librarians, as well as the students, can utilize the system in an effective manner for ease of work.

4. Delivery Requirement - There is always some time duration specified to develop a project. Similarly, this system is expected to be complete within 6 months of time. This launch will be used for improving the performance, as it will be evaluated by the users and then the problems that are occurring with the system will be solved.

5. **System Implementation Requirement -** To develop this system PHP, the server side scripting language has been used along with HTML 5 for designing the system layout. Also, PHP has been used since it is easy and effective for database connectivity. For the backend part which includes the database itself, MySQL has been used.

6.

- **Functional Requirement of library management system -**

1. Register

- Description : First the user will have to register/sign up. There are two different type of users.
- The library manager/head : The manager have to provide details about the name of library ,address, phone number, email id.
- Regular person/student : The user have to provide details about his/her name of address, phone number, email id.

1. Sign up

- Input: Detail about the user as mentioned in the description.
- Output: Confirmation of registration status and a membership number and password will be generated and mailed to the user.
- Processing: All details will be checked and if any error are found then an error message is displayed else a membership number and password will be generated.

2. Login

- Input: Enter the membership number and password provided.
- Output : User will be able to use the features of software.

2. Manage books by user.

1. : Books issued

- Description : List of books will be displaced along with data of return.

2. **: Search**

- Input : Enter the name of author's name of the books to be issued.
- Output : List of books related to the keyword.

3. **: Issues book**

- State : Searched the book user wants to issues.

- Input : click the book user wants.
- Output : conformation for book issue and apology for failure in issue.
- Processing : if selected book is available then book will be issued else error will be displayed.

4. : Renew book

- State : Book is issued and is about to reach the date of return.
- Input : Select the book to be renewed.
- Output : conformation message.
- Processing : If the issued book is already reserved by another user then error message will be send and if not then conformation message will be displayed.

5. : Return

- Input : Return the book to the library.
- Output : The issued list will be updated and the returned book will be listed out.

6. : Reserve book

- Input : Enter the details of the book.
- Output : Book successfully reserved.
- Description : If a book is issued by someone then the user can reserve it ,so that later the user can issue it.

7. :Fine

- Input : check for the fines.
- Output : Details about fines on different books issued by the user.
- Processing : The fine will be calculated, if it crossed the date of return and the user did not renewed if then fine will be applied by Rs 10 per day.

3. **Manage book by librarian**

3.1 :Update details of books:

3.1.1 Add books

- Input : Enter the details of the books such as names ,author ,edition, quantity.
- Output : confirmation of addition.

3.1.2 Remove books

- Input : Enter the name of the book and quantity of books.
- Output : Update the list of the books available.

- **Conclusion :**

After performing this experiment, students are now able to gather any information on any software system.

Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-04
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	**Requirement Specification**				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

- **Objective of the Experiment**: Understanding of SRS preparation for any specified project.
- **Outcome:** Must be able to write Specification Document.
- **Hardware & Software Specification :**
  Operating System – Ubuntu 18.04.3 LTS
  Processor – Intel Core i3 7th Gen.
  RAM – 4G
- **Theory :**

What is Software Requirement Specification – (SRS)

A **System Requirements Specification (SRS)** (also known as a Software Requirements Specification) is a document or set of documentation that describes the features and behaviour of a system or software application. It includes a variety of elements (see below) that attempts to define the intended functionality required by the customer to satisfy their different users.

A good SRS defines the how Software System will interact with all internal modules, hardware, communication with other programs and human user interactions with wide range of real life scenarios. Using the Software requirements specification (SRS) document on QA lead, managers creates test plan. It is very important that testers must be cleared with every detail specified in this document in order to avoid faults in test cases and its expected results.
It is highly recommended to review or test SRS documents before start writing test cases and making any plan for testing. Let's see how to test SRS and the important point to keep in mind while testing it.

Qualities of SRS:

- Correct
- Unambiguous
- Complete
- Consistent
- Ranked for importance and/or stability
- Verifiable
- Modifiable
- Traceable

An SRS is correct if every requirement included in the SRS represents something required in the final system. An SRS is complete, if everything the software is supposed to do and the responses of the software to all classes of input data are specified in the SRS. Correctness ensures that what is specified is done correctly, completeness ensures that everything is indeed specified.

An SRS is unambiguous if and only if every requirement stated has one and only one interpretation. Requirements are often written in natural language, which are inherently ambiguous.

An SRS is verifiable if and only if every stated requirement is verifiable. A requirement is verifiable if there exists some cost-effective process that can check whether the final software meets that requirement. An SRS is consistent if there is no requirement that conflicts with another.

Terminology can cause inconsistencies; for example, different requirements may use different terms to refer to the same object. All the requirements for software are not of equal importance. Some are critical, others are important but not critical, and there are some, which are desirable, but not very important. An SRS is ranked for importance and the stability of the requirement are indicated. Stability of requirement reflects the chances of it changing in future. An SRS is traceable if the origin of each of its requirements is clear and if it facilitates the referencing of each requirement in future development. Forward traceability means that each requirement should be traceable to some design and code elements. Backward traceability requires that it be possible to trace design and code elements to the requirements they support. Traceability aids verification and validation.

Why Use an SRS Document?

A software requirements specification is the basis for your entire project. It lays the framework that every team involved in development will follow.

It’s used to provide critical information to multiple teams — development, quality assurance, operations, and maintenance. This keeps everyone on the same page.

Using the SRS helps to ensure requirements are fulfilled. And it can also help you make decisions about your product’s lifecycle — for instance, when to retire a feature.

Writing an SRS can also minimize overall development time and costs. Embedded development teams especially benefit from using an SRS.

Software Requirements Specification vs. *System* Requirements Specification

A **software requirements specification (SRS)** includes in-depth descriptions of the software that will be developed.

A **system requirements specification (SyRS)** collects information on the requirements for a system.

"Software" and "system" are sometimes used interchangeably as SRS. But, a software requirement specification provides greater detail than a system requirements specification.

Template for SRS

Table of Contents

Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-05
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	**UML Diagram**				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

- **Objective :** Understanding processes for designing a software system.
- **Outcome :** Must be able to translate a specification into a design.
- **Hardware & Software Specification :**
  Operating System – Ubuntu 18.04.3 LTS
  Processor – Intel Core i3 7th Gen.
  RAM – 4GB
- **Theory :** Unified Modeling Language (UML) is a general purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite similar to blueprints used in other fields of engineering. UML is not a programming language, it is rather a visual language.
  - **Use Case Specification –** Use case diagrams are used to depict the functionality of a system or a part of the system. They are widely used to illustrate the functional requirements of the system & its interaction with external agent (actors). A use case is basically a diagram representing different scenarios where the system can be used. A use case diagram gives us a high level view of what the system or a part of the system does without going to implementations details.

1. **Use Case Diagram Library System Project –**

   In the library system project there are two users librarian and student. Both librarian and student can do all activities after login in to library management system.

   Librarian Activity –
   - Add publication.

- Add books.
- Add branch.
- Add student.
- Issue book.
- Return book.
- Apply Penalty.
- Change Password.
- View Report.

Student Activity –

- Search book
- Issue/return book report.
- Penalty report.
- Change Password.

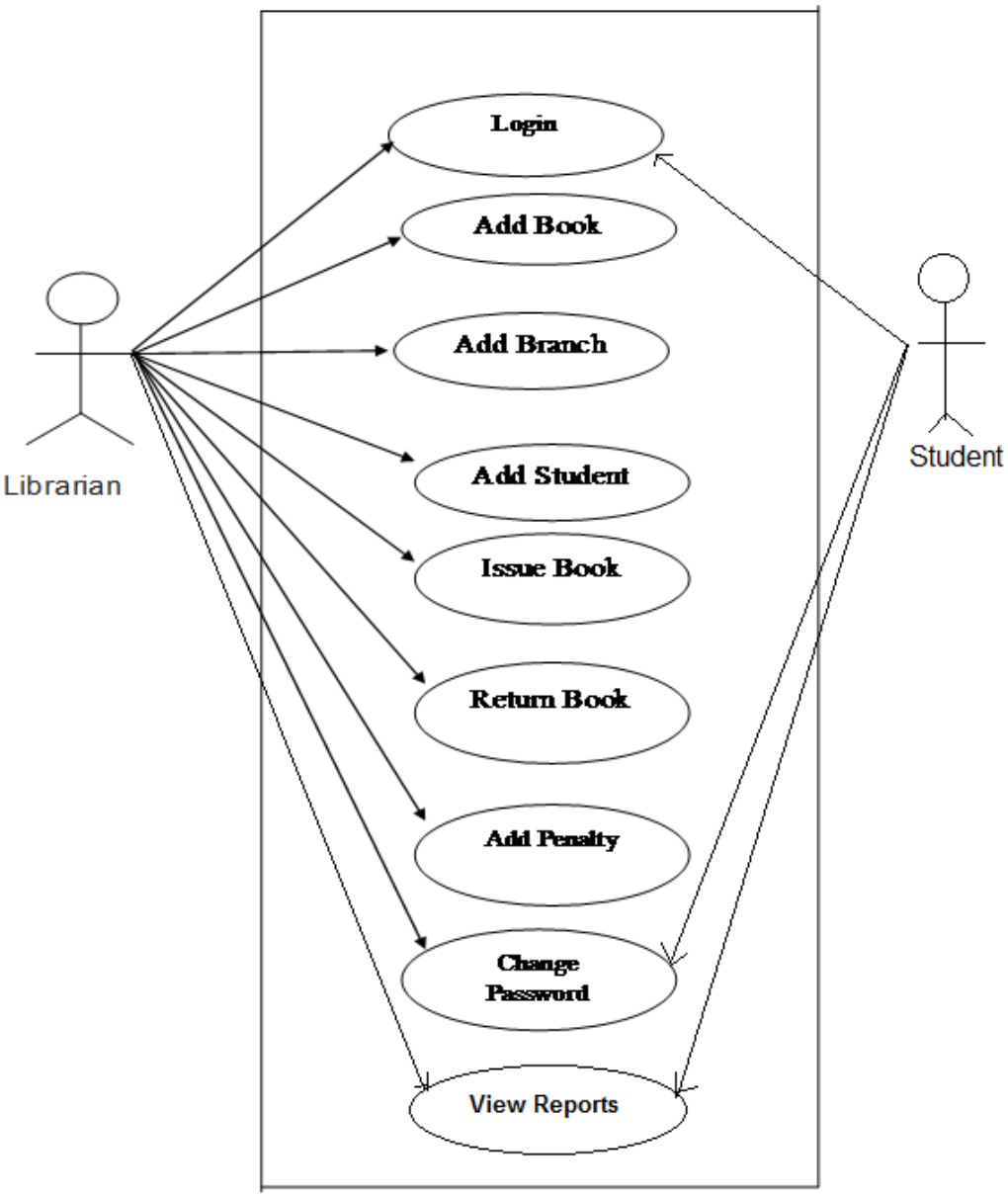


Fig.1. Use Case – Library Management System

2. **UML Class Diagram for library management contains classes such as –**
   - Book Class
   - Librarian Class
   - Transaction Class
   - Member record class
   - Bill class

Each class contains various attributes & methods (functions) which call other class attributes to share data.

1. Book class contains attributes such as author, book name, price status, rack number, edition, & function such as display book details, update status.
2. Librarian class contains attributes such as name, password & functions such as search book, issue book, calculate fine, calculate bill.
3. Transaction class contains attributes such as transaction ID, member ID, bill ID, date of issue, etc. & functions such as create transaction, delete transaction.
4. Member record class contains attributes such as member ID, type, date of issue, no. of books, etc. &functions such as increase book issue, decrease book issue, pay bill.
5. Bill class contains attributes such as bill no., date, member, amount & functions such as create bill, update bill.

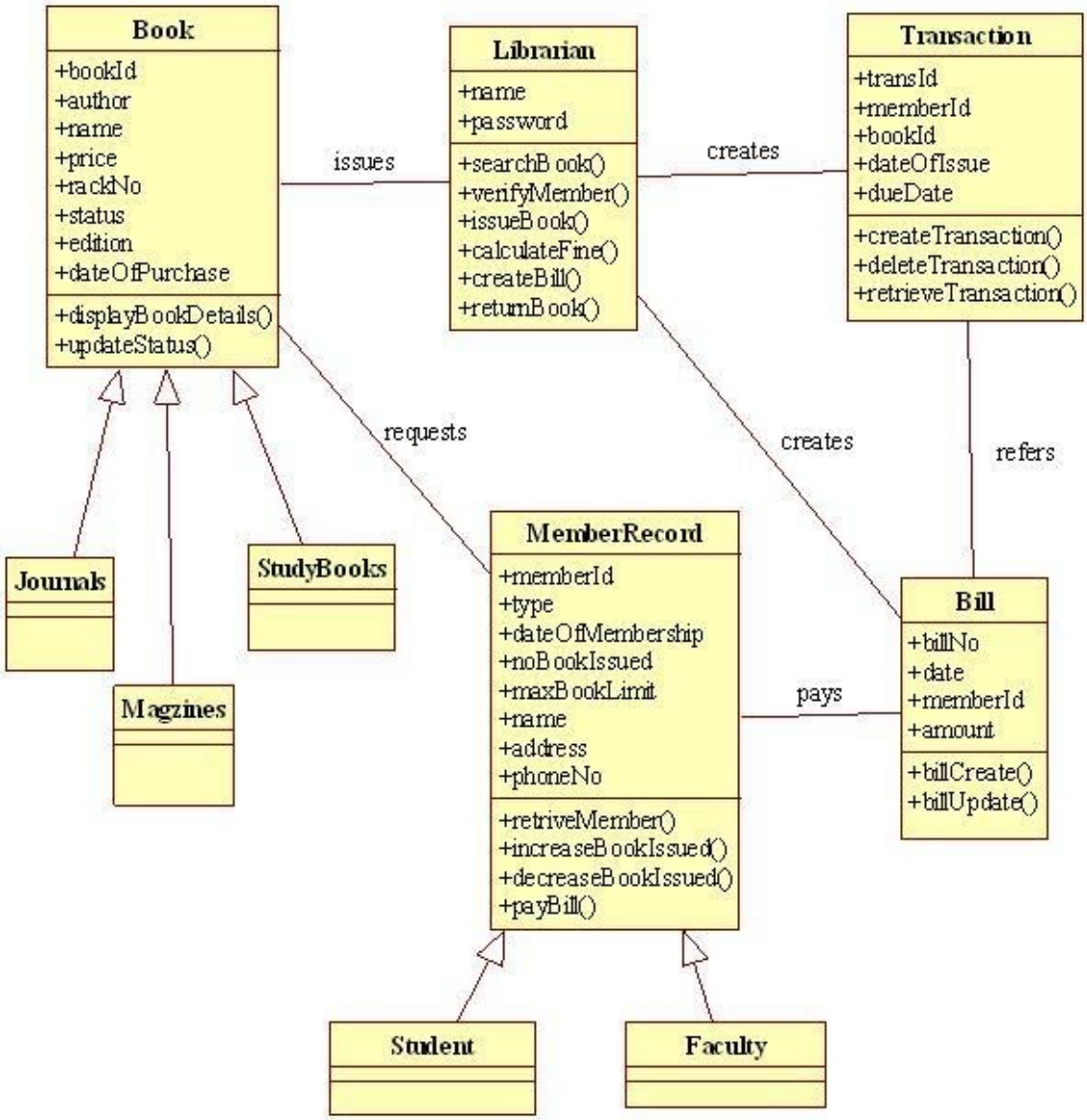


Fig.2. UML Class Diagram – Library Management System

3. UML Activity diagram for library management system –

Librarian is a responsible person who run the system, is an administrator of the whole system. Librarian has a full
Rights to handle the project.

Librarian Functionalities :

Here are list of activities of librarian –

- Add publication.
- Add book stock.
- Add branch.
- Add student.
- Issue books.
- Return books.
- Penalty.

The student is a registered member of library system. All students has unique username & password to access his or her account details, they can see the borrowed book report & penalty report.

Student functionalities :

- Book Reports.
- Account.
- Penalty status.

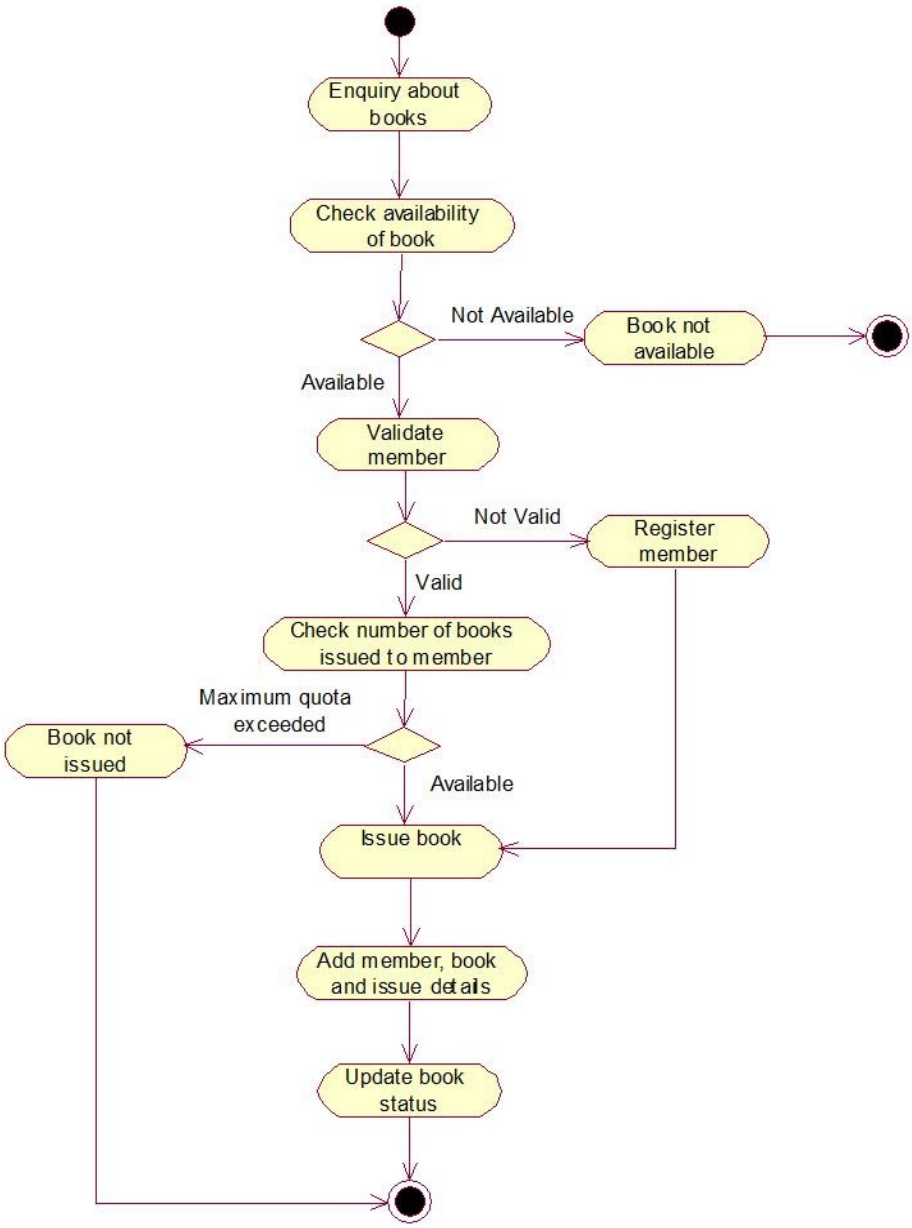


Fig.3. Activity Diagram – Library Management System

4. Sequence diagram for library management system –

Librarian issues a book to member – Whenever the member asks for a book, the librarian checks the availability of book, if the book is available then the rack number of that book will be returned to librarian. Librarian then checks the validity of member by verifying the library card, if member is valid then the number of books issued to him is less than maximum allowed books, then after the new books is issued to him and transaction completed. Librarian then update the number of books issued to the member & status of book.

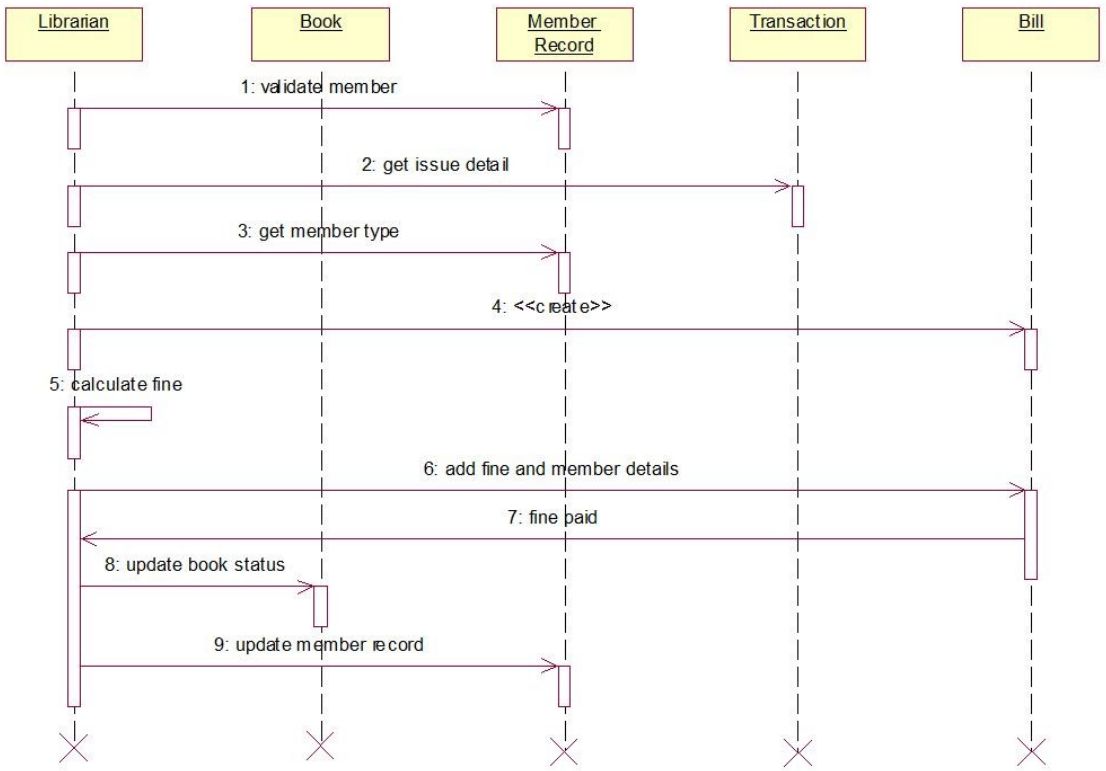


Fig.4. Sequence Diagram – Library Management System

Conclusion – Students are able to translate specifications into a design, i.e. now students can draw designs with the given SRS for any project.

Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-06
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	UML Diagram.				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

Objective of the Experiment - To understand designing of ER diagram.

Problem Statement - Developer wants to know how many entities are involved, what are their respective attributes and relationship between the entities for implementation phase. Draw the ER diagram for the respective project.

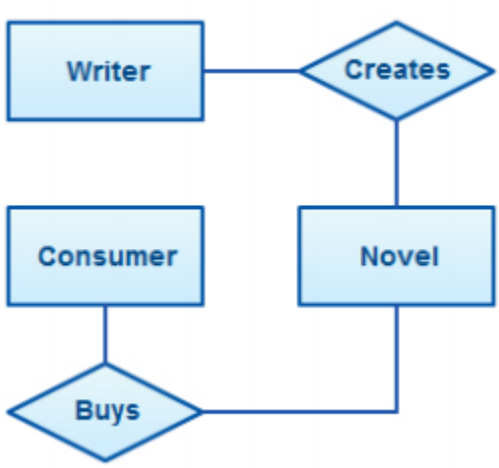
Outcome of the Experiment - Must be able to design ER diagram.

Description - UML is an acronym that stands for Unified Modeling Language. Simply put, UML is a modern approach to modeling and documenting software. In fact, it's one of the most popular business process modeling techniques. It is based on diagrammatic representations of software components. As the old proverb says: "a picture is worth a thousand words". By using visual representations, we are able to better understand possible flaws or errors in software or business processes.

Mainly, UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modeling

workflows as well as a wider range of features to improve readability and efficacy.

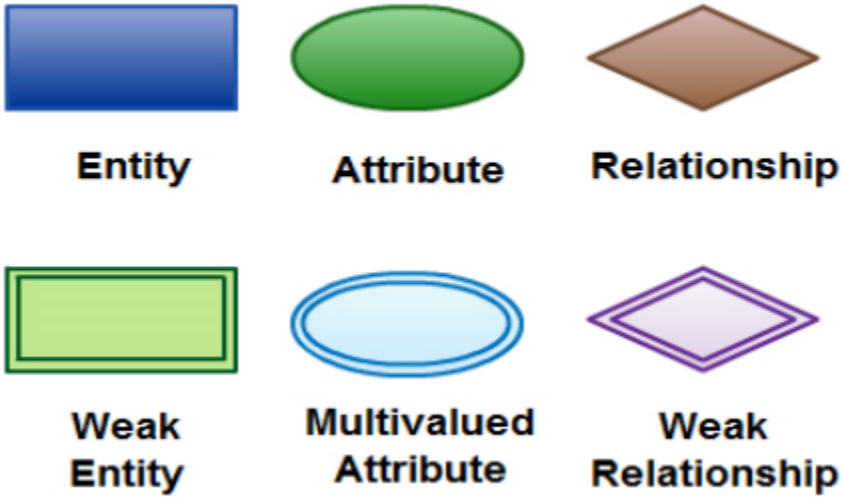
ER Diagram - Entity Relationship Diagrams are a major data modelling tool and will help organize the data in your project into entities and define the relationships between the entities. This process has proved to enable the analyst to produce a good database structure so that the data can be stored and retrieved in a most efficient manner.



Entity - A data entity is anything real or abstract about which we want to store data. Entity types fall into five classes: roles, events, locations, tangible things or concepts. E.g. employee, payment, campus, book. Specific examples of an entity are called instances. E.g. the employee John Jones, Mary Smith's payment, etc.

Relationship - A data relationship is a natural association that exists between one or more entities. E.g. Employees process payments. Cardinality defines the number of occurrences of one entity for a single occurrence of the related entity. E.g. an employee may process many payments but might not process any payments depending on the nature of her job.

Attribute - A data attribute is a characteristic common to all or most instances of a particular entity. Synonyms include property, data element, field. E.g. Name, address, Employee Number, pay rate are all attributes of the entity employee. An attribute or combination of attributes that uniquely identifies one and only one instance of an entity is called a primary key or identifier. E.g. Employee Number is a primary key for Employee.



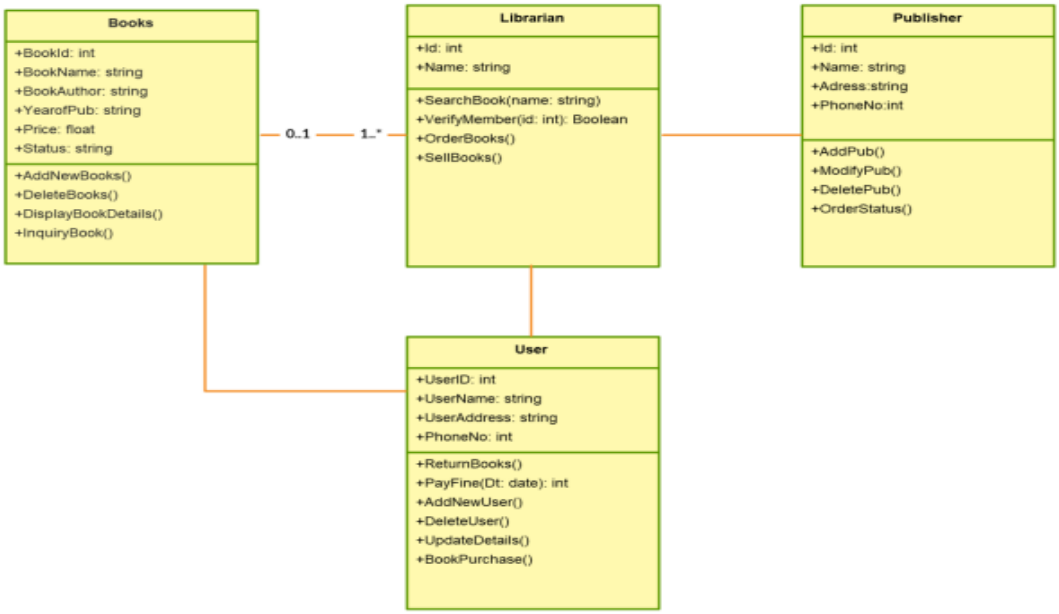
How to Draw ER Diagrams -

Below points show how to go about creating an ER diagram.

1. Identify all the entities in the system. An entity should appear only once in a particular diagram. Create rectangles for all entities and name them properly.

2. Identify relationships between entities. Connect them using a line and add a diamond in the middle describing the relationship.

3. Add attributes for entities. Give meaningful attribute names so they can be understood easily.



Benefits of ER diagrams -

ER diagrams constitute a very useful framework for creating and manipulating databases. First, ER diagrams are easy to understand and do not require a person to undergo extensive training to be able to work with it efficiently and accurately. This means that designers can use ER

diagrams to easily communicate with developers, customers, and end users, regardless of their IT proficiency. Second, ER diagrams are readily translatable into relational tables which can be used to quickly build databases. In addition, ER diagrams can directly be used by database developers as the blueprint for implementing data in specific software

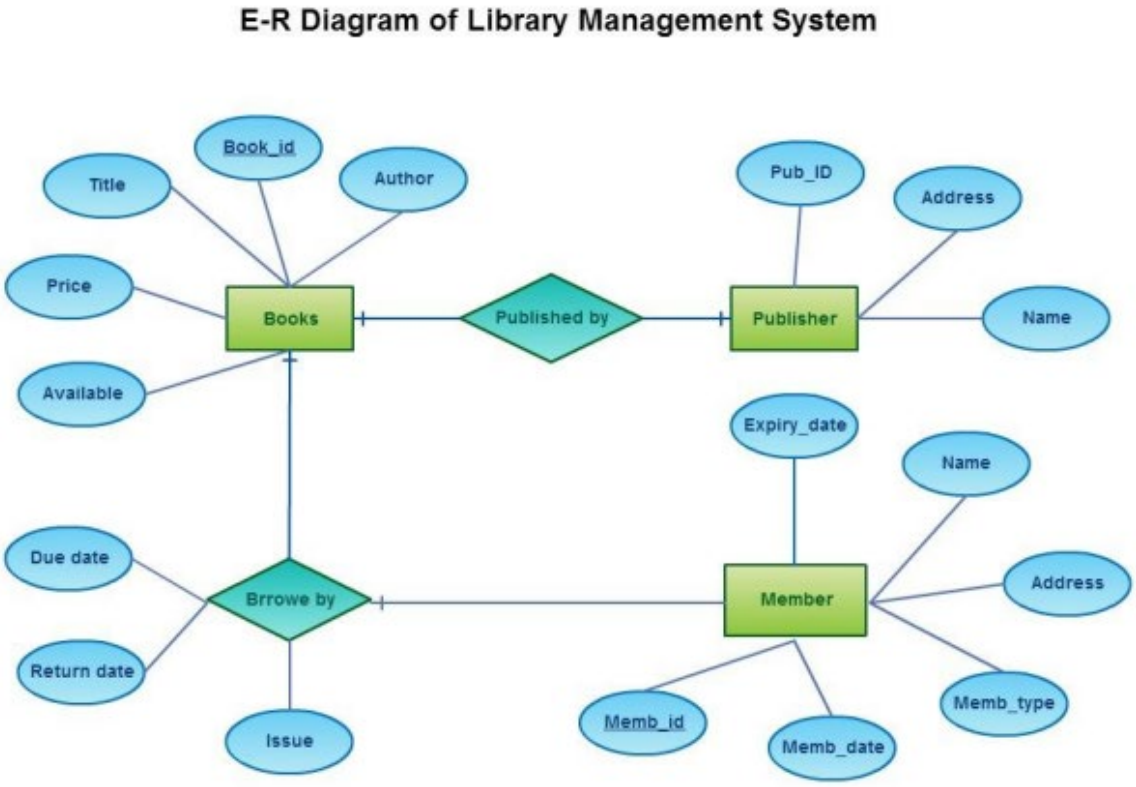
applications. Lastly, ER diagrams may be applied in other contexts such as describing the different relationships and operations within an organization.

Disadvantages of E-R Data Model -

Following are disadvantages of an E-R Model:

1. No industry standard for notation: There is no industry standard notation for developing an E-R diagram.

2. Popular for high-level design: The E-R data model is especially popular for high level.



Result - Hence the ER Diagram is designed.

Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-07
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	**Architectural Styles**				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

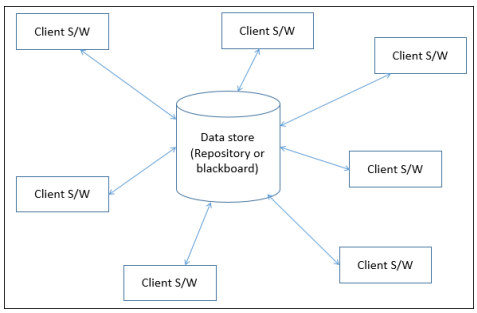
Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

- **Objective :** To know about type of Architectural Designs Methodologies.
- **Outcome of the Experiment :** Competent to identify the components to build the Architecture.
- **Theory :**The software needs the architectural design to represents the design of software. IEEE defines architectural design as "the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system."

Different types of Architectural styles

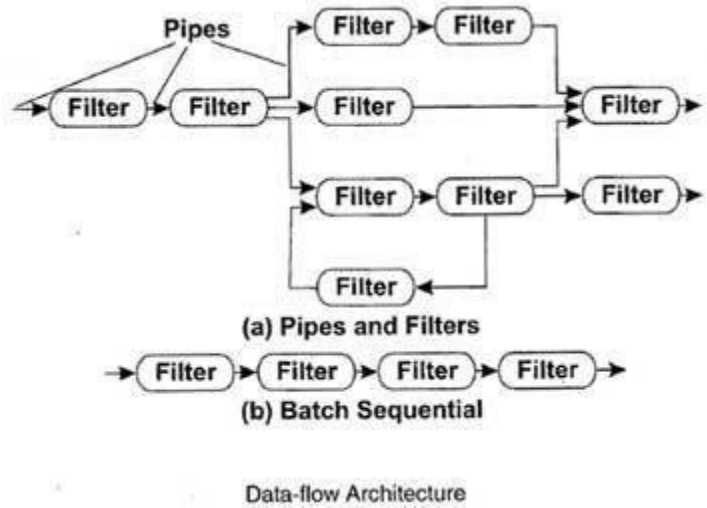
Data centred architectures:

1. A data store will reside at the center of this architecture and is accessed frequently by the other components that update, add, delete or modify the data present within the store.
2. The figure illustrates a typical data centered style. The client software access a central repository. Variation of this approach are used to transform the repository into a blackboard when data related to client or data of interest for the client change the notifications to client software.
3. This data-centered architecture will promote integrability. This means that the existing components can be changed and new client components can be added to the architecture without the permission or concern of other clients.
4. Data can be passed among clients using blackboard mechanism.



Data flow architectures:

5. This kind of architecture is used when input data to be transformed into output data through a series of computational manipulative components.
6. The figure represents pipe-and-filter architecture since it uses both pipe and filter and it has a set of components called filters connected by pipes.
7. Pipes are used to transmit data from one component to the next.
8. Each filter will work independently and is designed to take data input of a certain form and produces data output to the next filter of a specified form. The filters don't require any knowledge of the working of neighbouring filters.
9. If the data flow degenerates into a single line of transforms, then it is termed as batch sequential. This structure accepts the batch of data and then applies a series of sequential components to transform it.



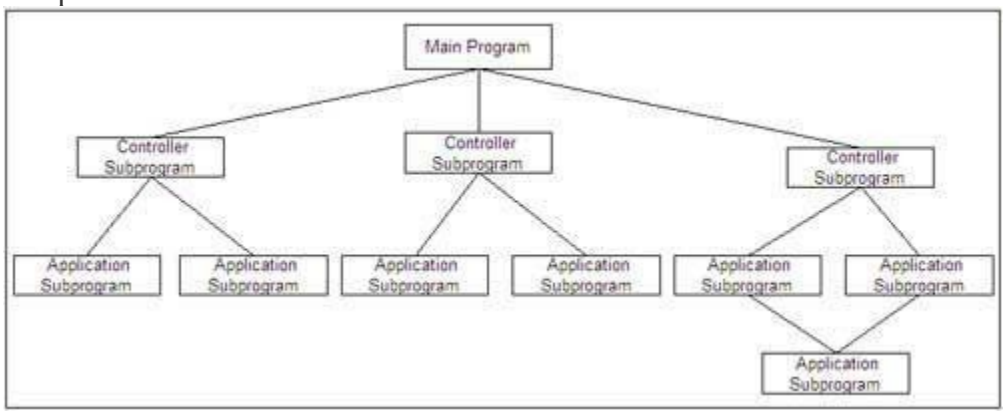
Data-flow Architecture

Call and Return architectures:

It is used to create a program that is easy to scale and modify. Many sub-styles exist within this category. Two of them are explained below.

- **Remote procedure call architecture:** This components is used to present in a main program or sub program architecture distributed among multiple computers on a network.
- **Main program or Subprogram architectures:** The main program structure decomposes into number of subprograms or function into a control hierarchy. Main program contains number of subprograms that can invoke other

components.

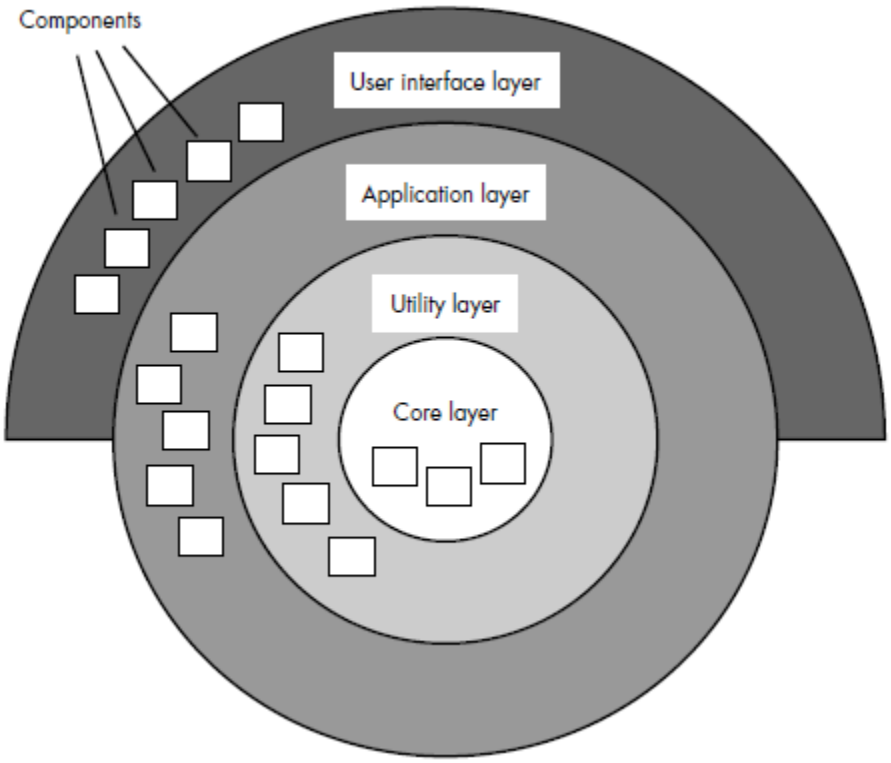


Object Oriented architecture:

The components of a system encapsulate data and the operations that must be applied to manipulate the data. The coordination and communication between the components are established via the message passing.

Layered architecture:

- A number of different layers are defined with each layer performing a well-defined set of operations. Each layer will do some operations that becomes closer to machine instruction set progressively.
- At the outer layer, components will receive the user interface operations and at the inner layers, components will perform the operating system interfacing(communication and coordination with OS)
- Intermediate layers to utility services and application software functions.



Exercise- Choose one of the architecture to build/implement your library management system. Explain why you have choosen that particular style? What are the advantages of that particular style.

Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-08
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	Implementation				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

Objective of the Experiment - To implement the previous designed diagrams.

Problem Statement - After the design phase apply implementation phase over the designs of library management system. Divide the design into modules and implement the modules, find the dependent modules.

Outcome of the Experiment - Must be able to implement the design modules.

Description - The various modules of this system areas:

User Module: This module is further divided into various sub-modules describing the user in a better way:

New user register: To sign up a new user to this system.

Student Login: So as to confirm that only an authenticated user is using the project.

Search book: The user can search book based on book id, book name, or by author name.

Issue Book: To help the user get the required books issued.

Return Book: To return the book before the last date without fine, or after the specified time duration with a late fine.

Admin Module: It is to be operated by the admin with unique id and password. The admin is the person who decides authentication and authorization for all the different users of the application. It further can be subdivided as:

Register user:

Issue Book: Maintain books in a stack, means record the availability at regular time interval.

Librarian: Includes all the library staff who are required to enter the records in the system and keep an eye on the various activities like the issue of the book, the return of the book, non-availability of books etc. through the developed system. Library: The main part of the organization for which this application has been designed. It has attributes like:

Name: The name of the library to distinguish it from all the libraries available in any campus, uniquely.

Address: This defines the address of the library as such the block number or lane number etc.

Books: These are the basic building block of this system as well as any library. In other words the main purpose of any library and the cause to develop systems like this.

Book_Name: The name of the book which is almost unique in some way.

Book_Code: A number to use for sorting and arranging the book, as well as identifying it in the library.

Author: The one who has written the book. As sometimes the book's series become more popular by the author's name rather than the book name.

Price: The market value of the book is also required to maintain in the record, as sometimes it is needed to arrange and sort based on this, secondly, it is also required for compensation in case of loss or damage, as fine charges.

Quantity: This is to indicate the availability of each book individually, so as to know whether last copy should be issued or kept as a reference piece. Also to maintain the number of books.

Rack_No: To get the exact location of the book, so as it becomes easy to search it and sort it at the time of binding up work.

Subject_Code: As there are various further division and subcategories of any subject. So, in that case, this is the unique id to distinguish the books, arrange them, and sort them. Like in computer science there are further many specialities like core java, advanced java, HTML, html5 etc.

User: The next is the beneficiary, by whom the library is being accessed and who serves as a purpose for this system. Its attributes include:

Name: The name of the student or teacher, who will get the book issued, or who will return the book.

Id: The user's unique college or university roll number i.e. the id. The same is applicable to teachers also, with their unique id.

Address: This refers to the user's physical area of residence. It is a composite attribute. As it further contains the house number and lane number.

Fine_Amount: To indicate the amount of fine he/she has to deposit and keep it up to date so that he/she is aware of the payment to be made at the end of the year or session.

Issue Status: It makes to the notice of the librarian as well as to the student or teacher that ow many books they have already got issued and how much more can they get at the current point of time. It includes attributes as:

Book_Name: The name of the book which is almost unique in some way.

Book_Code: A number to use for sorting and arranging the book, as well as identifying it in the library.

Id: The user's unique college or university roll number i.e. the id. The same is applicable to teachers also, with their unique id. To know which user has been issued the book and for what time limit, that is what time the user is supposed to return the book, and if not will be charged fine.

Date_Issue: The date on which user got the book issued to read from it.

Return_Date: It indicates the date on which user is supposed to be returning the book, that is it is the date after the duration completed for which the user has been issued the book.

Return Status: This tells the library management authority about the status of returned books per user. Whether a particular user has returned the book or not, on or before the last date. If not, in that case, the fine will be charged from him/her as a penalty for late submission.

Result - Hence, students are able to implement design modules.

Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-09
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	Testing.				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

Title - Testing.

Objective of the Experiment - Understanding how testing id done.

Problem Statement - For the library management system of your college after the implementation phase applying all testing methods and document all the respective test cases.

Outcome of the Experiment - Must be able to do different testing approaches.

Description - There are some basic and essential software testing steps every software developer should perform before showing someone else their work, whether it's for shift-left testing, formal testing, ad hoc testing, code merging and integration, or just calling a colleague over to take a quick look. The goal of this basic testing is to detect the obvious bugs that jump out immediately. Otherwise, you get into an expensive and unnecessary cycle of having to describe the problem to the developer, who then has to reproduce it, debug it, and solve it, before trying again.

Here are the essential software testing steps every software engineer should perform before showing their work to someone else.

1. Basic functionality testing - Begin by making sure that every button on every screen works. You also need to ensure that you can enter simple text into each field without crashing the software. You don't have to try out all the different combinations of clicks and characters, or edge conditions, because that's what your testers do—and they're really good at that. The goal here is this: don't let other people touch your work if it's going to crash as soon as they enter

their own name into the username field. If the feature is designed to be accessed by way of an API, you need to run tests to make sure that the basic API functionality works before submitting it for more intensive testing. If your basic functionality testing detects something that doesn't work, that's fine. Just tell them that it doesn't work, that you're aware of it, and that they shouldn't bother trying it. You can fix it later, just don't leave any surprises in there.

2. Code review - Another pair of eyes looking at the source code can uncover a lot of problems. If your coding methodology requires peer review, perform this step before you hand the code over for testing. Remember to do your basic functionality testing before the code review, though.

3. Static code analysis - There are tools that can perform analysis on source code or bytecode without executing it. These static code analysis tools can look for many weaknesses in the source code, such as security vulnerabilities and potential concurrency issues. Use static code analysis tools to enforce coding standards, and configure those tools to run automatically as part of the build.

4. Unit testing - Developers will write unit tests to make sure that the unit (be it a method, class, or component) is working as expected and test across a range of valid and invalid inputs. In a continuous integration environment, unit tests should run every time you commit a change to the source code repository, and you should run them on your development machine as well. Some teams have coverage goals for their unit tests and will fail a build if the unit tests aren't extensive enough. Developers also work with mock objects and virtualized services to make sure their units can be tested independently. If your unit tests fail, fix them before letting someone else use your code. If for any reason you can't fix them right now, let the other person know what has failed, so it won't come as a surprise when they come across the problem.

5. Single-user performance testing - Some teams have load and performance testing baked into their continuous integration process and run load tests as soon as code is checked in. This is particularly true for back-end code. But developers should also be looking at single-user performance on the front end and making sure the software is responsive when only they are using the system. If it's taking more than a few seconds to display a web page taken from a local or emulated (and therefore responsive) web server, find out what client-side code is slowing things down and fix it before you let someone else see it.

6. Finding the right balance - Make time to run as many of these tests as possible before you hand your code over to anyone else, because leaving obvious bugs in the code is a waste of your time and your colleagues' time. Of course, you'll need to find the balance between writing code vs. testing that suits you. "Here's the mix that worked for me," said Igor Markov, LoadRunner R&D Manager at HP Software. "40 percent of my time is spent designing and writing code; 5 percent is spent on code review and static code analysis; 25 percent on unit testing and integration testing; and 30 percent on basic functionality testing and single user performance testing," Leaving obvious bugs in the code isn't going to do your reputation any good either. "A developer who doesn't find the obvious defects is never going to shine," continued Markov. "Developers need to produce working software that's easy to use.

Result - Hence, Different Testing appoarches are studied.

Shivajirao Kadam Institute of Technology & Management, Indore

Computer Science & Engineering Department

Experiment-10
Laboratory Performance Evaluation

Academic Session:	Jan-June 2021	Semester	4th	Batch	
Name of Lab:	Software Engineering	Course Code		CS-403	
Name of Experiment	Testing.				**CO No.**
Name of Student	**Kartik Kulshreshtha**	Enrolment No		**0875CS191048**	

Date of Experiment	**Date of Submission**
01/06/2021	02/06/2021

Grade and remark by the tutor	**Score (0-10)**	**Remark / Reason**
Clarity about the Objective and Outcome of experiment		
2. Submitted the work in desired format		
3. Shown capability to solve the problems		
Average (out of 10)		

Title - Maintainence.

Objective of the Experiment - To know how to software product ismaintained after deployment.

Problem Statement - Use different methods to maintain your library management software. Use different types of maintainence methods.

Outcome of the Experiment - Must be able to understand how software is maintained.

Description - Software maintenance is widely accepted part of SDLC now

a days. It stands for all the modifications and updations done after the delivery of software product. There are number of reasons, why modifications are required, some of them are briefly mentioned below:

Market Conditions - Policies, which changes over the time, such as taxation and newly introduced constraints like, how to maintain bookkeeping, may trigger need for modification.

Client Requirements - Over the time, customer may ask for new featuresor functions in the software.

Host Modifications - If any of the hardware and/or platform (such as operating system) of the target host changes, software changes are needed to keep adaptability.

Organization Changes - If there is any business level change at client end, such as reduction of organization strength, acquiring another company, organization venturing into new business, need to modify in the original software may arise.

Types of maintenance -

In a software lifetime, type of maintenance may vary based on its nature. It may be just a routine maintenance tasks as some bug discovered by some user or it may be a large event in itself based on maintenance size or nature. Following are some types of maintenance based on their characteristics:

- **Corrective Maintenance -** This includes modifications and updations done in order to correct or fix problems, which are either discovered by user or concluded by user error reports.

- **Adaptive Maintenance -** This includes modifications and updations applied to keep the software product up-to date and tuned to the ever changing world of technology and business environment.

- **Perfective Maintenance -** This includes modifications and updates done in order to keep the software usable over long period of time. It includes new features, new user requirements for refining the software and improve its reliability and performance.

- **Preventive Maintenance -** This includes modifications and updations to prevent future problems of the software. It aims to attend problems, which are not significant at this moment but may cause serious issues in future.

- **Cost of Maintenance -** Reports suggest that the cost of maintenance is high. A study on estimating software maintenance found that the cost of maintenance is as high as 67% of the cost of entire software process cycle.

Test Cases For Library Management System - The test cases for library management system is an application that explains the test cases for library management system.

- **Login form:** The test cases involved are whether valid password and name are entered or invalid name and password entered.
- **Book entry form:** The test cases included are-on the click of add button, delete button, update button, search button, clear button, exit button and next button.
- **User account form:** The test cases included are-on the click of add button, delete button, update button, search button, clear button, exit button and next button.
- **Book return form:** The test cases included are-on the click of add button, delete button, update button, search button, clear button, exit button and next button.

LOGIN FORM:

S.N.	Test Case	Excepted Result	Test Result
1.	Enter valid name and password & click on login button	Software should display main window	Successful
2.	Enter invalid	Software should not display main window	Successful

BOOK ENTRY FORM:

S.N.	Test Case	Excepted Result	Test Result
1.	On the click of ADD button	At first user have to fill all fields with proper data , if any Error like entering text data	Successful

		instead of number or entering number instead of text..is found then it gives proper message otherwise Adds Record To the Database	
2.	On the Click of DELETE Button	This deletes the details of book by using Accession no.	Successful
3.	On the Click of UPDATE Button	Modified records are Updated in database by clicking UPDATE button.	Successful
4.	On the Click of SEARCH Button	Displays the Details of book for entered Accession no. Otherwise gives proper Error message.	Successful
5.	On the Click of CLEAR Button	Clears all fields	Successful
6.	On the Click of EXIT Button	Exit the current book details form	Successful

USER ACCOUNT FORM:

S.N.	**Test Case**	**Excepted Result**	**Test Result**
1.	On the click of ADD button	At first user have to fill all fields with proper data , if any Error like entering text data instead of number or	Successful

		entering number instead of text..is found then it gives proper message otherwise Adds Record To the Database	
2.	On the click of DELETE button	This deletes the details of student by using Register no.	Successful
3.	On the click of UPDATE button	Modified records are Updated in database by clicking UPDATE button.	Successful
4.	On the click of SEARCH button	Displays the Details of book for entered Register no. Otherwise gives proper Error message.	Successful
5.	On the click of CLEAR button	Clears all fields	Successful
6.	On the click of EXIT button	Exit the current book details form	Successful

BOOK ISSUE FORM:

S.N.	**Test Case**	**Excepted Result**	**Test Result**
1.	On the click of ADD button	At first user have to fill all fields with proper data ,if the accession number book is already issued then it will giving proper msg.	Successful

2.	On the click of DELETE button	This deletes the details of book by using Register no.	Successful
3.	On the click of UPDATE button	Modified records are Updated in database by clicking UPDATE button.	Successful
4.	On the click of SEARCH button	Displays the Details of issued book..Otherwise gives proper Error message.	Successful
5.	On the click of CLEAR button	Clears all fields	Successful
6.	On the click of EXIT button	Exit the current book details form	Successful

BOOK RETURN FORM:

S.N.	**Test Case**	**Excepted Result**	**Test Result**
1.	On the click of ADD button	At first user have to fill all fields with proper data , if any Error like entering text data instead of number or entering number instead of text..is found then it gives proper message otherwise Adds Record To the Database	Successful

2.	On the click of DELETE button	Which deletes the details of book by using Register no.	Successful
3.	On the click of UPDATE button	Modified records are Updated in database by clicking UPDATE button.	Successful
4.	On the click of SEARCH button	Displays the Details of returned book … Otherwise gives proper Error message.	Successful
5.	On the click of CLEAR button	Clears all fields	Successful
6.	On the click of EXIT button	Exit the current book details form	Successful

Result - Able to understand how software is maintain.