A MINOR PROJECT REPORT ON

**“Library Management System”**

**Submitted**

*In the partial fulfilment of the requirements for*

*The award of the degree of*

**BACHELOR OF TECHNOLOGY**

In

**COMPUTER SCIENCE & ENGINEERING**

By

K.Naga Chaitanya (171FA04026)

M.Eswar (171FA04036)

Sk. Waseem Galab (171FA04050)

Under the esteemed guidance of

Sri Lakshmi Uppalapati, Assistant Professor

****

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**VIGNAN'S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH**

(**Accredited by NAAC “A” grade**)

**Vadlamudi, Guntur.**

**VIGNAN’S FOUNDATION FOR SCIENCE TECHNOLOGY AND RESEARCH**

**(Accredited by NAAC “A” grade)**

****

**CERTIFICATE**

This is to certify that the Minor project Report entitled “LIBRARY MANAGEMENT SYSTEM” that is being submitted by **K.Naga Chaitanya (171FA04026), M.Eswar (171FA04036), Sk.WaseemGalab (171FA04050)** in partial fulfilment for the award of B.Tech degree in Computer Science and Engineering to the Vignan’s Foundation for Science, Technology and Research, Deemed to be University, is a record of bonafide work carried out by them under my supervision.

Sri Lakshmi Uppalapati   Dr. D. Venkatesulu

    Assistant Professor External Examiner     HOD, CSE

**DECLARATION**

I hereby declare that the project entitled “**Library Management System**” submitted for the DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING. This dissertation is our original work and the project has not formed the basis for the award of any degree, associate-ship and fellowship or any other similar titles and no part of it has been published or sent for publication at the time of submission.

By

K. Naga Chaitanya (171FA04026)

M. Eswar (171FA04036)

Sk. Waseem Galab (171FA04050)

Date:  -11-2019

**ACKNOWLEDGEMENT**

We are very grateful to our beloved Chairman **Dr. Lavu Rathaiah**,and Vice Chairman

**Mr. Lavu Krishna Devarayalu,** for their love and care.

It is our pleasure to extend our sincere thanks to Vice-Chancellor **Dr. M.Y.S. Prasad** for providing an opportunity to do my academics in VFSTR.

It is a great pleasure for me to express my sincere thanks to **Dr. D. Venkatesulu, HOD, CSE** of VFSTR**,** for providing me an opportunity to do my Mini Project.

We feel it our responsibility to thank **Sri Lakshmi Uppalapati** under whose valuable guidance that the project came out successfully after each stage.

We extend our whole-hearted gratitude to all our faculty members of Department of Computer Science and Engineering who helped us in our academics throughout course.

Finally we wish to express thanks to our family members for the love and affection overseas and forbearance and cheerful depositions, which are vital for sustaining effort, required for completing this work.

With Sincere regards,

By

K.Naga Chaitanya (171FA04026)

M.Eswar (171FA04036)

Sk.WaseemGalab (171FA04050)

**Table of Contents**

**1. Introduction**

**1.1 Purpose………………………………………………………………………....... 7**

**1.2 Scope……………………………………………………………………………... 7**

**1.3 Definitions, Acronyms, and Abbreviation……………………………………... 7**

**1.4 Overview…………………………………………………………………………. 8**

**2. Overall Description**

**2.1 Product Prescriptive…………………………………………………………….. 8**

**2.2 Software Requirements…………………………………………………………. 11**

**2.3 Hardware Requirements…................................................................................... 12**

**2.4 Functional Requirement**

**2.4.1 Actors...................................................................................................................... 12**

**2.4.2 Functionalities of actors .......................................................................................12**

**2.5 Non Functional Requirements**

**2.5.1 Security Requirements.......................................................................................... 13**

**2.5.2 Performance Requirements.................................................................................. 13**

**2.5.3 Safety Requirements............................................................................................. 13**

**2.6 User Characteristics..............................................................................................14**

**3. Entity Relationship Diagram................................................................................ 16**

**4. UML Diagrams**

**4.1 Use Case Diagram.................................................................................................. 17**

**4.2 Sequence Diagram................................................................................................. 18**

**4.3 Class Diagram........................................................................................................ 19**

**4.4 Activity Diagram.................................................................................................... 20**

**4.5 State Chart Diagram............................................................................................. 21**

**4.6 Object Diagram..................................................................................................... 21**

**4.7 Component Diagram............................................................................................. 22**

**4.8 Deployment Diagram............................................................................................. 23**

**5. Conclusion ............................................................................................................. 24**

**6. Reference ............................................................................................................... 24**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to present a detailed description of **“Library Management System**. It will explain the purpose and features of the system that it will provide. The document also describes the nonfunctional requirements of the system.

## Scope

This Project is used by two kinds of modules

**1) Librarian Module**

**2) Customer/User Module**

**3) Vendor Module**

1. **Librarian Module**

* A)Customer Module: This student module is under Admin module. In this module Admin can add, update and delete the student
* B)Books Module: In this module Admin can add, delete and update the books.
* C)Issue Of Books Module: In this module Admin will issue books to the students.
* D)Return Of Books Module: In this module issued books will be taken back by admin.

1. **Customer Module**

* In this module student can check the existing CUSTOMERS who are registered , available books and reserved books.
* Borrow Books
* Renew Books
* Return Books
* Pays Fine If any delay in returning

1. **Vendor Module**

* vendor can issue the books to library when they need it.

**1.3 Definitions, Acronyms, and Abbreviations**

• **UML –** Unified Modelling Language

• **JDK –** Java Development Kit

• **JRE –** Java Run time Environment

• **IDE –** Integrated Development Environment

**2. Overall Description**

Describes the general factors that affect the product and its requirements. This section does not state specific requirements. Instead it provides a background for those requirements, which are defined in section 3, and makes them easier to understand.

## 2.1 Product Perspective

## The proposed system is used in the student relationship management by the consultant people. This system will provide the information to students about different aspects. As the sales executives are in move; they may not have internet connectivity. When in offline mode the customer details and the client visit details will be stored in the SQL database of the android smart phone after receiving access to internet, may be after reaching the office WIFI, the customer data is synced to the server. Using the client browser the data can be queried from the Apache server.

### 2.1.1 System Interfaces

### Once the register Event is created, it needs to be storedsomewhere.The System supports development of a SQL database.Once the event is created then the data is stored in the database.

### 2.1.2 User Interfaces

The user interface requirements include the necessary Graphical User Interface that is required forth e user to easily communicate with the system. An efficient and accepted system has a very good Graphical User Interface provided that makes it well accepted and efficient. The admin that work on the system should not bed is gusted with the system appearance for working with the system.

### 2.1.2 Communications Interfaces

The system shall be a standalone product that does not require any communication interfaces.

### 2.1.3 Memory Constraints

Library Management requires a minimum of 400 MB of primary memory and 500 MB for designing the software.

### 2.1.4 Operations

The actions performed by librarian are

* He can add the customer by giving id and password
* He can add, delete, update the customers
* He can add book details, update the books, delete the books
* He can issue the books
* He can take the books

The actions performed by customer are

* He can register for books
* He can take books
* He can renew the books
* He can return the books

### 2.1.5 Site Adaptation Requirements

Argo Uml version 0.34 should be installed and start execution for design and modelling tools.

## 2.2 Product Functions

In software engineering and systems engineering, a functional requirement defines a function of a system or its component ,where a function is described as a specification of behavior between outputs and inputs

This section provides the functional overview of the product. The project will require the HTML and CSS as a front end and at the back end the database MYSQL will be running.

Keeping records of books.

Keeping the records of customers(staff and students).

Checking Availability Of books.

Keeping details about the book renewed

## 2.3 User Characteristics

Educational level of MSMS computer software–Low Experience of MSMS software–None.

Technical Expertise–Little.

**This application is for :**

* Students
* Faculty

## 2.5 Assumptions and Dependenciesfor auditing

The personal information –Id,username,user’srole,password is previously determined for all the users of the Library Management.Access to the system is restricted only to the pre-assigned usernames.higher level users are aware of the restrictions and limitations entitled to all the levels.every action taking place in the system is required to be saved in a separate database, purposes.

## 2.6 Apportioning of Requirements.

The requirements that are delayed during doing this project is database connectivity.

The software requirements are not upto the mark.testers should understand the problem with the software and its designing.

Library Managementdoenst contains any version issues the only issue with this project is database connectivity while implementing the servlets.

# Specific Requirements

This section contains all the software requirements at a level of detail, that when combined with

the system context diagram, use cases, and use case descriptions, is sufficient to enable

designers to design a system to satisfy those requirements, and testers to test that the system

satisfies those requirements.

1. Specific requirements should be stated with all the characteristics of a good SRS
2. correct
3. unambiguous
4. complete
5. consistent
6. ranked for importance and/or stability
7. verifiable
8. modifiable
9. traceable

**correct:**

the software which we are going to design should be free from errors.

**Unambiguous:**

Project should be understandable, easy to rectify the mistakes and confusion free.

**Complete:**

The project should be complete at the time of testing by test engineers.

**Verifiable**: Project should be verified.

## 3.1 External Interfaces

Library Management will use the standard input/output devices for a personal

computer. This includes the following:

* Keyboard
* Mouse
* Monitor
* Printer

**3.1.1 Software Requirements**

* • Argo UML 0.36 (IDE)
* • Jre 1.8.0\_211
* • Windows 7/8/10
* **3.1.2 Hardware Requirements**
* • Intel Pentium/i3/i5/i7 processor.
* • 512 MB RAM or above.
* • 40 GB hard disk space or above

## 3.2 Functions

## 3.2.1 Functional Requirements

**Actor**: An Actor models a type of role played by an entity that interacts with the subject but which is external to the subject.In this application there are three actors. There are Vendor, Librarian, User .

* Vendor:He is the one who provides or vends the books.
* Librarian:He is the one who keeps track of all the books.
* User:These are the persons who uses the application and get their required books.

## 3.3 Performance Requirements

Performance requirements define acceptable response times for system functionality.

1. The load time for user interface screens shall take no longer than ten seconds.
2. The log in information shall be verified within five seconds.
3. Queries shall return results within five seconds.

## 3.4 Logical Database Requirements

The logical database requirements include the retention of the following data elements. This list is not a complete list and is designed as a starting point for development.

## 3.5 Design Constraints

The Library Management shall be a stand-alone system running in a Windows environment. The system shall be developed using HTML, CSS, Javascript.

### 3.5.1 Standards Compliance

There shall be consistency in variable names within the system. The graphical

user interface shall have a consistent look and feel.

**3. Non-Functional Requirements**

### 3.6.1 Reliability

Specify the factors required to establish the required reliability of the software system at time of delivery.

### 3.6.2 Availability

The system shall be available during normal times.

### 3.6.3 Maintainability

The Library Management is being developed inNetbeans and text editors . It shall be easy to maintain.

### 3.6.5 Portability

The Library Management System shall run in any Microsoft Windows environment that

contains Microsoft Access database.

* **Correctness** - extent to which program satisfies specifications, fulfills user’s mission objectives
* **Efficiency** - amount of computing resources and code required to perform function
* **Flexibility** - effort needed to modify operational program
* **Interoperability** - effort needed to couple one system with another
* **Reliability** - extent to which program performs with required precision
* **Reusability** - extent to which it can be reused in another application
* **Testability** - effort needed to test to ensure performs as intended
* **Usability** - effort required to learn, operate, prepare input, and interpret output

## 3.7 Organizing the Specific Requirements

## There are different ways to organize the requirements by knowing the system mode.

### 3.7.1 System Mode

Some systems behave quite differently depending on the mode of operation. When organizing by mode there are two possible outlines. The choice depends on whether interfaces and performance are dependent on mode.

### 3.7.2 User Class

Some systems provide different sets of functions to different classes of users.

### 3.7.3 Objects

Objects are real-world entities that have a counterpart within the system. Associated with each object is a set of attributes and functions. These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

### 3.7.4 Feature

A feature is an externally desired service by the system that may require a sequence of inputs to effect the desired result. Each feature is generally described in as sequence eof stimulus-response pairs.

### 3.7.5 Stimulus

Some systems can be best organized by describing their functions in terms of stimuli.

### 3. 7.6 Response

Some systems can be best organized by describing their functions in support of the generation of a response.

### 3.7.7 Functional Hierarchy

When none of he above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by either common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be use dot show the relationships between and among the functions and data.

## 3.8 Additional Comments

Whenever a new SRS is contemplated, more than one of the organizational techniques given in 3.7 may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

Three are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.

In any of the outlines below, those sections called “Functional Requirement i” may be described in native language, in pseudocode, in a system definition language, or in four subsections titled: Introduction, Inputs, Processing, Outputs.

# 4.Change Management Process

Change management process indicates the changes that are happen in SRS document. We should update the changes of project in SRS document.

# 5.Supporting Information

The supporting information makes the SRS easier to use. It includes:

1. Table of Contents
2. Index
3. Appendices

The Appendices are not always considered part of the actual requirements specification and are not always necessary. They may include:

(a) Sample I/O formats, descriptions of cost analysis studies, results of user surveys

(b) Supporting or background information that can help the readers of the SRS

(c) A description of the problems to be solved by the software.

(d) Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When Appendices are included, the SRS should explicitly state whether or not the Appendices are to be considered part of the requirements.

Tables on the following pages provide alternate ways to structure section 3 on the specific requirements. You should pick the best one of these to organize section 3 requirements

**3. Entity Relationship Diagram (ER Diagram)**

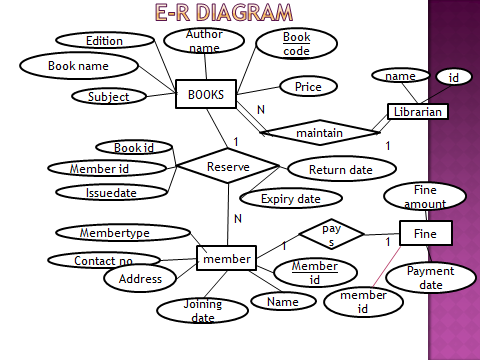


Fig 3.1 Entity Relationship Diagram

**4. UML Diagrams**

**4.1 Use case diagram**

To model a system, the most important aspect is to capture the dynamic behaviour. Dynamic behaviour means the behaviour of the system when it is operating.

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 the 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 interaction.

These internal and external agents are known as actors. Use case diagrams consist 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. Hence to model the entire system, a number of use case diagrams are used.

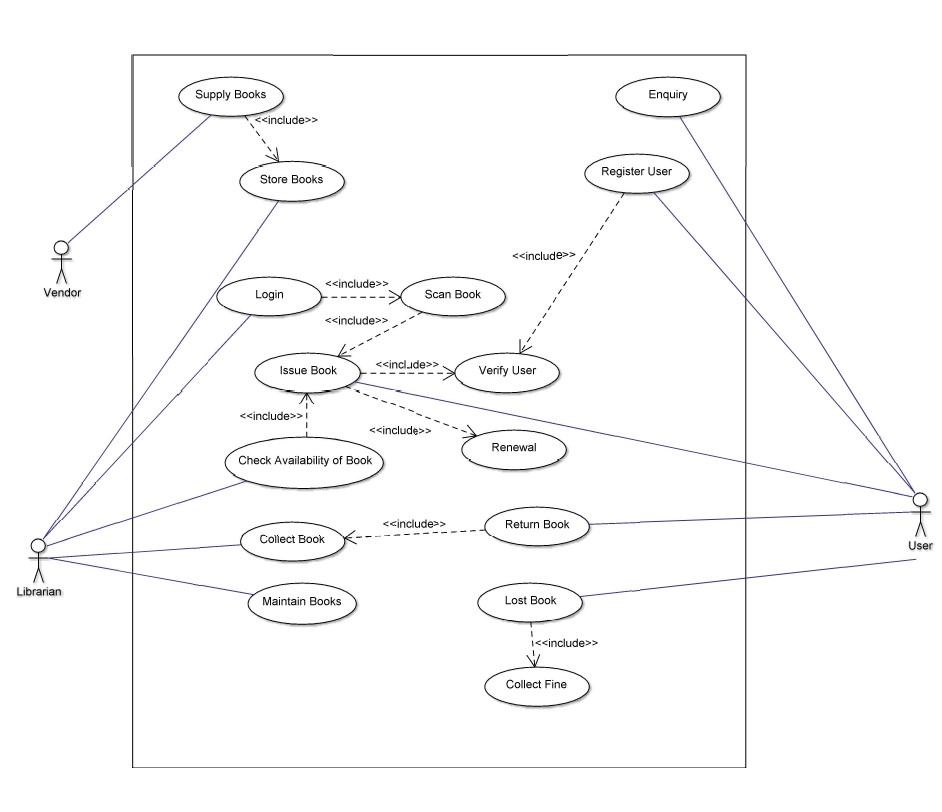
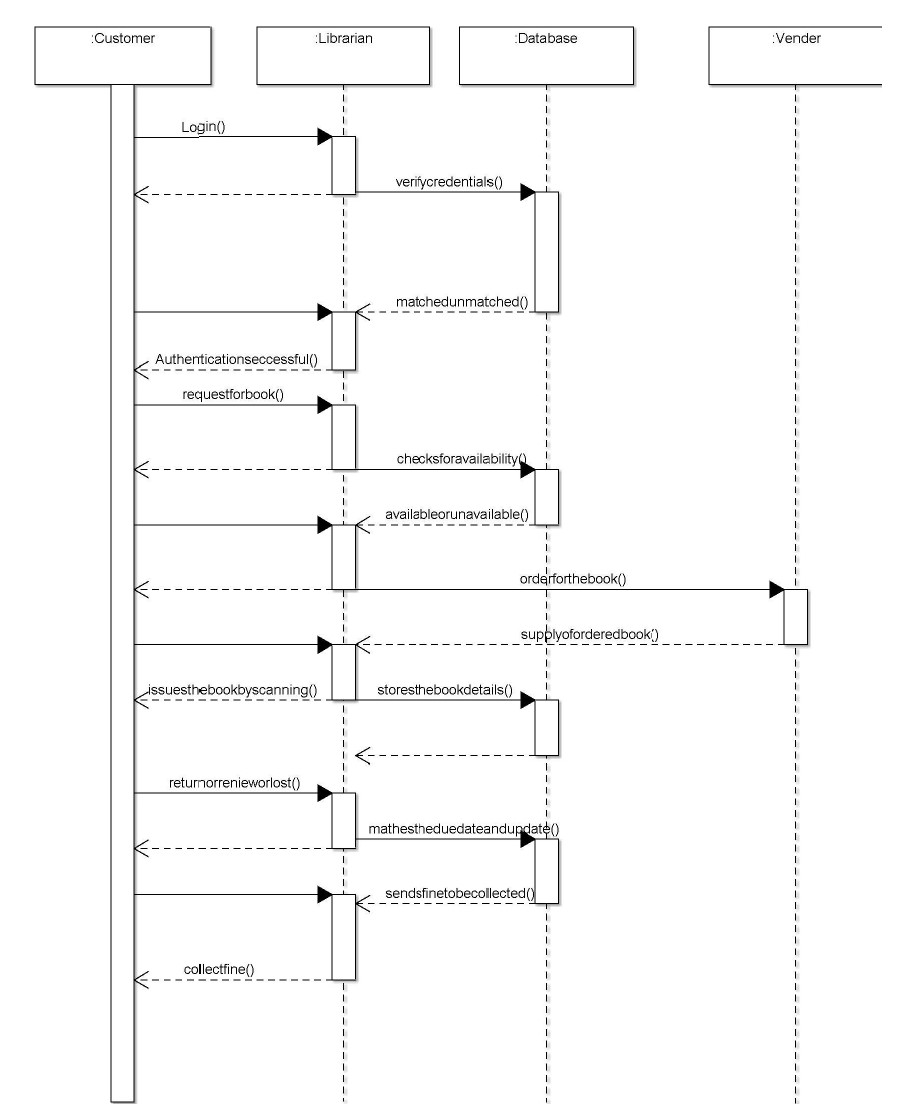


Fig 4.1.1 Use case diagram for library Management System

* 1. **Sequence Diagram**

A sequence diagram in Unified modelling Language (UML) is a kind of Interaction diagram it shows how processes operate with one another and in what order. It shows how object behaves with another object during its life line.

Sequence diagram describe interactions among classes in terms of an exchange of messages overtime. These can help to predict how a system will behave and discover responsibilities a class may need to have in the process of modelling a new system.

Fig 4.2.1 Sequence Diagram

**4.3 Class Diagram**

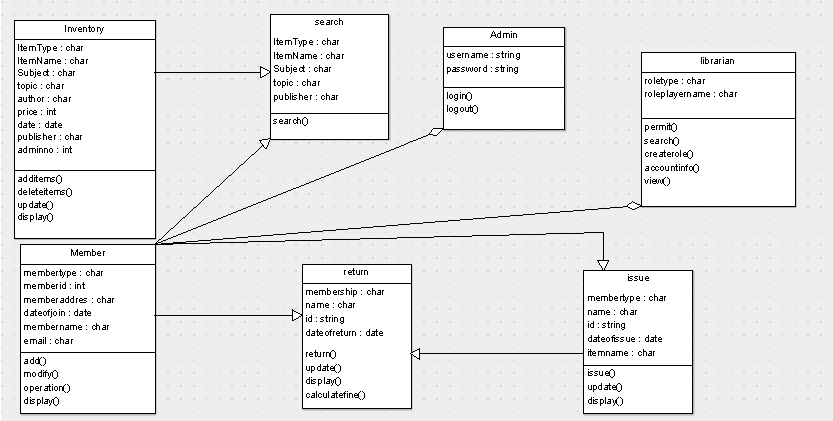
Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

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.

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

Class diagrams are used for describing the static view of the system. Describing the functionalities performed by the system. Construction of software applications using object oriented languages.

.

Fig4.3.1 Class Diagram

**4.4 Activity Diagram**

Activity diagram are graphical representation of stepwise activities and actions with support for choice, interaction and concurrency. Here in the activity diagram the Buyer login to the system and view the details of the Product. The seller login to the system for entering the product and update the details in the database. The final interaction is to sell the product.

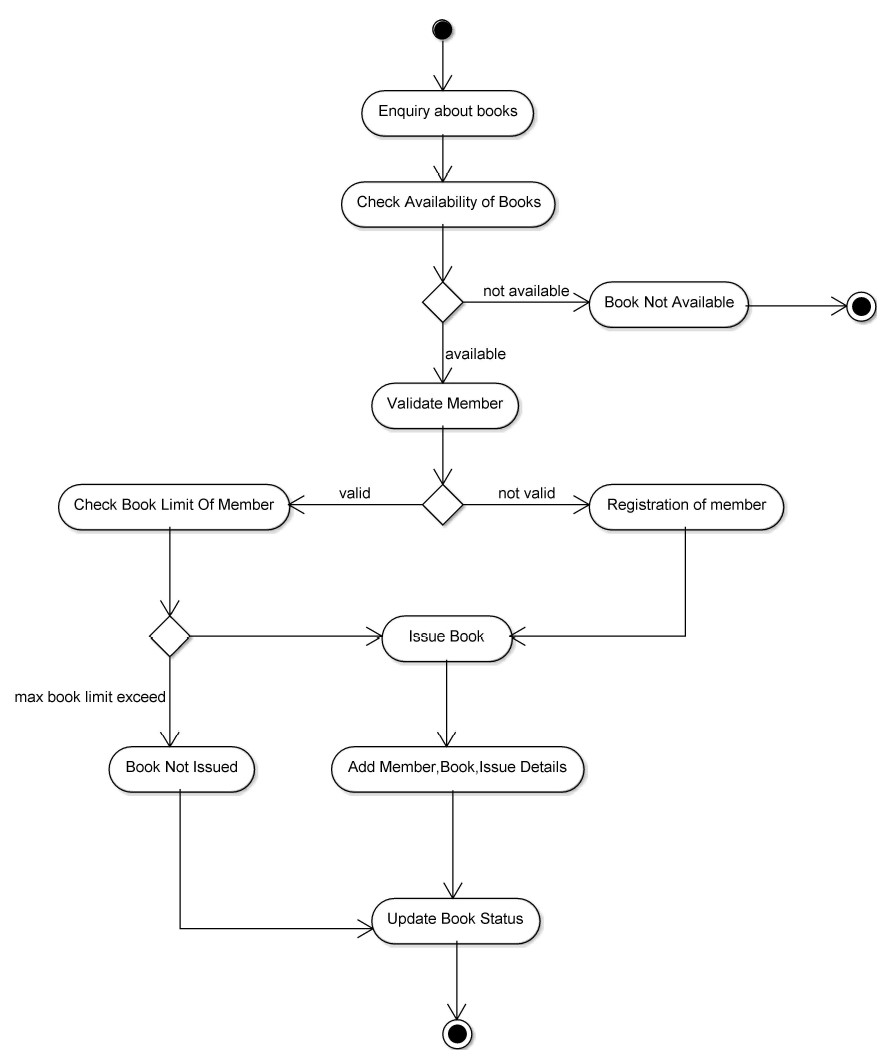


Fig 4.4.1 Activity Diagram

**4.5 State Chart Diagram**

A state chart diagram shows a state machine, emphasizing the flow of control from state to state. A state machine is a behaviour that specifies the sequences of states an object goes through during its lifetime in response to events, together with its responses to those events. A state is a condition or situation in the life of an object during which it satisfies some condition, performs some activity, or waits for some event.

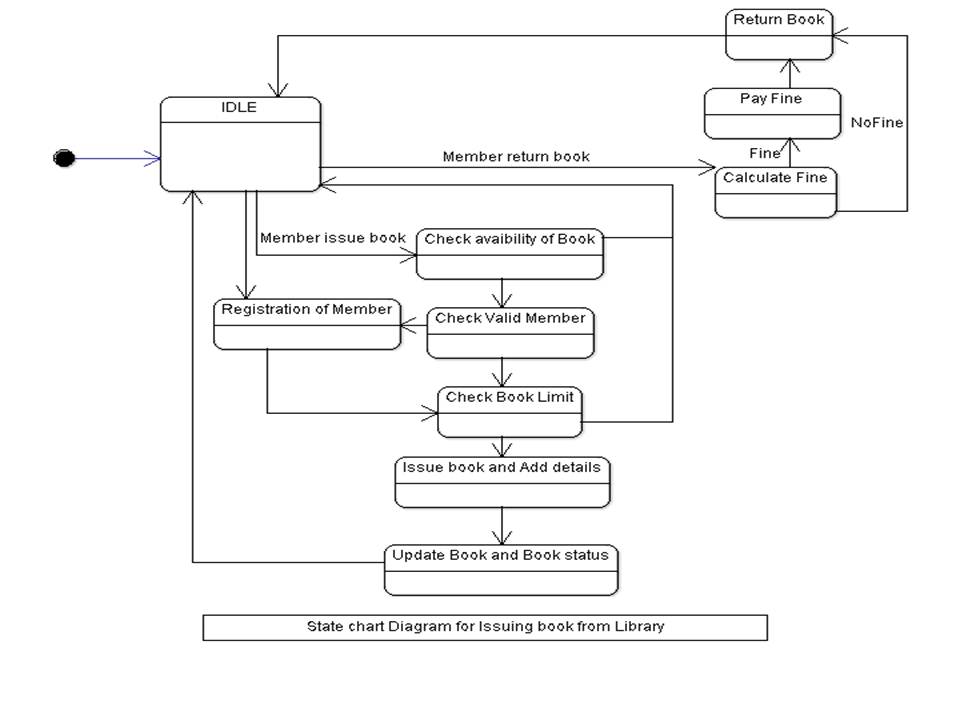


Fig 4.5.1 State Chart Diagram

**4.6 Object Diagram**

Object diagrams are derived from class diagrams so object diagrams are dependent upon class diagrams. Object diagrams represent an instance of a class diagram. Object diagrams also represent the static view of a system but this static view is a snapshot of the system at a particular movement. So the purpose of object diagram is to capture the static view of a system at a particular moment.

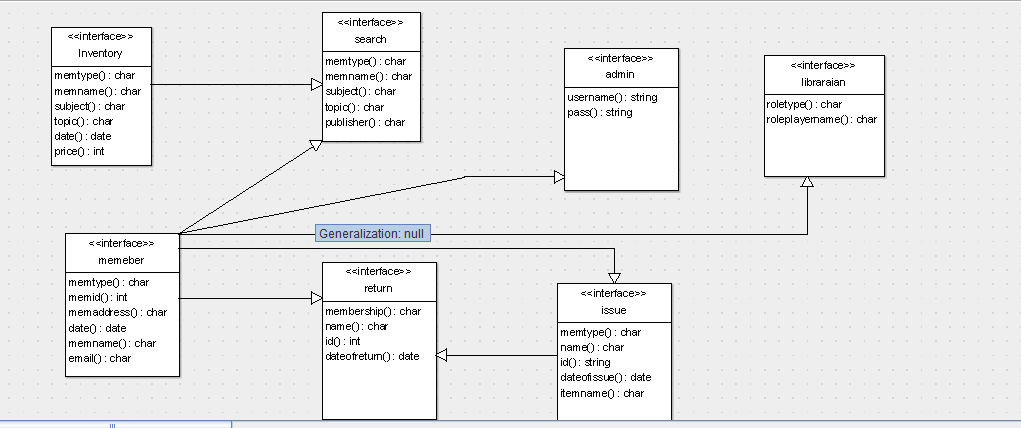


Fig 4.61Object Diagram

**4.8 Component Diagram**

Component diagrams are different in terms of nature and behaviour. Component diagrams are used to model the physical aspects of a system. Physical aspects are the elements such as executable, libraries, files, documents, etc. which reside in a node.

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems.

We have already described that component diagrams are used to visualize the static implementation view of a system. Component diagrams are special type of UML diagrams used for different purposes.

These diagrams show the physical components of a system. To clarify it, we can say that component diagrams describe the organization of the components in a system.

As we have already discussed, those components are libraries, files, executables, etc. Before implementing the application, these components are to be organized. This component organization is also designed separately as a part of project execution.

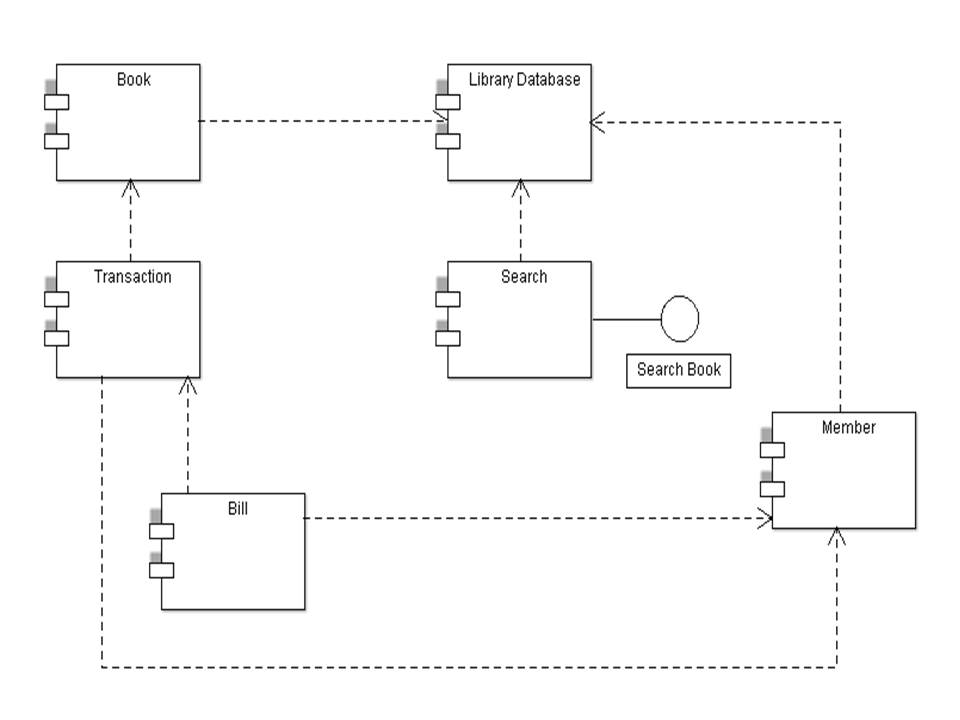


Fig 4.8.1 Component Diagram

**4.9 Deployment Diagram**

Deployment diagrams are used to visualize the topology of the physical components of a system, where the software components are deployed.

Deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

Deployment diagrams are mainly used by system engineers. These diagrams are used to describe the physical components (hardware), their distribution, and association.

Deployment diagrams can be visualized as the hardware components/nodes on which the software components reside.

Software applications are developed to model complex business processes. Efficient software applications are not sufficient to meet the business requirements. Business requirements can be described as the need to support the increasing number of users, quick response time, etc.

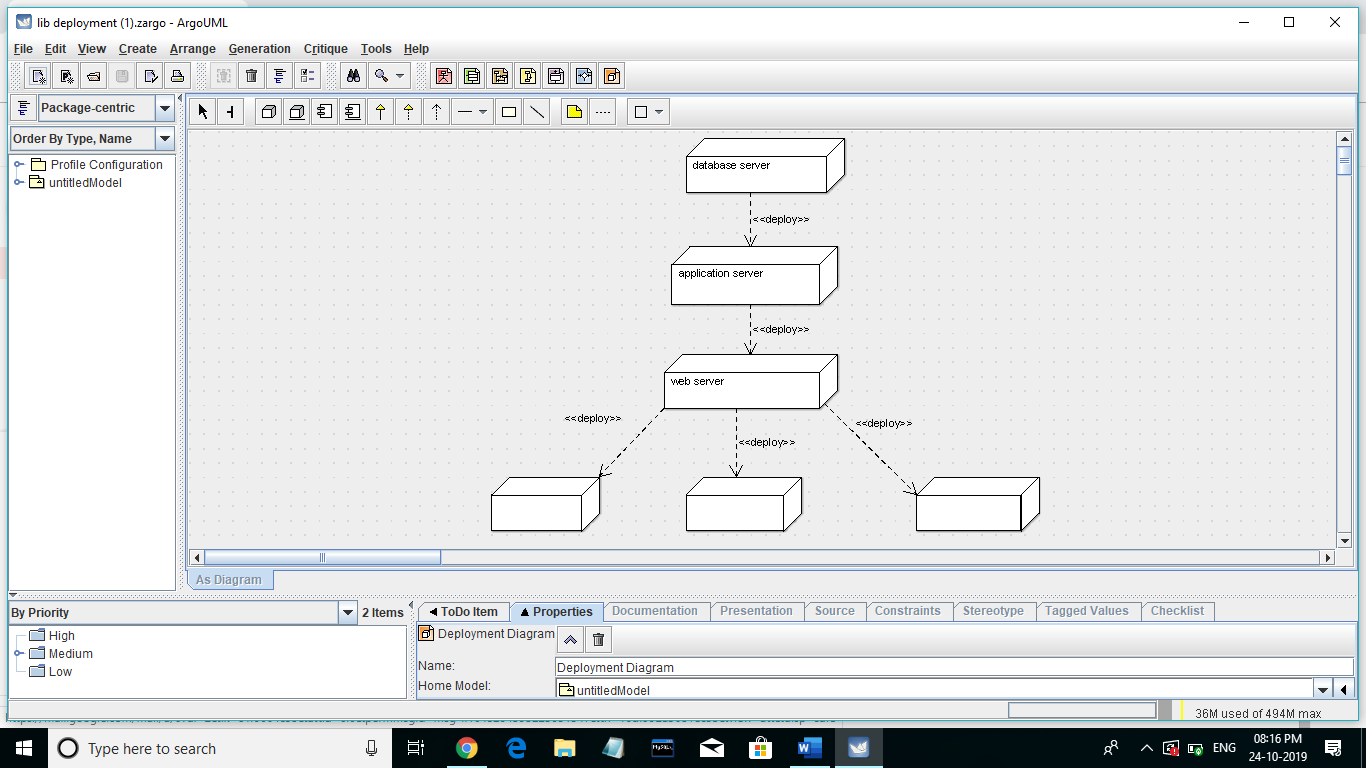


Fig 4.9.1 Deployment Diagram

**5.conclusion**

With regard to the Requirements specified, we completed the project. This system satisfies the users and it is a user–friendly application which is easy to operate.

The hypothesis was that Xpress visa would last the longest in all of the devices tested. My results do support my hypothesis.

### 5.1 Scope for future work

The system is developed in such a way that it can be further modified &new update can be added to the system. So that the system can have flexibility.

The following enhancements that the system provides compared to the existing one:

* Technologies used for the entire system improves the efficiency.
* It provides a friendly user interface which proves it better when compared to existing system.
* System security, data security is provided.

**6. References**

Simon Sennet, Steve McRobb and Ray Farmer, “Object Oriented System Analysis and Design using UML”, 2nd edition, McGraw Hill, 2002

Roger S. Pressman, “Software Engineering, A practitioner’s Approach”, 6th edition, McGrawHill International Edition, 2008

James Martin and James J. Odell, Principles of Object-Oriented Analysis and Design, Jan 17, 2008

Grady Booch, Robert A. Maksimchuk, Michael W. Engel, and Bobbi J. Young, Object-Oriented Analysis and Design with Applications, 3rd Edition, April 30, 2007

<https://www.tutorialspoint.com/uml/uml_standard_diagrams.htm>

<https://www.tutorialspoint.com/uml/uml_use_case_diagram.htm>

<https://www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/>

<https://creately.com/diagram/example/ila9idhl2/Mini%20Project%20UML>

<https://1000projects.org/wedding-event-planner-java-project.html>