TRAIN SIGNAL

A Project Report Submitted

In Partial Fulfilment of the Requirements

For the Degree of

MASTER OF COMPUTER APPLICATION

By

DEEPIKA

University Roll No. 1900290149041

Under the Supervision of

Mr. Ankit Verma

Assistant Professor

KIET Group of Institutions

Submitted to

DEPARTMENT OF COMPUTER APPLICATION

Affiliated to

DR. A. P. J ABDUL KALAM TECHNICAL UNIVERSITY

LUCKNOW

JULY,2021

DECLARATION

I hereby declare that the work presented in this report entitled **"Institute Presidency Application"**, was carried out by US. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution.

I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results. We shall be fully responsible and answerable.

DEEPIKA

University Roll No.- 1900290149041

CERTIFICATE

Certified that **DEEPIKA** **(Univ. Roll No.- 1900290149041)** have carried out the project work having **"Institute Presidency Application"** for Master of Computer Application from Dr.A.P.J.Abdul Kalam Technical University (AKTU),Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date: 08/08/2021

DEEPIKA **(Univ. Roll No - 1900290149041)**

This is to certify that the above statement made by the candidate is correct to the best of our knowledge.

Date: 08/08/2021

Mr. Ankit Verma
Assistant Professor
Department of Computer Application
KIET Group of Institutions, Ghaziabad

Signature of External Examiner **Signature of Internal Examiner**

Dr. Ajay Kumar Shrivastava
Head, Department of Computer Application
KIET Group of Institutions, Ghaziabad

ACKNOWLEDGEMENTS

Success in life is never attained single handedly. My deepest gratitude goes to my thesis supervisor **Mr. Ankit Verma**, for his guidance, help and encouragement throughout our project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to **Dr. Ajay Kumar Shrivastava**, Professor and Head, Department of Computer Application, for his insightful comments and administrative help at various occasions.
Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

DEEPIKA

Univ. Roll No. 1900290149041

Abstract

Main aim of this Project is to illustrate the concepts and usage of pre-built functions in OpenGL.

Train signal project simulated the railway crossing where two tracks are connected.

When there's just single railway track then a signal is used to stop one train and let other train to cross over using crossing track.

Finally once the train has crossed over then the signal is turned back to green and the stopped train is allowed to go normally.

We have used input devices like mouse and key board to interact with program

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

Introduction

As a software interface for graphics hardware, OpenGL's main purpose is to render two- and three-dimensional objects into a frame buffer.

These objects are described as sequences of vertices or pixels.

OpenGL performs several processing steps on this data to convert it to pixels to form the final desired image in the frame buffer.

OpenGL Fundamentals

This section explains some of the concepts inherent in OpenGL.

Primitives and Commands

OpenGL draws primitives—points, line segments, or polygons—subject to several selectable modes.

You can control modes independently of each other; that is, setting one mode doesn't affect whether other modes are set .Primitives are specified, modes are set, and other OpenGL operations are described by issuing commands in the form of function calls.

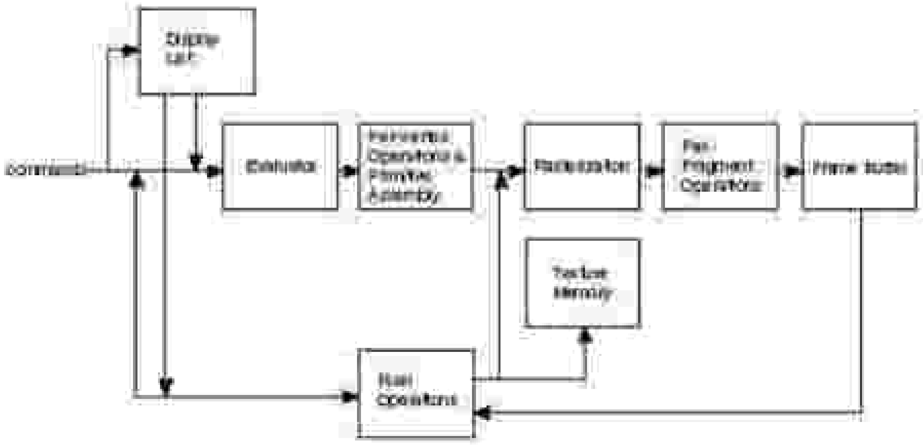
Primitives are defined by a group of one or more vertices. A vertex defines a point, an endpoint of a line, or a corner of a polygon where two edges meet. Data is associated with a vertex, and each vertex and its associated data are processed independently, in order, and in the same way. The type of clipping depends on which primitive the group of vertices represents.

Commands are always processed in the order in which they are received, although there may be an indeterminate delay before a command takes effect. This means that each primitive is drawn completely before any subsequent command takes effect. It also means that state-querying commands return data that's consistent with complete execution of all previously issued OpenGL commands.

Basic OpenGL Operation

The figure shown below gives an abstract, high-level block diagram of how OpenGL processes data. In the diagram, commands enter from the left and proceed through what can be thought of as a processing pipeline. Some commands specify geometric objects to be drawn, and others control how the objects are handled during the various processing stages.

Figure . OpenGL Block Diagram



As shown by the first block in the diagram, rather than having all commands proceed immediately through the pipeline, you can choose to accumulate some of them in a display list for processing at a later time.

Rasterization produces a series of frame buffer addresses and associated values using a two-dimensional description of a point, line segment, or polygon.

Each fragment so produced is fed into the last stage,

per-fragment operations, which performs the final operations on the data before it's stored as pixels in the frame buffer. These operations include conditional updates to the frame buffer based on incoming and previously stored z-value s (for z-buffering) and blending of incoming pixel colors with stored colors, as well as masking and other logical operations on pixel values.

All elements of OpenGL state, including the contents of the texture memory and even of the frame buffer, can be obtained by an OpenGL application.

Implementation

This program is implemented using various openGL functions which are

shown below.

Various functions used in this program.

glutInit() : interaction between the windowing system and OPENGL is initiated

glutInitDisplayMode() : used when double buffering is required and depth information is required

glutCreateWindow() : this opens the OPENGL window and displays the title at top of the window glutInitWindowSize() : specifies the size of the window

glutInitWindowPosition() : specifies the position of the window in screen co-ordinates

glutKeyboardFunc() : handles normal ascii symbols

glutSpecialFunc() : handles special keyboard keys

glutReshapeFunc() : sets up the callback function for reshaping the window

glutIdleFunc() : this handles the processing of the background

glutDisplayFunc() : this handles redrawing of the window

glutMainLoop() : this starts the main loop, it never returns

glViewport() : used to set up the viewport

glVertex3fv() : used to set up the points or vertices in three dimensions glColor3fv() : used to render color to faces

glFlush() : used to flush the pipeline

glutPostRedisplay() : used to trigger an automatic redrawal of the object

glMatrixMode() : used to set up the required mode of the matrix

glLoadIdentity() : used to load or initialize to the identity matrix

glTranslatef() : used to translate or move the rotation centre from one point to another in three dimensions

glRotatef() : used to rotate an object through a specified rotation angle

Interaction with program

This program includes interaction through keyboard.

- P Start the Project
- 1 -> Toggle signal number 1 which is at the top.
- 2-> Toggle signal number 2 which is at the bottom.
- 3 -> toggle both signals at once.
- X/x-> rotate the scene about x axis.
- Y/y-> rotate the scene about y axis.
- Z/z-> rotate the scene about z axis.
- Q-> Quit

1.3 IDENTIFICATION OF NEED

User need identification and analysis are concerned with what user needs rather than what he/she wants. Not until the problem has been identified, defined, and evaluated should the analyst think about solutions and whether the problem is worth working. This step intended to help the user and analyst understand the real problem rather than its symptoms. The user or the analyst may identify the need for a candidate system or for enhancement in the existing system.

An analyst is responsible for performing following tasks:

- Studied strength and weakness of the current system.
- Determined "what" must be done to solve the problem.
- Prepared a functional specifications document.

These modules are developed with the aim of reducing time, reducing manpower so that everything can be easily maintained and. The volume of work and complexity are increasing year by year. This system reduces complexity and time. Also provide availability 24*7.

1.4 PROBLEM STATEMENT

In the existing system all the work is done manually. This is chance of committing errors and it will take more time to perform or checkout any information. There are so many limitations in the existing system. So the existing system should be automized. If the system is carried over manually, for everything it take more time. So it is difficult to take immediate decisions.

- In the traditional system, if you wish to analyze any record you have to turn pages many time.
- Existing systems are time consuming as it requires too much planning and so much human involvement.
- As it involves much human involvement, the cost of the system automatically gets increased.
- Existing systems require paper use, which isn't good for the environment.
- With too much human involvement, there are high chances of risk as well.
- There is too much of paper work too, which makes the tasks in the existing system, very tedious.

System specifications

SOFTWARE REQUIREMENTS :

- MICROSOFT VISUAL C++
- OPENGL

HARDWARE REQUIREMENT :

- GRAPHICS SYSTEM,
- Pentium P4 with 256 of Ram(Min)

1.5.3 SOME REQUIREMENTS

Performance Requirements:

To achieve good performance the following requirements must be satisfied

- Scalability: The ease with which a system or component can be modified to fit the problem area.
- Portability: The ease with which a system or component can be transferred from one hardware or software environment to another.
- Security: It is the ideal state where all information can be communicated across the internet / company secure from unauthorized persons being able to read it and/or manipulate it..
- Maintainability: The ease with which a software system or component can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment.
- Reliability: The ability of a system or component to perform its required functions under stated conditions for a specified period of time.
- Reusability: The degree to which a software module or other work product can be used in more than one computing program or software system.

Safety Requirements:

In case scenarios where data integrity can be compromised, measures should be taken to ensure that all changes are made before system is shutdown. The user must have a registered account to use all facility of the web application.

G. Security The salesforce.com is providing system level and application level security. At the system level security, developer can set the permissions. Developer can generate the access levels as per the role of employees. So, as pet role base access, only authorised users can get access to their own developer org. Violation of security access will be reduced.

H. GUI The salesforce.com has a simple GUI. New user can easily understand the flow of application. For new users, salesforce.com is providing a beginner module in trail heads. From that user can easily work on the GUI related queries. User can learn from trailheads about the GUI.

1.5.4.2 APEX

Apex is strongly typed object-oriented, on-demand programming language. It is compiled, stored, and run entirely on the Force.com platform (multi-tenant environment and is very controlled in its invocations and limits).

Apex syntax looks mostly like Java and acts like stored procedures.

Apex allows developers to attach business logic to the record save process

Apex has built-in support for unit test creation and execution.

As a language apex is Integrated, Easy to use, Data focused, Rigorous, Hosted, Multi-tenant aware, automatically up-gradable, easy to test and versioned.

Apex has certain features like listed below.

- Integrated
- Easy to use
- Data Focused
- Hosted
- Multitenant aware
- Easy to test
- Versioned
- Object-Oriented

1.6 PROJECT SCHEDULE

The objective of software project planning is to provide a framework that enables the manager to make reasonable estimates of resources, costs and schedule. These estimates are made within a limited time frame at the beginning of a software project and should be updated regularly as the project progresses. In addition, estimates should attempts to define "best case" and "worst case" scenarios so that project outcomes can be bounded.

The first activity in software project planning is the determination of software scope. Function and performance allocated to software during system engineering should be assessed to establish a project scope that is ambiguous and understandable at Presidency and technical levels. Software scope describes function, performance, constraints, interfaces and reliability.

During early stages of project planning, a microscopic schedule is developed. This type of schedule identifies all major software engineering activities and the product functions to which they are applied. As the project gets under way, each entry on the macroscopic

schedule is refined into detailed schedule. Here specific software tasks are identified and scheduled.

Scheduling has following principles:

1. Compartmentalization: the project must be compartmentalized into a number of manageable activities and tasks.
2. Interdependency: the interdependencies of each compartmentalized activity or tasks must be determined.
3. Time allocation: each task to be scheduled must be allocated some number of work units.
4. Effort validation: every project has a defined number of staff members.
5. Defined responsibilities: every task that is scheduled should be assigned to a specific team member.
6. Defined outcomes: every task that is scheduled should have a defined outcome.

1.6.1 Pert chart

Program evaluation and review technique (pert) is a project scheduling method that is applied to software development.

Pert provide quantitative tool that allow the software planner to-Determine the critical path-the chain of tasks that determines the duration of the project; Establish "most likely" time estimates for individual tasks by applying statistical models; and

Calculate "boundary times" that defines a time "window" for a particular task.

Pert chart(program evolution review technique) for project-

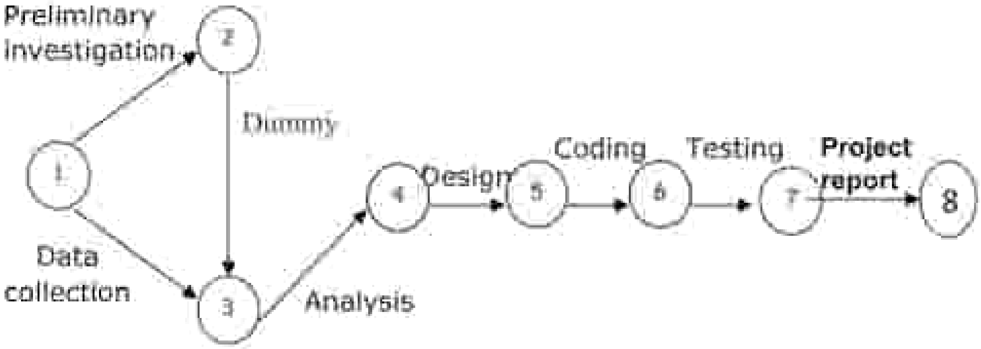


Figure 1.1 Pert chart

1.6.2 Gantt Chart

When creating a project schedule, the planner begins with a set of tasks (the work breakdown structure). If automated tools are used, the work breakdown is input as a task network. Effort, duration and start dates are input are each task network. As a consequence of this input, a timeline chart also called a Gantt chart is generated. A timeline chart is developed for entire project.

Gantt chart for project:

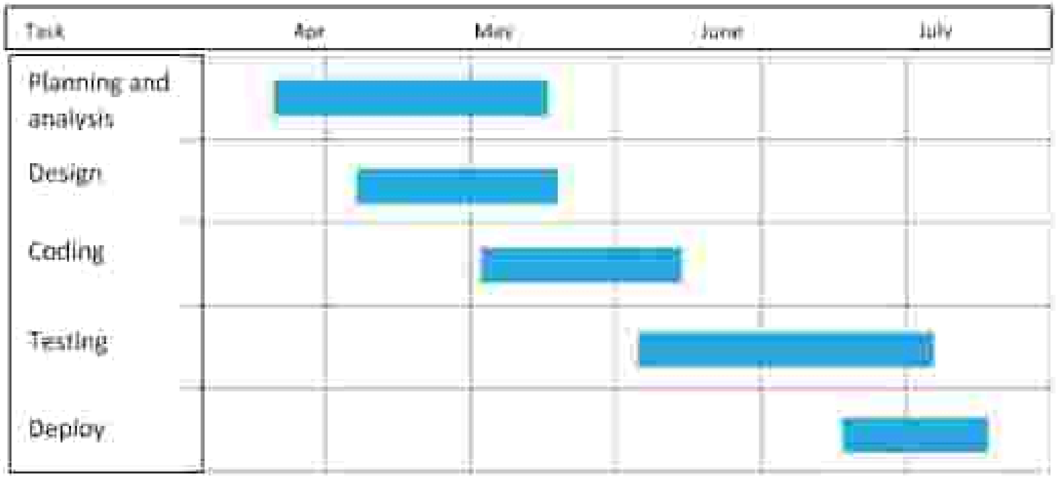


Figure 1.2:Gantt chart for project

Here horizontal bars indicate the duration of each task.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

ABSTRACT

trending cloud computing technology, It's a CRM (Customer relationship management), which is available 24*7 on cloud, no need to install any extra software, we can do our work on cloud with case and flexibility as well as availability. Our software mainly reduces the load tasks and it is efficient to maintain the data for a longer time than the hand written data as well give cloud case.

Our project is for institutes Is Intranet based software with cloud flexibility, our system is for managing, manipulating, reporting etc, information for the Institutes. Our project is created for a building institute to up & simple access to information. For accessing this the user should be enrolled with the system, after that they could change or manipulate data by the authorizations provided to the user. Institute presidency application is based an intranet application that provide giving information to management with in firm of all level.

KEYWORDS:- Cloud, Salesforce, cloud

programming concept can use our system easily[10].

"Institute Presidency Application" is based on cloud computing, a Salesforce application.

With the traditional database we use to store data in databases respectively with any software, coding or manually, it's quite hectic job.[15] that managing the database separately, but this time we have cloud-based storage mechanism which is easier and simpler to manage the database, that is salesforce[13].

Salesforce is customer relationship management (CRM) suite give applications for different size of organizations like small, midsize, large, with a proper focus on support and sales. It is an on-demand service. It contains managing all thing related of relationship between an firm and its user.also provide security in different ways[14].

So, the maintenance and maintenance of institute became very easy, it provide:-

- It makes searching of data easier and faster.
- Easy availability.
- liberates lengthy manual records.

So our project is providing convenience in different aspects,

2.2 RELATED WORK

2.2.1 College Management System

It's a system which manages the information of college, information of student, information of placement, various event going in college. It keep tracking record of all the data. It has also a notice board which have data about various programme either its cultural or any sports which is supposed to be held soon or technical[5] .

2.2.2 Campus Cloud – A Management Information System On Cloud

This paper has find several problem with the systems intended for management of institute and other systems similar. It is successful implementation of such type of systems makes for consistently better performance, institutions gets several problem, such as issues of revenue, issues of technological, complexity, technical knowledge lacks, Campus Cloud focuses on educational firms The best level of integration requirements of a given organization would be defined as per, with the on-demand nature and modular of the Cloud service, give the best possible advantages to the users[6].

2.2.3 Institute Management System (Ims)

Institute Management System (IMS), a system developed with the help of Java ,that could be used for presidency of different things in various firms like colleges, schools and universities. It have different access level for people of different categories like administrators, special users, normal users. It use SQL for management of database and can be connected so that accessible from any location ,that is to cloud database storage. It will be developed using Netbeans IDE. It contains different module which is available for different section[7].

2.2.4 Cloud Based College Management System

This system work mainly on Cloud Platform Techniques, Microsoft Azure to handle the database and code with monitoring of network for reasons of security and handling PHP extensions using PHP libraries inbuilt and Database MySQL and windows server 2016,virtual server. The now work can improve by PHP Code of each page method of ajax calling technique of JavaScript/JQuery Programming Language[8].

2.2.5 Salesforce.Com-A Cloud Provider

The proposed system work is about the using cloud stage which would change all the customary perspectives of system, product improvement Technologies and application. Salesforce.com, the best cloud providers. There are numerous reasons because of which IT industries are moving to cloud. Furthermore, quantities of reasons , industries need to taught to adopting salesforce.com cloud. The system work is focusing on regular and essential highlight of salsforce.com The aim of this proposed system is to show the resources available in the salesforce.com[2].

2.2.6 Extremely Effective Crm Solution Using Salesforce

Salesforce is one of the hot cloud computing tech. in computer industry, which is available on the cloud, there is no need to install any software. Salesforce.com is one of the on demand CRM, which runs on the force.com platform. we are talking about Introduction to the Cloud Computing, different Service models in the Cloud Computing, Types of it, Architecture of it and MVC, Introduction to the Salesforce, SOQL and Its Operators and also Force.com IDE and CRM[11].

In this we define the projects structure, feature, criteria for success the main objective is to create the desired project. our system is based on cloud.

2.3.2 Implementation And Evaluation

The implementation of the system is done with the help of salesforce and apex.by using of force.com cloud platform. With the help of apex language.by the using of force.com pages leveraging and functionalities the information on our account of salesforce[1].

With the help of the custom domain, user can use this domain to access the website.

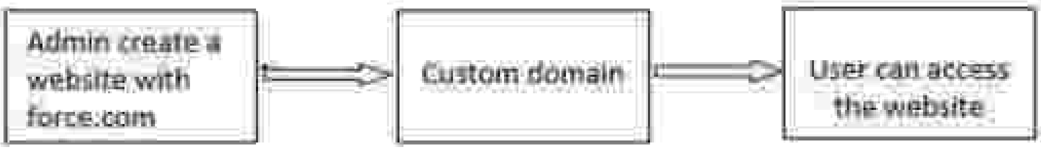

Admin create a website with force.com
Custom domain
User can access the website

Figure 2.2 flow of website

2.3 DESIGN

2.3.1 Design

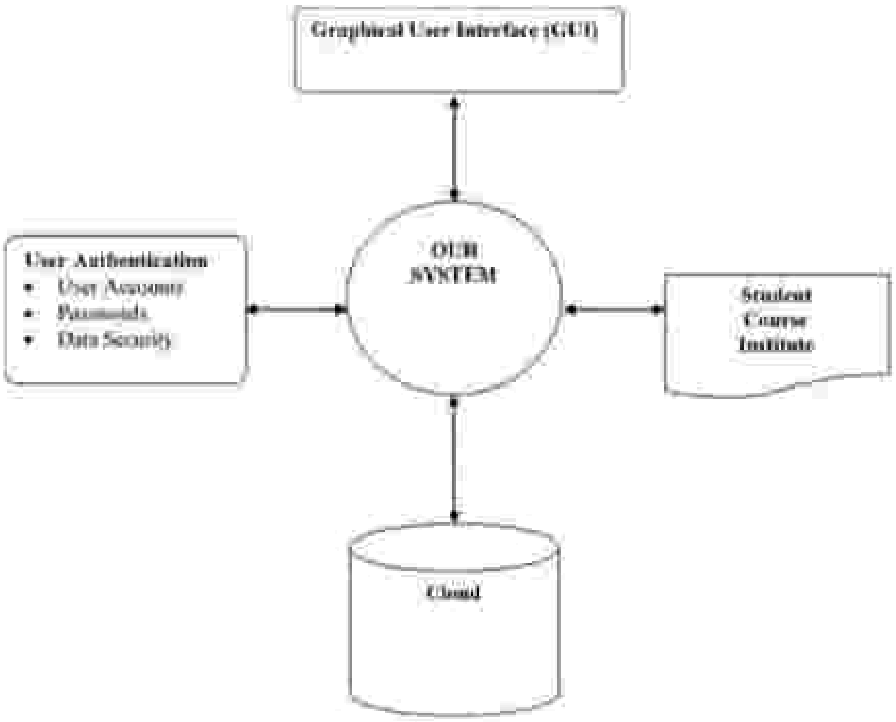


Figure 2.3 Design

2.3.2.1 Salesforce

Salesforce customer relationship management (CRM) which is an on-demand suite offering applications for different level of organization like small, midsize and large organizations, with a complete focus on support and sales as well.

Salesforce initially started for CRM solution as a cloud-base. CRM full form is Customer Relationship management.

It basically used for establishing a better connectivity between user and the organisation. With the help of cloud based technology, where all stuffs related to user available for 24*7. To form a ideology of trust between both.

2:3.2.2 Apex

Apex is object-oriented strongly typed, programming language ,it is on-demand. It is stored ,compiled, and run entirely the platform of the Force.com.

On which user can perform different tasks as they needed for making the project more effective and ready to use.

From the java syntax ,the Apex syntax assemble mostly and show as stored procedure.

The apex, programming language allow the coder to write their enterprises logic and can also attach records to save process. The apex has unit test execution and creation built-in support as well though which it get whopping effect.

As a programming language apex is Easy to use, Integrated, Data focused, Hosted, Rigorous, automatically up-gradable Multi-tenant aware,, easy to test and easily versioned.

2.3.3 GUI

The graphical user interface is one important factor by which the user can easily access the website to understand what he need to do in order to use the software effectively[3]. A easy and effective GUI is good for both user as well as developer.

Some of the GUI are as follow:

Figure 2.3 some screens

2.4 RESULT

We have come on result that our system working fine in each case. It is user friendly as well as efficient to use. we have done different things to verify the performance.

TEST CASE RESULT

TestCase#	Description	Result
TC#1	Loading the	Passed
TC#2	Login	Passed
TC#3	Validating	Passed
TC#4	Content	Passed
TC#5	Course page	Passed
TC#6	Reports page	Passed
TC#7	Logout	Passed

2.5 DISCUSSION

System was completely done after was duly coded, each modules of the project were checked to ensure they are fully functional units.[4] This was done by checking each unit to give assurance that it functions as required and that it performed exactly as defined. The success of each individual gave us go ahead to carryout testing properly[9].

The defined system was validated by the using a series of short questionnaire that was completely filled by representatives users who have used the system and give suggestion according to the need .This was done to the assess if the system met their respective needs and requirements. It was also find that it is easy to access the data as well as available when needed. With the flexibility of the cloud it is quite useful and better version of management system.

2.6 CONCLUSION & FUTURE SCOPE

2.6.1 CONCLUSION

A system means a lot of experience. I learned a lot of thing this project development. This project has also sharpened my concept cloud computing.

2.6.2 FUTURE SCOPE

To make our system more user friendly.

I will add Helping BOT in the system.

Adding of Online examination module.

2.7 REFRENCES

[1] Lubomir prech,yuri y.,vladimir, "Overview of apex project results",frontiers,published on 21 December 2018 .

[2] Sabahi (2011),. Salesforce.com-A cloud provider, 2011 IEEE International Conference on Communication Software and Networks...

[3] R. B. Guin, S. Chakrabarti, C. Tarafdar, and S. Mandal, "A smart architectural concept for the making of a university education system using cloud computing paradigm," in Proc. 2011 World Congress on Information and Communication Technologies, Mumbai, 2011, pp.48-52.

[4] R. M. Leod, management Information Systems, Third Ed., Science Research Associates, 1986, pp. 17-19.

[5] L. Long, college management System, Prentice Hall, 1989, pp.116-117.

[6] Campus Cloud – A Management Information System On Cloud Niraj Khot1, Ketan Mudur2, Omkar Thorat3, Yogesh Doulatramani4

[7] Institute Management System (Ims) Ijarcce ,Mustafiz Sharique pp 1-6

[8] A Research Paper On College Management System Lalit Mohan Joshi M.Tech Scholar.

[9] IJCSMC, Vol. 7, Issue. 7, July 2018, pg.20 – 31 Design and Implement a Novel Student Information Management System – Case Study Ihsan Ali Hassan

[10] Shimozono K., Itsukiy M., Harasaka Y. and Furukawa Z., (2010), "User Management in an Educational Computer System: Personal Information Management," Fukuoka, Japan, pp. 1-6.

[11] Rakesh kumar.sonu, Agarwal. pragya -Extremely Effective Crm Solution Using Salesforce, JETIR,

[12] Norasiah M. A. and Norhayati A., (2003), "Intelligent Student Information System," in 4th National Conference on Telecommunication Technology Proceedings, Shah Alam, Malaysia, pp. 212-215

[13] Bharamagoudar S. R., Geeta R. B. and Totad S. G., (2013), "Web Based Student Information Management System," International Journal of Advanced Research in Computer and Communication Engineering, vol. 2, no. 6, pp. 2342-2348.

[14] Jensen, J. Schwenk, N. Gruschka, & L. LO, Iacono (2009). On technical security issues in cloud computing. CLOUD 2009 - 2009 IEEE International Conference on Cloud Computing, 109–116.

[15] Connolly T. M. and Begg C. E., (2010), Dabases Systems: A Practical Approach to Design, Implementation and Management, 5th ed., Boston: Pearson, pp. 340-344.

CHAPTER 3

FEASBILITY STUDY

3.1 INTRODUCTION

Feasibility of the system is an important aspect, which is to be considered. The system needs to satisfy the law of economic, which states that the maximum output should be yielded in minimum available resources.

A feasibility analysis evaluates the project's potential for success; therefore, perceived objectivity is an essential factor in the credibility of the study for potential investors and lending institutions. There are five types of feasibility study—separate areas that a feasibility study examines, described below.

1. Technical Feasibility

This assessment focuses on the technical resources available to the organization. It helps organizations determine whether the technical resources meet capacity and whether the technical team is capable of converting the ideas into working systems. Technical feasibility also involves the evaluation of the hardware, software, and other technical requirements of the proposed system. As an exaggerated example, an organization wouldn't want to try to put Star Trek's transporters in their building—currently, this project is not technically feasible.

2. Economic Feasibility

This assessment typically involves a cost- benefits analysis of the project, helping organizations determine the viability, cost, and benefits associated with a project before financial resources are allocated. It also serves as an independent project assessment and enhances project credibility—helping decision-makers determine the positive economic benefits to the organization that the proposed project will provide.

3. Legal Feasibility

This assessment investigates whether any aspect of the proposed project conflicts with legal requirements like zoning laws, data protection acts or social media laws. Let's say an organization wants to construct a new office building in a specific location; A feasibility study might reveal the organization's ideal location isn't zoned for that type of business. That organization has just saved considerable time and effort by learning that their project was not feasible right from the beginning.

4. Operational Feasibility

This assessment involves undertaking a study to analyze and determine whether—and how well—the organization's needs can be met by completing the project. Operational feasibility

studies also examine how a project plan satisfies the requirements identified in the requirements analysis phase of system development.

5. Scheduling Feasibility

This assessment is the most important for project success; after all, a project will fail if not completed on time. In scheduling feasibility, an organization estimates how much time the project will take to complete.

When these areas have all been examined, the feasibility analysis helps identify any constraints the proposed project may face, including:

- Internal Project Constraints: Technical, Technology, Budget, Resource, etc.
- Internal Corporate Constraints: Financial, Marketing, Export, etc.
- External Constraints: Logistics, Environment, Laws, and Regulations, et

3.2 MAIN ASPECTS

There are three aspects of feasibility to be considered namely.

1. Technical
2. Operational
3. Economical

TECHNICAL:

In the technical aspects one may consider the hardware equipment for the installation of the software. The system being centralized will required very little hardware appliances. Hence this helps the system to work smoothly with limited amount of working capitals.

OPERATIONAL:

In the operational aspects may think of the benefits of the workload that many a personal may have to share. This is eased out and the required output may be retrieved in a very short time. Thus there is accuracy in the work on time is also saved there will be very little work that needs to be performed.

ECONOMICAL:

Economical system is definitely feasible because the hardware requirement is less and the operational working for the system requires less number of recruits. This help introduction over-staffing and wastage funds.

We studied on the position to evaluate solution. Most important factors in this study were tending to overlook the confusion inherent in system Development the constraints and the assumed studies. It can be started that it the feasibility study is to serve as a decision document it must answer three key questions.

1. Is there a new and better way to do the job that will benefit the user?
2. What are the costs and savings of the alternatives?
3. What is recommended?

On these questions it can be explained that feasibility study of the system includes following different angles.

3.2.1

Technical feasibility:

This centers on the existing computer system (hardware, software etc.) and to what extent it can support the proposed additional equipment. In this stage of study, we have collected information about technical tools available by which I could decide my system design as the technical requirements.

3.2.2 Operational Feasibility:

In this stage of study we have checked the staff availability. I concentrate on knowledge of end users that are going to use the system. This is also called as behavioral feasibility in which I have studied on following aspects; people are inherently resistant to change, and computers have been known to facilitate change. An estimate has been made to how strong a reaction the user staff is having toward the development of a computerized system. It is common knowledge that computer installations have something to do with turnover. I had explained that there is need to educate and train the staff on new ways of conducting business.

3.2.3 Economical feasibility:

Economical analysis is the most frequently used method for evaluating the effectiveness of candidate system. More commonly known as cost benefit analysis, the procedure is to determine the benefits and savings that benefits outweigh costs. The decision was to design and implement system because it is for having chanced to be approved. This is an on going effort that improves the accuracy at each phase of the system life cycle.

In developing cost estimates for a system I need to consider several cost elements. Among these is hardware personal facility. Operating and supply costs.

3.3 BENEFITS

Benefits of conducting a feasibility study:

- Improves project teams' focus
- Identifies new opportunities
- Provides valuable information for a "go/no-go" decision
- Identifies a valid reason to undertake the project
- Enhances the success rate by evaluating multiple parameters
- Aids decision-making on the project.
- Identifies reasons not to proceed.

3.4 SYSTEM REQUIREMENT SPECIFICATION

Any system can be designed after specifies the requirement of the user about that system. For this first of all gathered information from user by the preliminary investigation which is starting investigation about user requirement.

The data that the analysts collect during preliminary investigation are gathered through the various preliminary methods.

Documents Reviewing Organization

The analysis conducting the investigation first learn the organization involved in, or affected by the project. Analysis can get some details by examining organization charts and studying written operating procedures.

Collected data is usually of the current operating procedure:

- The information relating to clients, projects and students and the relationship between them was held manually.
- Managing of follow-ups was through manual forms.
- Complaints require another tedious work to maintain and solve.
- Payments details had to be maintained differently.

Gathering Information By Asking Questions

Interviewing is the most commonly used techniques in analysis. It is always necessary first to approach someone and ask them what their problems are, and later to discuss with them the result of your analysis.

Questionnaires

Questionnaires provide an alternative to interviews for finding out information about a system. Questionnaires are made up of questions about information sought by analyst. The questionnaire is then sent to the user, and the analyst analyzes the replies.

Electronic Data Gathering

Electronic communication systems are increasingly being used to gather information. Thus it is possible to use electronic mail to broadcast a question to a number of users in an organization to obtain their viewpoint on a particular issue.

In my project, with the help of Marg software solutions, I have send questionnaire through electronic mail to twenty employees of the company and retrieved the information regarding the problem faced by existing system.

Interviews

Interview allows the analysis to learn more about the nature of the project request and reason of submitting it. Interviews should provide details that further explain the project and show whether assistance is merited economically, operationally or technically.

One of the most important points about interviewing is that what question you need to ask.

It is often convenient to make a distinction between three kinds of question that is

- Open questions
- Closed question
- Probes

Open questions are general question that establish a persons view point on a particular subject.

Closed questions are specific and usually require a specific answer. .

Probes are question that follow up an earlier answer

CHAPTER 4

DESIGN

4.1 INTRODUCTION

System is created to solve problems. One can think of the systems approach as an organized way of dealing with a problem. In this dynamic world, the subject system analysis and design, mainly deals with the software development activities.

Since a new system is to be developed, the one most important phases of software development life cycle is system requirement gathering and analysis. Analysis is a detailed study of various operations performed by a system and their relationship within and outside the system. Using the following steps it becomes easy to draw the exact boundary of the new system under consideration.

All procedures, requirements must be analysed and documented in the form of detailed DFDs, logical data structure and miniature specifications.

System analyses also include sub-dividing of complex process involving the entire system, identification of data store and manual processes.

4.2 SYSTEM DESIGN

System design is the process of planning a new system or to replace the existing system. Simply, system design is like the blueprint for building, it specifies all the features that are to be in the finished product.
System design phase follows system analysis phase. Design is concerned with identifying functions, data streams among those functions, maintaining a record of the design decisions and providing a blueprint the implementation phase.
Design is the bridge between system analysis and system implementation. Some of the essential fundamental concepts involved in the design of application software are:

- **Modulation**
- Verification

Abstraction is used to construct solutions to problem without having to take account of the intricate details of the various component sub problems. Abstraction allows system designer to make step-wise refinement, which at each stage of the design may hide, unnecessary details associated with representation or implementation from the surrounding environment.

Modularity is concerned with decomposing of main module into well-defined manageable units with well-defined interfaces among the units. This enhances design clarity, which in turn eases implementation, Debugging, Testing, Documenting and Maintenance of the software product. Modularity viewed in this sense is a vital tool in the construction of large software projects

Verification is fundamental concept in software design. A design is verifiable if it can be demonstrated that the design will result in implementation that satisfies the customer's requirements. Verification is of two types namely,

- Verification that the software requirements analysis satisfies the customer's needs.
- Verification that the design satisfies the requirement analysis.

Some of the important factors of quality that are to be considered in the design of application software are:

Reliability:
The software should behave strictly according to the original specification and should function smoothly under normal conditions.

Extensibility:
The software should be capable of adapting easily to changes in the specification.

Reusability:

The software should be developed using a modular approach, which permits modules to be reused by other application, if possible.

The System Design briefly describes the concept of system design and it contains four sections. The first section briefly describes the features that the system is going to provide to the user and the outputs that the proposed system is going to offer.

The second section namely Logical Design describes the Data Flow Diagrams, which show clearly the data movements, the processes and the data sources, and sinks, E-R diagrams which represent the overall logical design of the database, and high-level process structure of the system.

Preliminary Design:

Preliminary design is basically concerned with deriving an overall picture of the system. Deriving entire system into modules and sub-modules while keeping Cohesion and Coupling factors in mind. Tools, which assist in preliminary design process, are Data Flow Diagrams.

Code design:

The purpose of code is to facilitate the identification and retrieval for items of information. A code is an ordered collection of symbols designed to provide unique identification of an entity or attribute. To achieve unique identification there must be only one place where the identified entity or the attribute can be entered in the code; conversely there must be a place in the code for every thing that is to be identified. This mutually exclusive feature must be built into any coding system.

The codes for this system are designed with two features in mind. Optimum human oriented use and machine efficiency They are also operable i.e., they are adequate for present and anticipate data processing both for machine and human use.

Input /Output design :

is a part of overall system design, which requires very careful attention. The main objectives of input design are:

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

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also to provide a permanent hard copy of these results for later consultation.

The various types of outputs are required by this system are given below:

- External outputs, whose destination is outside the concern and which require special attention because they, project the image of the concern.
- Internal outputs, whose destination is within the concern and which require careful design because they are the user's main interface within the computer.
- Operation outputs, whose use is purely within the computer department. E.g. program listings, usage statistics etc.

4.3 SDLC

Software Development Life Cycle (SDLC) is a framework that defines the steps involved in the development of software at each phase. It covers the detailed plan for building, deploying and maintaining the software

SDLC defines the complete cycle of development i.e. all the tasks involved in planning, creating, testing, and deploying a Software Product.

4
5
6
1
2
3
THE
SOFTWARE
DEVELOPMENT
CYCLE

Figure 4.1: Above image depicting the planning step

SDLC Phases

Given below are the various phases:

- Requirement gathering and analysis
- Design
- Implementation or coding
- Testing
- Deployment
- Maintenance

Requirement Gathering and Analysis

During this phase, all the relevant information is collected from the customer to develop a product as per their expectation. Any ambiguities must be resolved in this phase only.

Business analyst and Project Manager set up a meeting with the customer to gather all the information like what the customer wants to build, who will be the end-user, what is the purpose of the product. Before building a product a core understanding or knowledge of the product is very important.

Once the requirement gathering is done, an analysis is done to check the feasibility of the development of a product. In case of any ambiguity, a call is set up for further discussion.

Once the requirement is clearly understood, the SRS (Software Requirement Specification) document is created. This document should be thoroughly understood by the developers and also should be reviewed by the customer for future reference.

Design

In this phase, the requirement gathered in the SRS document is used as an input and software architecture that is used for implementing system development is derived.

Implementation or Coding

Implementation/Coding starts once the developer gets the Design document. The Software design is translated into source code. All the components of the software are implemented in this phase.

Testing

Testing starts once the coding is complete and the modules are released for testing. In this phase, the developed software is tested thoroughly and any defects found are assigned to developers to get them fixed.

Deployment

Once the product is tested, it is deployed in the production environment or first UAT (User Acceptance testing) is done depending on the customer expectation.

Maintenance

After the deployment of a product on the production environment, maintenance of the product i.e. if any issue comes up and needs to be fixed or any enhancement is to be done is taken care by the developers.

4.4 SOFTWARE ENGG. PARADIGM APPLIED

Software engineering is a layered technology. The foundation for software engineering is the process layer. Software engineering processes the glue that holds the technology layers together and enables rational and timely development of computer software. Process defines a framework for a set of key process areas that must be established for effective delivery of software engineering technology.

Software engineering methods provide the technical how-to's for building software. Methods encompass a broad array of tasks that include requirements analysis, design, program construction, testing and support. Software engineering tools provide automated or semi-automated support for the process and the methods. When tools are integrated so that information created by one tool can be used by another tool, a system for the support of software development, called computer-aided software engineering is established.

The following paradigms are available:

1. The Waterfall Model

2. The Prototyping Model

3. The Spiral model

Etc.

4.4.1 The Prototype model

The prototype model requires that before carrying out the development of actual software, a working prototype of the system should be built. A prototype is a toy implementation of the system. A prototype usually turns out to be a very crude version of the actual system, possible exhibiting limited functional capabilities, low reliability, and inefficient performance as compared to actual software. In many instances, the client only has a general view of what is expected from the software product. In such a scenario where there is an absence of detailed information regarding the input to the system, the processing needs, and the output requirement, the prototyping model may be employed.

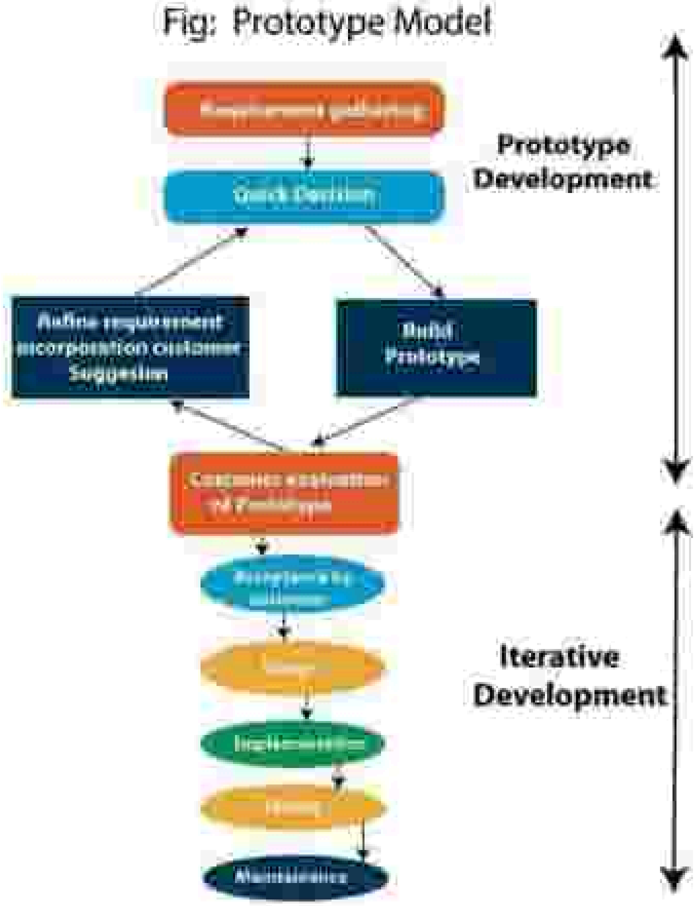


Figure 4.7 Prototype model

4.4.1.1 Advantage of Prototype Model

1. Reduce the risk of incorrect user requirement
2. Good where requirement are changing/uncommitted
3. Regular visible process aids. Presidency
4. Support early product marketing.
5. Reduce Maintenance cost.

6. Errors can be detected much earlier as the system is made side by side.

4.4.1.2 Disadvantage of Prototype Model

1. An unstable/badly implemented prototype often becomes the final product.
2. Require extensive customer collaboration
   - Costs customer money
   - Needs committed customer

   - May be too customer specific, no broad market
3. Difficult to know how long the project will last.
4. Easy to fall back into the code and fix without proper requirement analysis, design, customer evaluation, and feedback.
5. Prototyping tools are expensive.
6. Special tools & techniques are required to build a prototype.
7. It is a time-consuming process.

4.5 DFD

DFD is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present. Specific operations depending on the type of data can be explained by a flowchart. Data Flow Diagram can be represented in several ways.

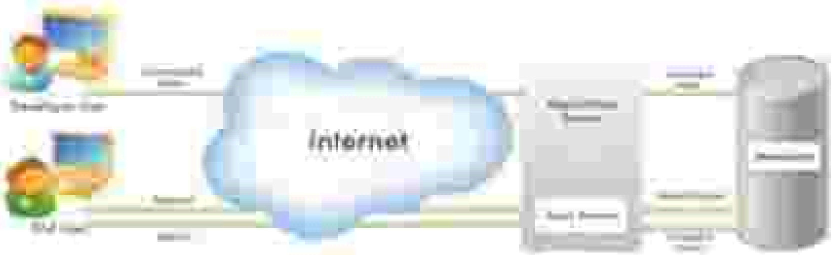


Figure 4.3 [illegible]

Level0 DFD :

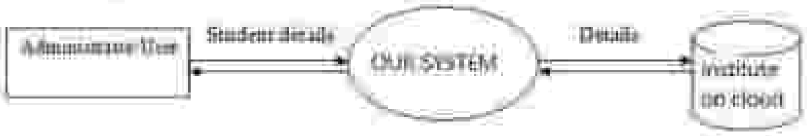


Figure 4.4 Context level

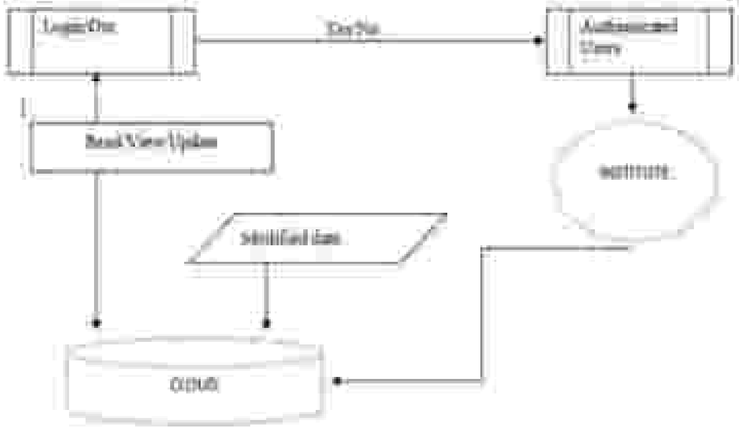


Figure 4.5 DFD 2 level

4.6. UML use case diagram

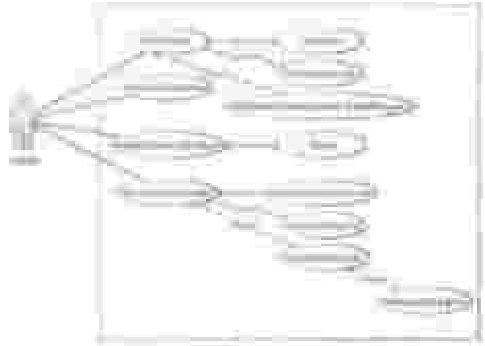


Figure 4.6 UML use case

4.7 ER DIAGRAM

An Entity -relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram has three main components:

1. Entity.
2. Attribute
3. Relationship

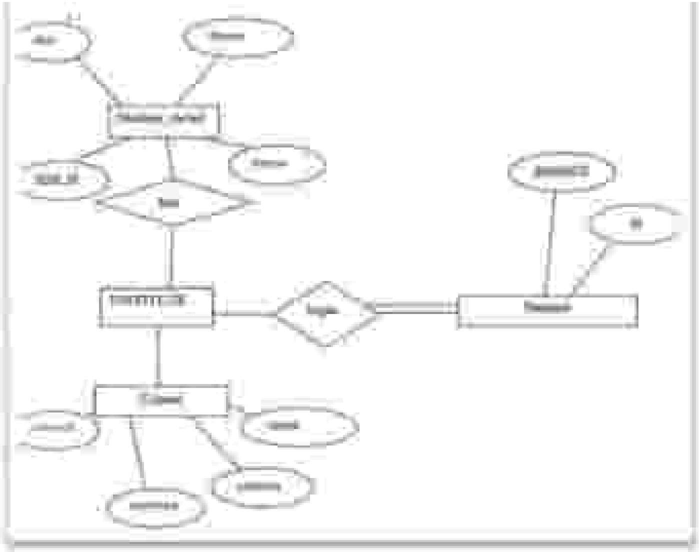


Figure 4.7:ER diagram of system

4.7.1 ER- Diagram Notations

ER- Diagram is a visual representation of data that describe how data is related to each other.

- **Rectangles:** This symbol represent entity types
- **Ellipses :** Symbolrepresent attributes
- **Diamonds:** This symbolrepresents relationship types
- **Lines:** It links attributes to entity types and entity types with other relationship types
- **Primary key:** attributes are underlined
- **Double Ellipses:** Represent multi-valued attributes

CHAPTER 5

REPORT

5.1 GIST

The diagram **figure 5.1,** depicting our system.
We have designed and developed an easy, Useful, reliable system.

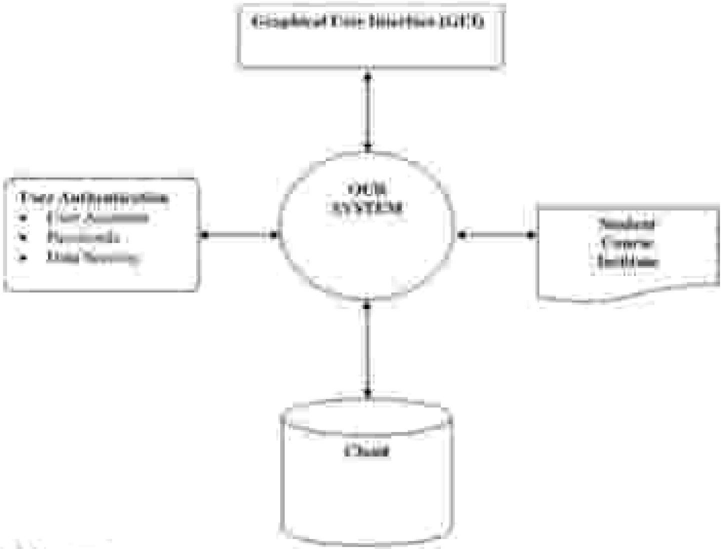
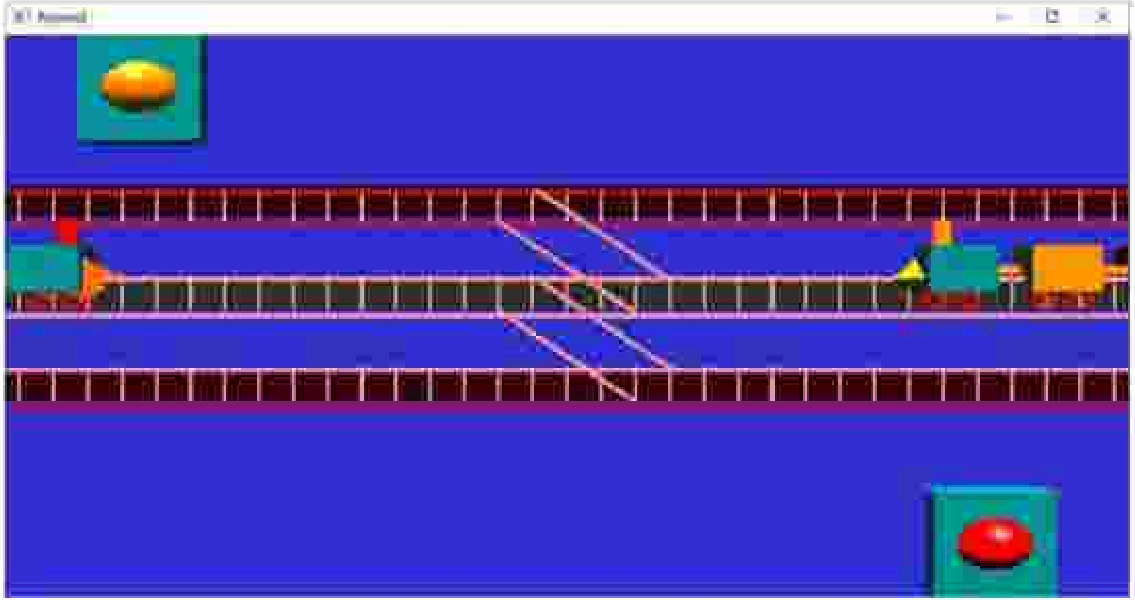
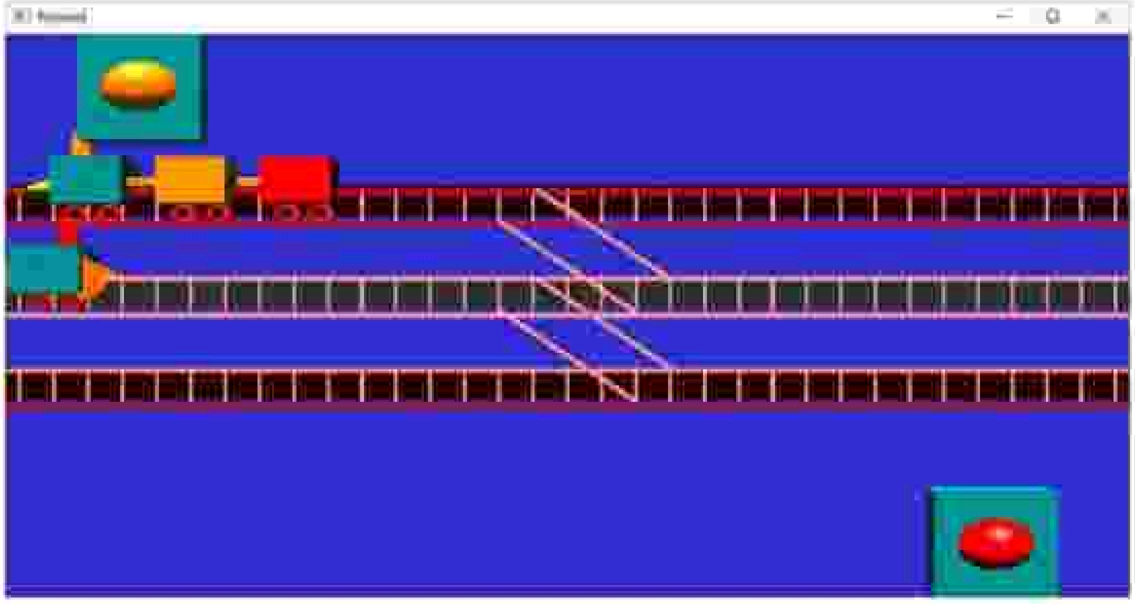


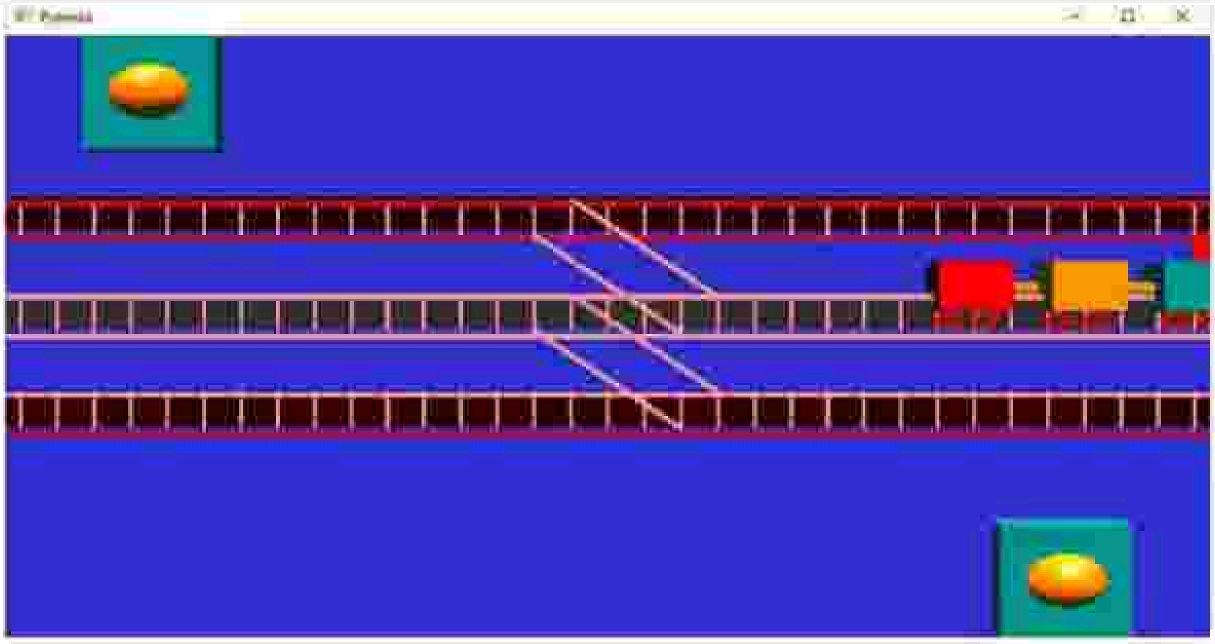
Figure 5.1 System

This gives a high level view of the system with the main components and the services they provide and how they communicate. It consists of the general graphical user interface facilities.

OUTPUT OF THE PROGRAM







Conclusions

The project "Train signal" clearly demonstrates the usage of OpenGL library.

Train signal simulation has been shown clearly and train crossing has been demonstrated using OpenGL.

Finally we conclude that this program clearly illustrate the concepts of openGL and has been completed successfully and is ready to be demonstrated.

Chapter 6:Coding

Source Code

```
#include
<windows.h>
#include<string.h>
#include<stdarg.h>
#include<stdio.h>
#include <glut.h>
#include <math.h>


static double x=0.0;
static double move=-60;
bool newModel=false;
bool sig_1=false;
bool


goUp=false;
```

```
bool goDown=false;


float x1=11,x2=-11; //starting train location
static float y_1=0;
static float y_2=0;

void
stroke_output(GLfloat x, GLfloat y, char *format,...)
{
        va_list args;
        char buffer[200], *p;
        va_start(args, format);
        vsprintf(buffer, format, args);
        va_end(args);
        glPushMatrix();
        glTranslatef(-2.5, y, 0);
        glScaled(0.003, 0.005, 0.005);
        for (p = buffer; *p; p++)
   glutStrokeCharacter(GLUT_STROKE_ROMAN,
   *p);
        glPopMatrix();
```

Train Signal

```
}

void track(){

glPushMatrix();
glScaled(100.8,0.1,0.1);
glutSolidCube(1)
;glPopMatrix();



}

void mat(){
        glPushMatrix();
glScaled(100,0.35,1);
glBegin(GL_POLYGO
N);glVertex2f(-1,-1);
glVertex2f(1,-1);
glVertex2f(1,1);
glVertex2f(-1,1);
glEnd();
```

```
glPopMatrix();

}

void stripes(){

glPushMatrix();
glScaled(0.1,1.57,0.1);
glutSolidCube(.5);
glPopMatrix();

}

void tyre(){



glPushMatrix();
glScaled(0.5,0.5,0.5);
glutSolidTorus(0.1,0.3,20,20);
```

```
glPopMatrix();

}


void train(){

// engine

// chimni

glPushMatrix();
glTranslatef(-
3.7,1.5,0);

//glutSolidSphere(0.5,20,20);
glScaled(0.2,3,0.2);
glRotatef(90,1,0,0);

glutSolidTorus(0.2,0.6,50,50);
glPopMatrix();
```

```
glPushMatrix();
glTranslatef(-
3.5,1,0);
glColor3f(0.0,1,1);
glutSolidCube(1);
glPopMatrix();


glPushMatrix();
glTranslatef(-
4.0,0.8,0.0);
glRotatef(-45,0,0,1);
glScaled(0.6,.6,.6);
glColor3d(1,1,0);
glutSolidTetrahedron();
glPopMatrix();
glPushMatrix();
glTranslatef(0.3,1.0,0);
glRotatef(135,0,0,1);
glScaled(3.2,0.1,0.1);
glutSolidCube(0.5);
glPopMatrix();
```

```
}

void signals(){

glPushMatrix();
glScaled(0.7,0.7,0.7);
glutSolidSphere(0.5,20,20);
glPopMatrix();

glPushMatrix();

glColor3f(0,1,1);
```

```
glScaled(1.0,1.3,0.3);
glutSolidCube(1.0);
glPopMatrix();




}


int angX,angY,angZ=0;



// Main Drawing


void trainSimulation()
{
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glLoadIdentity();
   glTranslatef(0.0f,0.0f,-
   13.0f);



        glRotatef(angX,0,1,1);
        glRotatef(angY,1,0,1);

        glRotatef(angZ,1,1,0);

//    track 1

        glPushMatrix();

glTranslatef(0,-1,0,0);

glPushMatrix();
glTranslatef(0,3.7,-
10);track();
glPopMatrix();

glPushMatrix();
glTranslatef(0,3.5,-
10);
```

```
mat();
glPopMatrix()
;

glPushMatrix();
glTranslatef(0,3,-
10);track();
glPopMatrix();
float j;
// Stripes
for(j=0;j<=50;j+=0.5)
{
glPushMatrix();
glColor3f(1,1,1);
glTranslatef(-10+j,3.4,-
10);stripes();
glPopMatrix();
}

glPopMatrix();
```

```
          glPushMatrix();



//2nd track
glPushMatrix();
glTranslatef(0,0.8,-
10);track();
glPopMatrix();

glPushMatrix();
glTranslatef(0,0.5,-
10);mat();
glPopMatrix();

glPushMatrix();
glTranslatef(0,0,-
10);track();
glPopMatrix();
```

```
//2nd Stripes
for(j=0;j<=50;j+=0.5)
{
glPushMatrix();
glColor3f(1,1,1);
glTranslatef(-10+j,0.5,-
10);stripes();
glPopMatrix();
}

glPopMatrix();



// track 3
        glPushMatrix();

glTranslatef(0,1.0,0);



//3rd track
glPushMatrix();
glTranslatef(0,-2.2,-
10);
```

```
track();
glPopMatrix()
;

glPushMatrix();
glTranslatef(0,-2.5,-
10);glColor3f(1,0,0);
mat();
glPopMatrix()
;

glPushMatrix();
glTranslatef(0,-3,-
10);track();
glPopMatrix();

//3rd Stripes

for(j=0;j<=50;j+=0.5){
glPushMatrix();
glColor3f(1,1,1);
glTranslatef(-10+j,-2.5,-
10);stripes();
```

```
glPopMatrix();
}

glPopMatrix();
/******** Track ENDS *************/
glPushMatrix();
joiner();
glPopMatrix()
;

glPushMatrix();
glTranslatef(-
3.5,2.8,0);

if(sig_1 || sig_3){
glColor3f(1,1,0);
}
else{
glColor3f(1,0,0)
;

}
```

24 Dept. of Computer Science & Engineering.

```
signals();
glPopMatrix()
;


glPushMatrix();
glTranslatef(3.5,-
2.8,0);


if(sig_2 || sig_3){
glColor3f(1,1,0);
}
else{
glColor3f(1,0,0)
;


}


signals();
glPopMatrix()
;



/********** Train Construction ************/
```

```
glPushMatrix();


if(x1<=5 && goUp){
        if(y_1<=1.9)
{y_1+=0.01;
glRotatef(-10,0,0,1);
}
glTranslatef(x1,y_1,-10);


}


else
{       glTranslatef(x1,y_1,-10);

}
glColor3f(1,1,0)
;train();


glPopMatrix();


//move second Train
```

```
glPushMatrix();



if(x2>=-5 &&
        goDown){
        if(y_2>=-2.1){
y_2-=0.01;
glRotatef(-10,0,0,1);
}
glTranslatef(x2,y_2,-10);


}
else
{       glTranslatef(x2,y_2,-10);

}
glRotatef(180,0,1,0);
train();
glPopMatrix();


   glFlush();
   glutSwapBuffers()
   ;
}
```

```
void p()
{




        if(sig_1){
                if(x2>=-11 && x2<=-4)
        {
                goUp=true;
        }

                x1-=0.01;
                }else{
                goUp=false;
                }
```

```
        if(sig_2){
        if(x1<=11 &&
                x1>=4){
                goDown=true;
        }x2+=0.01;
        }else { goDown=false;}


        if(sig_3){
                goUp=true;
                goDown=true;
        x1-=0.01;
        x2+=0.01
        ;
        }



trainSimulation();
```

29 Dept. of Computer Science & Engineering.

```
 }

void P(){

}

void doInit()
{

       /* Background and foreground color */
   glClearColor(0.2,0.2,0.8,0.0);
   glColor3f(.0,1.0,1.0);
   glViewport(0,0,640,480);

       /* Select the projection matrix and reset it then
    setup our view perspective */

   glMatrixMode(GL_PROJECTIO
   N);glLoadIdentity();
   gluPerspective(30.0f,(GLfloat)640/(GLfloat)480,0.1f,200.0f);
```

```
  /* Select the modelview matrix, which we alter with rotatef() */
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  glClearDepth(2.0f);

      glEnable(GL_COLOR_MATERIAL)
  ;glEnable(GL_DEPTH_TEST);
  glDepthFunc(GL_LEQUAL);
}

void display()
{

      glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
      glLoadIdentity();
      glTranslatef(0.0f,0.0f,-13.0f);
      stroke_output(-2.0, 1.7, "Welcome");
          stroke_output(-2.0, 1.0, "p--> Start the Simulation");

          stroke_output(-2.0, -1.1, "q--> quit");
```

```
GLfloat mat_ambient[]={0.0f,1.0f,2.0f,1.0f};
GLfloat mat_diffuse[]={0.0f,1.5f,.5f,1.0f};
GLfloat mat_specular[]={5.0f,1.0f,1.0f,1.0f};
GLfloat mat_shininess[]={50.0f};
glMaterialfv(GL_FRONT,GL_AMBIENT,mat_ambient);
glMaterialfv(GL_FRONT,GL_DIFFUSE,mat_diffuse);
glMaterialfv(GL_FRONT,GL_SPECULAR,mat_specular);
glMaterialfv(GL_FRONT,GL_SHININESS,mat_shininess);

GLfloat lightIntensity[]={3.7f,0.7f,0.7f,1.0f};

GLfloat light_position[]={0.0f,3.0f,2.0f,0.0f};

glLightfv(GL_LIGHT0,GL_POSITION,light_position);
glLightfv(GL_LIGHT0,GL_DIFFUSE,lightIntensity);
```

```
        glFlush();
        glutSwapBuffers()
        ;


}

void menu(int id)
{
        switch(id)
        {
        case
                1:glutIdleFunc(
                P);break;
        case 2:
                glutIdleFunc(p);
                break;


                break;
        case 5:exit(0);
                break;


        }
        glFlush();
        glutSwapBuffers()
        ;
```

```
        glutPostRedisplay();
}

void mykey(unsigned char key,int x,int y)
{



        if(key=='p')
        {
                glutIdleFunc(p);
        }
                if(key=='x')
        {
                angX++;
                glutIdleFunc(p)
                ;
        }
                if(key=='y')
        {
                        angY++;
                glutIdleFunc(p)
                ;
        if(key=='z')
{
                angZ++;
        glutIdleFunc(p)
        ;
}



        if(key=='X')
{
        angX--;
        glutIdleFunc(p)
        ;
}
        if( key=='Y')
{
                angY--;
        glutIdleFunc(p)
        ;
}
        if(key=='Z')
{
```

```
}

if(key=='1')
{
sig_1=!sig_1;
}

if(key=='2')
{
sig_2=!sig_2;
}

if(key=='3')
{
```

Train Signal

```
sig_1=false;
        sig_2=false
;sig_3=!sig_3;


}




if(key=='P')
{
        glutIdleFunc(P);
}



if(key=='r')
{x1=11;
x2=-11;
y_1=0
;
y_2=0
;
angX=angY=angZ=0;
```

```
        }



        if(key=='q'||key=='Q')
        {
                exit(0);
        }






}

int main(int argc, char *argv[])
{
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_DOUBLE|GLUT_RGB);
   glutInitWindowSize(1000,480);
   glutInitWindowPosition(0,0);
   glutCreateWindow("Pyramid");
```

```
  glutDisplayFunc(display);
      glEnable(GL_LIGHTING);
      glEnable(GL_LIGHT0);
      glShadeModel(GL_SMOOT
      H);
      glEnable(GL_DEPTH_TEST
      );
      glEnable(GL_NORMALIZE)
      ; glutKeyboardFunc(mykey);
      glutCreateMenu(menu);
  glutAddMenuEntry("Start    'p'",2);
      glutAddMenuEntry("Reverse The Simulation
      'P'",1);

      glutAddMenuEntry("Quit          'q'",5);
      glutAttachMenu(GLUT_RIGHT_BUTT
      ON);doInit();
  glutMainLoop()
      ;return 0;
}
```

CHAPTER 7

TESTING

7.1 INTRODUCTION

Software Testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The increasing visibility of software as a system element and the attendant "costs" associated with a software failure are motivating forces for well planned, thorough testing.

7.1.1 Testing Objectives

The following are the testing objectives:

-Testing is a process of executing a program with the intent of finding an error.

-A good test case is one that has a high probability of finding an as-yet-undiscovered error.

-successful test is one that uncovers an as yet undiscovered error.

7.1.2 Testing Principles

The basic principles that guide software testing are as follows:

-All tests should be traceable to customer requirements.

-Tests should be planned long before testing begins.

-The parate principle applies to software testing.

Pareto principle states that 80 percent of all errors uncovered during testing will likely be traceable to 20 percent of all program components.

Testing should begin "in the small "and progress toward testing "in the large."

Exhaustive testing is not possible.

7.2 LEVEL OF TESTING

There are different levels of testing

->Unit Testing

->Integration Testing

->System Testing

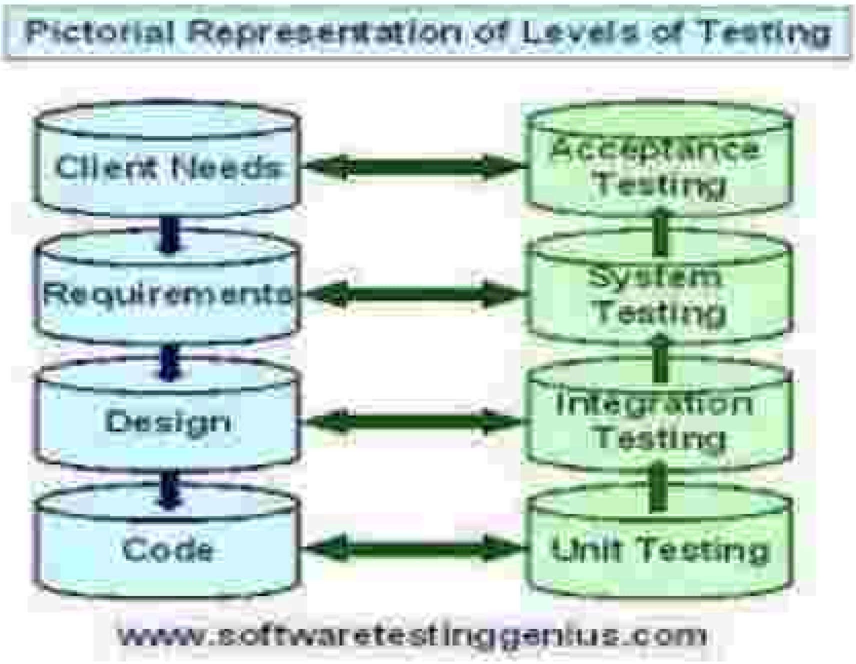


Figure 7.1: Testing pyramid

7.2.1 Unit testing

Unit testing focuses verification effort on the smallest unit of software design, the module. The important control parts are tested to uncover with in the boundary of the module. The module interface is tested to ensure that the information properly flows into and out of the program unit and boundary conditions are tested to ensure that the modules operate properly at boundaries established to limit or restrict processing. Test date is provided through testing screens.

7.2.2 Integration testing

Integrating testing is a systematic technique for constructing Program structure while conducting tests to uncover error associates with interfacing .The objective is to take unit modules and built a program structure that has been directed by design.

• Integration Testing will test whether the modules work well together.

• This will check whether the design is correct.

• Integration can be done in 4 different ways:

7.2.3 System testing

System testing is the process of testing the completed software as a part of the environment it was created for. It is done to ensure that all the requirements specified by the customer are met. System testing involves functional testing and performance testing.

• System Testing will contain the following testing :

- ➢ Functional Testing.
- ➢ Performance Testing.

• Function Testing will test the implementation of the business needs.

• Performance Testing will test the non-functional requirements of the system like the speed, load etc

7.3 SOME IMPORTANT OBSERVTIONS

7.3.1 System Testing and Validation Results.

System testing was done after the system was duly coded. Individual modules of the system were checked to ensure they are fully functional units before the integrating them. This was done by examining each unit; each script was checked to ensure that it functions as required and that it performed exactly as intended. The success of each individual unit gave us the go ahead to carryout integration testing.

The system was validated using a short questionnaire that was filled by representatives of the users who were let to interact with the system using test data and provided feedback about the system features. This was done to assess if the system met their needs and requirements as regards. It was found out that the system performed in conformance to the then defined user needs and requirements. Results of the validation are shown as percentages of respondents against each requirement.

7.3.2 Testing Test Scenarios

1.Check if the page load time is within the acceptable range.
2.Check the page load on slow connections.
3.Check the response time for any action under a light, normal, moderate, and heavy load conditions.
4.Check the performance of database stored procedures and triggers.
5.Check the database execution time.
6.Check for load testing of the application.
7.Check for the Stress testing of the application.
8. Check CPU and memory usage under peak load conditions.
We have checked for scenarios and find that our system performing well in the circumstances.

7.4 TEST CASE RESULT SUMMARY

TestCase#	Description	Result
TC#1	Loading the home page	Passed
TC#2	Login	Passed
TC#3	Validating	Passed
TC#4	Content	Passed
TC#5	Course page loading	Passed
TC#6	Reports page loading	Passed
TC#7	Logout	Passed

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

8.1 CONCLUSION

A software project means a lot of experience. I learned a lot through this project. This project has sharpened our concept cloud computing.
It provides easy methods to manage the load of work easily for the users.
It is much fast and more efficient as the data once entered can be used and accessed easily.
This project has given me an ample opportunity to design, code, test and implements an application. This has helped in putting into practice of various Software Engineering principles concepts like maintaining integrity and consistency of data.

8.2 FUTURE SCOPE

- The Future scope is to make the system more user friendly and enhanced.
- Further, the faculty can upload the videos of their lectures on to this site and students who had missed those classes can view those videos.
- And we will make Mobile app for our system.
- I will add Helping BOT in the system.
- Online examination module would be introduced to conduct online examination.

Bibliography

WE HAVE OBTAINED INFORMATION FROM MANY RESOURCES TO DESIGN AND IMPLEMENT OUR PROJECT SUCCESSIVELY. WE HAVE ACQUIRED MOST OF THE KNOWLEDGE FROM RELATED WEBSITES. THE FOLLOWING ARE SOME OF THE RESOURCES :

TEXT BOOKS :

INTERACTIVE COMPUTER GRAPHICS A TOP-DOWN APPROACH

-By Edward Angel.

COMPUTER GRAPHICS,PRINCIPLES & PRACTICES

- Foley van dam

- Feiner hughes

WEB REFERENCES:

http://jerome.jouvie.free.fr/OpenGl/Lessons/Lesson3.php
http://google.com
http://opengl.org