INDEX

INTRODUCTION OF SOFTWARE ENGINEERING

Software Engineering is the process of designing, developing, testing, and maintaining software. It is a systematic and disciplined approach to software development that aims to create high-quality, reliable, and maintainable software.

1. Software engineering includes a variety of techniques, tools, and methodologies, including requirements analysis, design, testing, and maintenance.
2. It is a rapidly evolving field, and new tools and technologies are constantly being developed to improve the software development process.
3. By following the principles of software engineering and using the appropriate tools and methodologies, software developers can create high-quality, reliable, and maintainable software that meets the needs of its users.
4. Software Engineering is mainly used for large projects based on software systems rather than single programs or applications.
5. The main goal of Software Engineering is to develop software applications for improving quality, budget, and time efficiency.
6. Software Engineering ensures that the software that has to be build should be consistent, correct, also on budged, on time, and within the required requirements.

EXPERIMENT - 1

1. **Identify project scope and objective of given problem:**
   a. College Automation System
   b. Banking Management System

COLLEGE AUTOMATION SYSTEM

INTRODUCTION:

College Automation System is a software that helps both the students and the management authorities of the college. Our College Automation System is capable of storing the details of he students and the teachers and also maintain their details in a dynamic order. This software can help us explore all the activities happening inside the college which we as students do not have any knowledge about. It can handle the details of students, teachers and head of the departments. In this system the HOD can maintain every detail of a particular student in his department. He can also post any notice corresponding to his department. He can also grant a student his attendance and allowance to appear for the examination. There is also a question and answer portal in our project in where anybody is able to post a question and anybody is able to answer that question. In case of a student the name and the department of the student will also be displayed in the question bar. This system provides detailed structure of the departments of the college and the facilities of the college. Our system synchronizes the working of all departments

•SCOPE:

1) Attendance Module:

i. Attendance of students can be evaluated on weekly basis.
ii. Instead of taking attendance on notebooks, each faculty can update the attendance of their class through this application.

2) Internal Marks Module: Highest or average of the two internal marks will be calculated directly.

3) Routine Module: While updating the routine the application must provide the names of the faculties available at that allocated time only.

4) E-Learning:

i. PDFs of notes can be uploaded by faculties.

ii. Videos of important lectures of esteemed faculties can be uploaded as well.

iii. Previous Year Questions can be made available here.

5) Notice already published can be updated.

6) Details of the events held or about to be held can be read.

7) Semester marks of students can also be accessed by this application.

8) Assignment

i. Submitted Assignment: It is a sub module of assignment where submitted assignments can be viewed by respective owner of assignment.

ii. Create Assignments: In this module teacher or any users which have access to this module can create assignment and once this assignment is created it will be visible to all the students after that student can upload their assignments once the assignment is open for acceptance.

OBJECTIVE:

The main objective of the College Automation Project is to manage the details of faculties, branches, registration, students, courses, etc. It manages all the information about the faculties, students, login and courses. The project is totally built at administrative end and thus, only the administrator is guaranteed the access. The purpose of the project is to build an application program to reduce the manual work for managing the faculties, branches, login, and registration. It tracks all the details about the registration, students, and courses

WHY IS IT REQUIRED?

Teachers have to maintain records of every student on paper which leads to huge paper work. Sometimes, it is hard to find record of some student through huge pile of papers. The idea mainly focuses on making the reducing the workload of teachers by using centralized database from where every teacher has access to the records and they can add, update or remove information

EXISTENING SYSTEM

Currently, teachers maintain records on papers. Different information is stored in different registers etc. For marks, there is different register or papers, for attendance there is different. This makes the job more time consuming to maintain records at different places. Further, it is difficult to find records of students. Sometimes, if some information of student say contact number is changed, is has to be updated everywhere where contact number of the student is written. Again, this is very tedious task, finding where contact number is stored and updating it

CONCLUSION:

With time schools and colleges need to get digitized. They need to adapt to new technologies and trends in the market and make the education process more comfortable. It is essential to check all the software details before choosing and see if they cater to your needs. It likewise gives the clients to create report of specific modules which required manual mediation beforehand. It eliminates numerous redundancies which occurred in past application with not ready to give adequate information about the activities which were performed by him.

BANKING MANAGEMENT SYSTEM

INTRODUCTION:

Banking system is a way to maintain few records which bank holds in order to keep a track of everything in the bank so a software application is required in order to make the work easier, for example- maintenance of international value of INR and other currency are also a part of the job of banking system. The bank management is also required to act as the currency distributor and to serve the work for the nation's well-being. This application is built to make it easier for the customers to track every transaction that is being made

AIM:

The main aim of an application is to somewhere automate records on the system. It gives all sorts of functions which are required by the bank in order to run a stable system. In addition to that it also helps to manually check the records of the pre-existing system like transactions that are made in the past. The application also changes or manipulates the new data that is being added and is then re-recorded. One can also check their present transactions that are in process and keep a check on their accounts via this application. It's not only useful for the customers but also for the admin

SCOPE:

1) Creating New Accounts: The application can be used to create two different types of accounts by the customers, which are Savings Account and Current Account. It helps save the hustle for the customer to visit the bank physically and create/use these accounts.

2) Depositing Money : As the world is moving towards the limited use of paper currency, depositing or transferring money from one bank to the other will become as easy as clicking a few buttons using this application.

3) Withdrawing Money: Requests can be sent through the application to ask for money transfer as well.

4) Account Holder List: This is a feature for the admin. The admin can view the list of all the account holders.

5) Balance Enquiry: The customer can check their balance via this application.

6) Changing Passwords/PIN: The customer can easily change the passwords and pin numbers using the application.

7) Closing: The customer can close their accounts too using this application

OBJECTIVE:

The main objective of bank management is to build an organic and optimal interaction system between the elements of banking mechanisms with a view to profit. Efficient management can offer high-quality service, and efficient management can be ensured by efficient organization management. So, professional management is impossible without crystallizing the authority & responsibility of all the personnel employed in a bank.

EXISTING SYSTEMS:

The existing bank system is slow as every task is being performed by the human being and comparing the computer task speed with a computer is not fair. The complexity of this systemis increased when an increase in the number of customers and with that there will be a number of transactions will be performed now everything needs to log in to a file for reference in the future which is simply not the kind of scenario we need at this time.

DRAWBACKS OF THE EXISTING SYSTEMS:

- Less security of customer and bank information.
- Require more physical work and manpower.
- All the manual entry and editing will take more time.
- No level of clearance for the different levels of employees.
- Safety of paper documents from the disaster.
- No backup of the information

CONCLUSION:

"BANK MANAGEMENT SYSTEM" undertaken as a project is based on relevant technologies. The main aim of this project is to develop software for bank management system. This project is to develop software for bank management system. This project has been developed to carry out the processes easily and quickly, which is not possible with the manuals systems, which are overcome by this software. This project is developed using VB language and. Hence, it provides the complete solution for the current management system.

EXPERIMENT – 2

2. Develop software requirements specification for (1a.) and (1 b.) problem

SRS FOR COLLEGE AUTOMATION SYSTEM

INTRODUCTION:

The title of the project is "COLLEGE MANAGEMENT SYSTEM" (CMS). CMS is an Internet based application that aims at providing information to all the levels of management within an organization. This system can be used as an information management system for the college. For a given user, the administrator will create a login id & password, using this user can access the system to either upload or download some information from the database. The front-end will be HTML pages with Java Script for client-side validation whereas all business logics will be in PHP reside at middle layer. And these layers will interact with third layer of database, which will be MySQL database. The web server will be Apache. To start working on this project environment required is a server having Apache as web server, MySQL as database and XAMPP as development environment

PROJECT SCOPE AND PRODUCT FEATURES:

This software system will be a College management system for the members of an organization. This system will be designed to maximize the administrative, academic and overall productivity by providing tools to assist in automating the technical procedures and processes, which would otherwise have to be performed manually. By maximizing the users work efficiency and production the system will meet the users' needs while remaining easy to understand and use. It is a user-friendly portal to interact, manage, access the information.

USER CLASSES AND CHARACTERISTICS:

The target audience for CMS product is the college Administrator

- Administrator/students/faculty/staff (Technical/Nontechnical) .The users for this system are: Student

- The Super user of the system. Mainly focuses on administrative and academic related issues. Staff

Introduction

The title of the project is COLLEGE MANAGEMENT SYSTEM (CMS). CMS is an Internet based application that aims at providing information to all the levels of management within an organization. This system can be used as a information management system for the college I For a given user, the administrator will create a login id & password, using this user can access the system to either upload or download some information from the database. The front-end will be HTML pages with Java Script for client-side validation whereas all business logics will be in PHP resides at middle layer. Andthese layers will interact with third layer of database, which will be MySQL database. The web server will be Apache. To start working on this project environment required is a server having Apache asset server, MySQL as database and XAMPP as development environment

1.1 Purpose

The purpose of this document is to present a detailed description of the College Management System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the client and the developers of the system and will be proposed to the Administrative head for its approval.

1.2 Project Scope and Product Features

This software system will be a College management system for the members of an organization. This system will be designed to maximize the administrative, academic and overall productivity by providing tools to assist in automating the technical procedures and processes, which would otherwise have to be performed manually. By maximizing the users work efficiency and production the system will meet the users' needs while remaining easy to understand and use. It is a user-friendly portal to interact, manage, access the information. the users' needs while remaining easy to understand and use. It is a user-friendly portal to interact, manage, access the information.

1.3 Intended Audience and Document Overview

There are different types of intended audience for this document, such as developers, testers, documentation writers and most importantly the users. We have divided the rest of the document into different subsections. We suggest that you begin with understanding the definitions, acronyms and abbreviations, then sequentially go through contents, overview section and proceeding through the detailed description sections that is most pertinent

1.4 Definition's, Acronyms, and Abbreviations

CMS- College Management System,SRS- Software Requirement Specification, IDLEPython, students data entry Ex- Name, Father name, Contact No. etc

2 Overall Description

2.1 Product Perspective

The product will be a standalone application and maybe run on multiple systems within an Internet network. The product will require a keyboard, mouse and monitor to interface with the users. The minimum hardware requirements for the product are specified in this document

2.2 User Classes and Characteristics

The target audience for CMS product is the college Administrator/students/faculty/staff (Technical/Nontechnical) .The users for this system are :

SOFTWARE REQUIREMENT SPECIFICATIONS:

EXTERNAL INTERFACES:

a) Hardware platform:
b) ▪PIII or above
c) ▪RAM of 512 or above MB - Hard Disk 20GB or above
d) Software Platform: Browser :

▪ Mozilla Fire-Fox v12.0 or higher

▪ Google Chrome v27.0.1453.116m or higher

FUNCTIONAL REQUIREMENTS:

a) Login: This section contains students' login menu where students have to login by their username as well as password

b) Mark sheet: This section contains student's stored data. Student can find their marks by entering detail in 'student detail' Option, and after feeling their data he/she may automatically get their marks in 'grades point option'.

c) Menu: This section includes menus for students' details such as student profile, library system, fee report and Mark sheet.

d) Search Page: Here student can search their stored data entering roll no..

e) Student Information: Here student can store their data in database form by entering data into 'student information' section

NON FUNCTIONAL REQUIREMENTS:

a) Performance Requirements: Performance should not be an issue because all of our server queries involve small pieces of data-changing screens will require very little computation and thus will occur very quickly. Server updates should only take a few seconds as long as the phone can maintain a steady signal.

b) Reliability: Must maintain data integrity. Computer crashes and misuse should not affect a user's history

c) Availability: The CMS Portal shall be available, up and running for 24*7 throughout the year except due to the routine maintenance activities.

d) Security Requirements: Administrator and Users with valid credentials will be able to log in to Portal. Administrator will have access to the database structures at back-end. Administrator will have the rights for modifications as well as any updating work for the datasets and website. Access to the various subsystems will be protected by a user log in screen that requires a user name and password to be updated in future.

SRS FOR BANKING MANAGEMENT SYSTEM

•INTRODUCTION:

Online banking system provides is specifically developed for internet banking for Balance Enquiry, Funds Transfer to another account in the same bank, Request for cheque book/change of address/stop payment of cheques, Mini statements (Viewing Monthly and annual statements). The Traditional way of maintaining details of a user in a bank was to enter the details and record them. Every time the user needs to perform some transactions he has to go to bank and perform the necessary actions, which may not be so feasible all the time. It may be a hard-hitting task for the users and the bankers too. The project gives real life understanding of Internet banking and activities performed by various roles in the supply chain. Here, we provide an automation for banking system through Internet. Internet banking system project captures activities performed by different roles in real life banking which provides enhanced techniques for maintaining the required in- formation up-to-date, which results in efficiency. The project gives real life understanding of Internet banking and activities performed by various roles in the supply chain

•USER CLASSES & CHARACTERISTIC:

1) Customers:

The normal users will have an account of fixed or savings and should have a minimum balance of Rs. 500. He can transfer funds to another account of the same bank & may view his monthly or annual statements.

2) Industrialists, Entrepreneur, Organizations academicians: These users will have all the three accounts & should have a minimum balance of20,000 Rs. He can view the statements of his organization or industry

•SOFTWARE INTERFACE:

1) Front End Client: The system is a web-based application client are requiring using modern web browser such as Mozilla Firefox 1.5, PHP.

2) Web Server: The web application will be hosted on one of the apache servers.

3) Back End: We use backend as MY SQL.

•OPERATING ENVIRONMENT:

1) Hard Drive: More than 160 GB.

2) RAM: More than 1 GB

3) Processor: Pentium 4 or Higher

•FUNCTIONAL REQUIREMENTS:

1) Login Process: This module allows valid customers to access the functionalities provided by the bank.

2) Balance Enquiry:. This module maintains the balance details of a particular account.

3) Update Profile: This module allows the customer to update profile of their account.

4) Funds. Transfer: This module allows the customers to transfer funds from one account to another within the same bank.

5) Password Change: This module allows customers to change their password.

6) Mini Statements: This module allows customers to view their transaction details

• **NON-FUNCTIONAL REQUIREMENTS:**

1) Security: The banking system must be fully accessible to only authentic user. It should require pin for entry to a new environment.

2) Reliability: The application should be highly reliable and it should generate all the updated information in correct order.

3) Availability: Any information about the account should be quickly available from any computer to the authorized user. The previously visited customer's data must not be cleared.

4) Maintainability: The application should be maintainable in such a manner that if any new requirement occurs then it should be easily incorporated in an individual module.

5) Portability: The application should be portable on any windows based system. It should not be machine specific.

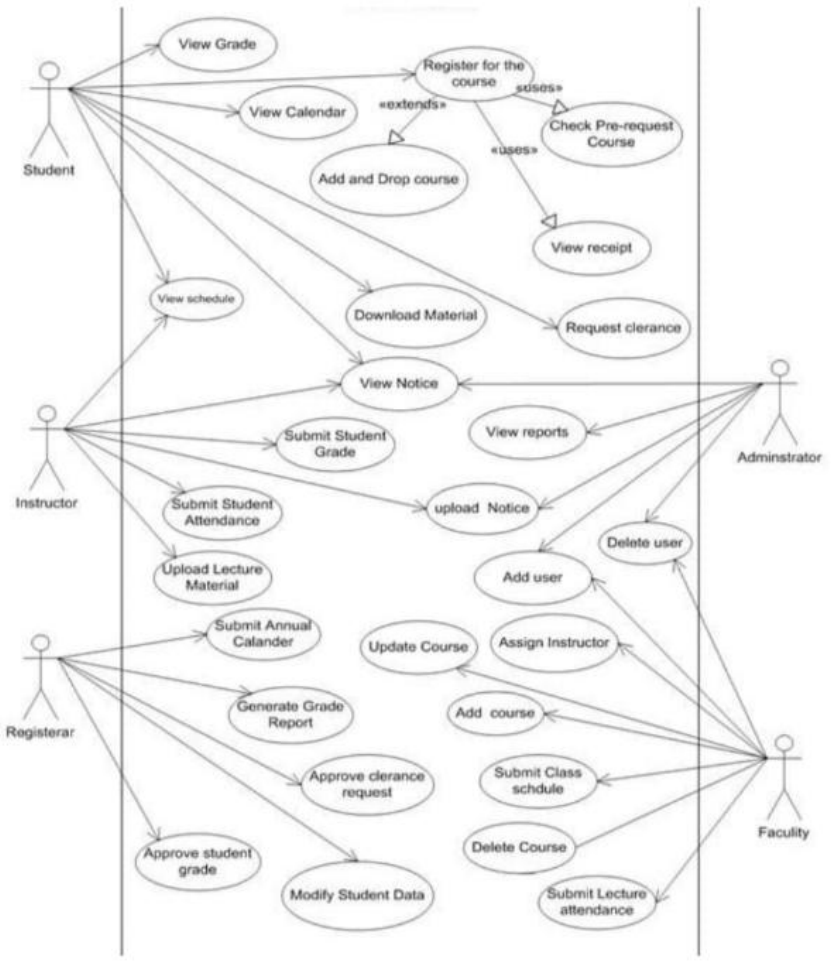
EXPERIMENT – 3

3. Develop UML Use case model for a problem.

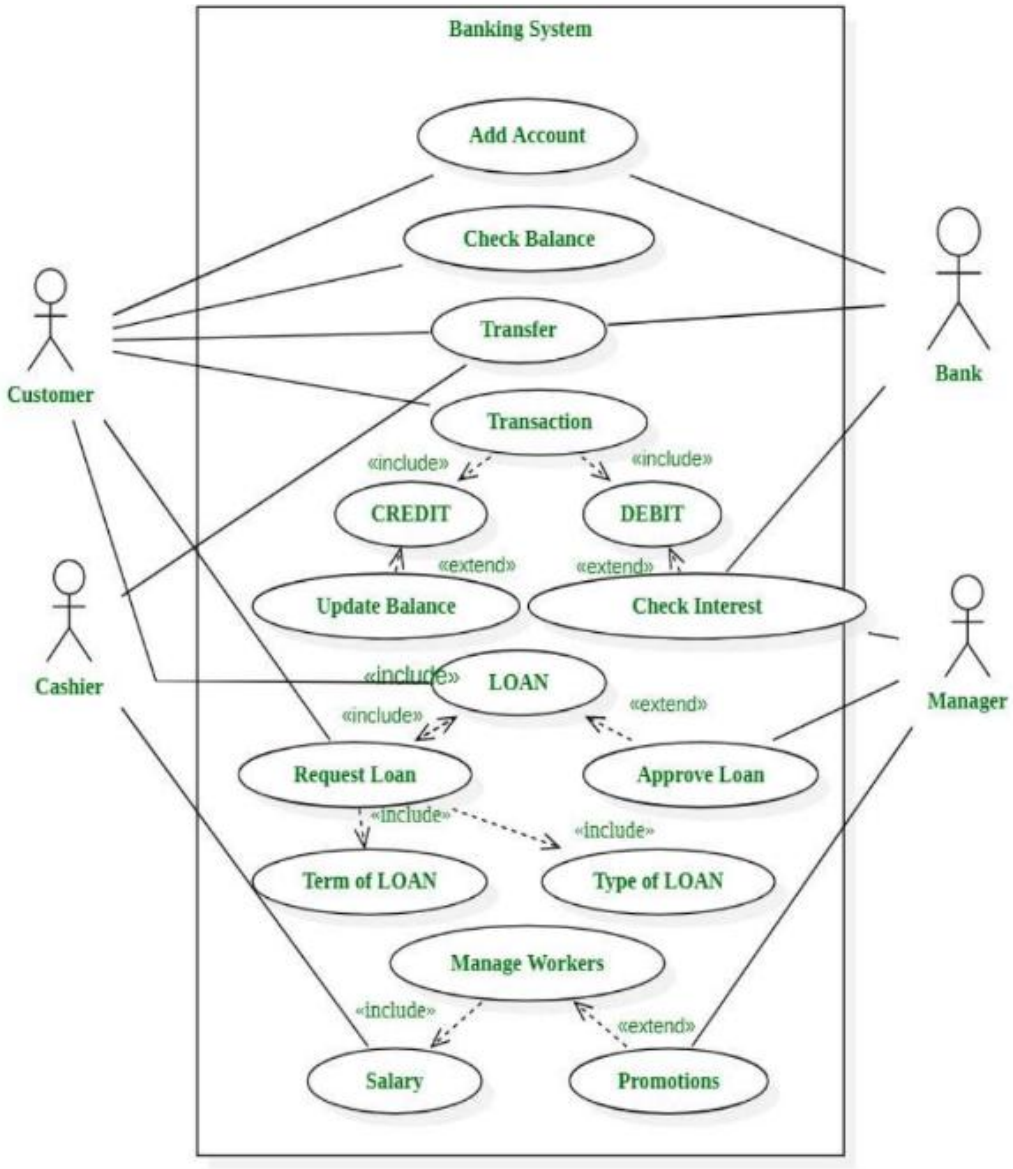
The Use Case Diagram is a UML Diagram where each use-case specifies the behavior expected fromsoftware from the perspective of end-user and relation as well as provides brief overview for different components concerning interaction between use-case, actors

and systems. The Use-Case Diagram is used to prepare, present and understand functional requirements of the system. Use-Case Diagram specifies exact context of the software being developed. It does not specify order in which actions must be performed. Each use-case represents function of system which is either process-automated or manual.

UML USE CASE MODEL FOR COLLEGE AUTOMATION SYSTEM



UML USE CASE MODEL FOR BANKING MANAGEMENT SYSTEM

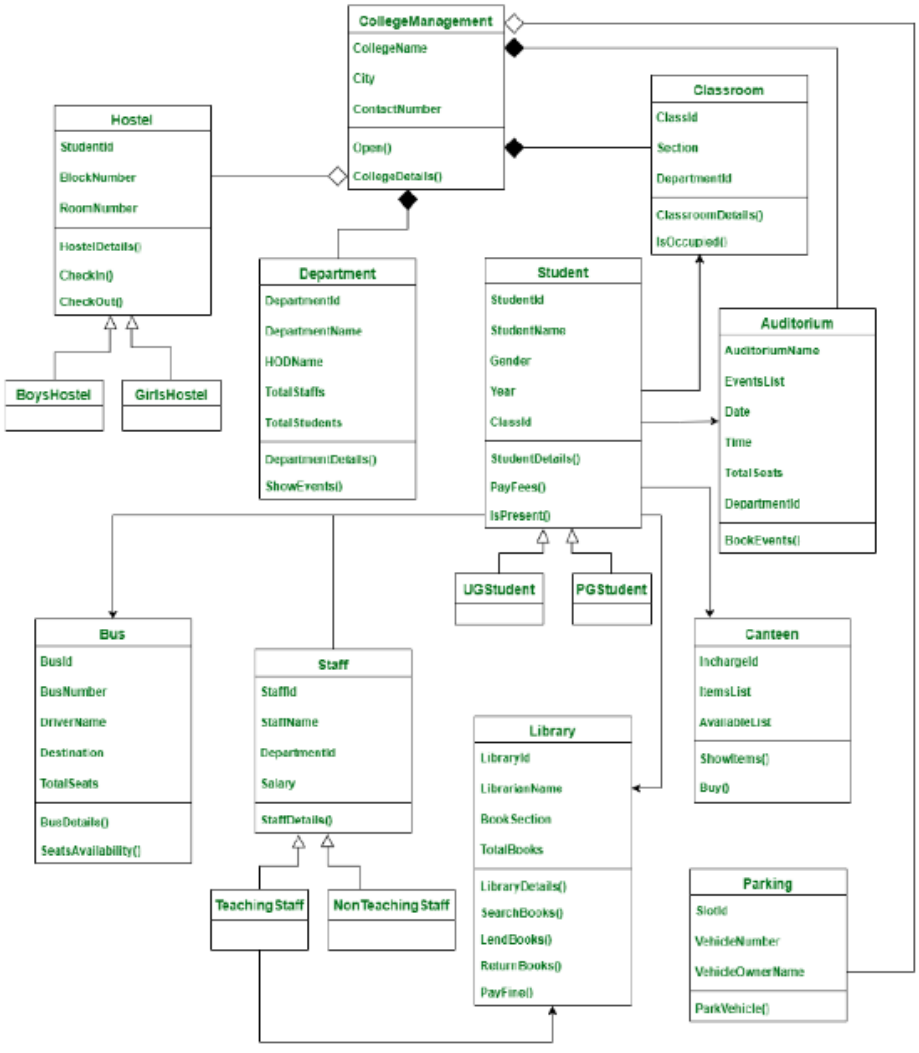

Banking System
Add Account
Check Balance
Transfer
Transaction
«include»
«include»
CREDIT
DEBIT
«extend»
«extend»
Update Balance
Check Interest
«include»
LOAN
«include»
«extend»
Request Loan
Approve Loan
«include»
«include»
Term of LOAN
Type of LOAN
Manage Workers
«include»
«extend»
Salary
Promotions
Customer
Cashier
Bank
Manager

EXPERIMENT – 4

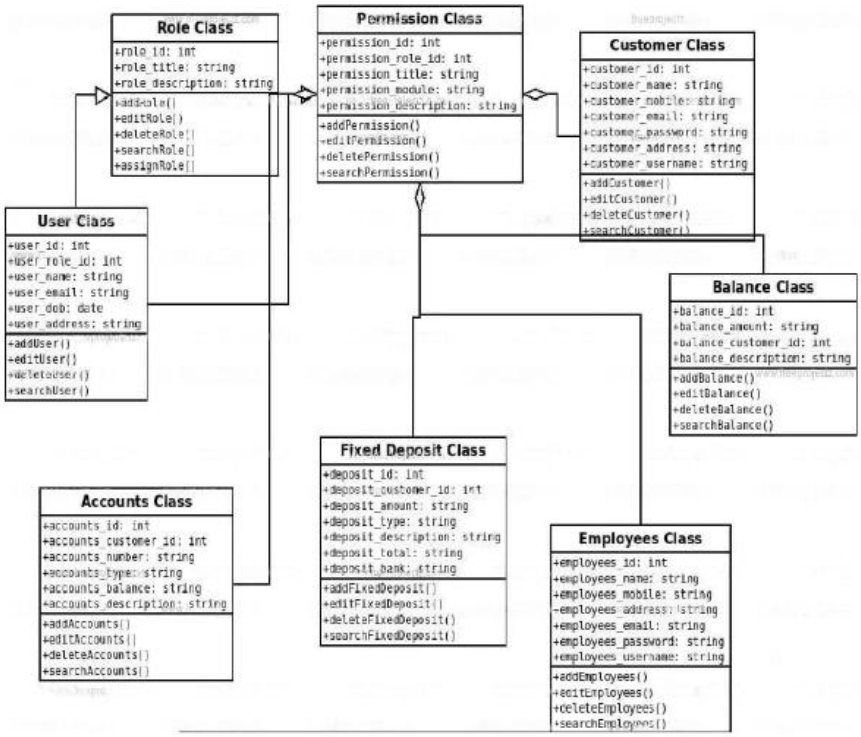
4. Develop Class Diagrams.

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

COLLEGE AUTOMATION SYSTEM



BANKING MANAGEMENT SYSTEM


Role Class
+role_id: int
+role_title: string
+role_description: string
+addRole()
+editRole()
+deleteRole()
+searchRole()
+assignRole()
Permission Class
+permission_id: int
+permission_role_id: int
+permission_title: string
+permission_module: string
+permission_description: string
+addPermission()
+editPermission()
+deletePermission()
+searchPermission()
Customer Class
+customer_id: int
+customer_name: string
+customer_mobile: string
+customer_email: string
+customer_password: string
+customer_address: string
+customer_username: string
+addCustomer()
+editCustomer()
+deleteCustomer()
+searchCustomer()
User Class
+user_id: int
+user_role_id: int
+user_name: string
+user_email: string
+user_dob: date
+user_address: string
+addUser()
+editUser()
+deleteUser()
+searchUser()
Balance Class
+balance_id: int
+balance_amount: string
+balance_customer_id: int
+balance_description: string
+addBalance()
+editBalance()
+deleteBalance()
+searchBalance()
Fixed Deposit Class
+deposit_id: int
+deposit_customer_id: int
+deposit_amount: string
+deposit_type: string
+deposit_description: string
+deposit_total: string
+deposit_bank: string
+addFixedDeposit()
+editFixedDeposit()
+deleteFixedDeposit()
+searchFixedDeposit()
Accounts Class
+accounts_id: int
+accounts_customer_id: int
+accounts_number: string
+accounts_type: string
+accounts_balance: string
+accounts_description: string
+addAccounts()
+editAccounts()
+deleteAccounts()
+searchAccounts()
Employees Class
+employees_id: int
+employees_name: string
+employees_mobile: string
+employees_address: string
+employees_email: string
+employees_password: string
+employees_username: string
+addEmployees()
+editEmployees()
+deleteEmployees()
+searchEmployees()

EXPERIMENT– 5

5. **Represent project scheduling of above-mentioned projects. COLLEGE AUTOMATION SYSTEM**

HIGHLIGHTS:

•College automation system is web-based software. It provides password facility for different users to ensure high level of security.

•Has inbuilt database back up facility for safety and reliability.

•Single database management system.

•Performance analysis of the students or staffs.

•It is user-friendly college management software.

•It regulates the fee management process completely.¬

MODULES:

•Admission or Registration

•User adding/Editing/Deleting

•Student administration

•Fee management

•Letter print

•Reports

•Time scheduling

•Library management

•Financial accounting

•Option poll and feedback¬

FEASIBILITY OF THE APPLICATION:

User- friendly features Easy to navigate environment Effective client-side validation Display of alerts can be deployed on the Web¬ENHANCEMENT:

- Results
- Online admission
- Online examination
- Discussion forum
- Study material

BANKING MANAGEMENT SYSTEM

REQUIREMENTS GATHERING: This involves understanding the requirements of the bank for the management system, such as the features required, the number of users, security requirements, etc.

DESIGNING THE SYSTEM ARCHITECTURE: This involves designing the overall system architecture, including the hardware and software components, database structure, and network configuration.

DATABASE DESIGN: This involves designing the database schema, tables, and relationships between them, based on the requirements of the bank.

USER-INTERFACE DESIGN: This involves designing the user interface for the bank management system, including the screens, forms, and reports required.¬

DEVELOPMENT OF THE SYSTEM: This involves coding the system, based on the design and requirements gathered.

TESTING: This involves testing the system for bugs and errors, and ensuring that it meets the requirements of the bank.

DEPLOYMENT: This involves deploying the system to the bank's infrastructure, including installation, configuration, and training for users.

MAINTENANCE: This involves ongoing maintenance and support for the bank management system, including bug fixes, updates, and enhancements. Based on these tasks, a typical project schedule for a Bank Management System might look like:

- Requirements gathering: 2 weeks
- Designing the system architecture: 2 weeks

- Database design: 4 weeks
- User interface design: 4 weeks
- Development of the system: 12 weeks
- Testing: 4 weeks
- Deployment: 2 weeks
- Maintenance: Ongoing

EXPERIMENT – 6

6. Use any model for estimating the effort, schedule and cost of software project.

CoCoMo (Constructive Cost Model) is a regression model based on LOC, i.e. number of Lines of Code. It is a procedural cost estimate model for software projects and is often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time, and quality. It was proposed by Barry Boehm in 1981 and is based on the study of 63 projects, which makes it one of the best-documented models. The key parameters which define the quality of any software products, which are also an outcome of the CoCoMo are primarily Effort & Schedule:

• Effort: It is the amount of labor that will be required to complete a task. It is measured in person-months units.

• Schedule: This simply means the amount of time required for the completion of the job, which is, of course, proportional to the effort put in. It is measured in the units of time such as weeks, and months.

Different models of CoCoMo have been proposed to predict the cost estimation at different levels, based on the amount of accuracy and correctness required. All of these models can be applied to a variety of projects, whose characteristics determine the value of the constant to be used in subsequent calculations. These characteristics pertaining to different system types are mentioned below. Boehm's definition of organic, semidetached, and embedded systems:

1. Organic

2. Semi-detached

3. Embedded

a) Basic CoCoMo Model

b) Intermediate CoCoMo Model

c) Detailed CoCoMo Model

Some basic formulas for the calculations of the parameters of the CoCoMo model are:

•Effort = a(KLOC)b

•Time = c(Effort)d

•Person required = Effort/ time

// C++ program to implement COCOMO

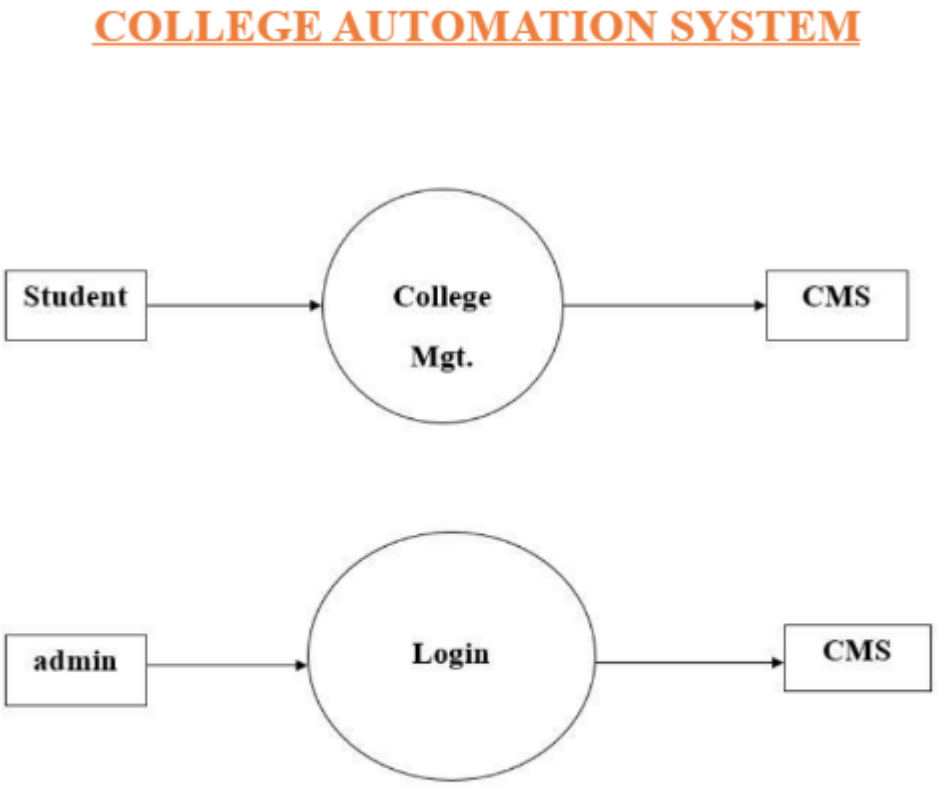
```
#include <iostream>

using namespace

 std;int fround(float x)// For rounding off float to int

{

int a;

x = x + 0.5;

a = x;

return (a);

}

void calculate(float table[][4], int n, char mode[][15], int size) //To calculate parameters

{

float effort, time, staff;

int model;

if (size >= 2 && size <= 50)

model = 0; // organic

else if (size > 50 && size <= 300)

model = 1; // semi-detached

else if (size > 300)

model = 2; // embedded

cout << "The mode is " << mode[model];

effort = table[model][0] * pow(size, table[model][1]);

time = table[model][2] * pow(effort, table[model][3]);

staff = effort / time;
```

```
cout << "\nEffort = " << effort << " Person-Month";
cout << "\nDevelopment Time = " << time << " Months";
cout << "\nAverage Staff Required = " << fround(staff) << " Persons";
}
int main()
{
float table[3][4] = { 2.4, 1.05, 2.5, 0.38, 3.0, 1.12, 2.5, 0.35, 3.6, 1.20, 2.5, 0.32 };
char mode[][15] = { "Organic", "Semi-Detached", "Embedded" };
int size = 4;
calculate(table, 3, mode, size);
return 0;
}
```

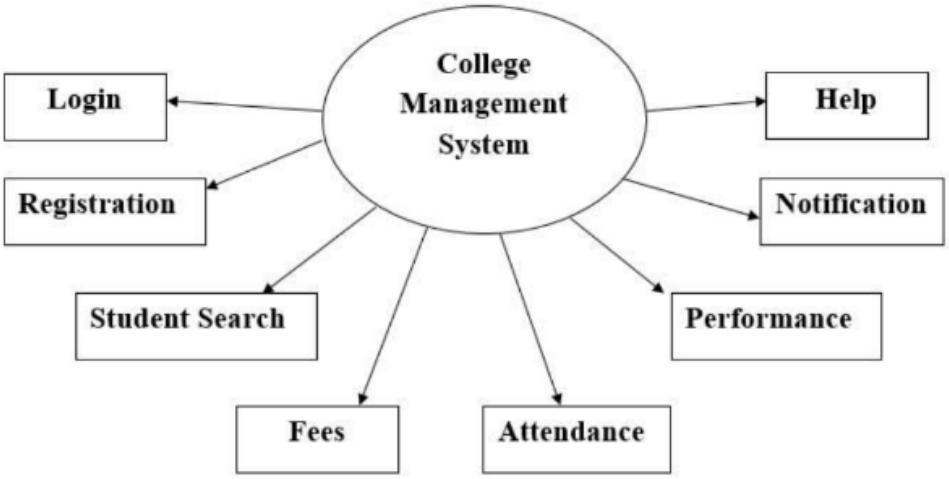
EXPERIMENT – 7

7. Develop DFD model (level-0, level-1 DFD and Data dictionary) of the project.

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyses an existing system or model a new one

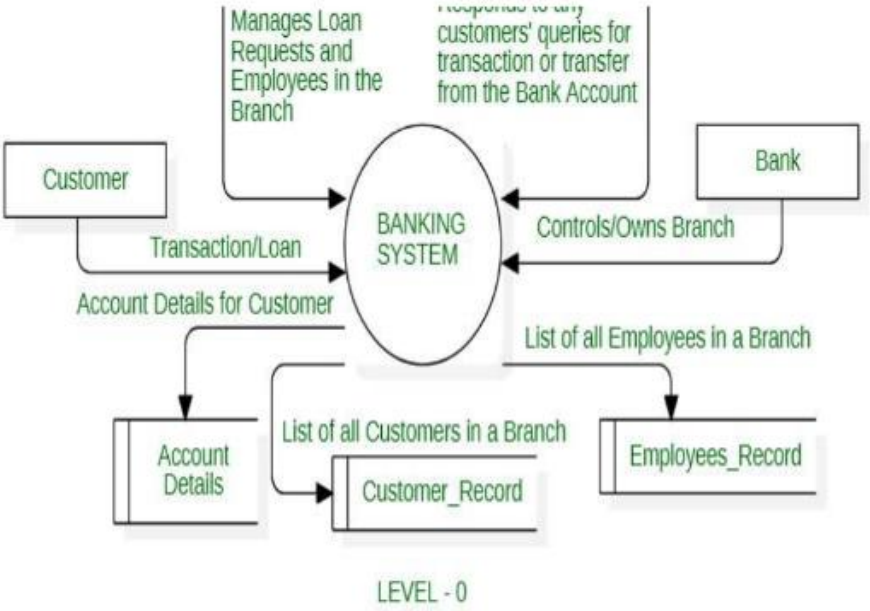


Level 0 DFD

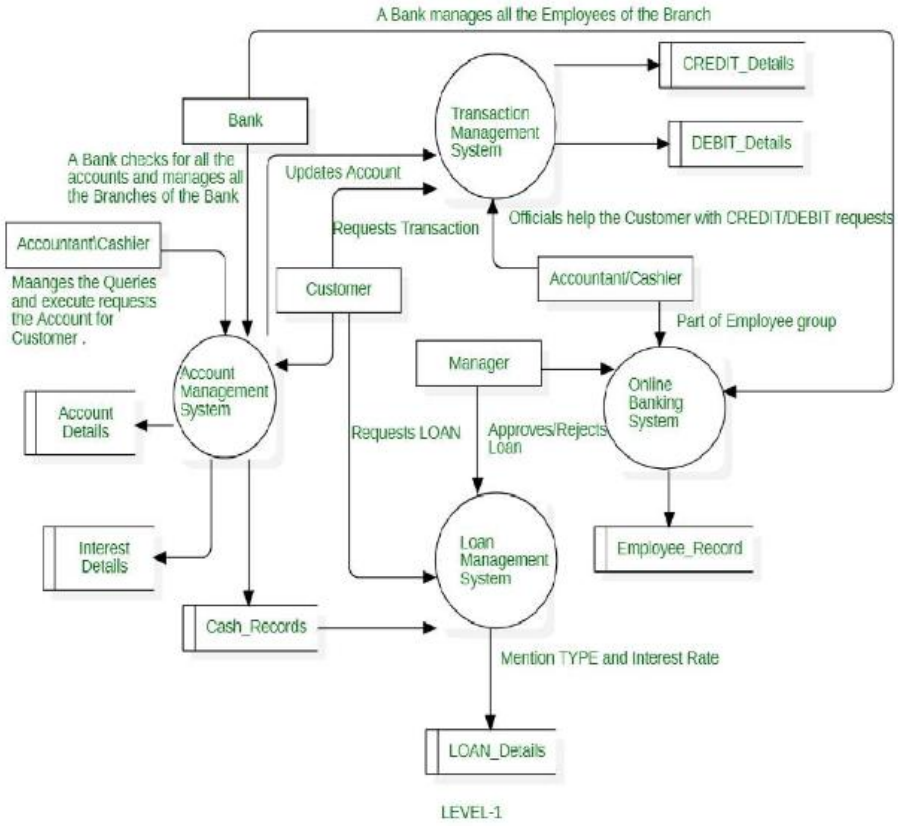


Level 1 DFD

BANKING MANAGEMENT SYSTEM



Level 0 DFD

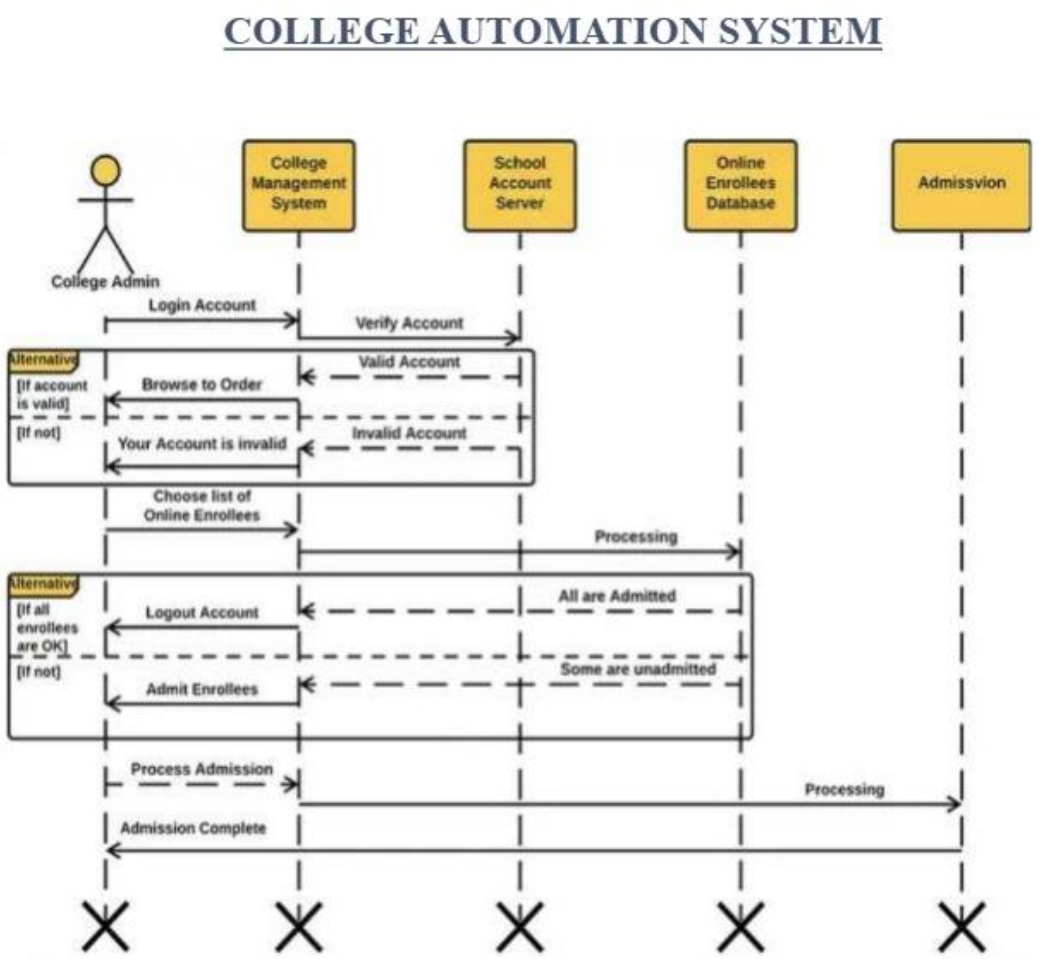

A Bank manages all the Employees of the Branch
Bank
A Bank checks for all the accounts and manages all the Branches of the Bank
Transaction Management System
CREDIT_Details
DEBIT_Details
Updates Account
Requests Transaction
Officials help the Customer with CREDIT/DEBIT requests
Accountant\Cashier
Maanges the Queries and execute requests the Account for Customer .
Customer
Accountant/Cashier
Part of Employee group
Account Management System
Account Details
Manager
Online Banking System
Requests LOAN
Approves/Rejects Loan
Interest Details
Loan Management System
Employee_Record
Cash_Records
Mention TYPE and Interest Rate
LOAN_Details
LEVEL-1

Level 1 DFD

EXPERIMENT – 8

8. Develop sequence diagrams.

UML Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.



EXPERIMENT – 9

9. **Develop Structured design for the DFD model developed.**

COLLEGE AUTOMATION SYSTEM

HOSTEL	
studentID	INT **PRIMARY KEY**
roomNumber	INT
blockNumber	INT
COLLEGE MANAGEMENT	
collegeName	VARCHAR **PRIMARY KEY**
City	VARCHAR
contactNumber	BIGINT
CLASSROOM	
classID	INT **PRIMARY KEY**
Section	VARCHAR
departmentID	INT
DEPARTMENT	
departmentID	INT **PRIMARY KEY**
departmentName	VARCHAR
HOD	VARCHAR
totalStaffs	INT
totalStudents	INT
STUDENT	
studentID	INT **PRIMARY KEY**
studentName	VARCHAR
Gender	VARCHAR
Year	INT
classID	INT
STAFF	
staffID	INT **PRIMARY KEY**
staffName	VARCHAR
departmentID	INT
Salary	INT
LIBRARY	
libraryID	INT **PRIMARY KEY**
LibrarianName	VARCHAR
bookSection	VARCHAR
totalBooks	INT

BANKING MANAGEMENT SYSTEM

CUSTOMER	
customerID	INT **PRIMARY KEY**
customerName	VARCHAR
customerMobile	BIGINT
customerEmail	VARCHAR
customerAddress	VARCHAR
customerUsername	VARCHAR
customerPassword	VARCHAR
EMPOLYEES	
employeeID	INT **PRIMARY KEY**
employeeName	VARCHAR
employeeMobile	BIGINT
employeeEmail	VARCHAR
employeeAddress	VARCHAR
employeeUsername	VARCHAR
employeePassword	VARCHAR
ACCOUNT	
accountID	INT **PRIMARY KEY**
accountNumber	BIGINT
accountType	VARCHAR
accountBalance	FLOAT

EXPERIMENT– 10

10. Develop the waterfall model, prototype model and spiral model of the product.

WATERFALL MODEL

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. The Waterfall model is the earliest SDLC approach that was used for software development. The Waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap. Design The whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. The sequential phases in Waterfall model are –

• Requirement Gathering and analysis − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

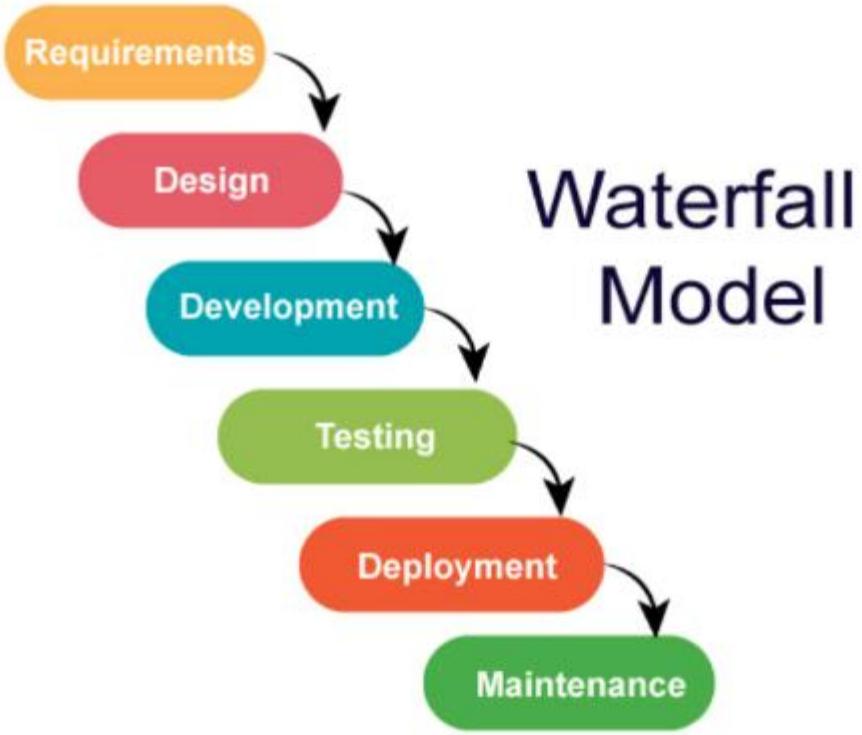
• System Design − the requirement specifications from first phase restudied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

• Implementation − with inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

• Integration and Testing − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

• Deployment of system − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

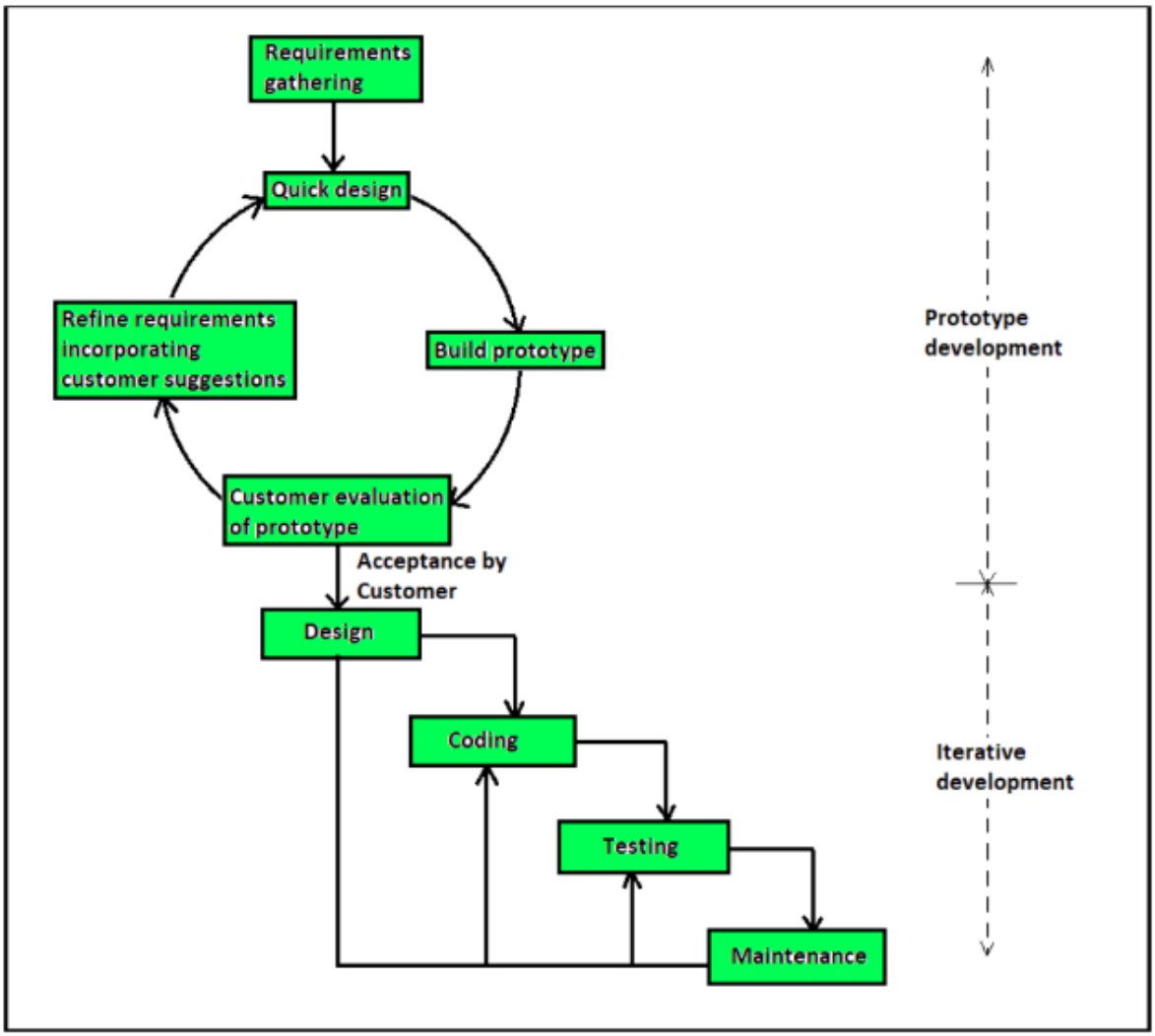
• Maintenance − There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released Maintenance is done to deliver these changes in the customer environment.



PROTOTYPE MODEL

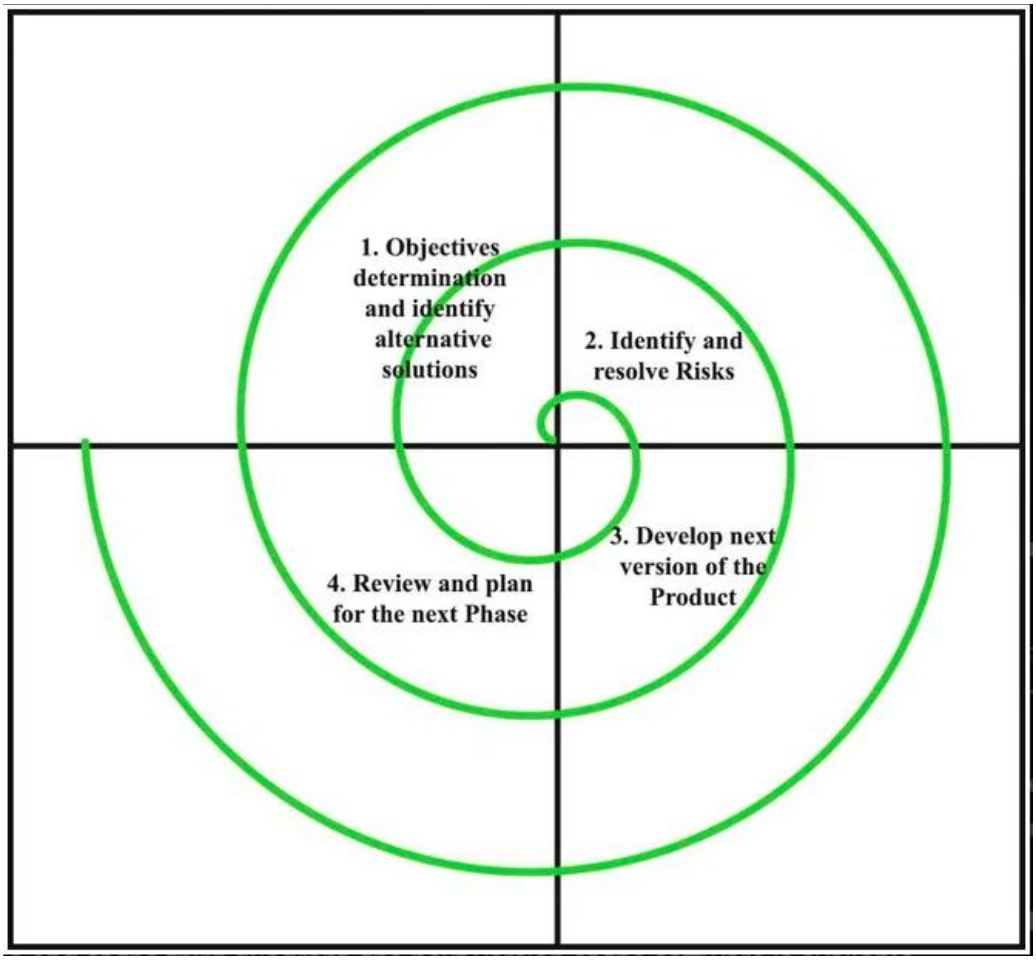
Prototyping is defined as the process of developing a working replication of a product or system that has to be engineered. It offers a small-scale facsimile of the end product and is used for obtaining customer feedback. The Prototyping Model is one of the most popularly used Software Development Life Cycle Models (SDLC models). This model is used when the customers do not know the exact project requirements beforehand. In this model, a prototype of the end product is first developed, tested and refined as per customer feedback repeatedly till a final acceptable prototype is achieved which forms the basis for developing the final product.

In this process model, the system is partially implemented before or during the analysis phase thereby giving the customers an opportunity to see the product early in the life cycle. The process starts by interviewing the customers and developing the incomplete high-level paper model. This document is used to build the initial prototype supporting only the basic functionality as desired by the customer. Once the customer figures out the problems, the prototype is further refined to eliminate them. The process continues until the user approves the prototype and finds the working model to be satisfactory



SPIRAL MODEL

Spiral model is one of the most important Software Development Life Cycle models, which provides support for Risk Handling. In its diagrammatic representation, it looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. Each loop of the spiral is called a Phase of the software development process. The exact number of phases needed to develop the product can be varied by the project manager depending upon the project risks. As the project manager dynamically determines the number of phases, so the project manager has an important role to develop a product using the spiral model. The Radius of the spiral at any point represents the expenses (cost) of the project so far, and the angular dimension represents the progress made so far in the current phase. The below diagram shows the different phases of the Spiral Model:



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

•Objectives determination and identify alternative solutions: Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.

•Identify and resolve Risks: During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.

•Develop next version of the Product: During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.

•Review and plan for the next Phase: In the fourth quadrant, the Customers evaluate the so far developed version of the software. In the end, planning for the next phase is started.

EXPERIMENT– 11

11. Explain with reason which model is best suited for the product.

COLLEGE AUTOMATION SYSTEM

There are various models for developing software for college automation system, and the best-suited model would depend on several factors, such as the size of the project, the development team's experience and skillset, and the project's budget and timeline. However, considering the requirements of a college automation system, the agile model would be the most suitable. The Agile model is a flexible and iterative approach to software development that involves collaboration between the development team and stakeholders. The Agile model focuses on delivering software quickly and responding to changes in requirements. The college automation system is a complex project that requires constant feedback and collaboration between the developers and the stakeholders. The Agile model allows for continuous testing and feedback, making it easier to identify and address any issues or changes needed in the software. Additionally, the agile model is adaptable to changing requirements, making it easier to accommodate changes that may occur during the project. Another advantage of the agile model is that it promotes communication and collaboration among team members. This is particularly important in a college automation system where multiple departments and stakeholders may be involved. The Agile model emphasizes teamwork and encourages the development team to work closely with the stakeholders to ensure that the software meets their needs. Overall, the agile model is the best-suited model for the software development of college automation system due to its flexibility, adaptability, and emphasis on collaboration and communication

BANKING MANAGEMENT SYSTEM

When it comes to developing a software system for banking management, the Waterfall model is the most suitable choice. The Waterfall model is a linear, sequential approach to software development that involves distinct phases, starting from requirements gathering to testing, deployment, and maintenance.

The Waterfall model is best suited for a banking management system due to the critical nature of the system. The system must be accurate, reliable, and secure, as it handles sensitive financial information. The Waterfall model's sequential nature ensures that each phase is completed before moving on to the next, reducing the risk of errors and ensuring that the system meets the required quality standards.

The banking management system requires a comprehensive and detailed set of requirements, as it is subject to strict regulatory requirements. The Waterfall model's requirement gathering phase ensures that all requirements are captured and documented before development begins, reducing the risk of misunderstandings or misinterpretations.

Additionally, the Waterfall model's testing phase ensures that the system is thoroughly tested and validated before deployment. This is crucial in a banking management system, where any errors or bugs can lead to significant financial losses or security breaches.

Furthermore, the Waterfall model provides a clear project timeline and well-defined milestones, making it easier to track progress and ensure that the project stays on schedule. This is important for banking management systems, where the system's timely delivery is crucial for the organization's success.

In summary, the Waterfall model is the best-suited model for the software development of banking management systems due to its emphasis on accuracy, reliability, and security, as well as its detailed requirement gathering and testing phases

EXPERIMENT– 12

12. Develop a working protocol of any of two problem. Problem

1: Slow website loading times

•Identify the source of the problem: Use website speed testing tools like GTmetrix, Google Page Speed Insights or Kingdom to identify the source of the problem. The test results will show the specific areas that need to be optimized.

•Optimize website images: Optimize website images by compressing them to reduce their size. Use image compression tools like TinyPNG or Kraken.io to compress the images without losing their quality.

•Minify CSS and JavaScript files: Minify the website's CSS and JavaScript files to reduce the file size, which will result in faster loading times. Use tools like CSS Minified or UglifyJS to minify the files.

•Use a Content Delivery Network (CDN): Use a CDN to deliver the website's content to users from a server that is geographically closer to them, reducing the loading times. Popular CDN services include Cloud flare, Amazon Cloud Front and Akamai.

•Enable caching: Enable caching on the website to reduce server load and speed up website loading times. Use caching plugins like W3 Total Cache or WP Super Cache to enable caching on Word Press websites.

•Use a faster web hosting service: Switch to a faster web hosting service to improve website loading times. Look for hosting providers that offer faster page loading times, faster server response times, and better uptime guarantees.

Problem 2: Email not working

•Check server status: Verify if the email server is up and running by checking the server status page provided by the email service provider.

•Verify email settings: Verify email settings such as SMTP, POP3 or IMAP configurations, email address, username and password to ensure that they are correctly set up.

•Check email filters: Check email filters to see if any filters are blocking emails from certain senders or domains.

•Check email quota: Check the email quota to ensure that it has not reached the limit. If the email quota is full, the user will not be able to send or receive emails.

•Verify DNS settings: Verify DNS settings and make sure they are correctly set up. Incorrect DNS settings can cause email delivery issues.

•Check spam folder: Check the spam folder in the email client to see if the emails are being delivered there instead of the inbox.

•Contact email service provider: If the issue persists, contact the email service provider's support team for assistance. They can provide additional troubleshooting steps or escalate the issue to their technical team for resolution.

EXPERIMENT– 13

13. Use LOC, FP and Cycloramic Complexity Metric of above-mentioned problem. Problem

1: Slow website loading times

•LOC: The LOC metric could be used to measure the size of the code changes made to the website to optimize its loading times. For example, if the original code base was 500 lines of code and the optimization changes added 100 lines of code, the LOC metric would be 600.

•FP: The FP metric could be used to measure the functionality added to the website to optimize its loading times. For example, if the optimization changes included adding a caching plugin and enabling caching on the website, this could be considered a new function and counted towards the FP metric.

•cycloramic Complexity: The cycloramic Complexity metric could be used to measure the complexity of the code changes made to the website to optimize its loading times. For example, if the code changes included adding conditional statements or loops, this could increase the cycloramic Complexity of the code

Problem 2: Email not working

•LOC: The LOC metric could be used to measure the size of the code changes made to fix the email issue. For example, if the code changes added 50 lines of code to fix the issue, the LOC metric would be 50.

•FP: The FP metric could be used to measure the functionality added to fix the email issue. For example, if the issue was caused by an incorrect email configuration, fixing this configuration could be counted as a new function and added to the FP metric.

•Cyclomatic Complexity: The Cyclomatic Complexity metric could be used to measure the complexity of the code changes made to fix the email issue. For example, if the code changes involved adding conditional statements or loops to handle email filters or quotas, thiscould increase the Cyclomatic Complexity of the code.It's worth noting that while these metrics can be useful for measuring the size, functionality, and complexity of code changes, they should be used in conjunction with other metrics and qualitative analysis to fully evaluate the effectiveness of the solutions.

EXPERIMENT – 14

14. Find Maintainability Index and Reusability Index of above-mentioned problem.

Problem 1: Slow website loading times Maintainability

Index: Suppose the code changes made to optimize the website's loading times had a complexity score of 70, a code duplication score of 90, a code documentation score of 80, and a code structure score of 75. Using the formula provided by SonarQube, the Maintainability Index would be calculated as follows:

MI = (171 - 5.2 * ln(V) + 0.23 * (70) - 16.2 * ln(90) + 50 * sin(sqrt(2.4 * 80)) + 1.5 * 75) * 100 / 171

MI = 72.45

Reusability Index: To calculate the Reusability Index, we would need to analyzethe code changes made and determine how reusable they are. If the code changes included creating reusable functions or components that could be used elsewhere in the website, this would increase the Reusability Index. Problem

2: Email not working

Maintainability Index: Suppose the code changes made to fix the email issue had a complexity score of 60, a code duplication score of 80, a code documentation score of 70, and a code structure score of 65. Using the same formula as before, the Maintainability Index would be calculated as follows:

MI = (171 - 5.2 * ln(V) + 0.23 * (60) - 16.2 * ln(80) + 50 * sin(sqrt(2.4* 70)) + 1.5 * 65) * 100 / 171

MI = 67.38

Reusability Index: To calculate the Reusability Index, we would need to analyse the code changes made and determine how reusable they are. If the code changes included creating reusable functions or components that could be used elsewhere in the website, this would increase the Reusability Index

EXPERIMENT– 15

15. Using any Case Tool find number of statements, depth and complexity of the prototype.

Use of CASE tools accelerates the development of project to produce desired result and helps to uncover flaws before moving ahead with next stage in software development. Requirements of a prototyping CASE tool: Prototyping is beneficial to know the necessities of complicated package product, to demonstrate an idea, to plug new concepts, and so on. The vital options of a prototyping CASE tool are as follows:

- Define user interaction
- Define the system management flow
- Store and retrieve knowledge needed by the system
- Incorporate some processing logic

Features of a good prototyping CASE tool:

There are many complete prototyping tools. However, a tool that integrates with the information wordbook will create use of the entries within the data dictionary, facilitate in populating the information wordbook and make sure the consistency between the planning data and also the paradigm. A decent prototyping tool ought to support the subsequent features:

Since one among the most uses of a prototyping CASE tool is graphical program (GUI) development, prototyping CASE tool ought to support the user to make an interface employing a graphics editor.

- The user ought to be allowed to outline all knowledge entry forms, menus and controls.
- It ought to integrate with the information wordbook of CASE surroundings.
- If doable, it ought to be able to integrate with external user outlined modules written in C or some in style high-level programming languages.
- The user ought to be able to outline the sequence of states through that a created paradigm will run. The user ought to even be allowed to manage the running of the paradigm.
- The run time system of paradigm ought to support mock runs of the particular system and management of the input and output knowledge.