Thadomal Shahani Engineering College

Bandra (W.), Mumbai- 400 050.

CERTIFICATE

Certify that Mr./Miss Raghav. Vikram. Rathi of Computer Department, Semester 5 with Roll No. 2113208 has completed a course of the necessary experiments in the subject Software Engineering. under my supervision in the **Thadomal Shahani Engineering College** Laboratory in the year 2023 - 2024

Teacher In-Charge

Head of the Department

Date 6/10/2023

Principal

THADOMAL SHAHANI TSEC ENGINEERING SOLLEGE

CONTENTS

SR. NO.	EXPERIMENTS	PAGE NO.	DATE	TEACHERS SIGN.
1.	Write a detail problem statement for hotel management system		19/7	
2.	Application of Agile Process model on project (JIRA)		26/7	
3.	Develop SRS document in IEEE format for project.		26/7	
4.	Develop DFD for the project		3/8	
5.	Develop Activity & State diagram for the project		10/8	
6.	Identify scenarios & develop use case.		23/8	
7.	Create project scheduling using gantt chart		6/9	
8.	Conduct function point analysis for the project		13/9	
9.	Application of COCOMO model for cost estimation of project.		13/9	
10.	Develop a list mitigation monitoring & management plan.		20/9	

6/10/2023

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CONTENTS

SR. NO.	EXPERIMENTS	PAGE NO.	DATE	TEACHERS SIGN.
11.	Case Study: Github for Version Control		27/9	
12	Develop test case for project using white box testing.		4/10	
A1	Assignment 1: Architectural design explain in detail.		4/10	
A2	Assignment 2: Explain in detail a) Software maintenance b) RE - engineering. c) Reverse - engineering.		4/10	

:: Experiment 1:-

Aim:- Write a detail Problem statement for any case study Justify which process model would be best suited to apply on it

→ Hotel Management S/m is a comprehensive & transformative s/ln for the hospitality industry it streamline operations and enhances guest experience from booking to check out with features like online booking, real time update and automate tasks it boosts productivity & cost saving. Sealess communication among staff automated tasks it boosts productivity & cost saving Seamless comm. among staff automated guest comm and target marketing further improve guest serve & loyalty.

→ Implementing Hotel Management System s/w provides a competitive adv. streamline operations & enhance guest satisfaction.
→ It actually also overcomes various challenges faced by the hospitality industry. It automates manual and time consuming process such as reservation & check ins reducing errors & saving time. Also provide robust security safe guards guest info. and payment method.

- A hotel Management System typically in
the following key functionalities.

1. Reservation management: Allows guest to mak
analysis in real

2. Check in / out :- facilitate quick and effic
in and out process red
wait times and enhancing guest exp

3. Guest Profiles & pref: Store guest info. p
and history to perso
service & loyalty exp.

4. Room and inventory mang: Tracks room status, h
schedules and inv.

5: Billing and Invoices: Generate bills and in
including room charg
additional services & taxes.

6. Staff mang: facilitates staff scheduling
attendance tracking and task

7. housekeeping & maintenance: helps mana
housekeeping
task maintence request and room cleaning
Status for efficient room turnover.

1.3

1. Customers :- The table stores info about hotel guests and potential customers. Includes ID, name, contact info, addr, etc. It help customer experient, manage programs, etc.

2. Staff : It contains data related to hotel employees it includ, id, name, position dept, etc. It is also staff management, scheduling, trackers, payroll, etc.

3. Suits :- It store info about various types of suit or rooms available in the hotel it includes detail id, type, desc, occupancy, prices, etc. facilitate room reservation, mng & room availability.

4. Catering : Contain data about catering services offered by hotel it includes id, name, option pricing, etc. it Helps manage catering booking and arrangements for event

5. booking - Record all hotel room reservation made by customers, Include id of (hotel, suit), chek in and out date). This table is critical for managing room occupancy chek in / out.

6. Cancellation : Tracks cancellation of room reservations. Includes details like id of booking, cancellation reason and the staff who handle, in case of managing room.

7. Stock : Store info about hotel inventory such as housekeeping supplies, toiletries and other consumables. Includes desc, quality of stock, etc.

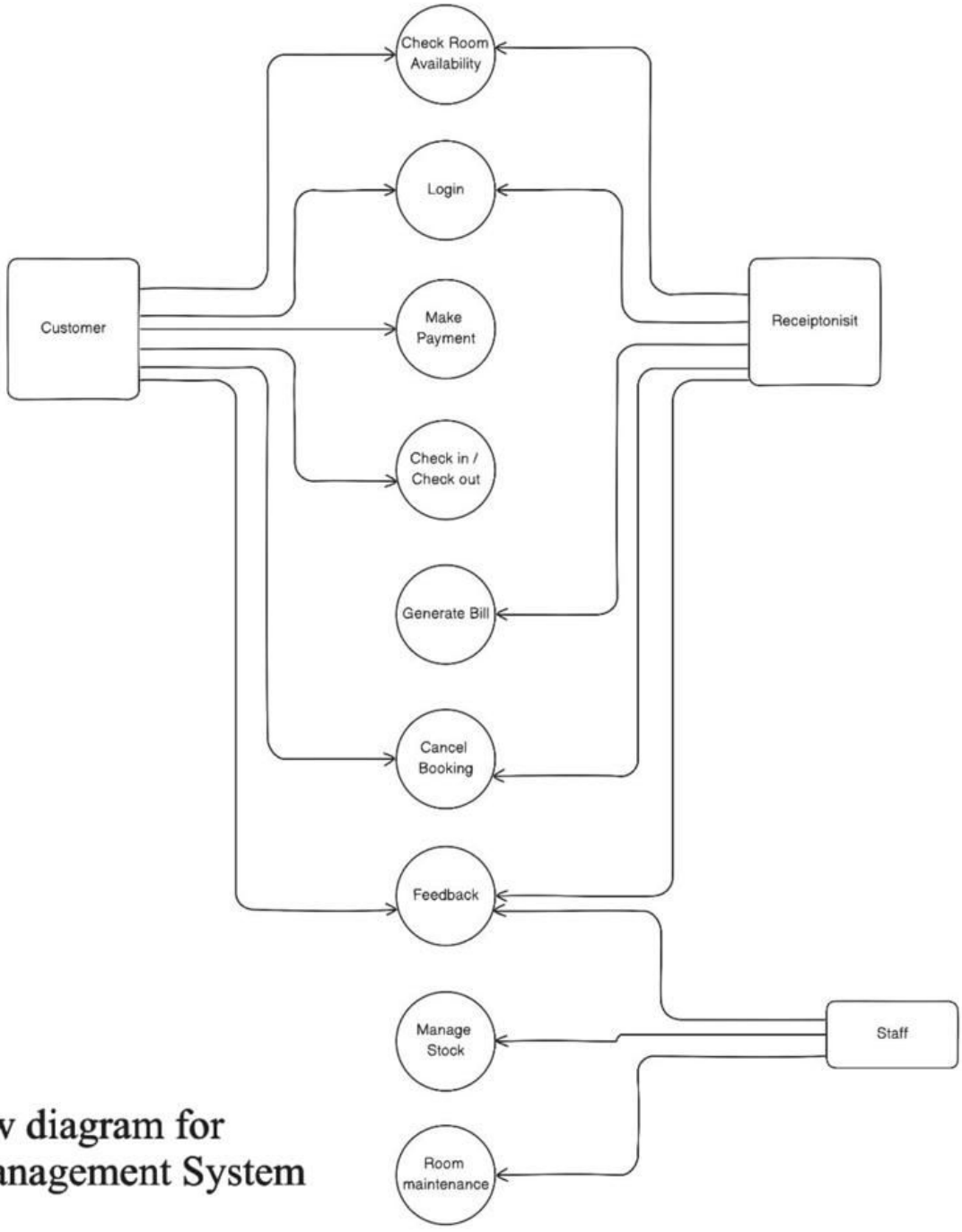
8. Events :- Contain data related to hotel events and functions. Includes id, date, organizer id, staff involved, etc.

9. Security : Store info about security - like activities & incidents in the hotel premises. Includes log in, time stamp, loc, etc.

10. Payment : all financial transaction related to services, room charges, bills. Includes payment & custom id, amount, method.

11. Feedback :- Stores guest feedback & reviews. Includes id of cust. food and time stamp, Help to track guest satisfaction.

12. Maintenance :- Data related to maintenance repairs in hotel. Includes desc, suit ra, staff involved. Help scheduling & tracking maintenance activities



Flow diagram for Hotel Management System

-: Experiment 2 :-

Aim :- Application of Agile Process model of project (JIRA).

An (A)

Subject: Roll No: 2213194, 2213196, 2113208

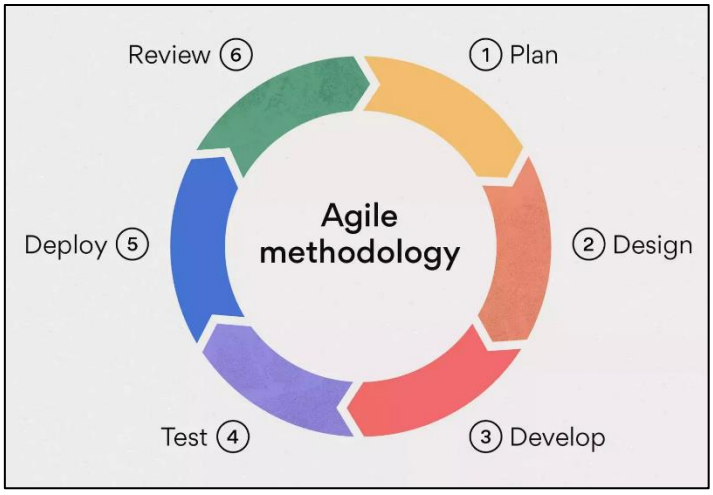
Experiment 2

Aim: Application of Agile Process Model on the project (JIRA)

Theory:

- What is Agile Process Model

  Agile is a methodology which consists of several iterative and incremental software development methodologies. The word agile refers to the ability to move quickly and easily. Therefore, an Agile approach enables project teams to adapt faster and easier compared to other project methodologies. The Agile methodology involves continuous planning, testing, integration and feedback from the stakeholders or clients throughout the life cycle of the software. The main objective of these models is to increase team involvement, and make quick decisions as per the situations. Agile methodology is mainly designed and developed to avoid common development issues during the software development life cycle and increase the overall efficiency of the development team. So, in a single sentence, the developments performed using Agile methodologies are normally developed and built iteratively and incrementally. In this methodology, a company can produce or deliver a quality product in less time and improve customer satisfaction at the end.



Need of Agile Process Model: -

a) Lower Cost
b) Enables clients to be happier with the end product by making improvements and involving clients with development decisions throughout the process.
c) Encourages open communication among team members, and clients.
d) Providing teams with a competitive advantage by catching defects and making changes throughout the development process, instead of at the end.
e) It keeps each project transparent by having regular consistent meetings with the clients and systems that allow everyone involved to access the project data and progress.

- Agile Process Model Types: -

  Each Agie method varies in the way it defines the steps of software development and the goal is to adapt the change while working on the software.

  1) Scrum: -

     Scrum, a dynamic Agile methodology, embodies a "inspect and adapt" philosophy. Unlike traditional linear approaches, Scrum emphasizes iterative progress and embraces change as a driving force. The heart of Scrum lies in its structured ceremonies:

     a) Sprint Planning: A collaborative session where the team selects a set of prioritized user stories for the upcoming sprint and outlines the tasks required to complete them.
     b) Daily Stand-up: A concise daily meeting where team members share updates on their work, discuss impediments, and align their efforts.
     c) Sprint Review: At the end of each sprint, the team showcases the completed features to stakeholders, eliciting feedback that fuels improvements and informs the product backlog.
     d) Sprint Retrospective: A reflective gathering where the team assesses their processes, celebrates achievements, and fine-tunes strategies for enhanced efficiency.

     Scrum's roles, including the Product Owner, who champions the customer's needs, and the Scrum Master, who safeguards the process, create a supportive ecosystem. By fostering transparent communication, encouraging self-organization, and embracing change, Scrum not only accelerates product development but makes it a powerful and unique Agile model.

  2) Kanban: -

     Kanban, a distinctive Agile methodology, revolves around visualizing and optimizing workflow. Unlike fixed sprint durations, Kanban emphasizes a continuous flow of work items, responding to real-time demands.

     Central to Kanban is the "kanban board," a visual representation of tasks progressing through different stages. This provides teams with clear visibility into their work, making bottlenecks and inefficiencies evident.

     Key features of Kanban include:

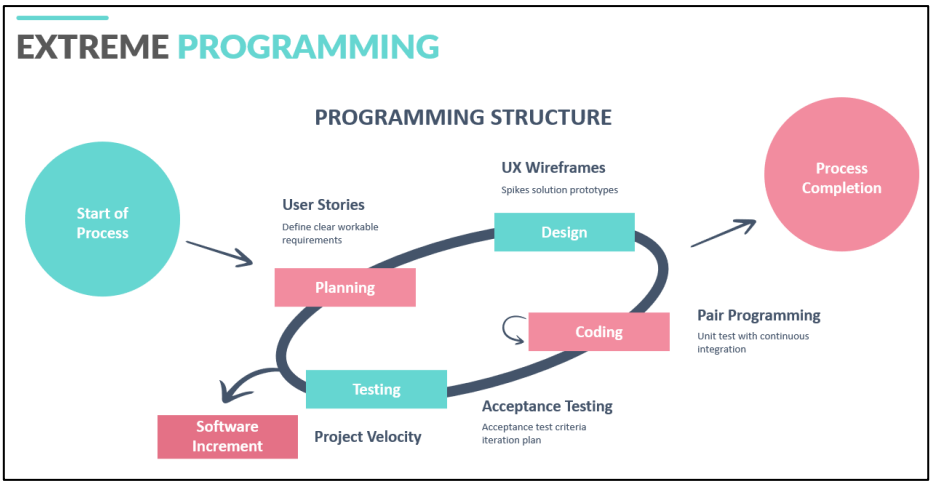
     1. Work-in-Progress Limits: Setting limits on tasks in each stage prevents overload and encourages focused effort, promoting smoother workflow.
     2. Continuous Improvement: Regular retrospectives empower teams to refine processes, enhance collaboration, and streamline performance.
     3. Metrics-Driven: Kanban employs quantitative data to measure lead time, cycle time, and other metrics, enabling informed decisions and strategic enhancements.

     Kanban's adaptability is particularly suited for support and maintenance teams, where incoming tasks vary. It thrives in environments where change is constant, enabling teams to remain responsive and effectively manage evolving priorities.

3) Extreme Programming (XP): -
Extreme Programming (XP), a cutting-edge Agile approach, is like a software craftsmanship workshop. It emphasizes creating top-notch code through Test-Driven Development (TDD), where tests are written before coding to ensure reliability. Pair Programming, like a collaborative duet, enhances code quality by sharing knowledge effectively.
The quick cycles allow rapid adjustments to changing requirements, akin to an artist refining their work. Continuous feedback loops, maintain alignment with customer needs. Frequent releases transform software development into a dynamic process, adapting to evolving expectations.
By embracing change, teamwork, and technical skill, Extreme Programming composes a remarkable performance in the software development world, where code becomes a work of art.

- Agile Model used in Existing model/system: -

Extreme Programming is well-suited for projects that require rapid iterations, close collaboration with customers, and a focus on delivering high-quality software. It provides a structured yet flexible approach that promotes responsiveness to change and continuous improvement throughout the development process.



Extreme Programming (XP) Agile model used to develop a hotel management system offered a tailored approach that addressed the unique demands of the hospitality industry: -

1. Test-Driven Development (TDD): XP's TDD ensures a dependable and thoroughly tested hotel management system. By writing tests before code, the system can maintain reliability and accuracy, critical for managing guest reservations, check-ins, and financial transactions.

2. Customer-Centric Iterations: XP's iterative approach aligns with the evolving needs of hotel staff and guests. Regular interactions with stakeholders facilitate rapid adjustments to accommodate changing requirements, whether it is integrating new services or optimizing the guest experience.
3. Pair Programming for Quality: In implementing core features, XP's pair programming method fosters collaboration between developers. This practice can result in higher-quality code for essential functions like room assignment algorithms, ensuring efficient guest accommodation.
4. Frequent Releases: XP's emphasis on small, frequent releases could mean the hotel management system can be deployed in stages. This allows the hotel to benefit from early functionality, such as reservations, and gradually incorporate additional modules like billing or housekeeping.
5. Refactoring for Adaptability: The dynamic nature of the hospitality industry requires system adaptability. XP's refactoring principle helps maintain code flexibility, making it easier to integrate new services, manage peak seasons, and adjust to changing market trends.
6. Continuous Improvement: XP's regular retrospectives encourage the hotel management team to continuously refine processes. This practice ensures that the system's performance and user experience consistently improve, enhancing guest satisfaction and operational efficiency.

Incorporating Extreme Programming into the development of a hotel management system fosters a collaborative, customer-focused, and adaptable approach, aligning technology with the ever-changing needs of the hospitality sector.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

:- Experiment 3:-

Aim:- Develop SRS document in IEEE format for project.

Experiment 3

Roll No

2113208: Raghav Rathi
2213196: Mohit Gangwani
2213194: Yash Gurnani

Aim: Develop SRS document in IEEE format for the project

What is SRS?

The production of the requirements stage of the software development process is Software Requirements Specifications (SRS) (also called a requirements document). This report lays a foundation for software engineering activities and is constructing when entire requirements are elicited and analyzed. SRS is a formal report, which acts as a representation of software that enables the customers to review whether it (SRS) is according to their requirements. Also, it comprises user requirements for a system as well as detailed specifications of the system requirements.

The SRS is a specification for a specific software product, program, or set of applications that perform particular functions in a specific environment. It serves several goals depending on who is writing it. First, the SRS could be written by the client of a system. Second, the SRS could be written by a developer of the system. The two methods create entirely various situations and establish different purposes for the document altogether. The first case, SRS, is used to define the needs and expectations of the users. The second case, SRS, is written for various purposes and serves as a contract document between customer and developer.
Some Characteristics of a good SRS include:

- Correctness: SRS should accurately capture all expected system needs, validated by user review.
- Completeness: SRS must include essential requirements, input data responses, references, and terms.
- Consistency: Ensure no conflicts in requirements regarding object characteristics, actions, or terminology.
- Unambiguousness: Every requirement should have a single interpretation, avoiding multiple definitions.
- Ranking for importance and stability: Identify the significance and stability of each requirement.
- Modifiability: SRS should allow easy system changes and be well-indexed for modifications.
- Verifiability: Requirements should be verifiable through cost-effective system checks.
- Traceability: SRS should clearly trace each requirement's origin and use forward/backward traceability.
- Design Independence: Avoid implementation details, allowing flexibility in selecting design alternatives.
- Testability: Ensure the SRS facilitates the generation of test cases and plans.
- Understandable by the customer: Use simple language and avoid complex notations for non-technical users.

- The right level of abstraction: Adjust the level of detail in the SRS based on its purpose and objectives.

Fig 3.1 Characteristics of a good SRS

Benefits of SRS:

Establish the basis of agreement between the customers and the suppliers on what the software product is to do.
It reduces the development effort
It also provides the basis for estimating costs and schedules
It provides a baseline for validation and verification
It is also useable during the maintenance phase
It works like a supplement for the project documentation
It helps building up a trustworthy relationship with the end-consumers of the project

Format of SRS:

Software Requirements Specification

for

Hotel Management System

Version 1.0 approved

Prepared by Group Raghav, Mohit, Yash

Thadomal Shahani Engineering College

30 August 2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

SRS for Hotel management software

1. Introduction

1.1 Purpose

The Hotel Management System will be a comprehensive software application (version 1.0) designed to manage various aspects of a hotel's operations, including reservations, guest services, room management, and more. This system will automate and streamline hotel management tasks, providing an efficient way to handle guest interactions, room bookings, billing, and staff management.

1.2 Document Conventions

Acronyms and Abbreviations: Acronyms and abbreviations are spelled out on their first occurrence in the document and are followed by their respective acronyms/abbreviations in parentheses, e.g., "Software Requirements Specification (SRS)."
Formatting: Requirement statements are formatted using bullet points for clarity and easy reference. Key terms, such as requirements types (e.g., functional, performance) or specific requirements (e.g., security, usability), may be highlighted or italicized for emphasis.

1.3 Intended Audience and Reading Suggestions

This project is designed for hotel owners, managers, staff, and IT professionals involved in hotel management. This document serves as a comprehensive guide to all the requirements of the Hotel Management System.

1.4 Product Scope

The main purpose of this project is to simplify and optimize hotel management processes, reducing manual efforts and enhancing guest experiences. The Hotel Management System will provide tools to manage reservations, room assignments, guest services, billing, and staff operations efficiently.

1.5 References

https://www.cloudbeds.com/articles/hotel-management-software-guide/
https://bosctechlabs.com/complete-hotel-management-system-development-guide/

2. Overall Description

2.1 Product Perspective

The Hotel Management System is a standalone application that does not require additional software or third-party plugins. It interacts with databases to manage guest and reservation information.

2.2 Product Functions

The major functions of the Hotel Management System include:

2.2.1 Reservation functions

- allow for a hierarchical organization of functions, providing a structured approach to documenting various aspects of the software.
- It's essential to define the frequency of use, technical expertise, privilege levels, and responsibilities associated with each user class or group (in this case, students). This information helps in tailoring the user experience.
- Describing user characteristics (e.g., educational level) assists in designing a user-friendly interface that aligns with the users' capabilities and expectations.

2.2.2 Guest Services Functions

- Provides a clear, structured breakdown of functions but for a different user class.
- Staff functions, as outlined in this subsection, may include administrative actions, data management, and responsibilities unique to staff roles.
- Ensuring that the software addresses the needs of staff members effectively and efficiently, supporting their roles within the organization.
- It is vital for developers and designers to design user interfaces and functionalities that align with the tasks staff members need to accomplish.

2.3 User Classes and Characteristics

The Hotel Management System will be used by two primary types of users:

Guests:
- Frequency of use: Occasional when making reservations or accessing guest services.
- Technical Expertise: Low
- Responsibilities: Make reservations, check-in, request services, view bills, and check-out.

Hotel Staff:
- Frequency of use: Regularly for managing reservations, room assignments, and guest services.
- Technical Expertise: Moderate to high
- Responsibilities: Manage room bookings, assign rooms, handle check-in/check-out, provide guest services, and manage billing.

2.4 Operating Environment

Hardware platform:

- Processor: Multi-core processor with sufficient processing power to handle concurrent user requests and data processing.
- Memory (RAM): A minimum of 4 GB of RAM is recommended to ensure smooth operation.
- Storage: Adequate storage capacity for database storage.
- Network Connectivity: Reliable internet connection.

OS:

- Windows: Compatible with Windows Server 2016 and later versions for server deployment. Supports client-side usage on Windows 10 and later versions.
- Linux: Compatible with various Linux distributions, including Ubuntu 18.04 LTS and CentOS 7 and later versions.
- macOS: Supports macOS 10.15 (Catalina) and later versions for client-side usage.

Software components:

- Web Browser: Users will access the Hotel Management System through web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, or Safari.

2.5 Design and Implementation Constraints

Hardware Limitations:

The system's performance may be affected by the hardware it runs on. Hardware should meet or exceed recommended specifications for optimal performance.
Data Security: Robust security measures, including encryption and access controls, must be implemented to protect sensitive guest and financial data.
Scalability: The system should be designed to accommodate future growth and additional features.

User Documentation:

- User manuals will provide guidance for using the Hotel Management System, catering to both staff and guests.
- Comprehensive documentation will cover system processes and development details.
- Online help and tutorials will assist users in navigating complex features.

2.7 Assumptions and Dependencies

Assumptions:

- Guest and reservation data will be available for migration into the system.

- Hardware and infrastructure meet minimum requirements.
- Third-party services, if used, will be available and reliable.

Dependencies:

- Database Management System (DBMS): The system relies on a DBMS for data storage and retrieval.

- Third-Party Services:

Integration with third-party services (e.g., payment gateways) may be required for specific functionalities.

3. External Interface Requirements

3.1 User Interfaces

Guest Interface

Description: The guest interface allows users to make reservations, access guest services, view bills, and check-out.

Characteristics:

- Authentication: Guests may log in using their reservation details.
- Reservation Management: Features for making reservations, modifying bookings, and viewing reservation details.
- Guest Services: Options for requesting room service, housekeeping, and other guest services.
- Billing: Access to view bills and make payments.
- Check-out: Features for checking out of the hotel.

Hotel Staff Interface

Description: The staff interface is designed for hotel staff responsible for managing reservations, room assignments, guest services, and billing.

Characteristics:

- Authentication: Staff members log in using their credentials.
- Reservation Management: Tools for managing room reservations, room assignments, and availability.
- Check-in/Check-out: Features for processing guest check-in and check-out.

- Guest Services: Options for handling guest requests and services.
- Billing: Tools for generating bills, processing payments, and managing accounts.

3.2 Hardware Interfaces

1. Server Hardware Interfaces

Description: The Hotel Management System runs on server hardware and interfaces with underlying server components.

Characteristics:

- Server Type: The system is designed to run on standard server hardware, including physical servers and virtual machines (VMs).
- Operating System: Compatible with various server operating systems, including Windows Server, Linux distributions, and macOS Server.
- Database Server: Interacts with a database server (e.g., MySQL, PostgreSQL, SQL Server) for data storage and retrieval.
- Communication Protocols: Uses standard protocols such as HTTP/HTTPS for web-based interactions and SQL for database operations.
- Load Balancing: In cases of high user loads, interfaces with load balancers to distribute incoming traffic for scalability.

2. Client Hardware Interfaces

Description: The Hotel Management System provides user interfaces accessible on various client devices.

Characteristics:

- Device Types: Supports desktop computers, laptops, tablets, and smartphones.
- Operating Systems: Accessible on client operating systems, including Windows, macOS, Linux, Android, and iOS.
- Web Browsers: Users access the system through modern web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.
- Mobile App Interfaces: May provide mobile applications for Android and iOS devices that interact with the system through APIs.

3. Network Interfaces
Description: The Hotel Management System relies on network interfaces for data transmission and communication.

Characteristics:

- Network Protocols: Uses standard network protocols such as TCP/IP, HTTP, and HTTPS for data transmission between clients and servers.
- Firewalls: Interfaces with network firewalls and security systems to ensure secure data exchange over the network.

- Load Balancers: In load-balanced environments, communicates with load balancers to distribute requests to backend servers.

4. Database Interfaces
Description: The Hotel Management System interfaces with a database management system (DBMS) for data storage and retrieval.

Characteristics:

- DBMS Compatibility: Compatible with multiple DBMS options, including MySQL, PostgreSQL, and Microsoft SQL Server.
- Database Connection: Establishes database connections using standard database communication protocols (e.g., JDBC).
- Data Retrieval: Retrieves data from the database using SQL queries and data manipulation commands.
- Data Storage: Inserts, updates, and manages data in the database using SQL transactions.

3.3 Software Interfaces

1. Database Management System (DBMS)
Description: The Hotel Management System interacts with a database management system (DBMS) for data storage and retrieval.

Characteristics:

- DBMS Options: Compatible with multiple DBMS options, including MySQL, PostgreSQL, and Microsoft SQL Server.
- Data Access: Uses standard database communication protocols (e.g., JDBC) to establish connections and interact with the database.
- Data Retrieval: Retrieves data from the database using SQL queries and data manipulation commands.
- Data Storage: Inserts, updates, and manages data in the database using SQL transactions.
- Shared Data: Data shared with the DBMS includes guest profiles, reservations, room assignments, billing information, and more.

2. Operating Systems

Description: The Hotel Management System is compatible with various operating systems on both server and client sides.

Characteristics:

- Server OS: Compatible with server operating systems, including Windows Server, Linux distributions, and macOS Server.
- Client OS: Supports client operating systems, including Windows, macOS, Linux, Android, and iOS.
- Compatibility: Ensures compatibility with the latest versions of these operating systems to support a wide range of users.

3. Web Browsers

Description: Users access the Hotel Management System through web browsers.

Characteristics:

- Supported Browsers: Compatible with popular web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.
- Cross-Browser Compatibility: Ensures that web-based interfaces function consistently across different browsers and versions.
- Data Exchange: Data is transmitted between the system and users' browsers using HTTP/HTTPS protocols.

4. Third-Party Services

Description: The Hotel Management System may integrate with third-party services and APIs for specific functionalities.

Characteristics:

- Communication: Communicates with third-party services using APIs and data exchange protocols.
- Examples: Third-party services may include payment gateways for financial transactions, messaging services for communication, and external booking platforms.
- Data Flow: Data exchanged with third-party services includes payment data, communication logs, and booking information.

3.4 Communication Interfaces

1. Web-Based User Interfaces

Description: The Hotel Management System provides web-based user interfaces accessible through standard web browsers.

Requirements:

- Supported Browsers: Compatible with modern web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari.
- HTTP/HTTPS: Uses HTTP/HTTPS for secure data communication between clients and the system.
- Data Transfer Rates: Web interfaces are designed to ensure efficient data transfer rates for a responsive user experience.
- Message Formatting: Data exchanged through web interfaces is formatted in standard HTML, JavaScript, and CSS for rendering in browsers.
- Synchronization: Ensures data synchronization between the server and clients to provide real-time updates and notifications.

2. Email Notifications

Description: The Hotel Management System sends email notifications to users for various events and alerts.

Requirements:

- SMTP Protocol: Uses the Simple Mail Transfer Protocol (SMTP) for sending email notifications.
- Message Formatting: Email notifications are formatted in plain text and HTML for user-friendly content.
- Security: Email communication includes standard security measures like secure socket layer (SSL) or transport layer security (TLS) for encryption.
- Data Transfer Rates: Email notifications are sent promptly to ensure timely communication with users.

3. API Interfaces

Description: The Hotel Management System exposes APIs for integration with external systems and mobile applications.

Requirements:

- API Endpoints: Defines specific API endpoints for data retrieval, updates, and interactions.
- Data Formats: APIs accept and provide data in common formats such as JSON or XML for ease of integration.
- Authentication: API endpoints require authentication, ensuring secure access to sensitive data.
- Data Transfer Rates: APIs support efficient data transfer rates to minimize latency in data exchange.
- Security: API communication includes security measures such as token-based authentication and encryption to protect data in transit.

4. System Features

4.1 Reservation Management

4.1.1 Description and Priority

Description: This feature allows guests to make reservations for rooms in the hotel.
Priority: Highest

4.1.2 Stimulus/Response Sequences

a: Guest visits the hotel website and selects "Make a Reservation."
b: Guest provides reservation details, including check-in/check-out dates and room preferences.

c: The system displays available rooms and their prices.
d: Guest confirms the reservation and provides payment details if required.
e: The system processes the reservation and sends a confirmation to the guest.

4.1.3 Functional Requirements

a: Provide an intuitive reservation interface for guests.
b: Check room availability based on specified dates and room preferences.
c: Calculate reservation costs and taxes.
d: Securely collect and process payment information.
e: Send confirmation emails to guests upon successful reservations.

4.2 Guest Services

4.2.1 Description and Priority

Description: This feature allows guests to request additional services during their stay, such as room service or housekeeping.
Priority: High

4.2.2 Stimulus/Response Sequences

a: Guest logs in to their account and selects "Request Services."
b: The guest chooses the desired service(s) and provides specific instructions if needed.
c: The system forwards the service request to the appropriate staff or department.
d: Staff acknowledges the request and provides the requested service.
e: The system updates the guest on the service status.

4.2.3 Functional Requirements

a: Offer a user-friendly platform for guests to request services.
b: Categorize services (e.g., room service, housekeeping) for easy selection.
c: Notify relevant staff or departments of service requests.
d: Track the status of service requests and provide updates to guests.
e: Maintain a record of requested services for billing purposes.

4.3 Billing and Payment

4.3.1 Description and Priority

Description: This feature manages billing and payment processes for guest stays, including check-out and invoicing.
Priority: High

4.3.2 Stimulus/Response Sequences

a: Guest selects "Check Out" or "View Bill" from their account.

b: The system calculates the total bill based on the guest's stay, additional services, and any outstanding charges.
c: Guest reviews the bill and selects a payment method.
d: The system processes the payment and generates an invoice.
e: The system sends the invoice to the guest via email or provides a printed copy.

4.3.3 Functional Requirements
a: Calculate the total bill accurately, including room charges and additional services.
b: Display the bill to guests for review and transparency.
c: Support multiple payment methods, including credit cards, cash, and mobile payments.
d: Generate invoices with detailed breakdowns of charges.
e: Email or provide printable invoices to guests for their records.

4.4 Room Management

4.4.1 Description and Priority

Description: This feature oversees room assignments, availability, and maintenance.
Priority: Medium

4.4.2 Stimulus/Response Sequences

a: Staff logs in to the hotel management system and selects "Room Management."
b: Staff views the room availability calendar and assigns rooms to guests.
c: The system updates room availability in real-time.
d: Housekeeping staff mark rooms as cleaned and ready for check-in.
e: The system alerts staff to maintenance requests or issues with rooms.

4.4.3 Functional Requirements

a: Maintain a real-time room availability calendar.
b: Assign rooms to guests based on their reservations.
c: Notify housekeeping staff of rooms that need cleaning.
d: Track room maintenance requests and prioritize them.
e: Provide status updates on room availability, cleaning, and maintenance.

4.5 Staff Management

4.5.1 Description and Priority

Description: This feature manages records of hotel staff members, including personal information and roles.
Priority: Medium

4.5.2 Stimulus/Response Sequences

a: Admin logs in to the hotel management system and selects "Staff Management."
b: Admin adds new staff members or updates existing staff information.

c: The system records and displays staff details accurately.

4.5.3 Functional Requirements

a: Maintain individual profiles for staff members with personal information.
b: Allow addition, editing, and deletion of staff records.
c: Assign and manage staff roles and responsibilities.
d: Ensure data security and restricted access based on roles.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- System Responsiveness: The system must respond to user actions within 3 seconds under normal load conditions.
- Concurrent Users: The system must support a minimum of 100 concurrent users during peak hours without performance degradation.

5.2 Safety Requirements

- Data Security: Implement encryption and access controls for data protection, complying with relevant regulations.
- Regular Backups: Perform daily automated off-site data backups for disaster recovery.
- User Authentication: Enforce strong, unique passwords and implement account lockout mechanisms.

5.3 Security Requirements

- User Authentication: Implement user authentication mechanisms, including strong password policies and account lockout features.
- Data Encryption: Encrypt sensitive data both in transit and at rest to protect against unauthorized access.
- Access Control: Enforce role-based access control (RBAC) to limit user privileges based on their roles.
- Vulnerability Assessment: Conduct regular vulnerability assessments and penetration testing to identify and mitigate security weaknesses.
- Compliance: Ensure compliance with relevant data protection regulations and industry security standards.
- Incident Response: Develop and document an incident response plan to address security breaches and data breaches promptly.

5.4 Software Quality Attributes

- Usability: The system should have a user-friendly interface with a usability score of at least 85 out of 100, as measured by user testing.
- Reliability: The system must have an uptime of 99.5%, ensuring it's available for use except during scheduled maintenance.

- Maintainability: Code changes and updates should be easily accomplished, with at least 80% code coverage for unit tests.
- Portability: The system should be accessible through major web browsers (Chrome, Firefox, Safari) and mobile devices (iOS and Android).

5.5 Business Rules

- User Roles: Only authorized administrators can access and modify staff records, while guests can access their own reservations and billing information.
- Payment Deadlines: Guests must settle their bills upon check-out or as per the hotel's policy to avoid additional charges.
- Access Control: Only authorized staff members can access sensitive financial data and guest profiles.
- Data Retention: Guest records are retained for legal and operational purposes as required by hotel management.
- Maintenance Requests: Maintenance requests from guests should be addressed promptly to ensure a comfortable stay.

6. Other Requirements

6.1 Database Requirements

- The system shall use a relational database management system (RDBMS) to store and manage data.
- Data backups shall be performed daily and stored securely.

6.2 Legal Requirements

- The system shall comply with all applicable data protection and privacy regulations (e.g., GDPR).
- Terms of service and privacy policy pages shall be accessible and up-to-date.

Appendix A: Glossary

SRS: Software Requirements Specification - A document that outlines the detailed requirements for a software project.
DBMS: Database Management System - Software for creating and managing databases.
JDBC: Java Database Connectivity - A Java-based API for connecting and interacting with databases.
SMTP: Simple Mail Transfer Protocol - A standard protocol for sending email messages.
API: Application Programming Interface - A set of rules and protocols for building and interacting with software applications.
RBAC: Role-Based Access Control - A security approach that restricts system access based on user roles and responsibilities.
GDPR: General Data Protection Regulation - A European Union regulation governing data protection and privacy for individuals.

Roll No. 2113208

-: Experiment 4 :-

Aim :- Develop data flow diagram (DFD) for the project (Smart door, Lock door).

- What is DFD, Why we draw it ?

A data flow diagram is a visual depiction used to depict how data moves within a system. DFS's are drawn for several important purposes.

a) Enhance clarity :- DFD's provide a clear and concise representation of complex data flow patterns in a system.

b) In depth Analysis : They aid in comprehending system functionality, making it better / easier to pinpoint potential bottlenecks or inefficiencies.

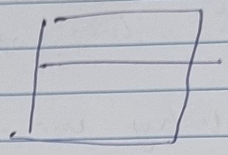
c) Effective Design :- DFD's play a pivotal role in designing new system or optimizing existing one by illuminating data interaction.

d) Facilitate communication : These diagram make it simpler for stakeholders such as users & developers to communicate & understand system intricacies.

e) Comprehensive Documentation : DFSs serves as comprehensive documentation for system requirements & captured ideas insights.

A+

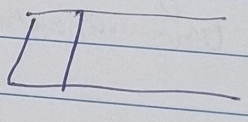
- Elements of DFA

1) Process 

→ Works or actions performs on data.
→ Label should be verb phrases
→ Receive o/p data & produces o/p.

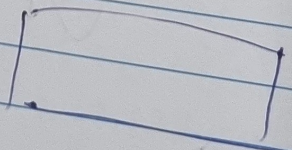
2) Data flow ⟶

→ is a part for data to move from one part to another.
→ arrows depicting movement of data.
→ Can represents flow between process & data store by 2 seperate arrows.

3) Data Store 

→ Uses in DFA to implement data that rystem stores.
→ Tables should be noun phrases.

4) Source / Sink / (External link)

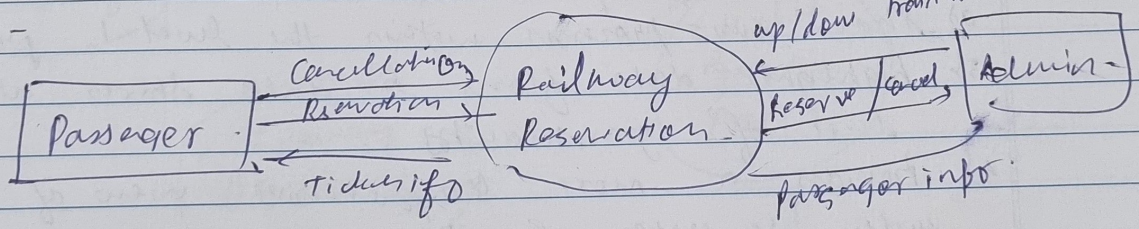


→ External entity that is origin or destination of data.
→ Labels should be noun phrases.
→ Source: Entity that supplies data to system.
→ Sink: Entity that receives data from system.

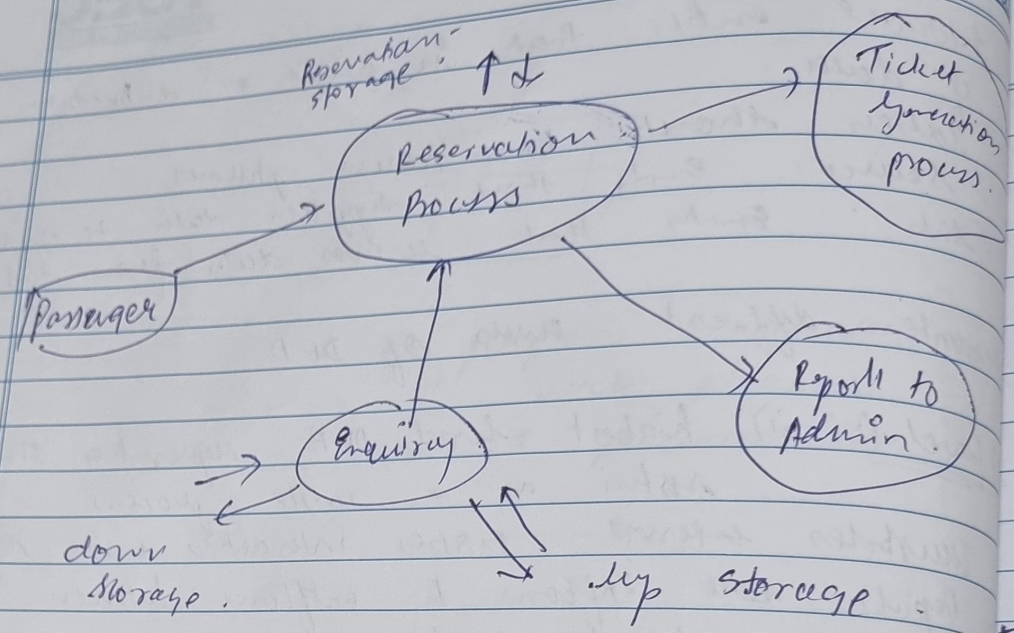
Describe different levels of DFD:

a) Level 0: i) Highest level DFA, represents the entire system as a single process.
2) Identifies external entities interacting with system.
3) Depicts data inflows & outflows between the system & external entities.
4) Offers a high level view of system boundaries & interactions.

ex:-



b) Level 1: 1) Shows the system major processes data flows & data stores at a high level of abstraction.
2) When context diag is expanded into DFA level-0, all the connections that flows into & out of process 0 needs to be retained.

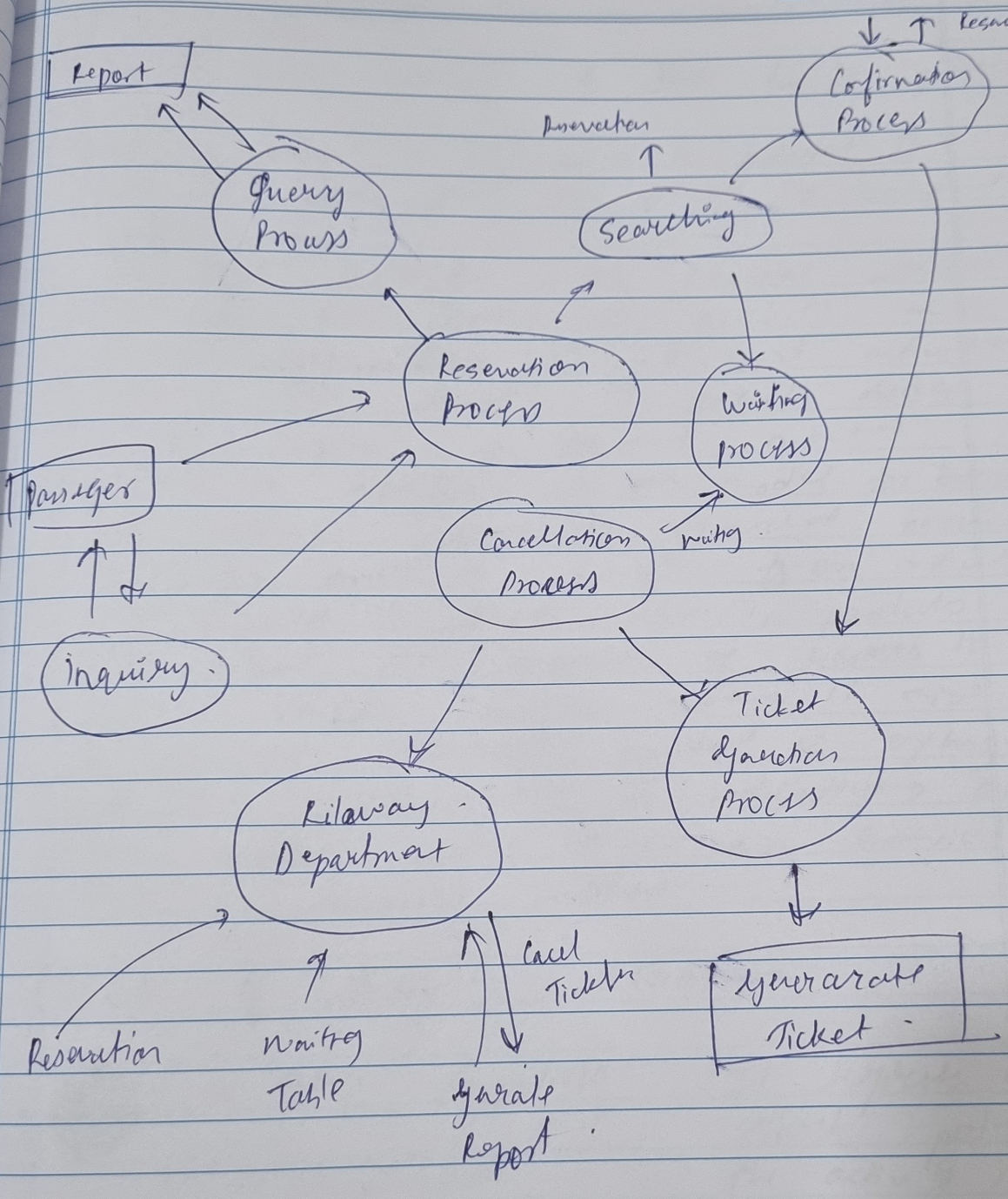


c) Level 2 :- a) A more details breakdown of from level 1.

a) Shows sub-process within the level 1 pro

3) highlights data flows & data stores at level of granularity.

4) Provides a more comprehensive view of in system processes & their interactions.



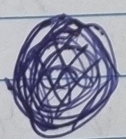
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

:- Experiment 5 :-

Aim: Develop activity & state for the project.

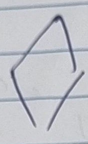
- Activity diag. can be used to understand the flow of work that an object or component perform
- Is often associated with several classes.
- one of the benefits of activity diag. is the representation of on current activities.
- one of the strengths of activity diag. is the representation of current activities.
- provide a workflow of the model for business
- are very serves to flowchart because you can create a workflow from activities.
- are used to show workflow in parallels.
- can depict the process of resources to share various activities making it easier to visualize resource consent & optimize.
- help in documenting & understanding complex business process making them an essential tool in business process.

- Elements of Activity diagram.

1)  : Start Symbole : It represents the start point of the activity flow. and indicates where the process or activity begins.

ii) [Activity] : Activity Symbol : Depicts the ... That ... model

These symbol include small descrip...

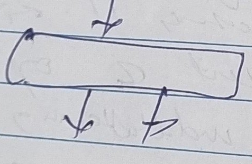
iii)  : decision symbol : decision nodes used to model decision point

the process depending upon the choice flow or control can take multiple pa...

iv) ⟶ : Connection symbol : Are use for c... action & decis... action and ...

process It shows the sequence of activi...

v)



: fork symbol : splits an storage acti...

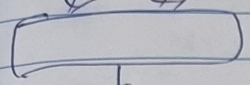
con current activities flow into two

showed using. synchronize with ...

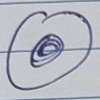
vi) [condition] : guard expression : Placed near decision

condi. an activity flow that split in know under direction.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE



: Join Sybole : Combins two concurrent activity and introduce them to a flow where only one activity.



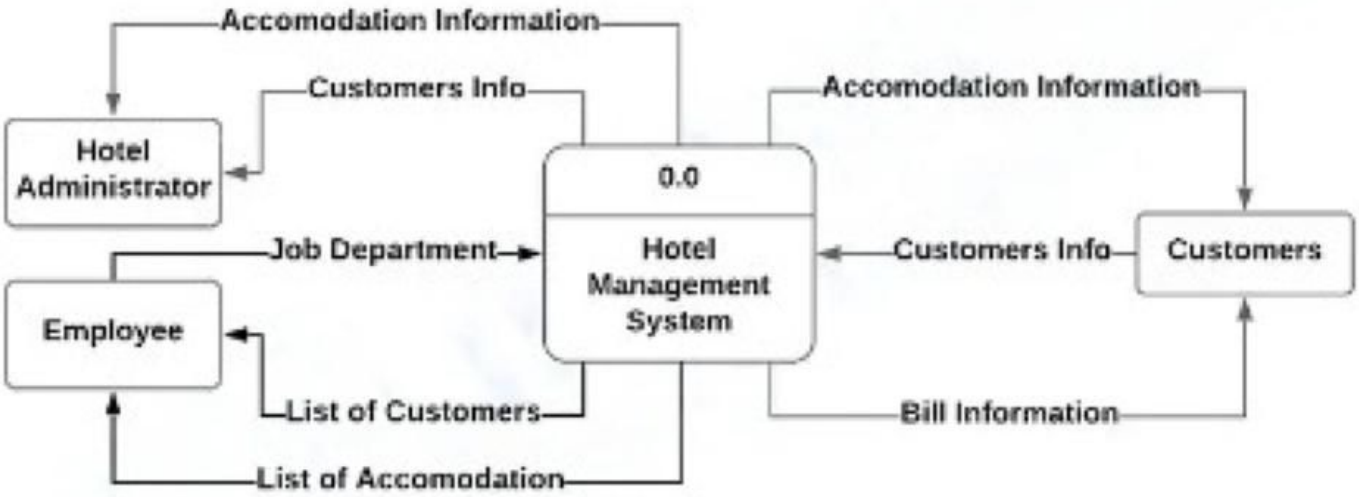
: End Sybole. : Masks the end state of an activity and represents the completion OR all flows at process

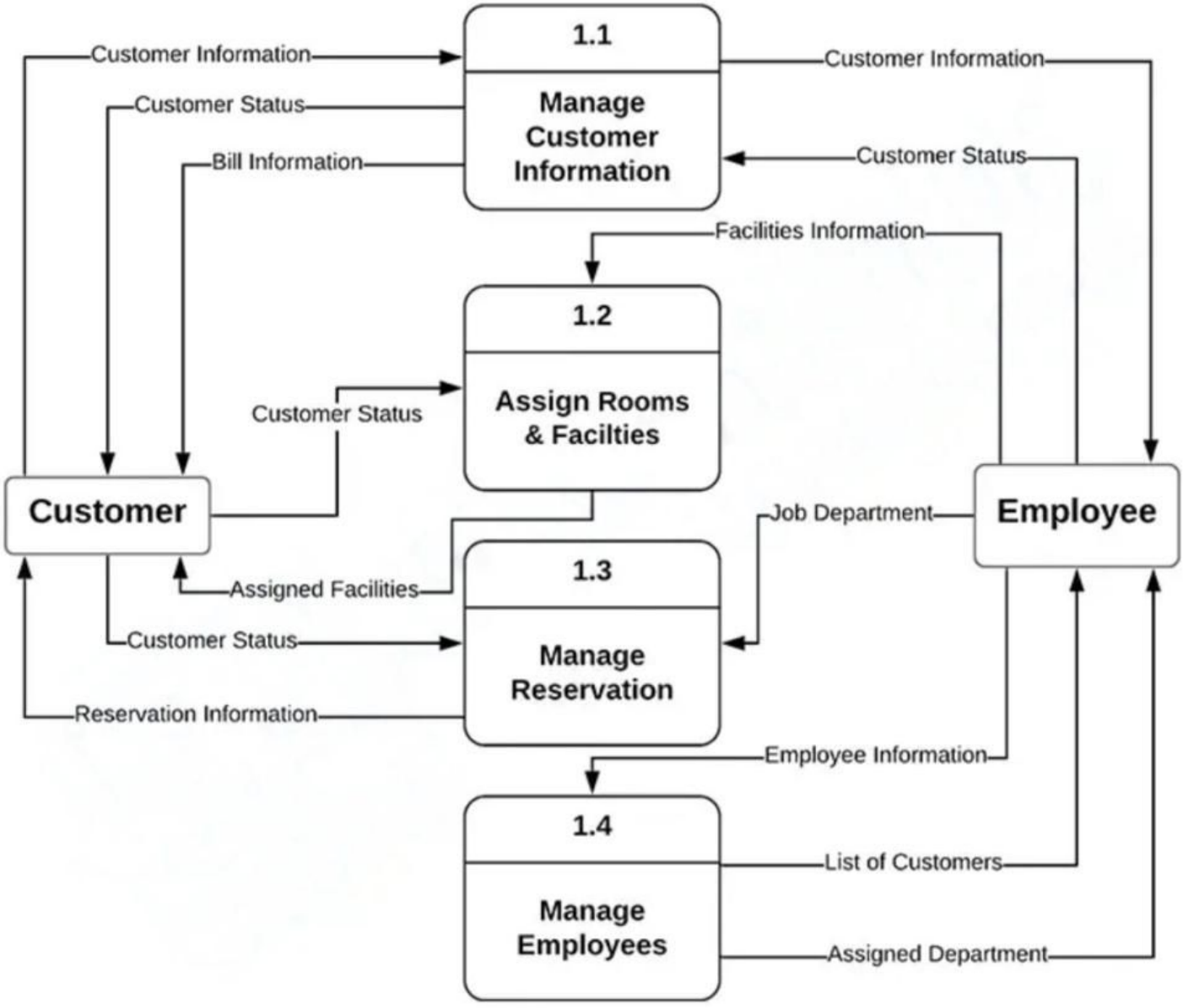
Explanation of Activity Diagram

It Represents the process of a user reserving a hotel room in a hotel management system.

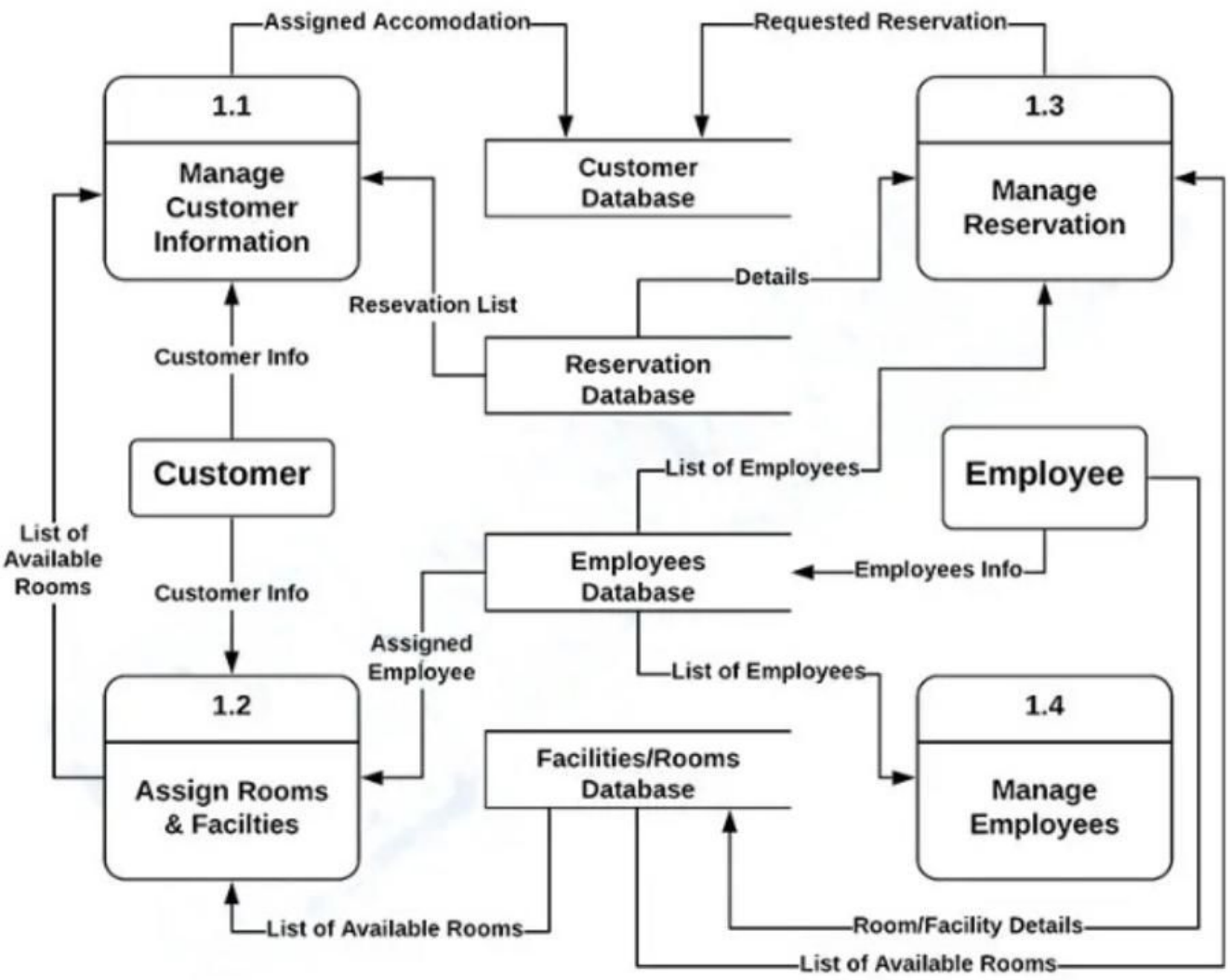
→ The user begins with the reservation option
→ the system checks there are available rooms if not display "no room" is available the user enters guest information confirms the reservation and proceeds for the payment
→ If the payment is successful if yes, it confirms the reservation and if not it display "Payment failed" message.

DFD Level 0 for
Hotel Management System





DFD Level 1 for
Hotel Management System



DFD Level 2 for
Hotel Management System

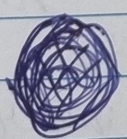
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

:- Experiment 5 :-

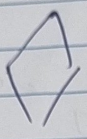
Aim: Develop activity & state for the project.

- Activity diag. can be used to understand the flow of work that an object or component perform
- Is often associated with several classes.
- one of the benefits of activity diag. is the representation of concurrent activities.
- one of the strengths of activity diag. is the representation of concurrent activities.
- provide a workflow of the model for business
- are very serves to flowchart because you can create a workflow from activities.
- are used to show workflow in parallels.
- can depict the process of resources to share various activities making it easier to visualize resource consent & optimize.
- help in documenting & understanding complex business process making them an essential tool in business process.

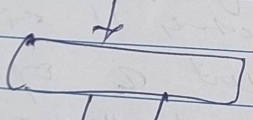
- Elements of Activity diagram.

1)  : Start Symbole : It represents the start point of the activity flow. and indicates where the process or activity begins.

ii) [Activity] : Activity : Indicates the
Symbol: That
model
These symbols include small descrip

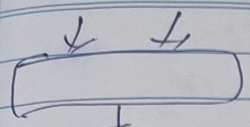
iii)  : decision symbol : decision nodes
used to mode
decision
the proces depending upon the choice
flow or control can take multiple pa

iv) → : Connection : Are use for
Symbol activo &
actions and
proces It shows the sequence of activ

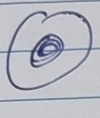
v)  : fork : Split an
Symbol. storage
flow into
on current activity Symbol with
observed thing.

vi) [condition] : guard : Placed
expression decision
know
condi. an activity flow Node split in
direction.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE



: Join Sybole : Combins two concurrent activity and introduce them to a flow where only one activity.



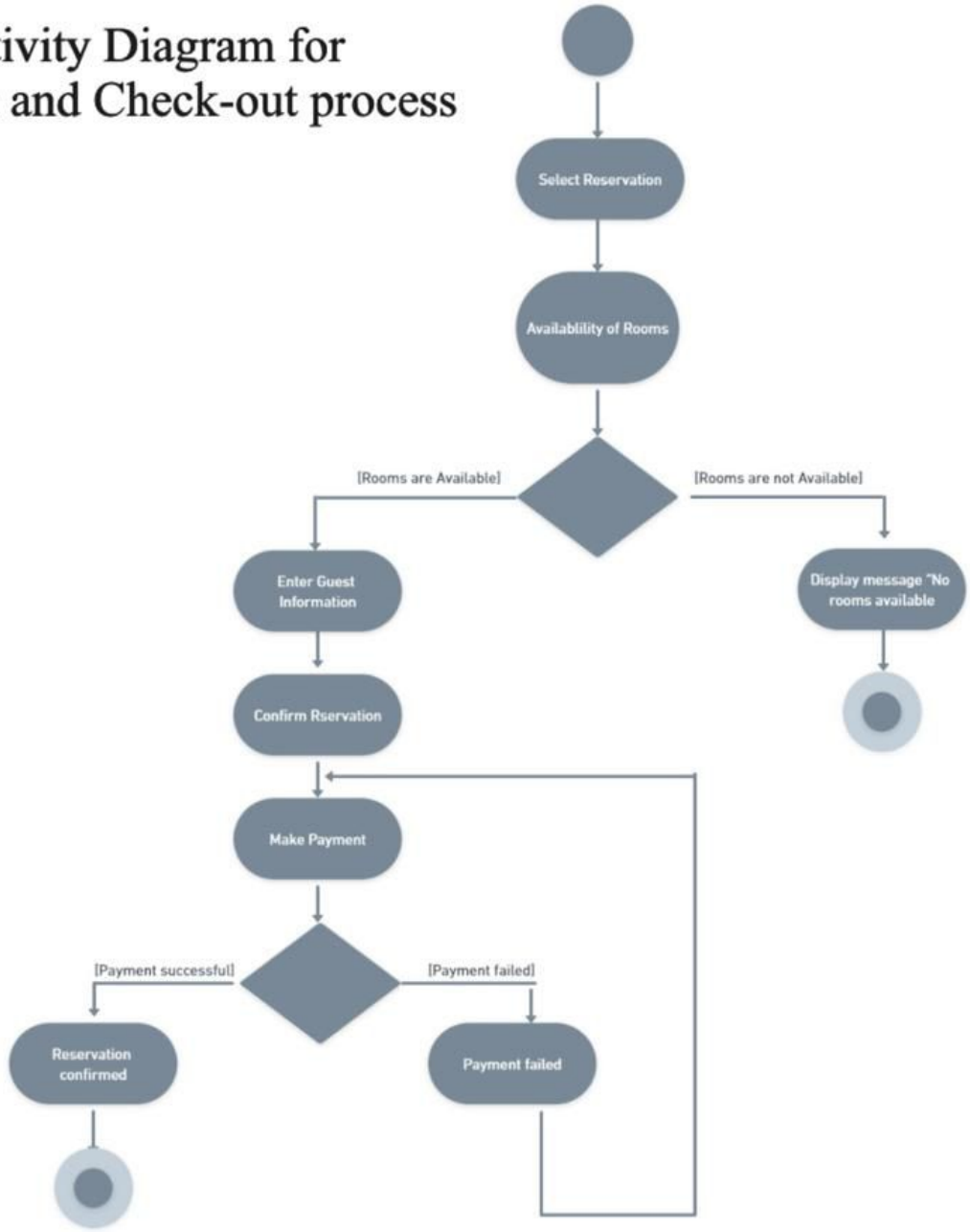
: End Sybole. : Masks the end state of an activity and represents the completion OR all flows at process

Explanation of Activity Diagram

It Represents the process of a user reserving a hotel room in a hotel management system.

→ The user begins with the reservation option
→ the system checks there are available rooms if not display "no room" is available the user enters guest information confirms the reservation and proceeds for the payment
→ If the payment is successful if yes, it confirms the reservation and if not it display "Payment failed" message.

Activity Diagram for Check-in and Check-out process



Payel
213208

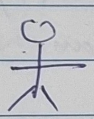
Experiment 6:-

Aim:- Identify scenarios develop use case diagram for the project (Smart Draw, Lucid chart).

1. What is use-case, why we draw it?

→ A use case diag. is a visual representation used in S/W engg. to illustrate how different actors interact with a system and its various func. It helps stakeholders understand system behaviour requirements & user interactions. aiding in clear communication, design & analysis of complex systems.

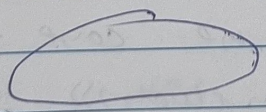
2. Elements of use case with example.

(a) Actor 

→ An actor represents the user of system including human users & other systems.

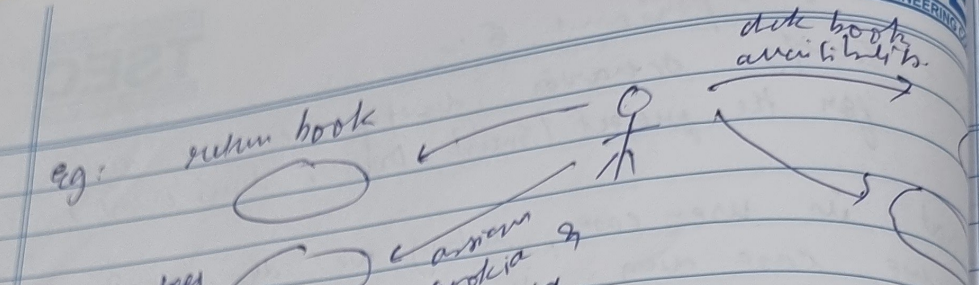
→ Three types
- i) users of system.
- ii) external application system.
- iii) External devices.

(b) usecase name of usecase.



→ use case represents the functionality provided by the system.

→ A use case represents a dialog between an actor & system.



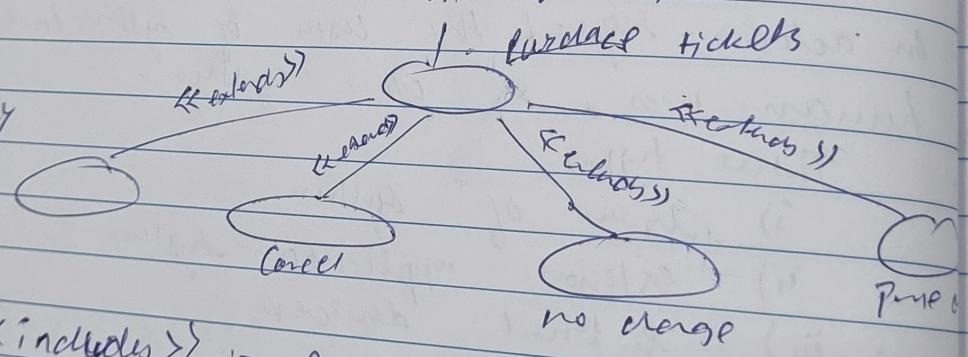
(3) Relationship in usecase with example.

→ i) <<extends>> :- This relationship models exceptional or seldom invoked

⇒ the exceptional event flows are factored of the main event flow.

⇒ Use case representing exceptional flows able to extend more than one use

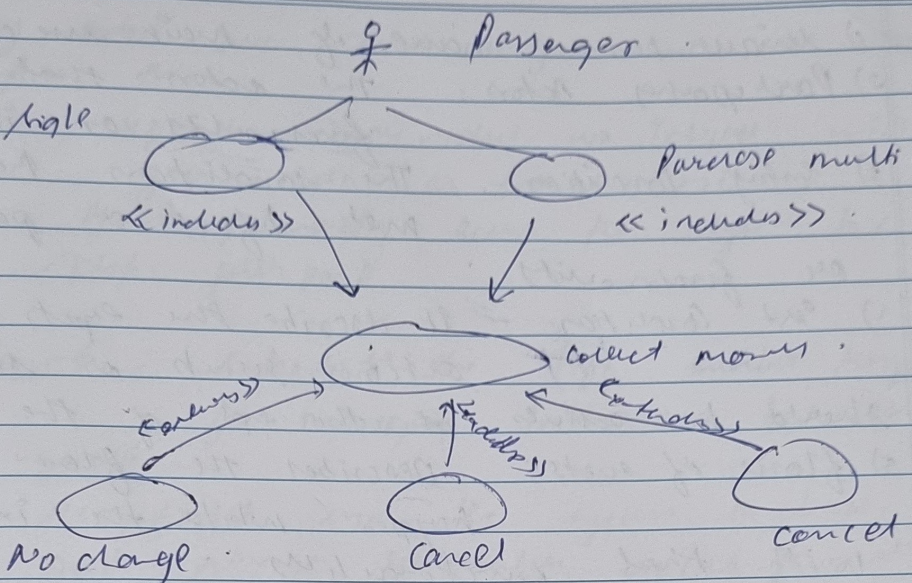
eg:



ii) <<includes>> :- Relationship represents common functionality needed in more one use case.

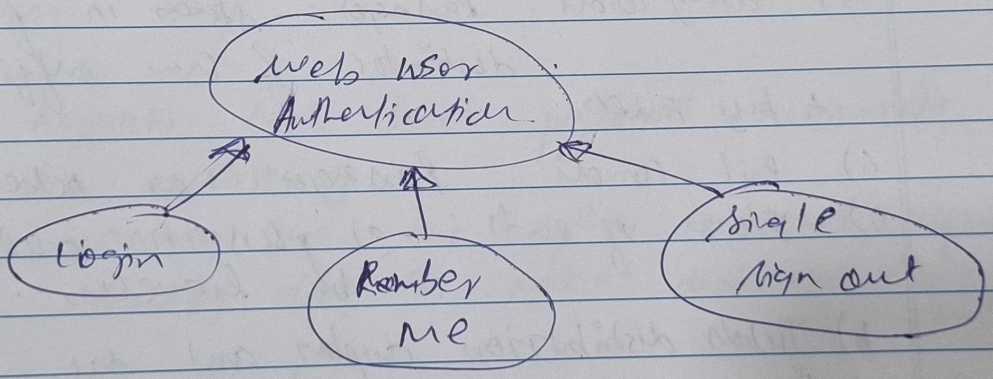
• The direction of a <<includes>> relationship is moving use cases (unlike the direction of the <<extends>> relationship):

Eg:



iii) Generalization :- its between use cases is similar to generalization bet^n classes child usecase inherits properties & behaviour of the parents use case & may override the behaviour of parents.

Eg:



(5) Six point doc. with eg.

→ The 6 point doc is the doc of the use-case diag. It represents the use-case diag. in text based descp. It contains of following 6 point.

1) Use case name: name of main use case
2) Participating actor: the actor that u
3) Entry condition: The conditions that
functionality or indu
met before a parti
or functionality.
4) Exit condition: - It describe the expects or
actions event or state
should be actives at the end of the pro
5) flow of events: Describes the flow of
happen while the interac
with that functionality.
6) Special requirements: Any special requirem
that need to be

eg: 1) Name: Purchase Ticket.
2) Participating Actor: Passenger.
3) Entry condi: Passenger stands in front of
destination & has sufficient
to buy tickets.
4) Exit condi: Passenger has ticket.
5) Flow of events: a) Passenger selects no of
be travelled.
b) Tickets distribution display and are.
c) Passenger inserts money, at least the amount
d) Ticket distributor returns change.
e) Ticket distributor issues ticket.
6) Special requirements: None.

6) How many actors are there in your use case name & explain them.

→ i) Guest: represent individual who interact with your system as external users.

Role: Guest primarily access the system for specific purpose.

ii) Receptionist :- Embodies the staff member responsible for managing interactions between system and guest.

Role :- receptionist use the system to check in guest manage reservation, process payments & add guest inquiries or issues.

iii) Staff: includes various employees within your org. such as housekeeping, maintenance & restaurant staff.

Role: Member utilize system for diff. purpose depending on their role.

iv) Manager: Represents high level personnel responsible for overseeing.

Role :- manager use the system to access reports analyze data & make decisions to improve.

7) List & explain the usecases of your diag.

i) Reservation Management: allows guest to book rooms online or through receptionist.

ii) Check in and check out :- guests can check in & out of standardly. The system handles reg

iii) Room allocation :- This use case ensures rooms are allocated based guest preference & availability. It may also special requests such as connecting room e

iv) Billing & payment :- The system calculates & bills for guests including room charges addi. service & taxes.

v) Inventory management :- This use case keep hotel inventory including type availability & pricing.

vi) Housekeeping & maintenance : use the system to room status & request

vii) Guest service & requests : can make service through system & hotel

viii) Reporting & Analysis : Managers & admin access various rep

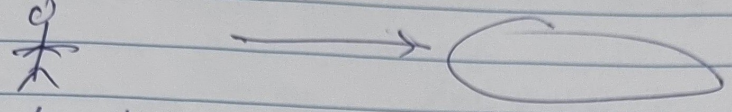
ix) Employee management : System manage emp. schedule. & access c

x) Security & Access control : The system ensures security of guest manage access rights for staff. maintains guest privacy. matters &

xi) Guest feedback & services : The system may e enabling for hotel to gather feedback from gue improve guest. valuable insigh

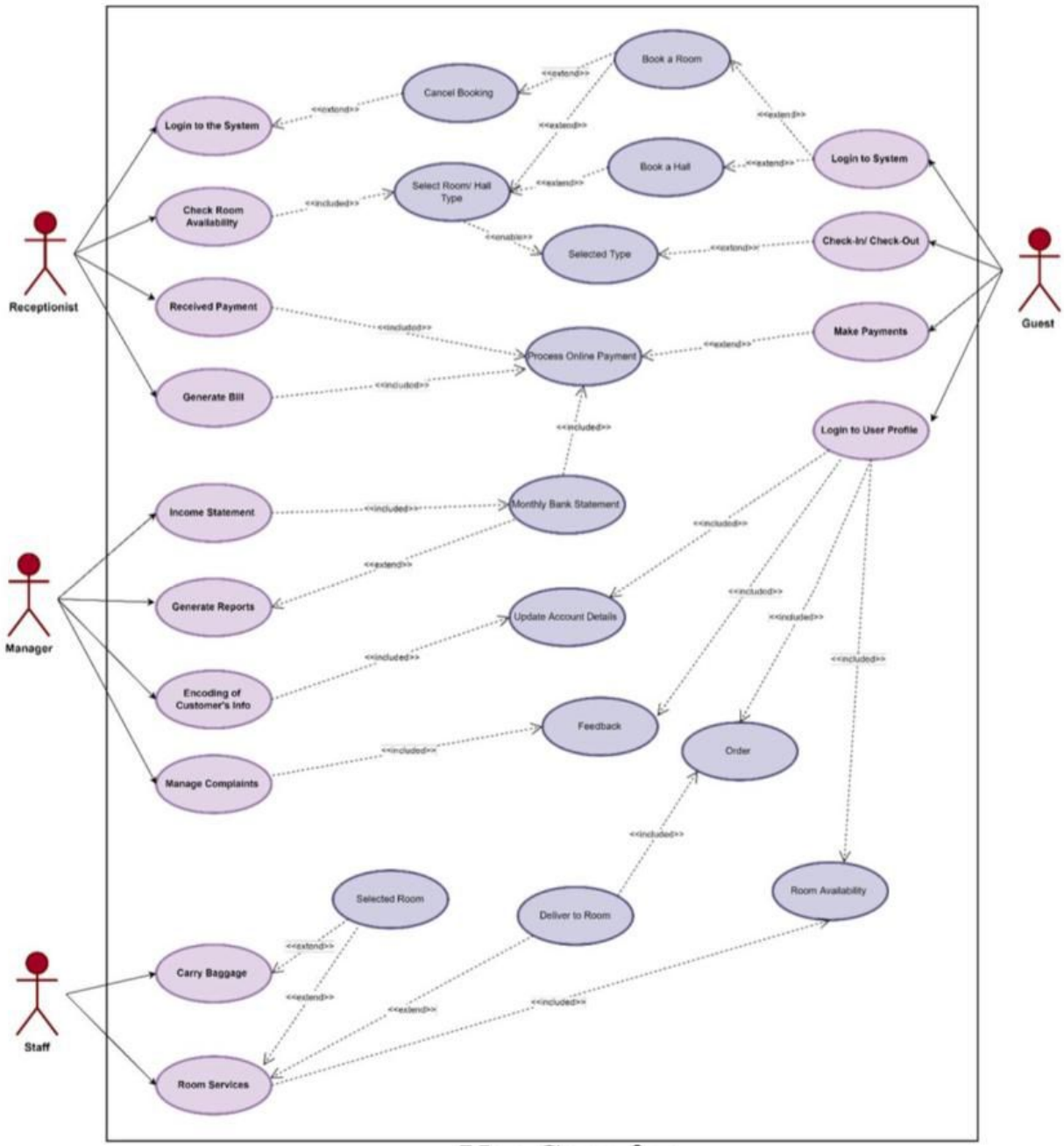
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

(5) 6 point plan. for any 2 inp. fuctionality & diag.



(1) name :- check in / out.
(2) Participating actor : Guest.
(3) Entry condi : ⇒ Logs in with valid in & out password visit the hotel.
→ Guest check the room availability.
(4) Exit Condi : → Guest is allow with a room
→ payment is made on check in date.
(5) Flow of events : → Guest login to system / visits
→ check the room avail. by the receptionist.
⇒ room details & amount
→ Guest selects appropriate room, is allocate the room & check in / out & accordingly due payment

(6) Special Requirement : None.



Use Case for
Hotel Management System

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

-: Experiment 7 :-

Aim: Create a project schedule using gantt chart.

1. What is Project Scheduling: It is the process of defining the tasks & duration of resources required to compare a project. It involves creating a timeline that outlines a project. It involves creating a timeline that outlines when each task should start & finish there an explanation in points.

i) Task Sequencing: Identify the order in which tasks should be perform to accomplish project goals.

ii) Duration Estimation: The time needed for each task.

iii) Resource allocation: assign necessary resources such as: personal, equipment, etc.

iv) Dependencies: Determining task dependencies. While one task may rely on another.

v) Critical path: Identifying the longest sequence of tasks that determines the projects duration.

vi) Gantt chart: Commonly used visual tools to represent project schedules task milestones etc.

vii) Resource leveling: Balancing resource allocation to prevent overallocation.

viii) Project timeline: Creating a timeline that outlines project milestones & deadlines.

ix) Monitoring & control: Continuously tracking progress adjusting schedule & managing risks.

2) How Gantt chart can be used in project
→ it is valuable tools for project scheduling
1) Visualization :- They provide project schedule
of project tasks and their
2) Task scheduling: Gantt charts display tasks
performed in the right order.
dependencies, ensuring tasks
3) Resource mng: They help allocate resources
by showing when specific res
are needed.
4) Critical path Analysis: Identify the critical path
the tasks that could delay the
5) Timeline Tracking: Easily monitor progress,
delays and make adjustments
needed throughout the projects.

eg:

Tasks	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M
Recruitment of projects staff											
PreKAP Evalt.											
Training the Teach.											
workshop											
School health clubs formation											
Activities in school health camp											
Annual meeting											

Task Description	Duration	Week 1					Week 2					Week 3					Week 4					Week 5					Week 6					Week 7					Week 8				
		M	T	W	T	F	M	T	W	T	F	M	T	W	T	F	M	T	W	T	F	M	T	W	T	F	M	T	W	T	F	M	T	W	T	F	M	T	W	T	F
Getting Started																																								
- Analysis	5																																							
- Requirement Gathering	4																																							
- Getting Approval	4																																							
System Design & Architecture																																								
- Detailed Analysis & Data Gathering	7																																							
- Prepare Design Documentation	3																																							
Implementation																																								
- Development	12																																							
- Setting up environment	9																																							
- Quality testing	7																																							
Deployment																																								
- Setup production	2																																							
- Run Beta	4																																							
- Release	1																																							

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

:- Exp. 8 :-

Aim :- Conduct function Point Analysis (FPA) for the project.

(1) What is cost Estimation?

a) Cost estimation in S/W engg is critical process that involves predicting the financial and resource
b) encompasses analyz. various factors such as project scope, complexity, team expertise & tech.
c) Accurate cost estimation is crucial for budget planning, resource allocation.
d) Effect. cost estimation helps org. make informed decisions avoid budget overruns & deliver

(2) List the diff. methods of cost estimation.

→ a) LOC (line of code) → a method that estimates S/W development cost based on the no. of lines of code. written. may inaccurate due to varying complexity & lang.
b) function Point (FP) Analysis :- Measure S/W fn. Estimate cost, complexity inputs, o/p, inquiries, files & interfaces.
c) COCOMO (Constructive Cost Model) :- Employs eqn to estimate cost, effort and time based on proj. charac. like size, complexity and dev. env.

(3) Explain in detail :- function point (FP): (a) FP is a S/W metric used in S/W engg. to measure the func. or feature provided by a SW

b) objective measurement :- is an objective measure technique that assess a functionality independent of the technology or prog.

c) components of FP: I/P → external data and
O/P → result produced for
Inquiries → User Interaction for info
files :- logical data group
Interfaces :- Interact with external

d) eg: for Airline reservation system, FP is calculated, assuming weighting factor as

Info Domain Value	Count		Weighting factor			
			Simple	Avg	Complex	
no of user i/p	4	=	3	4	6	=
no of user o/p	2	=	4	5	7	=
no of user Inquiries	3	=	3	4	6	=
No of files	3	=	7	10	15	=
No of external files	2	=	5	7	10	=
					Total	=

now assigning values to 14 Questions on scale of
so values are {4, 2, 3, 4, 3, 3, 5, 3, 2, 4, 5, 3,
= ΣF_i = 51

∴ FP = Total count * [0.65 + 0.01 * Σ
= 82 * [0.65 + 0.01 *
= 82 * 1.16
= 95.12.

e) Benefits:-
1) helps in estimating project size & complexity.
2) Enables cost & effort estimation.
3) Supports project planning & resources allocation.
4) Facilitates performance measurement & benchmarking.

f) Use cases:- FP often used in SW cost estimation, project management, and as a basic for SW quality assessment and improvement.

g) function point analysis for your application.
→ for Hotel management system, FP computation is calculated using weighting factor as "average"

Info Domain value	Count		Weight factor				Count
			Simple	Avg	Complex		
No of user I/P	5	=	3	4	6	=	20
no of user O/P	5	=	4	5	7	=	25
no of user user queries	6	=	3	4	6	=	24
no of files	5	=	7	10	15	=	50
no of external i/f	5	=	5	7	10	=	35
					Total	=	154

Now Assigned values to 14 questions on scale of 0-5
So values are :- {5, 4, 1, 2, 3, 4, 5, 4, 3, 2, 4, 5}
$= \Sigma f_i = 40$.

$\therefore$ FP = Total Count * [0.65 + 0.01 * Σf_i]
= 154 * [0.65 + 0.01 * 40]
= 154 * 1.05
$\therefore$ FP = 161.7

* Value Adjustment factors.

(1) Does the system require reliable backup recovery.
(2) Are data comm. required.
(3) Are there distributed processing functions
(4) Is performance critical?
(5) Will the system run in an existing heavily operational env.?
(6) Does the system require on-line data entry
(7) Does the online data entry require the transaction to be built over multiple screens or operations?
(8) Are the master files updated online?
(9) Are the I/p, O/p files or inquiries complex?
(10) Is the internal processing complex.
(11) Is the code designed to be reusable?
(12) Are conversion & installation included in design?
(13) Is the system designed for multiple installations in different orgn.
(14) Is the application designed to facilitate change & ease of use by user?

The ans. of all 14 Qns. are as $\Sigma_{i=1}^{14} f_i$.

Roll No 243208

-: Experiment 9 :-

Aim: Application of cocomo model for software estimation of the project.

What is cocomo?

cocomo short for constructive Cost Model is a family of sw cost estimation models developed by Barry Boehm in the 1980s it is used to estimate the effort, time and cost required for software development based on various project parameters. cocomo models are valuable tools for project managers and sw developers in planning and scheduling sw projects.

Three Main types of cocomo models

a. cocomo Basic:
- This is the original model.
- It is suitable for small to medium size project.
- It estimate effort & cost based on line of code (LOC)

b. cocomo Intermediate:
- An extension of the base model
- Suitable for medium to large size projects

c. cocomo II:
- A more detailed & flexible model.
- Suitable for a wide range of project size and type

- Takes into account numerous factors like product, HW, personnel & project attributes

- Stages of calculation:

a. Pre-processing
- Gather project-related data, such as complexity and team capabilities.
- Determine the type of module to use based on project charac.

b. Effort Estimation:
- Calculate the effort required for the project based on cocomo equations.
- Effort is often measured in person-months or person-hours

c. Cost Estimation:
- Once effort is estimated, determine the cost using factors like personnel salaries, equipment costs and overhead expenses.

d. Schedule Estimation:
- Calculate the expected project duration based on the effort estimate and project schedule factors.

Formulae:

The cocomo modf formula for effort estmation in its basic fom is.

Effort (E) = a* (KLOC) ^ b.

where.

E → Effort in person months.
a & b → Constants based on project attribute.
KLOC → represents the estimated size of the s/w in thousands of line of code.

Example of Three Steps

Us apply cocomo to estmate the cost of developing a Mobile Management Syst.

Pre processing:
Syst size. estmate: 100 KLOC.
Projet type: Organic

Effor Estmation.

ing cocomo basic; E = a*(100*b
o organic projct a = 2.4 and b = 1.05.

E = 2.4 * (100 ^1.05) =
∴ 242 person-months.

(c) Cost Estimation

Assuming a avg team member salary is $5000 per month.

Cost = Effort * Salary per month.
Cost = 242 person months * $ 5,000 =
= $ 1210,000

(d) Schedule Estimation.

Schedule = Effort / Team size
Assuming a team size of 10 people
Schedule = 21 months.
person-months / 10 = 24.2 months

:- Experiment 10 :-

Aim :- Develop a Risk Mitigation, Monitoring and Management Plan (RMMM) for the project

Theory :-

1. What is risk in SE Development ?

→ Risk in SE dev refers to the likelihood of unfavorable events or circumstances that can potentially disrupt the projects progress or impact its success. These risks can manifest in various forms and at diff. stages of the SE lifecycle. Effective risk management aims to minimize the impact of these risks and increase the likelihood of the project success

2. Different Types of Risk

→ 3 Important categories of risk

1) Project risks : That directly affect the schedule of the project and the resources involved in the development

Ex: after project loss : loss of an experienced developer and designer.

2) Product risk : The product risk of the application the quality and of built.

Ex of: failure of purchased component perform as per expectation.

3) Business Risk : Risk affecting the those develop and process the software.

Ex: A competitor of the organization int a new product.

- Risk Management Strategies.

Risk	Categories	Probability	Impact	RMMM
Developer Turnover	Personnel Risks	Medium	High	- Cross train member to dependency on experts. - Maintain d Repo. to do project info - Monitor Moral an address co proactivity.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE
Roolan
2113208

Data Security.	Operational Risk	low	very high	- Implement robust encryption and access control for sensitive data. - Regular security audits and penetration testing. - Incident response plans for data breaches.
Hard deadline	Project Risks	High	High	- Use agile development methodologies to accomodate changes - Regularly track project schedule and resource allocation. - Identify potential bottlenecks and contingency plans.

- Risks specific to Hotel Management System

Risk	Categories	Probability	Impact	RMMM
Integration Challenges	Technical Risks	Medium	High	- Conduct thorough integration testing Identify and address potential issues - Use standardized interfaces and protocols facilitate integration external systems - Allocate additional in the project schedule for unexpected challenges.
Regulatory Compliance	Business Risks	low	Very high	- Continuously monitor changes in regulatory requirements. - Establish clear and compliance - Conduct regular Compliance audits to ensure adherence to regulations

-: Experiment 11 :-

Aim:- Case Study: GitHub for version Control.

Experiment No 11

Aim: Case study of GitHub and version control

1. What is Github?

GitHub is a web-based version control and collaboration platform for software developers. Microsoft, the biggest single contributor to GitHub, acquired the platform for $7.5 billion in 2018. GitHub, which is delivered through a software as a service (SaaS) business model, was started in 2008.

Git is used to store the source code for a project and track the complete history of all changes to that code. It lets developers collaborate on a project more effectively by providing tools for managing possibly conflicting changes from multiple developers.

GitHub allows developers to change, adapt and improve software from its public repositories for free, but it charges for private repositories, offering various paid plans. Each public and private repository contains all of a project's files, as well as each file's revision history. Repositories can have multiple collaborators and can be either public or private.

*Benefits and features of GitHub:

GitHub facilitates collaboration among developers. It also provides distributed version control. Teams of developers can work together in a centralized Git repository and track changes as they go to stay organized.

Other products and features of note include the following:

- **GitHub Gist** lets users share pieces of code or other notes.
- **GitHub Flow** is a lightweight, branch-based workflow for regularly updated deployments.
- **GitHub Pages** are static webpages to host a project, pulling information directly from an individual's or organization's GitHub repository.
- **GitHub Desktop** lets users access GitHub from Windows or Mac desktops, rather than going to GitHub's website.
- **GitHub Student Developer Pack** is a free offering of developer tools for students. It includes cloud resources, programming tools and support, and GitHub access.
- **GitHub Campus Experts** is a program students can use to become leaders at their schools and develop technical communities there.

2. What is Version control?

Version control is a system that records changes to files over time so that you can recall specific versions later. It is primarily used in software development but can be applicable to any kind of files.

The main purposes of version control are:

1. Tracking Changes: Version control tracks changes made to files, providing a history of modifications, additions, and deletions. Each change is typically accompanied by a description of what was altered and why.

2. Collaboration: Multiple developers can work on the same files concurrently without interfering with each other's work. Version control systems allow for merging changes made by different contributors.

3. Reverting to Previous Versions: Developers can revert to a previous version of a file or project if a mistake is made, allowing for easy error correction and recovery.

4. Branching and Merging: Version control facilitates the creation of branches, which are independent lines of development. Branches allow for experimentation and the implementation of new features without affecting the main project. Merging combines the changes from one branch to another.

5. Documentation and Annotations: Developers can provide comments and annotations to describe changes made, aiding in understanding the evolution of the project.

There are various version control systems available, with Git being one of the most widely used in the software development industry. Git, for instance, is distributed, meaning that each user has a complete copy of the repository, allowing for work to be done offline and fostering a decentralized approach to collaboration. Other version control systems include Subversion (SVN) and Mercurial.

3. Case study of any one application using github

Topic: Development of the "Atom" Text Editor

Background:

Atom is a popular, open-source text editor developed by GitHub. It's known for its extensibility, ease of use, and customization features. The development of Atom involves a global community of contributors and relies heavily on GitHub for version control and collaboration.

GitHub Implementation:

1. Repository Setup:

GitHub hosts the "Atom" repository, making it accessible to the public. The repository contains all the source code, documentation, issues, and pull requests related to the project.

2. Collaborative Development:

A diverse community of developers from around the world contributes to the Atom project.

Contributors can fork the repository, make changes in their own branches, and then submit pull requests to the main Atom repository to propose changes.

3. Version Control:

GitHub's version control system tracks all changes made to the Atom codebase over time.

Developers commit code changes to their respective branches and provide descriptive commit messages to explain the purpose of each change.

4. Pull Requests and Code Review:

When a contributor wants to submit a change, they create a pull request (PR) on GitHub.

The PR serves as a discussion space where other developers can review the code, provide feedback, and discuss potential improvements.

Continuous integration (CI) tools are used to automate testing, ensuring that proposed changes do not introduce regressions.

5. Issue Tracking:

GitHub Issues is used for tracking bugs, feature requests, and enhancements.

Contributors and users can report issues, and maintainers assign, prioritize, and label them accordingly.

Issues often serve as a starting point for contributions.

6. Branching Strategy:

Atom's repository uses a branching strategy that includes a "master" branch for stable releases and a "beta" branch for pre-release versions.

Contributors create feature branches to work on specific enhancements or bug fixes.

7. Continuous Integration/Continuous Deployment (CI/CD):

GitHub Actions is used for automating various tasks, such as building, testing, and packaging Atom releases.

This ensures that the project maintains a high level of quality and is continuously available for users.

8. Documentation and Wiki:

GitHub's Wiki feature is used to maintain comprehensive documentation for Atom, including installation guides, customizations, and API references.

9. Community Involvement:

Atom's development thrives on community involvement. GitHub provides a platform for both GitHub employees and external contributors to collaborate and improve the project.

10. Releases and Changelogs:

GitHub facilitates the release management process, including the creation of release notes and changelogs, making it easy for users to understand what's new in each version.

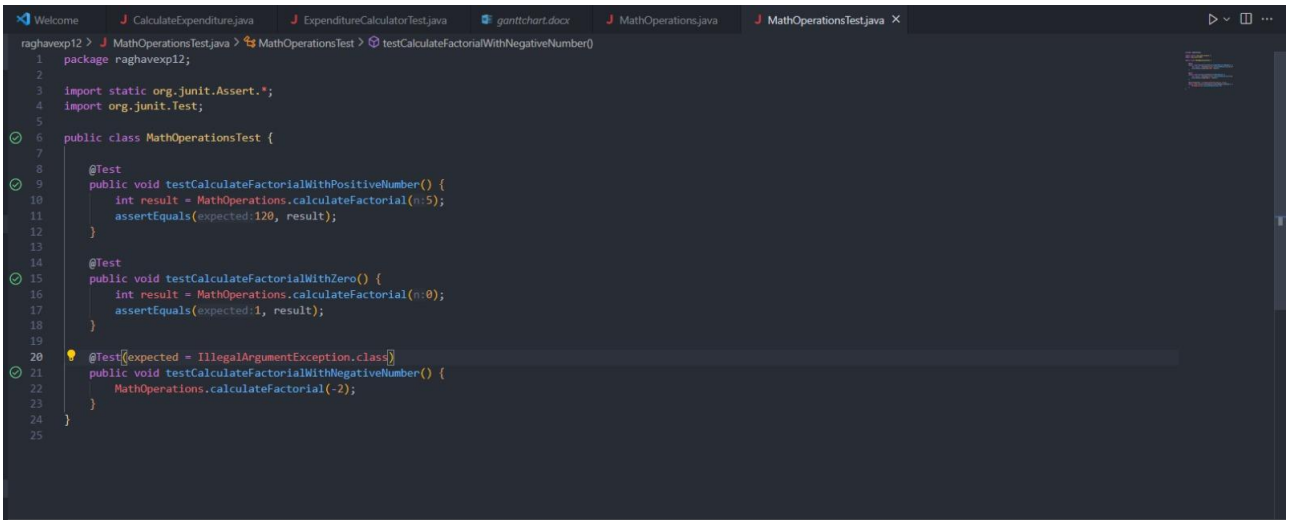
By leveraging GitHub for version control, collaboration, and issue tracking, the Atom text editor has become a widely adopted and highly extensible tool for developers and writers alike. The open-source nature of the project and the use of GitHub have fostered a thriving community and continuous improvement of the software.

Java Code

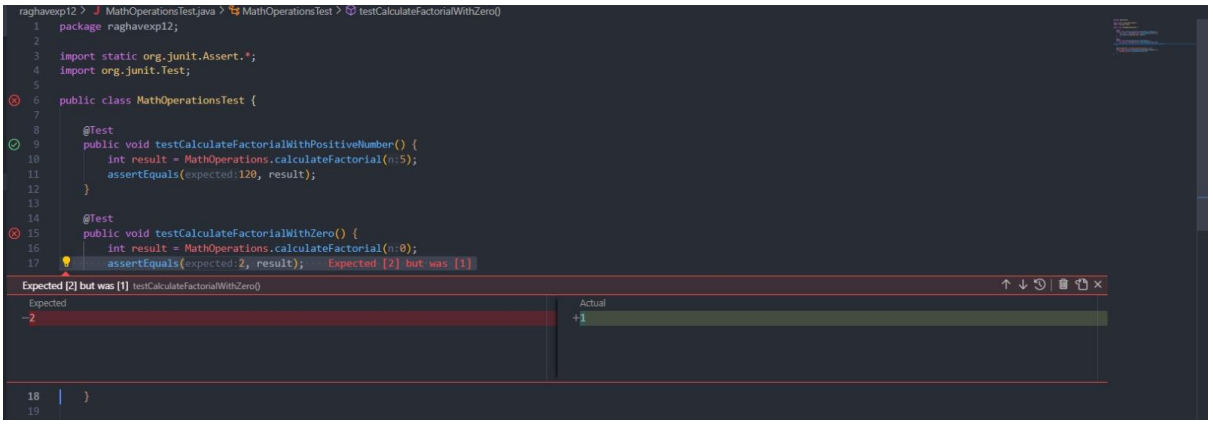
```
public class MathOperations {

    public static int calculateFactorial(int n) {
        if (n < 0) {
            throw new IllegalArgumentException("Input must be a non-negative integer.");
        } else {
            int result = 1;
            for (int i = 1; i <= n; i++) {
                result *= i;
            }
            return result;
        }
    }
}

```

```
import static org.junit.Assert.*;
import org.junit.Test;

public class MathOperationsTest {

    @Test
    public void testCalculateFactorialWithPositiveNumber() {
        int result = MathOperations.calculateFactorial(5);
        assertEquals(120, result);
    }

    @Test
    public void testCalculateFactorialWithZero() {
        int result = MathOperations.calculateFactorial(0);
        assertEquals(1, result);
    }

    @Test(expected = IllegalArgumentException.class)
    public void testCalculateFactorialWithNegativeNumber() {
        MathOperations.calculateFactorial(-2);
    }
}
```

Testing



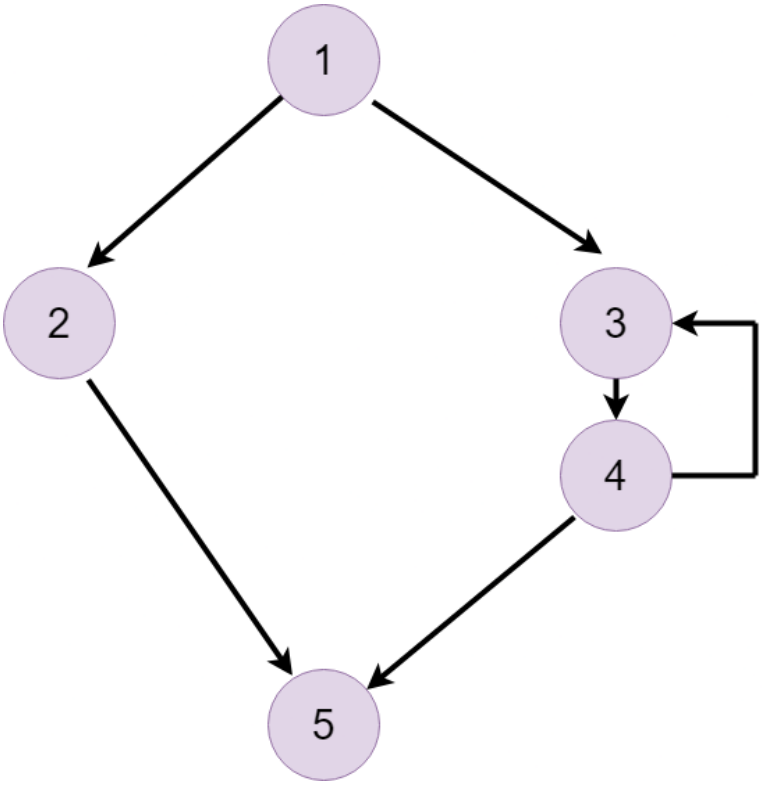
```
Welcome | CalculateExpenditure.java | ExpenditureCalculatorTest.java | ganttchart.docx | MathOperations.java | MathOperationsTest.java
raghavexp12 > MathOperationsTest.java > MathOperationsTest > testCalculateFactorialWithNegativeNumber()
1   package raghavexp12;
2
3   import static org.junit.Assert.*;
4   import org.junit.Test;
5
6   public class MathOperationsTest {
7
8       @Test
9       public void testCalculateFactorialWithPositiveNumber() {
10          int result = MathOperations.calculateFactorial(n:5);
11          assertEquals(expected:120, result);
12      }
13
14      @Test
15      public void testCalculateFactorialWithZero() {
16          int result = MathOperations.calculateFactorial(n:0);
17          assertEquals(expected:1, result);
18      }
19
20      @Test(expected = IllegalArgumentException.class)
21      public void testCalculateFactorialWithNegativeNumber() {
22          MathOperations.calculateFactorial(-2);
23      }
24  }
25
```



```
raghavexp12 > MathOperationsTest.java > MathOperationsTest > testCalculateFactorialWithZero()
1   package raghavexp12;
2
3   import static org.junit.Assert.*;
4   import org.junit.Test;
5
6   public class MathOperationsTest {
7
8       @Test
9       public void testCalculateFactorialWithPositiveNumber() {
10          int result = MathOperations.calculateFactorial(n:5);
11          assertEquals(expected:120, result);
12      }
13
14      @Test
15      public void testCalculateFactorialWithZero() {
16          int result = MathOperations.calculateFactorial(n:0);
17          assertEquals(expected:2, result);    Expected [2] but was [1]

Expected [2] but was [1] testCalculateFactorialWithZero()
Expected                    Actual
-2                          +1

18      }
19
```

Flow Graph

\



Cyclomatic Complexity for the above flowgraph

E(Number of Edges) = 5
N(Number of Nodes) = 4
P(No. of predicate Nodes) = 2
Cyclomatic complexity, V(G) = E-N+2
Therefore,
V(G) = 5-4+2 = 3

Or
V(G) = P + 1
V(G) = 2 + 1 = 3

Therefore, the cyclomatic complexity of the program code and it's drawn flowgraph is 3

Aayush 2113208

:- Experiment 12 :-

What is Testing.

Testing is a crucial phase in software development that involves the evaluation of a software application to identify defects ensure its functionality meets requirements and verify that it performs as expected. Testing helps improve software quality, reliability and security.

Different Types of Testing.

There are basically 10 Types of Testing:

Unit Testing.
Integration Testing.
System Testing.
Functional Testing.
Acceptance Testing.
Smoke Testing.
Regression Testing.
Performance Testing.
Security Testing.
User Acceptance Testing.

Types of software test →

- Automating Testing
- Manual Testing
  - Grey Box
  - White Box
  - Black box

Manual Testing

- Functional Testing
  - Unit Testing
  - Integration Testing
    - Incremental Testing
      - Top down
      - bottom up
    - Non-Incremental Testing
  - System Testing
- Non-functional Testing
  - Performance Testing
    - load Testing
    - Stress Testing
    - Scalability Testing
    - Stability Testing
  - Modularity Testing

Unit testing: It involves testing individual code components or functions in isolation, ensuring they produce the expected output. It is typically automated and helps identify defects early in the development process.

Integration Testing: It checks how different components or modules interact when combined. It ensures the data flows smoothly between them and that they work together as a cohesive system.

System testing: it evaluates the entire SW system as a whole. it assess whether the integrated components function correctly and meet specific requirements, focusing on the system behavior and features.

Functional Testing: Functional testing ensures the SW func and features to ensure they operate according to predetermined requirements. Testcases are designed to validate that the software performs its intended tasks accurately.

Acceptance Testing: Acceptance testing is the final phase before deployment and involves end-user or stakeholders testing the SW to determine if it meets their acceptance criteria. It verifies that the SW is ready for production use.

Smoke Testing: it is a preliminary test to check essential and critical functionalities of the software work without major issues. it helps quickly identify showstopper defects.

Regression Testing: it ensures that new code changes do not introduce defects or break existing functionality by re-running previously executed test cases often automated to save time.

Performance Testing: evaluates the sw speed, scalability and stability under various conditions. This includes load testing, stress testing and capacity testing to measure performance metrics.

~~Security~~ Testing.

Security Testing:- it assesses the s/w resistance to security threats and vulnerabilities it includes penetration testing, vulnerability scanning and authentication check to protect data and functionality.

User Acceptance Testing:- UAT involves end-users or stakeholders validating that the s/w meets their business needs and operates as expected in a real-world environment, ensuring it meets user expectations before deployment

Explaining White Box Testing:

White Box testing, also known as structural or glass box testing, examines the internal structure, code logic and paths within a software application. Testers use knowledge of the code to design test cases that exercise specific code paths. This testing method aims to ensure that all code branches, statements and conditions have been tested.

Key steps in White Box Testing:

Identify code paths: Analyze the source code to identify possible execution paths.

Design test cases: Create test cases that target specific code paths.

Execute test cases: Run the test cases and record the results.

Analyze results: Check if the code executes as expected and verify any discrepancies.

Repeat: Repeat the process until all code paths are tested and issues are resolved.

Junit is

Junit is widely used testing framework for Java that supports White Box testing by providing a framework for writing & running test cases.

- Cyclomatic complexity :-

It is a software metric that quantifies the no. of independent paths or decision points in a program. It helps assess code complexity by indicating the minimum no. of test cases required to cover all possible paths thus aiding in code quality analysis. One of the three ways to compute cyclomatic complexity.

V(G) for a flowgraph, G, is defined as

V(G) : E - N + 2

where E → no of edges in flowgraph
N → no of flowgraph nodes.

or V(G) = P + 1

where P is no of predicate nodes
edges.

A1-1

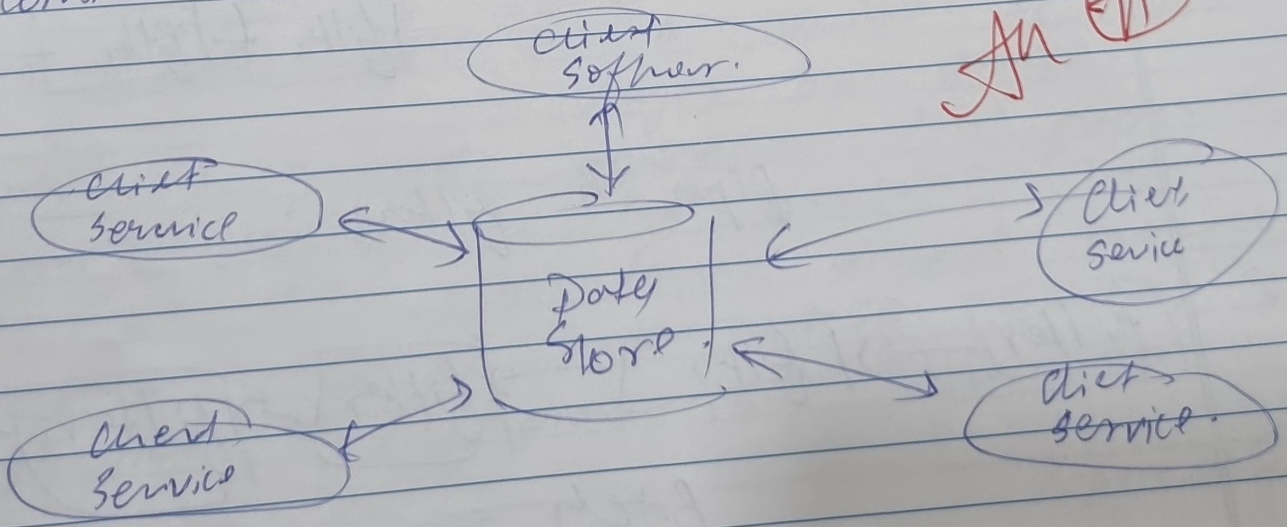
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Assignment 1:-

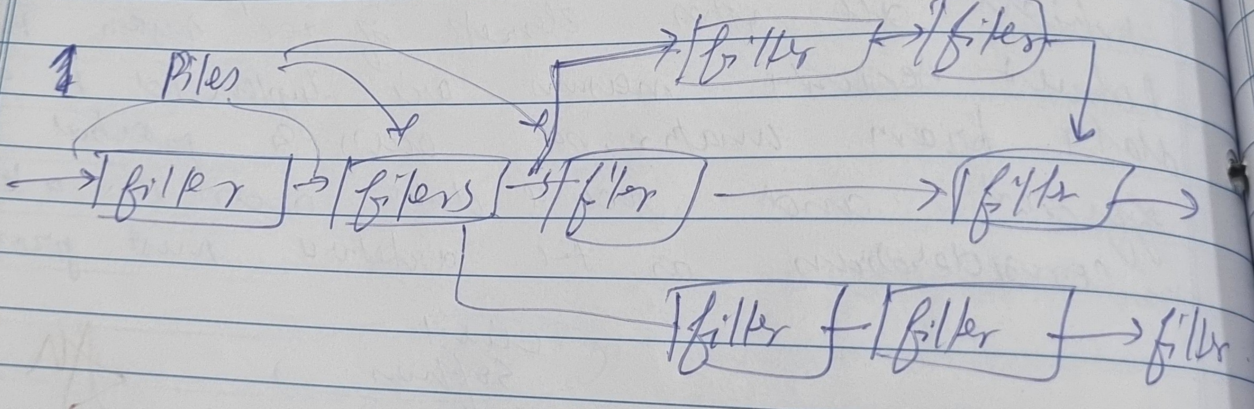
Architecture Design: Explain in detail each type with example.

The s/w architecture of a program or computing system is the structure or structures of the sy. which comprise the s/w components the externally visible properties of those components and the relationships among them.

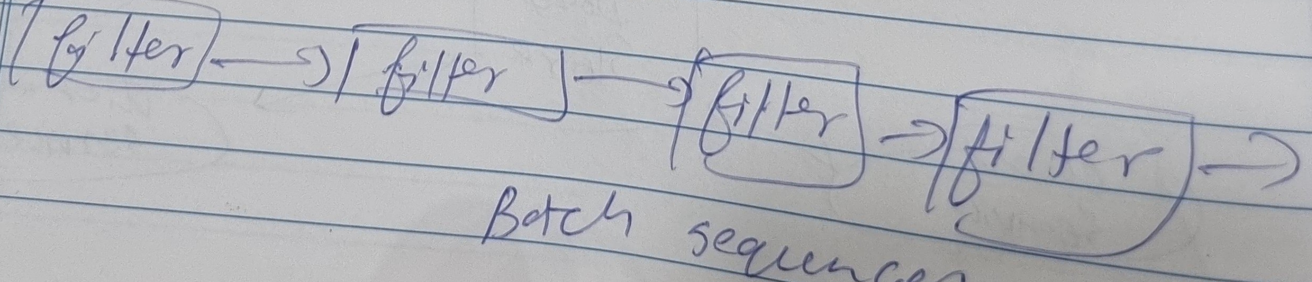
1) Data centered architecture: A data center architecture places data at the forefront of system design with data being stored and accessed as its imp. focus. In this architectural approach, data is considered the central and most valuable component around which all other elements of the system revolve. Robust security measures are implemented to safeguard data from unauthorized access & potential breaches. Efficient and secure data access is a key consideration as the architecture must provide.



(2) Data flow architecture: This archi. is applied when the i/p data are to be transformed through a series of computation or manipulative components into o/p data. A pipe and filter pattern has a set of components. Inadequately lie upstream and downstream and is design to expect data input. The filter does not require knowledge of the working of its neighbouring filter. It encourage reuse, repurposing and modification with this design concurrent execution is supported.



Pipes & filters.



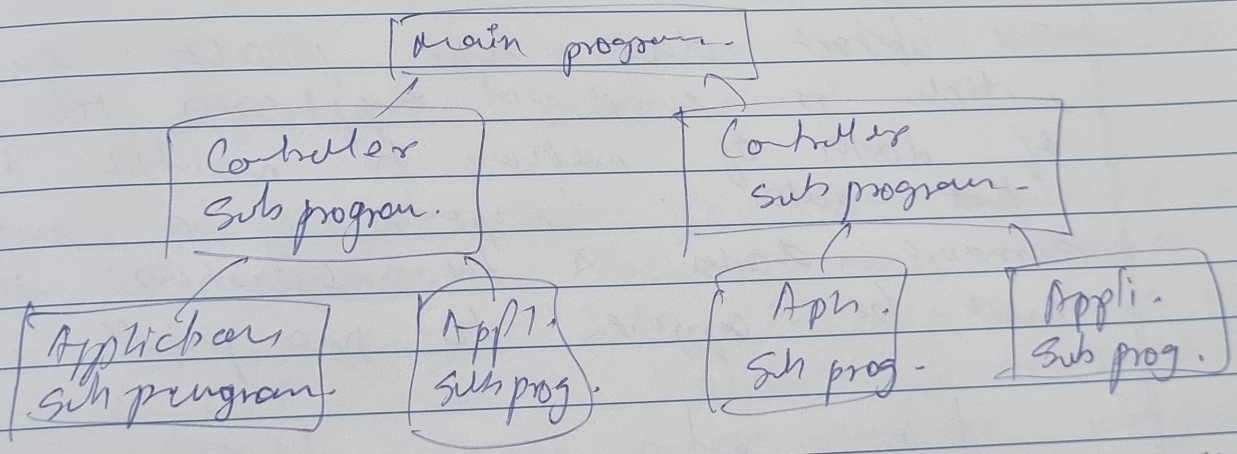
Batch sequences.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

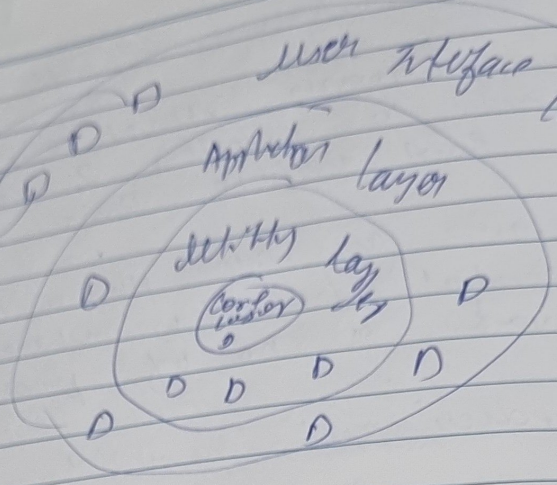
Call & return architecture:- Style enable a s/w design to achieve a program structure that relatively easy to modify & scale two sub-styles exist within this category.

① Main/sub program architecture: Program structure decomposes into a control hierarchy where a "main" program invoks a no of programs.

② Remote procedure call architecture: The components of main program. subprogram architecture are distributed across.

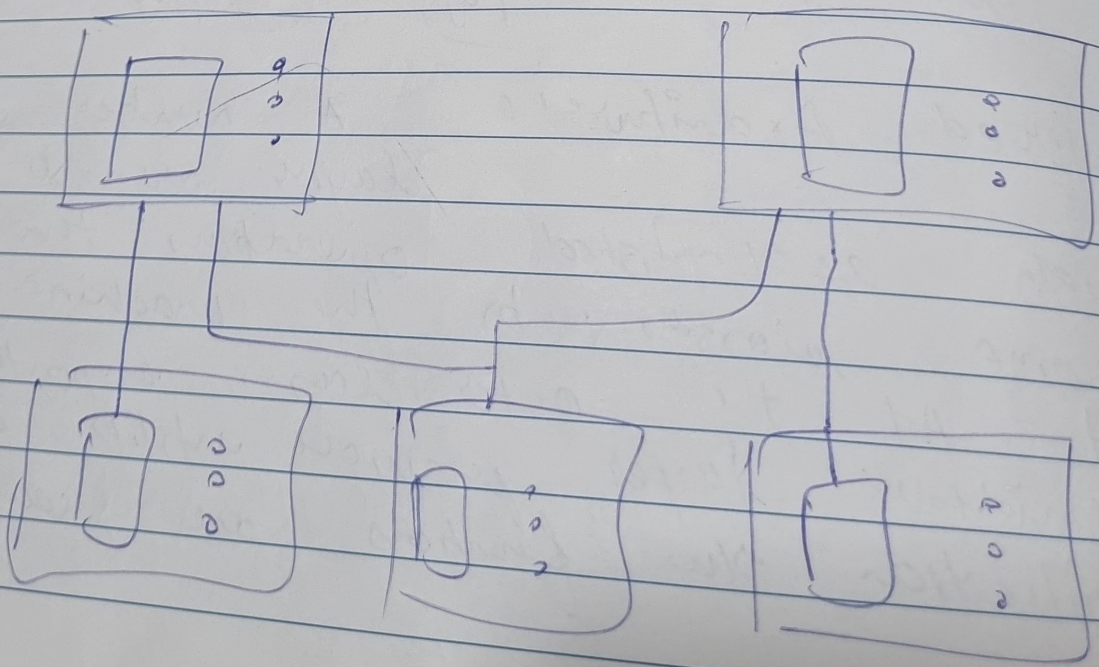


Layered Architecture: A number of diff. layers are design. each accomplished operations that progressively become closer to the machine inst. set. At the outer layer components. Intermediate layer provide utility services & application s/w functions are common.



(c) Object oriented architecture:

The abstract data type paradigm from which is improved emphasizes the building of data of methods to manipulate & access that data. Components of a system manage data & the operations that must be applied to manipulate.



THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Assignment 2

Aim :- Explain in detail :-

S/w Maintance :- It is the process of modifying a s/w product after it has been delivered to the customers.

The main purpose of s/w maintance is to modify and update s/w applications after delivery correct faults and to improve performance.

It is done after the product has s/w launched for several reasons including improv. of s/w. overall working issues or bugs, to boost performance and more.

Following are some reasons for s/w maintance -

1) Market Conditions: Policies, which changes over the times such as taxation and newly introduced constraints like, Reimbursement bookkeeping may trigger need for modifications.

2) Client Requirements: Over the time, customer may ask for new features or functions.

3) Host modification:- Any of h/w & s/w platform (Such as OS) of the host changes s/w changes are needed to keep adaptation.

4) Organisation changes: Any business level change at client side in origin s/w

There are 4 Types of s/w maintenance
a) Corrective maintence: A s/w product needs maintence to support the new features that the users want.

b) Adaptive maintance :- Includes modification and updating when the customer need the product to run on new platform or new OS.

c) Perfective maintance: A system or product needs maintenance to support the new features that the users want or to change diff. type of system

d) Preventive maintence :- This maintance includes modification and updation to prevent future problem of s/w. It goal to attend problem which are not seen at the moment

- **Re - Engineering :-** When we need to update the s/w to keep it to current market without impacting its functionality it is called s/w re-engineering

It is a thought process where the design of s/w is changed and programs are re-written.

ex. initially Unix was developed in assembly lang. When lang c. came into existence Unix was re-engg in c because working in assembly lang was difficult.

Other than this, sometimes programmers notice that few parts of s/w need more maintenance than others and they also need re-engineering.

The 5 steps are.

i) Decide what to re-engg. It is whole s/w or a part of it.
ii) Perform Reverse engg. in order to obtain specifications of existing s/w.
iii) Restructure Program if required.
iv) Re-structure data as required.
v) Apply forward engineering concept in order to get re-engineered s/w.

Reverse Engg :- It is a process to achieve system spec. by thoroughly analysing, understanding the existing sys. This process can be seen as reverse SDLC model. i.e we go backward from abstraction levels.

S/w Reverse engg is the process of recovering the design and the requirements specifications of a product from an analysis of its code.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

An existing system is previously implemented design about which we have known little or get it design. with design in mind they try to code to system specifications.

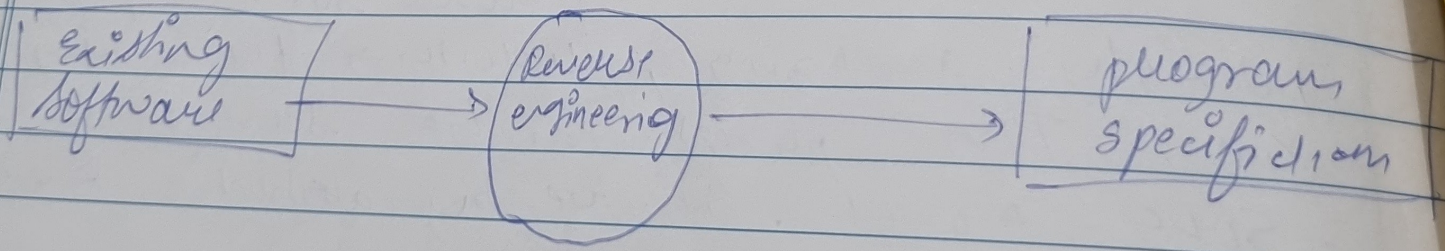
Some reasons for reverse engg.

- Providing proper system documentation.
- Recovery of lost info.
- Assisting with maintenance.
- Facing of sw decense.
- Discovering unexpled flow or faults.

S/w reverse engg is the process of recovering the design & the requirement specification of a product from an analysis of its code.

Reverse Engineering is usally done on previously implemented design engg. by looking at the code and try to get the design.

Diagram for the process



Reverse Engineering of s/w is a tedious part because usually the s/w is bundled via obfuscation.